Parameter estimation of multivariable processes represented by stochastic models in pseudo-canonical form

Veltmeijer, A.J.M.

Award date:
1985

4964

DEPARTMENT OF ELECTROTECHNICAL ENGINEERING
EINDHOVEN UNIVERSITY OF TECHNOLOGY
Group Measurement and Control

PARAMETER ESTIMATION OF MULTI-VARIABLE PROCESSES REPRESENTED BY STOCHASTIC MODELS IN PSEUDO-CANONICAL FORM.

by A.J.M. Veltmeijer

This report is submitted in partial fulfillment of the requirements for the degree of electrotechnical engineer (M.Sc.) at the Eindhoven University of Technology.
The work was carried out from sept.1984 untill aug.1985 in charge of Prof.dr.ir. P. Eykhoff and under supervision of Dr.ir. A.A.H. Damen and Ir. P.M.J. Van den Hof.

SUMMARY

The result of a multi-variable state space model parameter estimation is only unique if canonical forms are estimated. These canonical forms are unique in their parameters and in their structure. Besides the observability canonical forms, also the pseudo-canonical forms are described in this report, which do not have a unique structure.

If the estimated model has to have 'good' simulation properties, the error made by simulation, the output error, should be kept as small as possible according to some criterion.
For good one-step-ahead prediction performances a prediction error should be minimised. For ARMA-models a prediction error can be defined which is linear in the parameters and identical to the equation error. In this report equation- and prediction errors are generalised to the multi-variable case and translated to the state space representation.
To improve the one-step-ahead prediction error, a general innovation error is introduced in state space models and translated to input-output descriptions to complete the comparison of the two representations.

An inter-active program is implemented for estimation of a output-, equation-, prediction- or innovation error state space model in (pseudo-) canonical form. Many options are implemented to have different estimation conditions and different validations of the results.
Test runs are given to show all derived relations between different errors and structures. Practical data is used to show the practical applicability of the program.

SAMENVATTING

Het resultaat van een parameter schatting voor multi-variabele toestandsmodellen is alleen eenduidig als kanonieke vormen geschat worden. Deze kanonieke vormen zijn uniek in de parameters en in de struktuur. Behalve de observeerbare kanonieke vormen worden in dit verslag ook de pseudo-kanonieke vormen beschreven, die niet uniek zijn in de struktuur.

Als een geschat model 'goede' simulatie eigenshappen moet hebben, moet de simulatie fout, de output error, zo klein mogelijk gehouden worden volgens een gegeven criterium.
Voor een goede voorspelling van de eerst volgende uitgang moet een voorspellings fout, de one-step-ahead prediction error, geminimaliseerd worden. Voor modellen met één ingang en één uitgang kan een predictie fout gedefinieerd worden die linear is in de parameters en identiek aan de fout in de vergelijking, de quation error. In dit verslag zijn de predictie en de vergelijkings fout gegeneraliseerd naar het multi-variabele geval en vertaald naar het toestandsmodel.
Ter verbetering van de predictie fout is een algemene innovatie fout, de innovation error, geïntroduceerd in het toestandsmodel en vertaald naar input-output modellen om de vergelijking tussen de twee model representaties te completeren.

Een inter-actief programma is geïmplementeerd voor het schatten van output-, equation-, predictie- of innovatie error toestandsmodellen in (pseudo-)kanonieke vorm. Vele opties zijn geïmplementeerd voor het verkrijgen van verschillende schattings condities en verschillende mogelijkheden om de resultaten te laten zien.
Test schattingen zijn gegeven om de afgeleide relaties tussen de verschillende fouten en strukturen te laten zien. Data uit de praktijk is gebruikt om de praktische toepasbaarheid van het programma te laten zien.

Preface

This report is the result of a final project before masters degree at the department for Electrotechnical engineering at the university of technology in Eindhoven. The work was carried out at the group Measurement and Control. In this group a main research subject is 'System Identification' and 'Parameter Estimation'. A result of the research efforts in the last years in this field, is the interactive program package SATER. This program package enables the user to identify processes and estimate models and model order of Single-Input Single-Output (SISO) systems in an interactive way. Due to many other implemented feasibilities, and the users friendly implementation, SATER has also become an important educative tool (see van den Boom and Bollen 1983).

Because of the progress in recent years of multi-variable identification, a similar (simplified) program package as SATER, for Multi Input Multi Output (MIMO) systems has become a possibility. To make a first move towards such a package, several identification routines have been accomplished in the group for the last two years. In this report, the mathematical aspects, implementation and results of one such a routine are described.

This report and the implemented estimation routine could not have been accomplished without the help of the members of the group M>R. I want to thank everybody in this group that has been helpful to me in the last year, specially prof.dr.ir. P. Eykhoff and above all dr.ir. A.A.H. Damen and ir. P.M.J. Van den Hof, who supported and guided me very well.

CONTENTS

0. Introduction

For several purposes, it might be useful to have a mathematical description of a process/system.

- Simulation of the process
- Prediction of future behaviour of the process
- Process optimisation
- Stability analysis

To get such a description, the process has to be identified from available data and physical insight in the process.

System identification is a wide defined conception, and many aspects are covert by it. A rough distinction between different aspects of identification can be given by

1. Process definition
2. Structural identification
3. Parameter estimation
4. Model validation

First we have to know the input and output variables that are of any interest to us, from the estimation point of view (1.). After the inputs and outputs of the process are know, we have to determine the order/complexity of the (sub)system(s) to be able to choose a parameter set that might give good estimation results (2.).
If the structure of the process is identified, we have to choose a model type suitable for our purpose, and suitable for the available estimation techniques (3.). After estimation we have to check if the estimated model is suitable for its purpose (4.).

In this report we will assume that stage 1. and 2. have been accomplished with satisfying results. We will consider several stochastic models, each with its own specific advantages for a specific purpose.

The estimation is described and implemeneted for one mathematical representation, the State Space Model (SSM), but the relation with another model, the Matrix Fraction Description (MFD), is described extensively (see sect. 0.1).
In the state space representation 4 different stochastic models are described and implemented (see sect. 0.2). The Equation Error Model (EEM) because of its mathematical advantages, the Prediction Error Model (PEM) for its predictive advantage, the Output Error Model (OEM) for good simulations and the Innovation Error Model (IEM) because it covers the PEM and OEM (the EEM only in special cases).
The estimation program is implemented on a VAX computer system in Fortran'77

0.1 System Representation

Two of the many possible descriptions of a linear relation between in- and output variables of a process are treated extensively in this report

The Matrix Fraction Description (MFD) gives a relation in the next form

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) \tag{0.1}$$

with P(z) and Q(z) matrix polynomials in the time shift operator z.

$$\begin{aligned} P(z) &= \left[\ p_{ij}(z)\ \right] \\ Q(z) &= \left[\ q_{i\ell}(z)\ \right] \end{aligned} \tag{0.2}$$

If the degrees of the polynomials p(z) and q(z) (structure of the model) are well defined this system representation is unique in the parameters. This means that for a specific MFD no other MFD with the same number of parameters at the same places in p(z) and q(z) is able to describe the same input-output relation. If no restrictions are made on the parameters, the MFD may represent a non causal system.

Given eq.(0.1) the transfer function representation is given by

$$\underline{y}(k) = P^{-1}(z)Q(z)\underline{u}(k) \qquad (0.3)$$

The State Space Representation (SSM) of a system is given by

$$\begin{aligned} \underline{x}(k+1) &= F\underline{x}(k) + G\underline{u}(k) \\ \underline{y}(k) &= H\underline{x}(k) + D\underline{u}(k) \end{aligned} \qquad (0.4)$$

With F, G, H and D matrices without time shifts and $\underline{x}(k)$ the state vector that contains all past information.
Due to this intermediate vector $\underline{x}(k)$, a SSM is a very compact and clear way to describe a system and it allows the use of physical knowledge about the system (see Elgerd 1967).
The state space representation is not unique in the parameters. But every quantitively defined SSM (eq.(0.4)) can be brought into one or more pseudo-canonical forms that are unique in the parameters (no other SSM with the same structure is equivalent to this pseudo-canonical model). Also one canonical form exists that apart from uniqueness in the parameters, is unique in its structure. This means that no other canonical form with a different structure (but same order) is equivalent to this model. The (pseudo-) canonical representations are given by the matrices $\{A,B,C,D\}$ instead of $\{F,G,H,D\}$.
From eq.(0.4) it is clear that every SSM represents a causal system.

Eq.(0.4) can be written as (in pseudo-canonical form)

$$\underline{y}(k) = CA^{k}\underline{x}(0) + CA^{k-1}B\underline{u}(0) + \ldots + CB\underline{u}(k-1) + D\underline{u}(k) \qquad (0.5)$$

Given eq.(0.5), the impuls response or Markov-parameters given in SSM-parameters become:

$$\left(CA^{k-1}B \right) = M(k) \quad ; \quad M(0) = D \qquad (0.5\text{-a})$$

Where $M_{ij}(k)$ is the response of output i to an impulse on input j.
The (pseudo-)canonical form(s) of (0.4), with the parameters in matrices A, B and D, are the forms that will be estimated.

0.2 Parameter Estimation

During estimation one tries to find that parameter values for a given model representation, which fit the given I/O-data set as good as possible.
Because of all kind of noises and because we may try to model the process with a wrong order (see Damen et al, 1985), the fit will never be perfect.

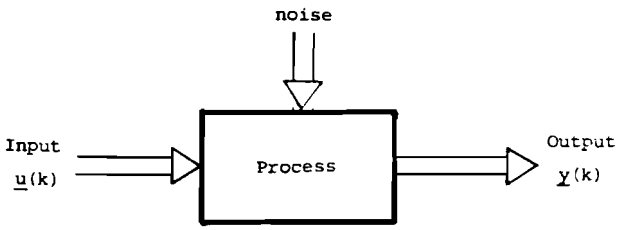


Fig. 0.6 The Process.

Because we can only measure q outputs at every sample moment, we won't be able to distinguish more then q (unknown) noise sequences at the output. If we introduce q error variables in our model and suppose that the input-output relation of our model is exactly the same as the process, the error sequence can be seen as an extra unknown input. We may then try to keep this unknown error sequence as small as possible according to some criterion. We then say that our model fits as good as possible to the data, for the given stochastic (with error sequence) representation. Dependent on the particular definition of the noise sequences, the resultant estimated model will differ and will be suitable for different applications.
To be able to say something about the model misfit, we want to know the model error for given parameter values. To find the error, the model can be drawn as done below, where outputs are used to find the errors:

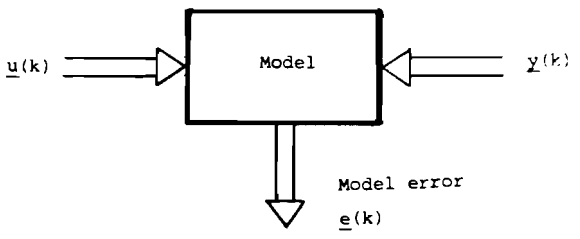


Fig. 0.7 The Stochastic Model.

If we want a model for simulation purposes, the error should be chosen as the error in that estimate of the output that is only dependent on inputs (Output Error Model (OEM)).

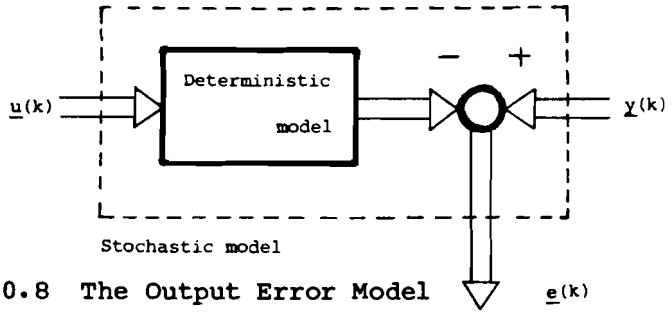


Fig. 0.8 The Output Error Model

If we want to predict the next outputs with the estimated model, the error should be the error in that estimate of the output that is dependent of all available data at sample moment k (Prediction Error Model (PEM), Equation Error Model (EEM))

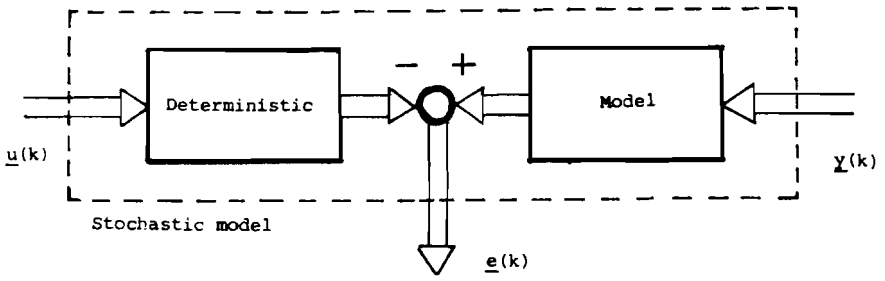


Fig. 0.9 The PEM and EEM

The output of a filter with white noise as input, is uncorrelated to variables not generated by the same white noise sequence. To incorporate this non-correlation of the noises, the noise parameters can also be estimated in order to decrease the error function of the model (Innovation Error Model (IEM))

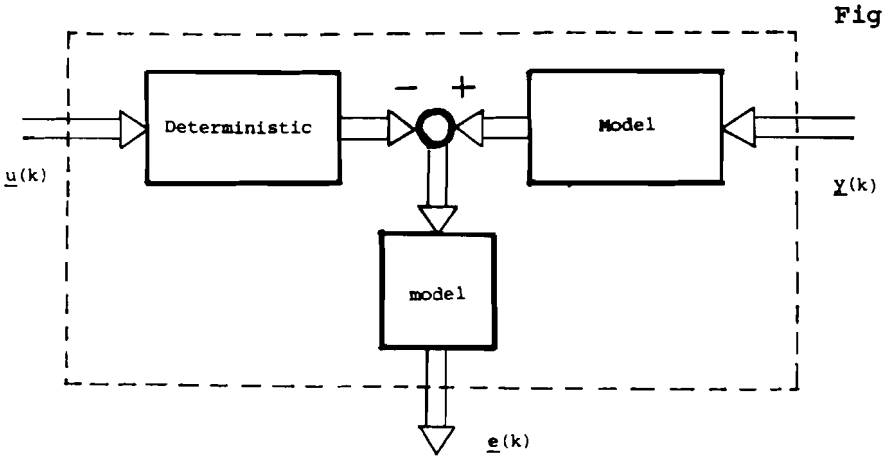


Fig. 0.10 Model with Noise Parameters

Because for a MFD the equation error can be given as a linear function of the model parameters, the model that minimises the equation error can be found easily and fast. Also for a SSM the function that must be minimised for a EEM is comparatively easy. This makes the EEM a possibly good start for minimsation routines that minimise the output, prediction or innovation error.

The PEM, OEM and IEM can be drawn in general as

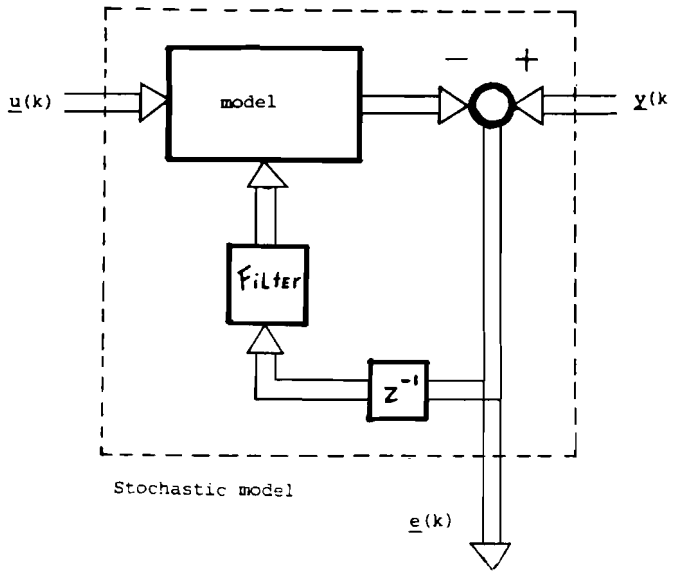


Fig. 0.11 General Model with Error Feedback

where for the OEM K equals 0, for the PEM K is dependent on the model parameters and for the IEM K represents the noise parameters.
Fig.(0.9) and (0.10) are derived from the MFD of a OEM, PEM, EEM and IEM, and fig.(0.11) is derived from the SSM of a OEM, PEM and IEM.

3 Different minimisation routines are implemented to estimate a EEM, PEM, OEM or IEM. Also offset and initial state estimation is implemented for a OEM and IEM estimation (see program guide).

1. Observability Pseudo-Canonical Forms

In general a state space representation of a system is not unique in the parameters. This means that by a regular transformation of the basis of the state vector to another basis the new model will have the same external behaviour.
For estimation purposes it is of great importance to have as less parameters as possible. Some specific transformations reduce the number of parameters considerably. Two closely related transformations will be treated here.
Certain transformations lead to observability pseudo-canonical forms, also known as an overlapping parametrisations. If the order of the model is greater then the number of outputs, more pseudo-canonical forms are possible. Every form being the result of a unique parameter transformation.
Another transformation, according to the method of Guidorzi, leads to the observability canonical form. For every MIMO-system only one canonical form exists. So except for uniqueness of the parameters, its structure is unique for the original system.
One of the possible pseudo-canonical forms will have the same structure as this canonical form, except for some possible structural zero's in the canonical form. For SISO models the only possible pseudo-canonical form coïncide with the canonical form.

1.1 Definitions

Consider a time-discrete observable system which can be represented by the state-space model

$$\begin{aligned} \underline{x}'(k) &= F\underline{x}'(k) + G\underline{u}(k) \\ \underline{y}(k) &= H\underline{x}'(k) + D\underline{u}(k) \end{aligned} \qquad (1.1\text{-a})$$

with :

$\underline{x}'(k)$: State-vector (dimension n)
$\underline{y}(k)$: Output vector (dimension q)
$\underline{u}(k)$: Input vector (dimension p)

k : Time moment indicator

F : System matrix (n×n)

G : Distribution matrix (n×p)

H : Output matrix (q×n)

D : Input-output matrix (qxp)

n : Order of the system and dimension of the state vector

We also assume that there is no static dependence between the output signals. So consequently

$$\text{rank}(H) = q$$
$$q \leq n \qquad (1.1\text{-b})$$

If we write the output matrix as

$$H = \begin{bmatrix} \underline{h}_1^T \\ \vdots \\ \underline{h}_q^T \end{bmatrix} \qquad (1.2)$$

the observability matrix can be written as

$$\Gamma = \left[\underline{h}_1, \ldots, \underline{h}_q, F^T\underline{h}_1, \ldots, F^T\underline{h}_q, (F^2)^T\underline{h}_1, \ldots\ldots \right]^T \qquad (1.3)$$

Because we assumed the system represented by this model to be observable, the observability matrix has rank n.

1.2 Transformation Matrix

To transform the n-dimensional state space to another n-dimensional state space, the transformation matrix must be regular. From (1.3) we see that this transformation matrix T can be composed of p.e. rows of Γ. A particular way to select n rows leads to pseudo-canonical forms of (1.1-a) (see sect. 1.3) and is called the nice-selection but does not necessarily lead to n independent vectors.

Nice-selection rule: Take q integers, the nice-selection indices $\mu_1,\ldots,\mu_q$, such that $(\mu_1+\ldots+\mu_q)=n$ and $\mu_i \geq 1$ for $i=1,\ldots,q$ (*). Select the following rows of Γ

$$\underline{h}_i^T \;,\; \underline{h}_i^T F \;,\; \ldots\ldots \;,\; \underline{h}_i^T F^{(\mu_i-1)} \qquad \text{(for } i=1,\ldots q\text{)}$$

So a general choice of the nice-selection indices does not guarantie that the system is representable in the particular pseudo-canonical form.

(*) :If the later restriction is not made so if one or more nice-selection indices are zero the selection is not called a nice-selection. This extension of the nice-selection rule is treated by Hajdasinski (1983)

Because the nice-selection indices are greater than or equal to 1, the first q rows are selected automatically. Besides these q rows, (n-q) rows have to be selected out of q groups.
As a result the number of possible nice-selections are (see J.Pfennings 1983)

$$\begin{bmatrix} n-1 \\ q-1 \end{bmatrix} = \frac{(n-1)!}{(q-1)!\,(n-q)!} \qquad (1.4)$$

A strictly prescribed way of obtaining a nice-selection for a given model leads to a structural unique canonical form of model (1.1.a) (see par. 1.3). This selection is called the Kronecker-selection and always leads to n independent vectors.

Kronecker-selection rule: Select from the observability matrix every next row independent on previous selected rows, starting at row number 1.

This Kronecker-selection also leads to the selection of $\underline{h}_1,\ldots,\underline{h}_q$ because of the assumed static independence between the output elements. Once there is found a dependent row $(F^T)^{\ell}\underline{h}_i$, all rows $(F^T)^{k}\underline{h}_i$ with $k>\ell$ are also dependent on their predecessors.
This can be proved as follows:

$(F^T)^{\ell}\underline{h}_i$ is dependent on previous selected rows so

$$(F^T)^{\ell}\underline{h}_i = \sum_{k=1}^{\ell-1}\sum_{j=1}^{q} \alpha_{ij,k}(F^T)^{k}\underline{h}_j + \sum_{j=1}^{i-1} \alpha_{ij,\ell}(F^T)^{\ell}\underline{h}_j \qquad (1.5)$$

Now the next dependence relation holds,

$$(F^T)^{\ell+1}\underline{h}_i = \sum_{k=1}^{\ell-1}\sum_{j=1}^{q} \alpha_{ij,k}(F^T)^{k+1}\underline{h}_j + \sum_{j=1}^{i-1} \alpha_{ij,l}(F^T)^{\ell+1}\underline{h}_j \qquad (1.6)$$

By the time $(F^T)^{\ell+1}\underline{h}_i$ is to be tested for independence, every vector of the right hand side of (1.6) has been selected before or is a linear combination of previously selected vectors.
As a result $(F^T)^{\ell+1}\underline{h}_i$ is dependent on its predecessors and consequently not selected.

The structural indices found by this Kronecker-selection criterion are referred to as the Kronecker indices and will be written as $\nu_1,\ldots,\nu_q$.

In this report no distinction is made between the nice-selected and Kronecker-selected structural indices when aspects are analysed which can be treated the same for both the canonical and pseudo-canonical form. Only when the canonical form is meant explicitly the Kronecker indices notation (ν) is used.

Once there are found n vectors by one of the two given selection rules, the transformation matrix can be made:

$$T = \left[\underline{h}_1 \ , \ \ldots \ , \ (F^T)^{(\mu_1-1)}\underline{h}_1 \ , \ \ldots\ldots \ , \ (F^T)^{(\mu_q-1)}\underline{h}_q\right]^T \qquad (1.7)$$

A ordering of the vectors in T as is shown above, leads to the output multi-companion forms of the next section.

Note that this transformation matrix is not necessarily regular for every possible nice selection, but it certainly is for the Kronecker-selection.
A different ordering of the independent vectors in eq.(1.7) will lead to models with the same number of parameters but will not lead to the output companion forms of the next section (see Overbeek and Ljung 1979).

1.3 Transformation to the Output Multi-Companion Form.

Now that we have found a transformation matrix given by (1.7), we can transfer the original model (1.1.a) into an output companion form. Of course this is only allowed if the transformation is regular.
The new form will have (n×q + n×p + q×p) parameters if we use a transformation matrix found by the nice-selection rule.
For a specific model (1.1-a) only the canonical transformation leads to a minimal number of parameters (possibly less then nxq+nxp+qxp) for the given model representation (SSM).

Given a model (1.1.a), nice-selection indices $\mu_1,\ldots,\mu_q$, the transformation matrix (1.7) and the relations

$$\begin{aligned} \underline{x}(k) &= T\,\underline{x}'(k) \\ A &= T\,F\,T^{-1} \\ B &= T\,G \\ C &= H\,T^{-1} \end{aligned} \qquad (1.8)$$

we get the observability (pseudo-)canonical form of (1.1-a)

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) \end{aligned} \qquad (1.9)$$

B and D are fully parametrised but A and C have structural zeroes and ones
If we take a closer look at the equations AT=TF and CT=H we find the structural zeroes and ones in matrices A and C.

Using the same notation as in (1.2) and (1.7) CT=H becomes:

$$C \begin{bmatrix} \underline{h}_1^T \\ \vdots \\ \underline{h}_1^T F^{(\mu_1-1)} \\ \vdots \\ \underline{h}_q^T F^{(\mu_q-1)} \end{bmatrix} = \begin{bmatrix} \underline{h}_1^T \\ \vdots \\ \underline{h}_q^T \end{bmatrix} \qquad (1.10)$$

so C is equal to

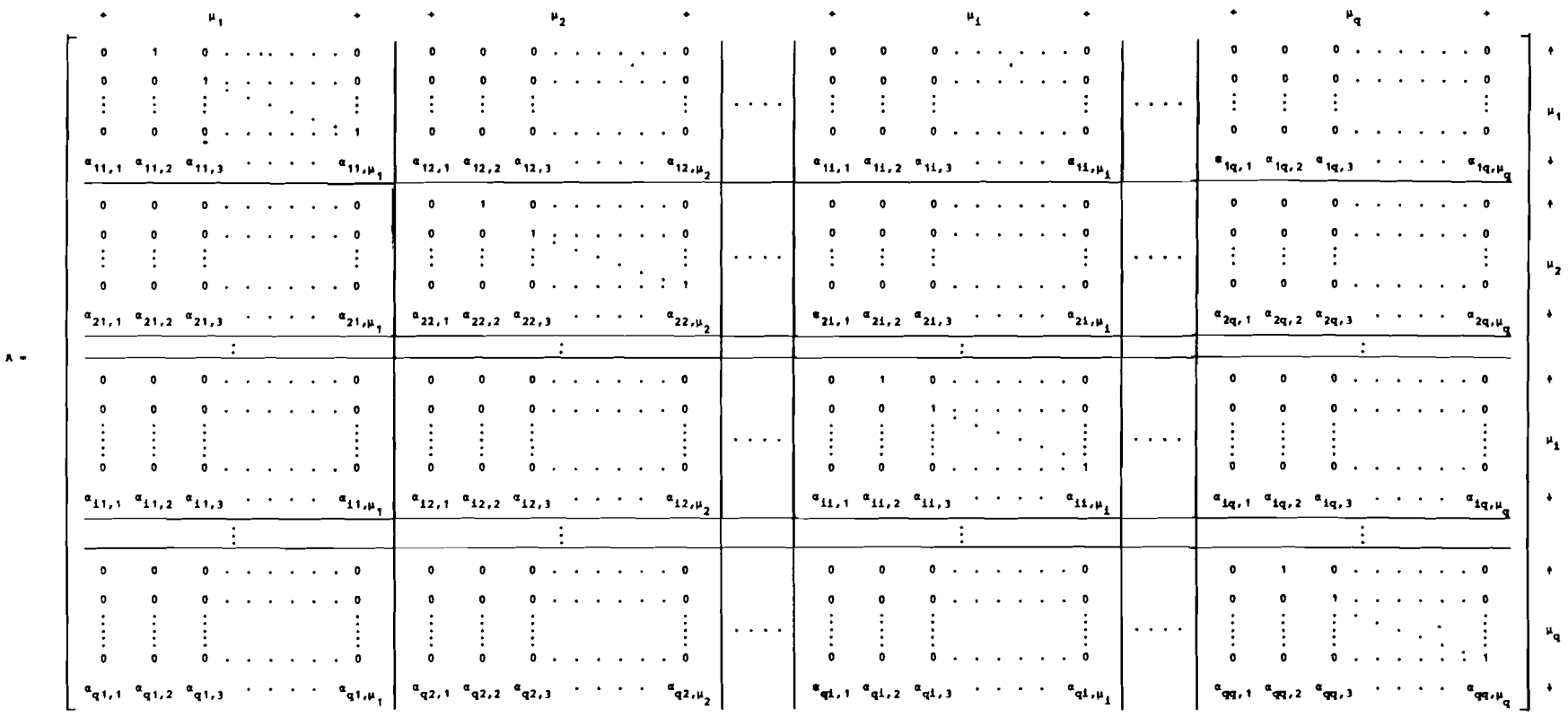
$$C = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 \cdots & 0 & 0 & 0 & \cdots \\ 0 & 0 & \cdots & 0 & 1 & 0 \cdots & 0 & 0 & 0 & \cdots \\ \vdots & & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 0 & 0 & \cdots & 0 & 0 & 0 \cdots & 0 & 1 & 0 & \cdots \end{bmatrix} \qquad (1.11)$$

column : 1, (μ_1+1), $(\mu_1+..+\mu_{q-1}+1)$

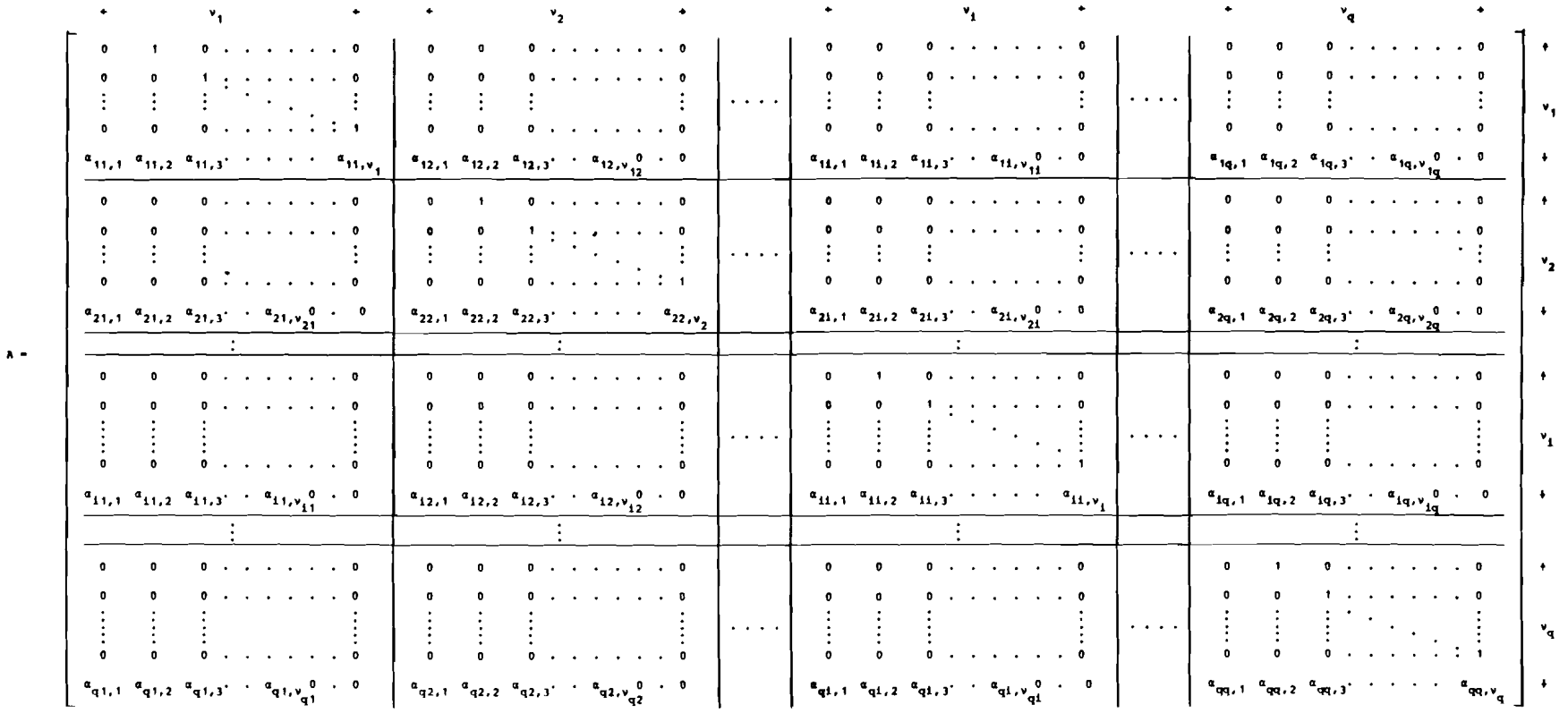
AT=TF becomes:

$$A \begin{bmatrix} \underline{h}_1^T \\ \vdots \\ \underline{h}_1^T F^{(\mu_1-1)} \\ \vdots \\ \underline{h}_q^T F^{(\mu_q-1)} \end{bmatrix} = \begin{bmatrix} \underline{h}_1^T F \\ \vdots \\ \underline{h}_1^T F^{(\mu_1)} \\ \vdots \\ \underline{h}_q^T F^{(\mu_q)} \end{bmatrix} \qquad (1.12)$$

So A has the output companion form given in (1.13-a). This is caused by the fact that the rows of matrix TF are all selected in T except for rows $\underline{h}_i^T F^{(\mu_i)}$,(i=1,...,q), which are linear combinations of rows of T. Compare the α-parameters with eq.(1.5). Note that for SISO-models the first q rows of the (pseudo-canonical) observability matrix form the identity matrix.



(1.13-a) Pseudo-canonical form of system matrix A.



(1.13-b) Canonical form of system matrix A.

In case of a Kronecker-selection we have some more knowledge about the α-parameters in (1.13) and (1.5).
In case of a Kronecker-selection, row $(F^T)^{(\nu_i)}\underline{h}_i$ for $(i=1,\ldots,q)$ can not be dependent on

$$(F^T)^{(k-1)}\underline{h}_j \qquad \text{with : } (k-1) > \nu_i \quad \text{or : } (k-1) = \nu_i \text{ if } j>i \tag{1.14}$$

If we look at the relation AT=TF these conditions can be written as,

$$\alpha_{ij,k}=0 \quad \text{for} \quad k > \min(\nu_i+1,\nu_j) \quad \text{if } j<i \quad \text{or} \quad k > \min(\nu_i,\nu_j) \quad \text{if } j>i \tag{1.15}$$

In literature a commonly used way to write these characteristic features of the observability canonical form is:

$$\alpha_{ij,k}=0 \text{ if } k>\nu_{ij} \text{ with } \nu_{ij}= \min(\nu_i+1,\nu_j) \text{ for } j<i, \quad \nu_{ij}= \min(\nu_i,\nu_j) \text{ for } j>i \tag{1.16}$$

The canonical form of the system matrix is given by (1.13-b).
Looking at the matrices of model (1.9) we see that the number of parameters in the pseudo-canonical form is q×n (in A) plus n×p (in B) plus q×p (in D). If however the model is given in the canonical form matix A can have less parameters. The more dependences are detected in the Kronecker-selection, the more structural zeroes will appear. For any n-th order model the minimal possible number of parameters in the system matrix is

$$(q\times n) - \sum_{i=2}^{q} \sum_{j=1}^{i-1} (\nu_j-\nu_i) \tag{1.17}$$

This number can only be obtained if the Kronecker indices are ordered like

$$\nu_1 \leq \ldots \leq \nu_q \tag{1.17-a}$$

A reordering of the output elements to get this canonical form will generally not be possible because the Kronecker-selection rule selects from $(F^T)^i\underline{h}_1$ to $(F^T)^i\underline{h}_q$, not the other way around.

Given the structural Kronecker indices of a system (we assumed them to be known) the canonical form of model (1.1-a) has a minimal number of parameters in the given model representation (SSM).

2 State Space Description - Matrix Fraction Description.

Let us again consider the time-discrete observable system that can be described by the State Space Model (SSM) (1.1.a). We have seen in the previous chapter that if we use the appropriate transformation (1.7), a nice-selection with structural indices $\mu_1,\ldots,\mu_q$ may result in a unique parametrisation, the observability pseudo-canonical form:

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) \end{aligned} \qquad (2.1)$$

For one specific selection, the Kronecker selection, with the unique structural indices, the transformation leads to the canonical form with a minimal number of parameters in a state space representation.

Another way to describe a time-discrete system is the Matrix Fraction Description (MFD). This I/O-representation of a system is given by (for more details see F.Bekkers 1985)

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) \qquad (2.2)$$

$$\text{with : } P(z) = \left[\, p_{ij}(z) \,\right] \quad (i,j=1,\ldots,q)$$
$$Q(z) = \left[\, q_{i\ell}(z) \,\right] \quad (\ell=1,\ldots,p)$$

If (2.1) and (2.2) are both models in the same model-set (see chapter 3), so that by a proper choice of the parameters in P(z) and Q(z) the MFD can describe every possible input-output relation that can be described by the SSM (and the other way around), both models must relate to each other in a well defined way.
To be able to transfer both parameter sets into each other, we want to find this relation between the equivalent models, but we will see that in the pseudo-canonical form the model sets of SSM and MFD may differ.

2.1 Matrix Fraction Description of a Pseudo-Canonical Form.

Because we already have derived unique parametrisations of the state space model (1.1-a), we want to know how to find the MFD-parameters if the pseudo-canonical SSM-parameters, including the structural indices, are known. To do this the state vector of (2.1), which contains information on past I/O-samples, has to be eliminated.
Introduction of the following notations will facilitate further derivations.

$$[\ A\ ]_{row(\mu_1+\ldots+\mu_i)} = \underline{\alpha}_i^T$$

$$= [\ \alpha_{i1,1}, \ldots, \alpha_{i1,\mu_1}, \ldots\ldots, \alpha_{iq,\mu_q}\ ]$$

$$[\ B\ ]_{row(\mu_1+\ldots+\mu_{i-1}+j)} = \underline{b}_{i,j}^T$$

$$= [\ b_{i,j1}, \ldots\ldots, b_{i,jp}\ ]$$

$$[\ D\ ]_{row(i)} = \underline{d}_i^T$$

$$= [\ d_{i,1}, \ldots\ldots, d_{i,p}\ ]$$

$$\underline{x}^T = [\ x_{1,1}, \ldots, x_{1,\mu_1}, \ldots\ldots, x_{q,\mu_q}\ ]$$

$$z^{\pm 1}\underline{x}(k) = \underline{x}(k\pm 1) \quad \text{(time-shift operation)}$$

Indices before the comma indicate the number of a subsystem.
Indices after the comma indicate an element in a subsystem.
The pseudo-canonical state-space model (2.1) now can be written as:

$$y_i(k) = x_{i,1}(k) + \underline{d}_i^T\underline{u}(k) \tag{2.4}$$

$$\begin{aligned} z\,x_{i,1}(k) &= x_{i,2}(k) + \underline{b}_{i,1}^T\underline{u}(k) \\ &\vdots \\ z\,x_{i,j}(k) &= x_{i,j+1}(k) + \underline{b}_{i,j}^T\underline{u}(k) \\ &\vdots \\ z\,x_{i,(\mu_i-1)}(k) &= x_{i,\mu_i}(k) + \underline{b}_{i,\mu_i-1}^T\underline{u}(k) \end{aligned} \tag{2.5}$$

$$z\, x_{i,\mu_i}(k) = \underline{\alpha}_i^T \underline{x}(k) + \underline{b}_{i,\mu_i}^T \underline{u}(k) \qquad (i=1, \ldots, q) \tag{2.6}$$

The state-vector can be eliminated as follows:
Shift both state vector elements in every equation of (2.5) to the opposite side and use eq.(2.4) to write every element as a linear function of $\underline{u}$ and $\underline{y}$. If we omit the time moment indicator, eq.(2.4) and (2.5) become

$$\begin{aligned}
x_{i,1} &= y_i - \underline{d}_i^T \underline{u} \qquad (2.7)\\
x_{i,2} &= zy_i - z\underline{d}_i^T \underline{u} - \underline{b}_{i,1}^T \underline{u}\\
&\vdots\\
x_{i,j} &= z^{j-1} y_i - z^{j-1} \underline{d}_i^T \underline{u} - z^{j-2} \underline{b}_{i,1}^T \underline{u} - \ldots - \underline{b}_{i,j-1}^T \underline{u}\\
&\vdots\\
x_{i,\mu_i} &= z^{(\mu_i-1)} y_i - z^{(\mu_i-1)} \underline{d}_i^T \underline{u} - z^{(\mu_i-2)} \underline{b}_{i,1}^T \underline{u} - \ldots - \underline{b}_{i,\mu_i-1}^T \underline{u}
\end{aligned}$$

(for i=1, ... ,q)

If we now insert these equations into (2.6) and shift all elements of $\underline{y}$ to the left side and take together all q equations in one expression we get

$$\boxed{\overline{A}(\alpha) V_\mu(z) \underline{y}(k) = \overline{A}(\alpha) W_\mu(z) B(b,d) \underline{u}(k)} \tag{2.8}$$

with $\overline{A}(\alpha)$, $B(b,d)$, $V_\mu(z)$, and $W_\mu(z)$ given at the next pages in (2.9) to (2.12).
The left hand side of (2.8) can easily be verified out of (2.9), (2.11), (2.7) and (2.6).
For verification of the right hand side of (2.8), $W_\mu(z)B(b,d)$ is given by (2.13).

$$\left[\begin{array}{cccc|c|cccc|c|c|cccc|c}
 & & \mu_1+1 & & & & & \mu_2+1 & & & & & \mu_q+1 & & \\
-\alpha_{11,1} & -\alpha_{11,2} & -\alpha_{11,3} & \cdots\cdots & -\alpha_{11,\mu_1} & 1 & -\alpha_{12,1} & -\alpha_{12,2} & -\alpha_{12,3} & \cdots\cdots & -\alpha_{12,\mu_2} & 0 & -\alpha_{1q,1} & -\alpha_{1q,2} & -\alpha_{1q,3} & \cdots\cdots & -\alpha_{1q,\mu_1} & 0 \\
-\alpha_{21,1} & -\alpha_{21,2} & -\alpha_{21,3} & \cdots\cdots & -\alpha_{21,\mu_1} & 0 & -\alpha_{22,1} & -\alpha_{22,2} & -\alpha_{22,3} & \cdots\cdots & -\alpha_{22,\mu_2} & 1 & -\alpha_{2q,1} & -\alpha_{2q,2} & -\alpha_{2q,3} & \cdots\cdots & -\alpha_{2q,\mu_2} & 0 \\
\hline
 & & \vdots & & & & & \vdots & & & & & & \vdots & & & & \\
\hline
-\alpha_{q1,1} & -\alpha_{q1,2} & -\alpha_{q1,3} & \cdots\cdots & -\alpha_{q1,\mu_1} & 0 & -\alpha_{q2,1} & -\alpha_{q2,2} & -\alpha_{q2,3} & \cdots\cdots & -\alpha_{q2,\mu_2} & 0 & -\alpha_{qq,1} & -\alpha_{qq,2} & -\alpha_{qq,3} & \cdots\cdots & -\alpha_{q2,\mu_q} & 1
\end{array}\right]$$

(2.9) Matrix $\overline{A}(\alpha)$.

$$V_\mu(z) = \left[\begin{array}{cccc|c|cccc|c|c|cccc|c}
 & & & q & & & & & & & & & & & & & \\
1 & 0 & 0 & \cdots & 0 & & & & & & & & & & & & \\
z & 0 & 0 & \cdots & 0 & & & & & & & & & & & & \\
z^2 & 0 & 0 & \cdots & 0 & & & & & & & & & & & & \\
\vdots & \vdots & \vdots & & \vdots & & & & & & & & & & & & \\
z^{\mu_1} & 0 & 0 & \cdots & 0 & \mu_1+1 & & & & & & & & & & & \\
\hline
0 & 1 & 0 & \cdots & 0 & & & & & & & & & & & & \\
0 & z & 0 & \cdots & 0 & & & & & & & & & & & & \\
0 & z^2 & 0 & \cdots & 0 & & & & & & & & & & & & \\
\vdots & \vdots & \vdots & & \vdots & & & & & & & & & & & & \\
0 & z^{\mu_2} & 0 & \cdots & 0 & \mu_2+1 & & & & & & & & & & & \\
\hline
 & & \vdots & & & & & & & & & & & & & & \\
\hline
0 & 0 & 0 & \cdots & 1 & & & & & & & & & & & & \\
0 & 0 & 0 & \cdots & z & & & & & & & & & & & & \\
0 & 0 & 0 & \cdots & z^2 & & & & & & & & & & & & \\
\vdots & \vdots & \vdots & & \vdots & & & & & & & & & & & & \\
0 & 0 & 0 & \cdots & z^{\mu_q} & \mu_q+1 & & & & & & & & & & &
\end{array}\right]$$

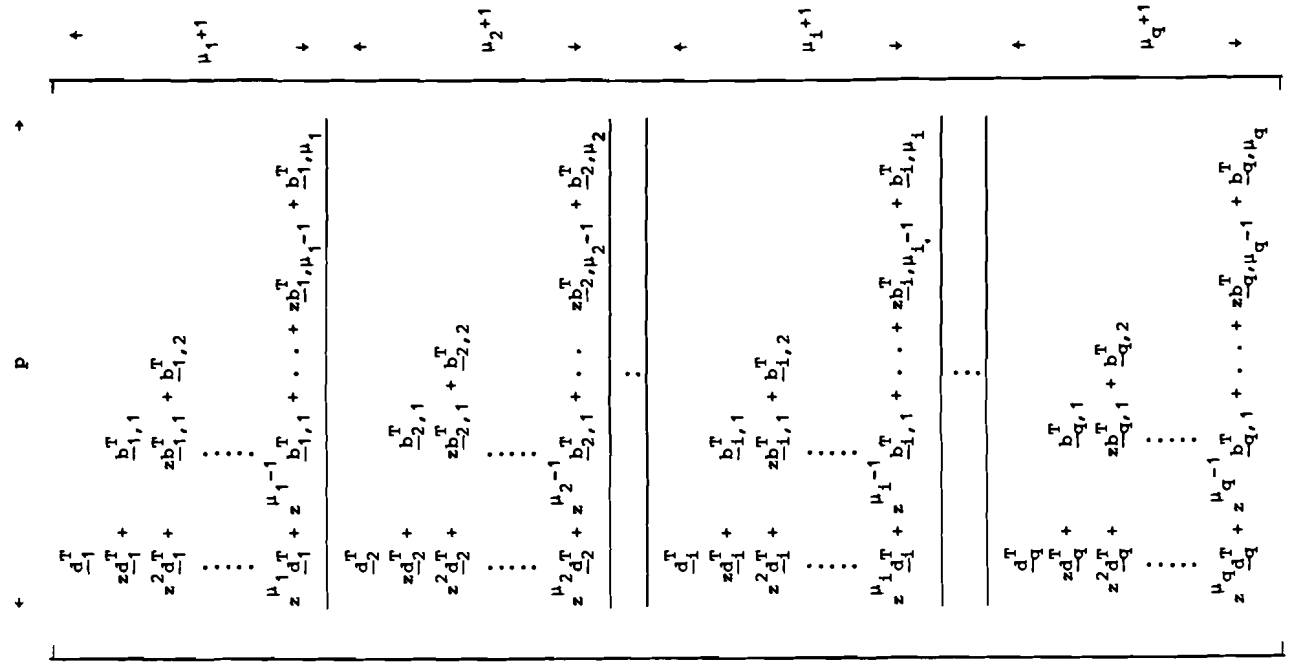
(2.11) Matrix $V_\mu(z)$.

(2.12) Matrix $W_\mu(z)$.

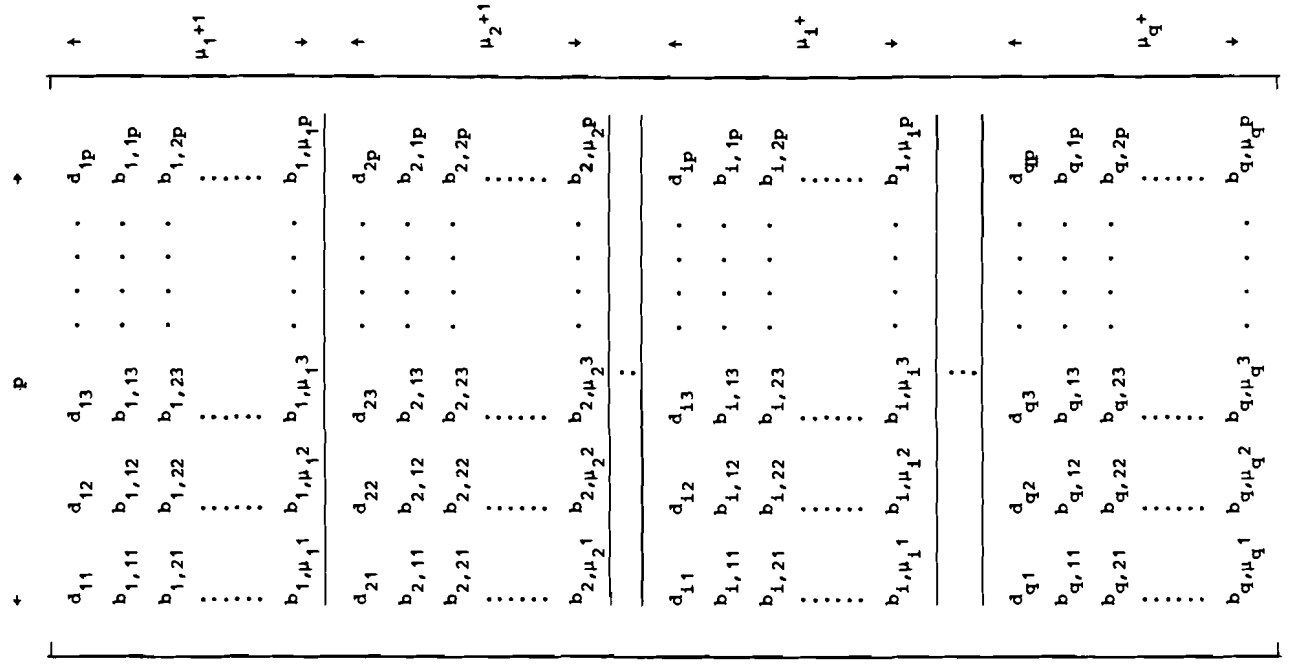
$$W(z) = [\, w_{ij}(z) \,] \qquad (i,j=1, \ldots, q)$$

$w_{ij}(z) = 0$ (dimension $(\mu_i+1)\times(\mu_j+1)$ and $i \neq j$

$$w_{ii}(z) = \left[\begin{array}{cccccc}
1 & 0 & 0 & \cdots & 0 & 0 \\
z & 1 & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\
z^{\mu_i} & z^{\mu_i-1} & \cdots & \cdots & z & 1
\end{array}\right]$$



(2.13) Matrix $W_{\mu}(z)B(b,d)$.



(2.10) Matrix B(b,d).

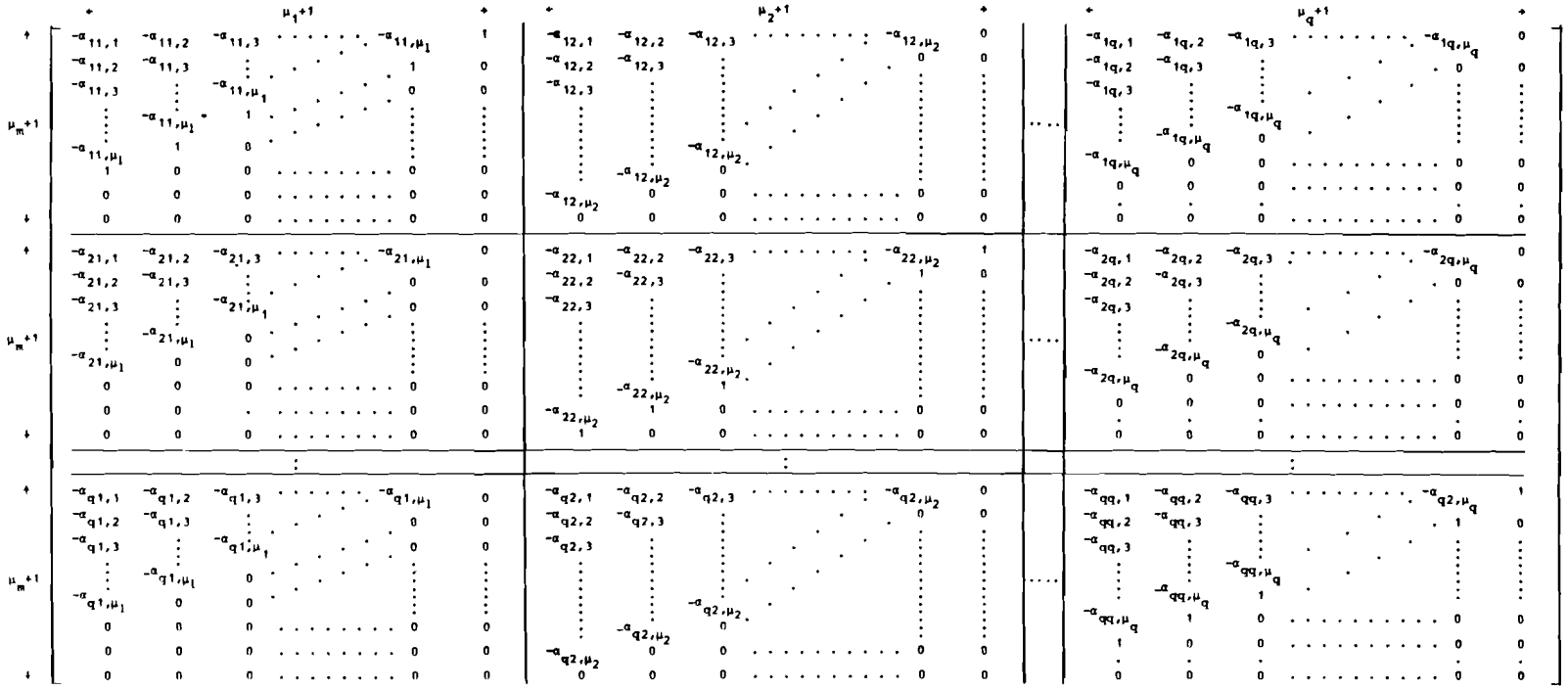
$$
\left[\begin{array}{ccccc}
\beta_{1,11} & \beta_{1,12} & \beta_{1,13} & \cdots & \beta_{1,1p} \\
\beta_{1,21} & \beta_{1,22} & \beta_{1,23} & \cdots & \beta_{1,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{1,\mu_m 1} & \beta_{1,\mu_m 2} & \beta_{1,\mu_m 3} & \cdots & \beta_{1,\mu_m} \\
\left[\beta_{1,(\mu_m+1)1}\right. & \beta_{1,(\mu_m+1)2} & \beta_{1,(\mu_m+1)3} & \cdots & \left.\beta_{1,(\mu_m+1)p}\right] \\
\hline
\beta_{2,11} & \beta_{2,12} & \beta_{2,13} & \cdots & \beta_{2,1p} \\
\beta_{2,21} & \beta_{2,22} & \beta_{2,23} & \cdots & \beta_{2,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{2,\mu_m 1} & \beta_{2,\mu_m 2} & \beta_{2,\mu_m 3} & \cdots & \beta_{2,\mu_m p} \\
\left[\beta_{2,(\mu_m+1)1}\right. & \beta_{2,(\mu_m+1)2} & \beta_{2,(\mu_m+1)3} & \cdots & \left.\beta_{2,(\mu_m+1)p}\right] \\
\hline
 & & \vdots & & \\
\hline
\beta_{i,11} & \beta_{i,12} & \beta_{i,13} & \cdots & \beta_{i,1p} \\
\beta_{i,21} & \beta_{i,22} & \beta_{i,23} & \cdots & \beta_{i,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{i,\mu_m 1} & \beta_{i,\mu_m 2} & \beta_{i,\mu_m 3} & \cdots & \beta_{i,\mu_m p} \\
\left[\beta_{i,(\mu_m+1)1}\right. & \beta_{i,(\mu_m+1)2} & \beta_{i,(\mu_m+1)3} & \cdots & \left.\beta_{i,(\mu_m+1)p}\right] \\
\hline
 & & \vdots & & \\
\hline
\beta_{q,11} & \beta_{q,12} & \beta_{q,13} & \cdots & \beta_{q,1p} \\
\beta_{q,21} & \beta_{q,22} & \beta_{q,23} & \cdots & \beta_{q,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{q,\mu_m 1} & \beta_{q,\mu_m 2} & \beta_{q,\mu_m 3} & \cdots & \beta_{q,\mu_m p} \\
\left[\beta_{2,(\mu_m+1)1}\right. & \beta_{2,(\mu_m+1)2} & \beta_{2,(\mu_m+1)3} & \cdots & \left.\beta_{2,(\mu_m+1)p}\right]
\end{array}\right]
\begin{array}{l}
\uparrow \\
\\
\mu_m \\
[\mu_m+1] \\
\downarrow \\
\\
\uparrow \\
\\
\mu_m \\
[\mu_m+1] \\
\downarrow \\
\\
\\
\uparrow \\
\\
\mu_m \\
[\mu_m+1] \\
\downarrow \\
\\
\\
\uparrow \\
\\
\mu_m \\
[\mu_m+1] \\
\downarrow
\end{array}
$$

(2.16-a) Pseudo-canonical form of matrix $\overline{B}(\beta)$.

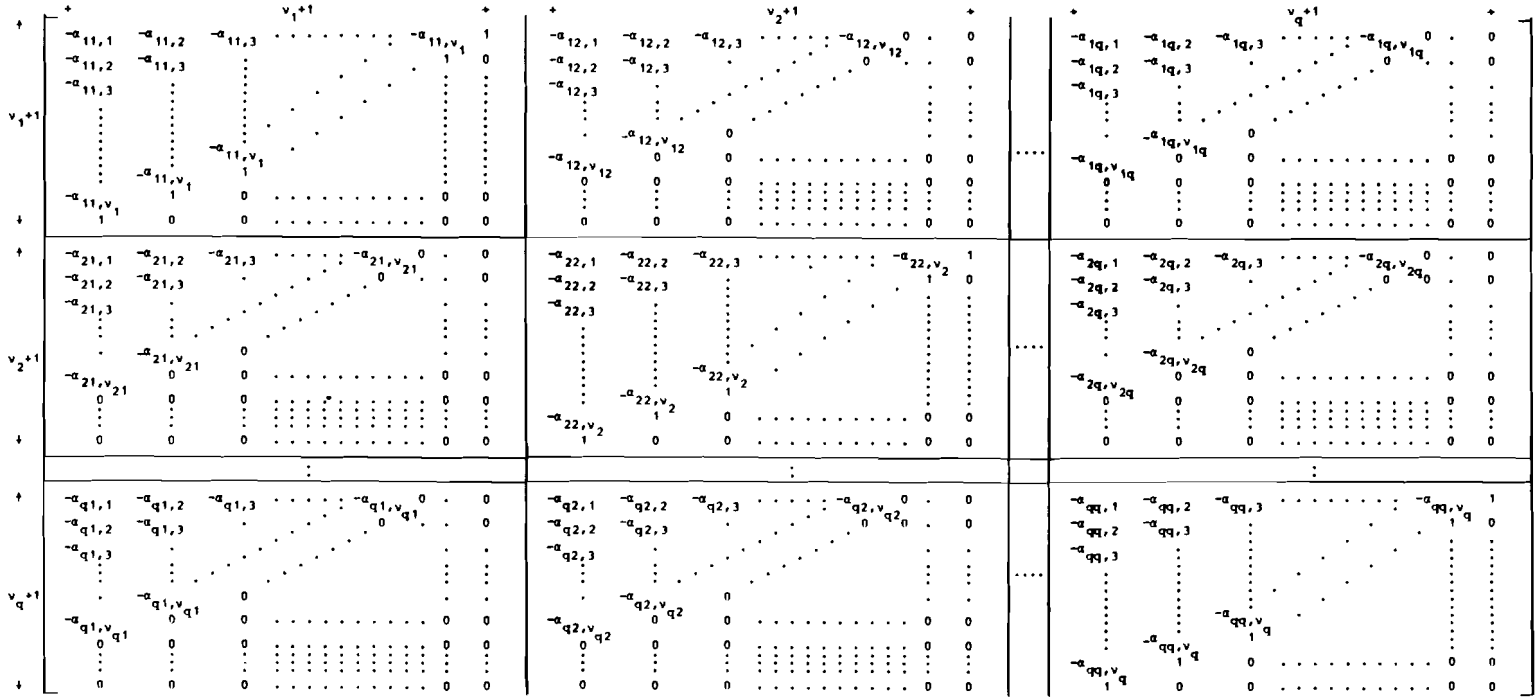
($\left[\beta_{i,(\mu_i+1)j}\right]$ indicates that this parameter is zero if $\mu_i=\mu_m$. $(\mu_m=\max_i(\mu_i))$

$$
\begin{array}{c}
\overset{\leftarrow}{} \qquad\qquad p \qquad\qquad \overset{\rightarrow}{} \\
\left[\begin{array}{lllll}
\beta_{1,11} & \beta_{1,12} & \beta_{1,13} & \cdots & \beta_{1,1p} \\
\beta_{1,21} & \beta_{1,22} & \beta_{1,23} & \cdots & \beta_{1,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{1,\nu_1 1} & \beta_{1,\nu_1 2} & \beta_{1,\nu_1 3} & \cdots & \beta_{1,\nu_1} \\
\beta_{1,(\nu_1+1)1} & \beta_{1,(\nu_1+1)2} & \beta_{1,(\nu_1+1)3} & \cdots & \beta_{1,(\nu_1+1)p} \\
\hline
\beta_{2,11} & \beta_{2,12} & \beta_{2,13} & \cdots & \beta_{2,1p} \\
\beta_{2,21} & \beta_{2,22} & \beta_{2,23} & \cdots & \beta_{2,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{2,\nu_2 1} & \beta_{2,\nu_2 2} & \beta_{2,\nu_2 3} & \cdots & \beta_{2,\nu_2 p} \\
\beta_{2,(\nu_2+1)1} & \beta_{2,(\nu_2+1)2} & \beta_{2,(\nu_2+1)3} & \cdots & \beta_{2,(\nu_2+1)p} \\
\hline
 & & \vdots & & \\
\hline
\beta_{i,11} & \beta_{i,12} & \beta_{i,13} & \cdots & \beta_{i,1p} \\
\beta_{i,21} & \beta_{i,22} & \beta_{i,23} & \cdots & \beta_{i,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{i,\nu_i 1} & \beta_{i,\nu_i 2} & \beta_{i,\nu_i 3} & \cdots & \beta_{i,\nu_i p} \\
\beta_{i,(\nu_i+1)1} & \beta_{i,(\nu_i+1)2} & \beta_{i,(\nu_i+1)3} & \cdots & \beta_{i,(\nu_i+1)p} \\
\hline
 & & \vdots & & \\
\hline
\beta_{q,11} & \beta_{q,12} & \beta_{q,13} & \cdots & \beta_{q,1p} \\
\beta_{q,21} & \beta_{q,22} & \beta_{q,23} & \cdots & \beta_{q,2p} \\
\vdots & \vdots & \vdots & & \vdots \\
\beta_{q,\nu_q 1} & \beta_{q,\nu_q 2} & \beta_{q,\nu_q 3} & \cdots & \beta_{q,\nu_q p} \\
\beta_{2,(\nu_q+1)1} & \beta_{2,(\nu_q+1)2} & \beta_{2,(\nu_q+1)3} & \cdots & \beta_{2,(\nu_q+1)p}
\end{array}\right]
\end{array}
\begin{array}{l}
\uparrow \\
\\
\\
\nu_1+1 \\
\\
\downarrow \\
\uparrow \\
\\
\\
\nu_2+1 \\
\\
\downarrow \\
\\
\uparrow \\
\\
\\
\nu_i+1 \\
\\
\downarrow \\
\\
\uparrow \\
\\
\\
\nu_q+1 \\
\\
\downarrow
\end{array}
$$

(2.16-b) Canonical form of matrix $\overline{B}(\beta)$.



(2.17-a) Pseudo-canonical form of matrix $M(\alpha)$.
In this example $\mu_1=\mu_m-2$, $\mu_2=\mu_m$. ($\mu_m=\max_i(\mu_i)$)



(2.17-b) Canonical form of matrix M(α).

In (2.8) we have found an expression of the I/O-representation in state space parameters (2.1). Because we wanted a representation in MFD-form (eq.(2.2)), we have to write the p(z) and q(z) polynomials explicitly. For matrix P(z) this can be done without any problem.

$$p_{ij}(z) = \quad -\alpha_{ij,\mu_j} z^{(\mu_j-1)} - \ldots - \alpha_{ij,2} z - \alpha_{ij,1} \qquad (i \neq j)$$

$$p_{ii}(z) = z^{\mu_j} - \alpha_{ii,\mu_i} z^{(\mu_i-1)} - \ldots - \alpha_{ii,2} z - \alpha_{ii,1} \qquad (2.14)$$

For matrix Q(z) the coefficients of the polynomials have to be given as a (non linear) function of α- and (b,d)-parameters.

$$\boxed{\bar{B}(\beta) = M(\alpha)B(b,d)} \qquad (2.15)$$

with $\bar{B}(\beta)$, $M(\alpha)$ and $B(b,d)$ given by (2.16-a), (2.17-a) and (2.10). For the canonical form $\bar{B}(\beta)$ and $M(\alpha)$ are given by (2.16-b) and (2.17-b). (The form of (2.16-b) is explained in the next section).
Now the polynomials can be written as

$$q_{i\ell}(z) = \quad \beta_{i,\mu_m \ell} z^{(\mu_m-1)} + \ldots + \beta_{i,1\ell} \qquad \text{for } \mu_i < \mu_m$$

$$q_{i\ell}(z) = \beta_{i,(\mu_m+1)\ell} z^{\mu_m} + \beta_{i,\mu_m \ell} z^{(\mu_m-1)} + \ldots + \beta_{i,1\ell} \qquad \text{for } \mu_i = \mu_m \qquad (2.18)$$

So now we know the equivalent MFD of an observability pseudo-canonical SSM.
Because in general $M(\alpha)$ in (2.15) is not a square matrix, it is impossible to transform a randomly chosen set of β-parameters to (b,d)-parameters. More about this will be said in the next section.

2.2 Properties of MFD's

In the last section it has been shown how to derive an equivalent MFD, given a pseudo-canonical SSM (2.1) with a given set of structural indices.
By (2.1) every possible n-dimensional SSM is represented, so from paragraph 2.1 we may conclude:

> Given a MFD in observability pseudo-canonical form, the equivalent SSM exists if the transformation from SSM- to MFD-parameters (eq.(2.15)) can be inverted.

The mentioned transformation matrix $M(\alpha)$ is only dependent on α-parameters. For later use we mention here that it can be proved that for all pseudo-canonical forms this matrix has full rank n+q (see J.Pfennings 1983).
In case of the canonical form (paragraph 2.1), we found the next restrictions on the α-parameters:

$$\alpha_{ij,k} = 0 \quad \text{if} \quad k > \min(\nu_i,\nu_j) \quad \text{for } j \geq i \qquad \text{or} \quad k > \min(\nu_i+1,\nu_j) \quad \text{for } j < i \tag{2.19}$$

Because of this, a number of equations in (2.15) become trivialities such that (see 2.16-b)

$$\beta_{i,j\ell}=0 \quad \text{for} \quad j > \nu_i+1 \tag{2.20}$$

and (2.18) can be written as

$$q_{i\ell}(z) = \beta_{i,(\nu_i+1)\ell} z^{\nu_i} + \beta_{i,\nu_i\ell} z^{(\nu_i-1)} + \ldots + \beta_{i,1\ell} \tag{2.21}$$

Given these restrictions, the number of MFD-parameters is equal to the number of SSM-parameters and matrix $M(\alpha)$ is a square matrix with dimension n+q.
The canonical form is a special kind of pseudo-canonical form so it also has full rank n+q (see above), so that this matrix can be inverted and the MFD-parameters can be transformed directly to SSM-parameters.

Of course the transformation to SSM-parameters can also be given for pseudo-canonical forms if the β-parameters have such specific values so that eq.(2.15) is valid for a specific set of SSM-parameters. However, if the MFD-parameters set is a direct result of a pseudo-canonical estimation, this restrictive condition on the β-parameters will generally not be fulfilled. So now we may conclude:

> Given a MFD in observability pseudo-canonical form, the equivalent SSM only exists if the transformation matrix from SSM- to MFD-parameters is a square and regular matrix (inversible) or if the MFD-parameters are chosen such that SSM-parameters can be found that fulfill eq.(2.15).

Let us now take a closer look at the MFD without any restrictive conditions on the parameters (eq.(2.2), (2.14) and (2.18).:

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) \tag{2.2}$$

with:
$\text{degree } p_{ij}(z) = (\mu_j - 1)$ for $i \neq j$
$\text{degree } p_{ii}(z) = \mu_i$
$\text{degree } q_{i\ell}(z) = (\mu_m - 1)$ for $(\mu_i < \mu_{max})$
$\text{degree } q_{i\ell}(z) = \mu_m$ for $(\mu_i = \mu_{max})$

We see that the highest sample number of output i in (2.2) is $y_i(k+\mu_i)$, because $p_{ii}(z)=z^{\mu_i} - \ldots$.
If we look at the i-th equation of (2.2) we see that $y_i(k+\mu_i)$ can be a function of future samples of $\underline{u}$ if $\mu_i<\mu_m$ and future samples of y_j if $\mu_j>(\mu_i+1)$.
From this we may conclude:

> Without any restictions on the parameters the observability pseudo-canonical MFD of (2.2) might be non causal.

However if we remember the canonical restrictions on the MFD-parameters (eq.(2.19) and (2.20)) we see that

MFD's in observability canonical form are always causal

This can also be seen as follows:
We have stated before that for the canonical representation $M(\alpha)$ is a square and regular matrix, so any set of canonical MFD-parameters can be transformed to a set of SSM-parameters. In this case both parameter sets represent equivalent model sets. Because every SSM is causal by definition the equivalent MFD also has to be causal.

Now we have shown that apart from the limitations on the parameter sets, the restrictions in (2.19) and (2.21) are sufficient for an MFD to be causal.
Because in state space representation all causal linear time-discrete time-invariant systems can be represented by (2.1), the restrictions are also necessary for the treated MFD's to represent causal linear time-discrete time-invariant systems.
So eq.(2.19) and (2.21) represent the causality restriction for the MFD.

3 Stochastic models

3.0 Introduction

Before we extend existing deterministic models, described in the previous chapters, with an error, we have to define the term 'model set' and evaluate the impact of this definition on the aspects of the treated models.

Model set: A collection of mathematical descriptions of relations between variables: A model set is characterised by the number of variables and further restrictions made by ourselves to indicate the bounds of the model set.

We will consider the model set (S) that contains all possible relations, i.e. external behaviour, between the input and output variables of a n-th order state space model given by eq.(1.1-a). A particular quantitatively defined (parameters) model within this set is called an element and denoted by $M^0=\{F,G,H,D\}$.
In chapter 1 we have seen that every element of S can be written in only one canonical form, which is unique in the parameters but also has a unique set of Kronecker indices. So the possible canonical forms exclude each other (see fig. 3.1-a)

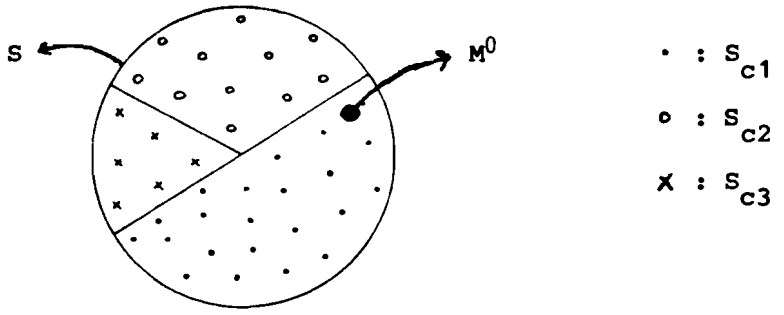


Fig. 3.1-a Canonical model sets for a 4-th order model with 2 outputs. S_{c1}: $\nu_1=2$, $\nu_2=2$; $S = \{S_{c1},S_{c2},S_{c3}\}$
S_{c2}: $\nu_1=3$, $\nu_2=1$; $M^0 \in S_{c1}$
S_{c3}: $\nu_1=1$, $\nu_2=3$

We have seen also that every element of S might be represented in more then one pseudo-canonical form. So the model sets defined by pseudo-canonical forms do not exclude each other (overlapping parametrisations). If the nice-selection indices are equal to the kronecker indices (so pseudo-canonical and canonical forms with the same structure-indices) then the canonical model set is part of the pseudo-canonical model set (see fig. 3.1-b).

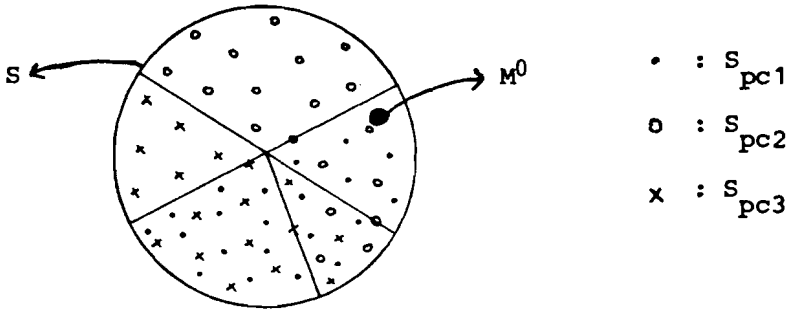


Fig. 3.1-b Pseudo-canonical model sets defined by SSM's.

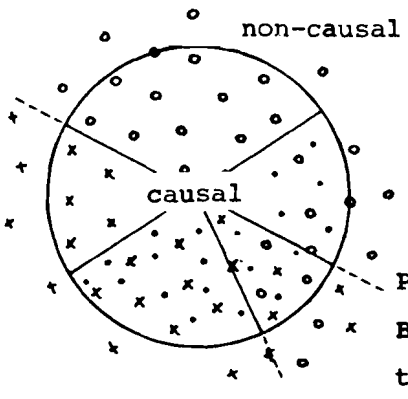
(Causal). S_{pc1}: $\mu_1=2$, $\mu_2=2$; $M^0 \in S_{pc1}$

S_{pc2}: $\mu_1=3$, $\mu_2=1$; $M^0 \in S_{pc2}$

S_{pc3}: $\mu_1=1$, $\mu_2=3$

Notice that it is impossible to represent a model with canonical structure (1,3) in the pseudo-canonical structure (3,1) and vice versa for structure (3,1) and (1,3). This because of the dependence relations derived in chapter 1.

In chapter 2 we have derived the equivalent (pseudo-)canonical forms for the MFD. The canonical model sets defined by a SSM and a MFD are equal. For the pseudo-canonical forms the model sets of SSM and MFD are equal only if we restrict ourselves to causal MFD's (see fig.3.1-b). Otherwise the pseudo-canonical model set of a SSM is just a part of the pseudo-canonical model set defined by the MFD (see fig.3.1-c).



M^0 • : $S_{MFD\ pc1}$

S o : $S_{MFD\ pc2}$

x : $S_{MFD\ pc3}$

Fig. 3.1-c

Pseudo-canonical model sets of MFD's

Because for structure (2,2) the model set is equal to the canonical model set, the pseudo-canonical MFD is causal.

In the previous chapters nothing is said, and in this chapter nothing will be said about the initial conditions of the model because these initial conditions will not be considered as model parameters. They only have to be considered for a given estimation problem (see chapter 4).

After this review in connection with the term model set, we will focus our attention on stochastic models.
In practice, processes and observations of process variables are contaminated with all kind of unknown disturbances not caused by in- or output variables. These disturbances will be introduced in a linear causal model of the process as noises uncorrelated with former and present input variables.
If a process without disturbances has been described by a deterministic model (an element of a deterministic model set), then the process with disturbances will be described by a stochastic model (an element of a stochastic model set).
The stochastic model set is characterised by its deterministic part, the stochastic properties of the noise (nois parameters), and the places where the noises are modelled to enter the process.

In the next four sections we will try to find stochastic SSM's and MFD's of the same stochastic model sets, just as done in chapter 2 for deterministic model sets. In this chapter nothing will be assumed about the stochastic properties of the noises. This will be done in chapter 4 in connection with the estimation algorithm.

In section 3.1 the deterministic model will be extended with the stochastic variable, the output error. This error can be introduced in SSM's and MFD's without any problem.

In section 3.2 and 3.3 the introduced stochastic variables are the prediction error and the equation error, which have proved their superiority for specific purposes. Both errors are introduced in ARMA-models and consequently they have to be generalised to MFD models first. In order to find the same stochastic SSM in the same stochastic model set

as this stochastic MFD, we have to transfer the MFD into a SSM. We have shown before that the transformation of the deterministic part is not possible if the MFD is not a causal model. We will assume this causality. The introduction of the prediction error and equation error in a SSM is shown in table 3.2. To show the procedure drawn in this table, the same procedure is used in section 3.1.

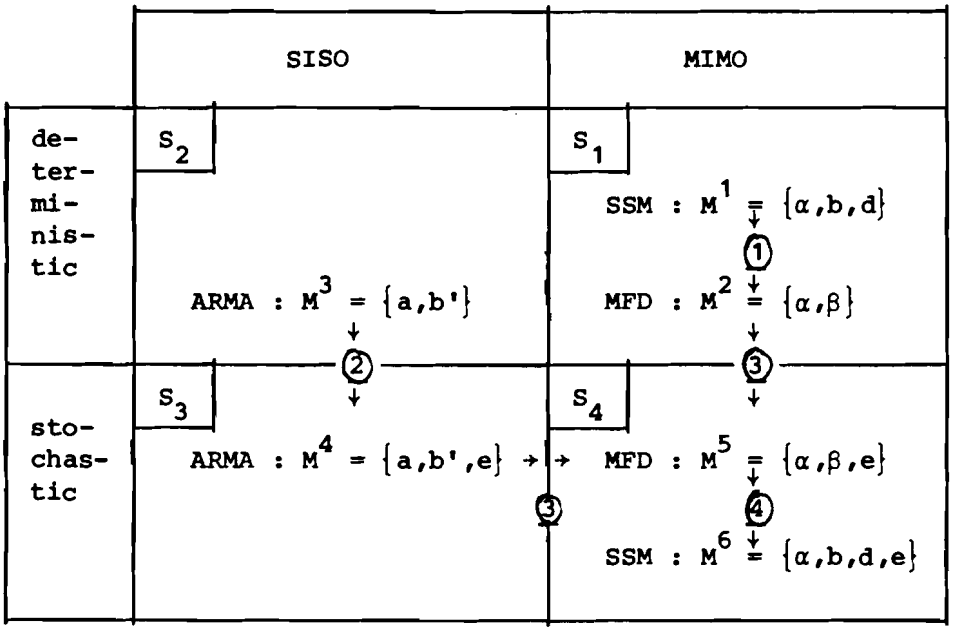


Table 3.2 1 Restriction to causal MFD's
2 Introduction of the error in an ARMA model.
3 Generalisation to MFD's
4 Transformation to SSM's

In section 3.4 the innovation error is introduced in a SSM together with an extension of the parameter set with (nxq) noise parameters. The transformation from this stochastic model to the equivalent MFD is always possible.

3.1 Output Eror Model (OEM)

For SISO systems the ARMA (Auto-Regressive Moving Average) model is usually given by

$$\left[\ 1+A(z^{-1})\ \right] y(k) = \left[\ b_0+B(z^{-1})\ \right] u(k) \qquad (3.4)$$

with : $A(z^{-1}) = a_1 z^{-1} + \ldots + a_{n'} z^{-n'}$

$B(z^{-1}) = b'_1 z^{-1} + \ldots + b'_m z^{-m}$

(Apostrophes are used to distinguish parameters and order from those used in previous chapters).
In notation used for MIMO-models this would be

$$P(z)y(k) = Q(z)u(k) \qquad (3.5)$$

with : $P(z) = z^n - \alpha_n z^{n-1} - \ldots - \alpha_1$

$Q(z) = \beta_{n+1} z^n + \beta_n z^{n-1} + \ldots + \beta_1$

$n = \max(n',m)$

Because eq.(3.4) is commonly used for SISO-models we will use this notation if SISO-models are discussed. The reader should be careful not to mix up parameters in the two different notations, in particular note the reverse order of indexing.

The output error is defined as the misfit in the simulated output. The simulated output is an estimate of y(k) knowing all inputs of the process to sample moment k.

$$y(k) = \mathrm{Func}\{a,b',u(k),u(k-1),\ \ldots\ \} + e_o(k) \qquad (3.6)$$

For ARMA representations of processes the extended model becomes:

$$y(k) = \frac{\left[\ b_0+B(z^{-1})\ \right]}{\left[\ \ 1+A(z^{-1})\ \right]} u(k) + e_0(k) \qquad (3.7)$$

The generalisation of the deterministic model (3.4) to MIMO-models is the

MFD given below (see also (2.2), (2.14) and (2.18)).

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) \tag{3.8}$$

The introduction of an output error conformable to (3.6) can be generalised to the vector case through

$$\underline{y}(k) = \text{Func}\{\alpha,\beta,\underline{u}(k),\underline{u}(k-1), \ldots\} + \underline{e}_0(k) \tag{3.9}$$

Combining (3.8) and (3.9) the MFD extended with an output error vector is,

$$\underline{y}(k) = P^{-1}(z)Q(z)\underline{u}(k) + \underline{e}_0(k) \tag{3.10}$$

This expression is valid for every sample moment and because no assumptions were made on the stochastic behaviour, the model may be considered as a perfect model for an output sample sequence $\underline{y}(k)-\underline{e}_0(k)$. Because the transformation from SSM- to MFD-parameters is given by (3.3) and because of the new interpretation of the output variable, the MFD (3.10) can directly be transformed to an equivalent SSM. In State Space representation the new temporarily combined outputs must be separated to get the model for the original I/O-samples

$$\boxed{\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{e}_0(k) \end{aligned}} \tag{3.11}$$

Because the state vector is only dependent on former input samples, the estimate of $\underline{y}(k)$ (= $C\underline{x}(k)+D\underline{u}(k)$) is only a function of present and former inputs. This knowledge would have been enough to introduce the output error directly in the State Space representation without help of an I/O-representation.

The next two sections treat two error extended models for which the error introduction is based on ARMA-models and therefore an equivalent SSM can not be found directly.

3.2 Prediction Error Model (PEM)

In this section the same notations for deterministic models will be used as used in section 3.2. These deterministic models can be extended with a stochastic variable, the one-step-ahead prediction error.
This prediction error is defined as the misfit in the best prediction of the next output sample. This output prediction must be a function of all available data at sample moment k. So

$$y(k) = \text{Func}\{a,b',u(k),u(k-1), \ldots \ldots ,y(k-1),y(k-2), \ldots\} + e_p(k) \tag{3.12}$$

For ARMA-models this extension to stochastic models becomes:

$$y(k) = - A(z^{-1})y(k) + [\ b_0 + B(z^{-1})\ ]\ u(k) + e_p(k) \tag{3.13}$$

If we want to extend MIMO-models with a prediction error we have to find a relation conformable to (3.12) in vector form for the MFD.

$$\underline{y}(k) = \text{Func}\{\alpha,\beta,\underline{u}(k),\underline{u}(k-1), \ldots \ldots ,\underline{y}(k-1),\underline{y}(k-2), \ldots\} + \underline{e}_p(k) \tag{3.14}$$

We will try to find such a relation in two steps. First we rewrite (3.8) into

$$\underline{y}(k) = - \Sigma_\mu^{-1}(z)P'(z)\underline{y}(k) + \Sigma_\mu^{-1}(z)Q(z)\underline{u}(k) \tag{3.15}$$

with :
$$P(z) = \Sigma_\mu(z) + P'(z)$$
$$p'_{ij}(z) = - \alpha_{ij,\mu_j} z^{\mu_j - 1} - \ldots - \alpha_{ij,2} z - \alpha_{ij,1}$$
$$\Sigma_\mu(z) = \begin{bmatrix} z^{\mu_1} & 0 & \cdots & 0 \\ 0 & z^{\mu_2} & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & z^{\mu_q} \end{bmatrix} \tag{3.16}$$

From this we see that in the right hand side of (3.15) I/O-samples for sample moments greater or equal to k may occur. However, if we restrict

our selves to canonical forms, only I/O-samples at sample moments less than or equal to k occur in (3.15). This canonical restriction will be made temporarily, because a pseudo-canonical expression conformable to (3.14) is hard to get directly.

For canonical forms polynomial matrix P'(z) becomes:

$$p'_{ij}(z) = -\alpha_{ij,\nu_{ij}} z^{\nu_{ij}-1} - \ldots - \alpha_{ij,2} z - \alpha_{ij,1} \qquad (3.17)$$

with : $\nu_{ij} = \min(\nu_i, \nu_j)$ for $j \geq i$

$\nu_{ij} = \min(\nu_i+1, \nu_j)$ for $j<i$

Because of the condition on ν_{ij} for $j<i$, and dependent on the structure of the model, still present samples of the output occur in the right hand side of (3.15), if $\nu_{ij}=\nu_i+1$.

So the second step is to rewrite (3.15) such that the prediction error can be introduced.

$$R(\alpha)\underline{y}(k) = -\Sigma_\nu^{-1}(z)P''(z)\underline{y}(k) + \Sigma_\nu^{-1}(z)Q(z)\underline{u}(k) \qquad (3.18)$$

with : $\Sigma_\nu^{-1}(z)P'(z) = \Sigma_\nu^{-1}(z)P''(z) + R'(\alpha)$

$R(\alpha) = R'(\alpha) + I$

$$R(\alpha) = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -\alpha_{21,\nu_2+1} & 1 & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ & & & 1 & 0 \\ -\alpha_{q1,\nu_q+1} & & \cdots & -\alpha_{q(q-1),\nu_q+1} & 1 \end{bmatrix} \qquad (3.19)$$

Note that $R(\alpha)$ is square and regular and $\alpha_{ij,(\nu_i+1)}=0$ if $(\nu_i+1)>\nu_j$ for $j<i$.

Now similar to the SISO case and consistent with (3.14) a one-step-ahead prediction error can be introduced.

$$\underline{y}(k) = -R^{-1}(\alpha)\Sigma_\nu^{-1}(z)P''(z)\underline{y}(k) + R^{-1}(\alpha)\Sigma_\nu^{-1}(z)Q(z)\underline{u}(k) + \underline{e}_p(k) \qquad (3.20)$$

If this is written in a proper stochastic canonical MFD we get

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + \Sigma_{\nu}(z)R(\alpha)\underline{e}_p(k) \qquad (3.21)$$

Now the equivalent SSM of this MFD has to be found.
If we consider the prediction error as an extra additive q-dimensional input vector with polynomial matrix $\Sigma(z)R$ instead of $Q(z)$, then the SSM equivalent, if it exists, must look like

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K\underline{e}_p(k) \qquad (3.22) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + L\underline{e}_p(k) \end{aligned}$$

We have seen before that for canonical forms the transformation from SSM- to MFD-parameters (from (b,d)- to β-parameters) is invertible and given by

$$\overline{B}(\beta) = M(\alpha)B(b,d) \qquad (3.13)$$

So also the transformation from (k,ℓ)- to the parameters of $\Sigma(z)R$ is invertible and given by

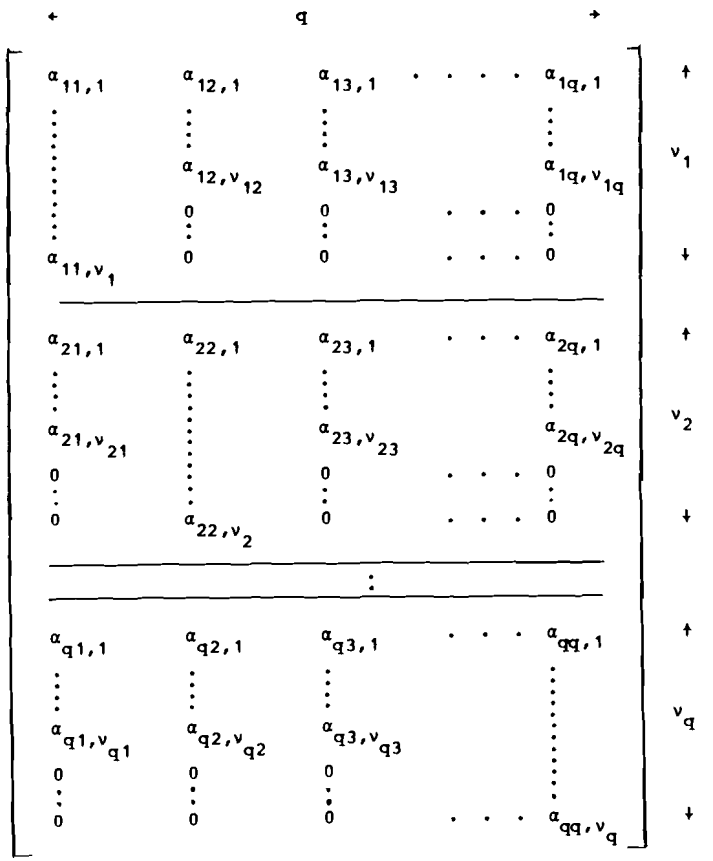
$$\overline{R}(\alpha) = M(\alpha)K(k,\ell) \qquad (3.23)$$

with $\overline{R}(\alpha)$ and $K(k,\ell)$ given by (3.24) and (3.25). $K(k,\ell)$ has the same structure as $B(b,d)$.

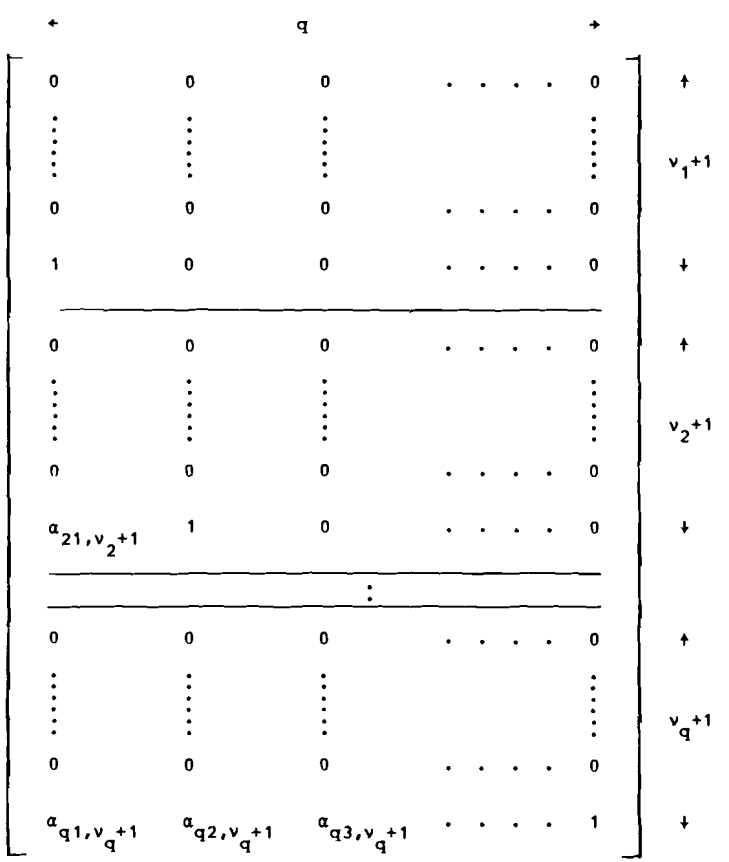
Given the parameters of $R(\alpha)$ we don't have to invert $M(\alpha)$ to solve (3.23). Using the nice properties of $R(\alpha)$ and the structure of $K(k,\ell)$, (3.23) is equivalent to

$$\begin{aligned} R(\alpha) &= R(\alpha)L \qquad (3.26) \\ 0 &= N(\alpha)K - O(\alpha)L \end{aligned}$$

with $N(\alpha)$ and $O(\alpha)$ given by (3.26-a) and (3.26-b).
Note that $N(\alpha)$ is a square matrix and has a similar structure as $M(\alpha)$, so $N(\alpha)$ is regular (see J. Pfennings 1983).
As a result the equivalent canonical SSM of the generalisation from the



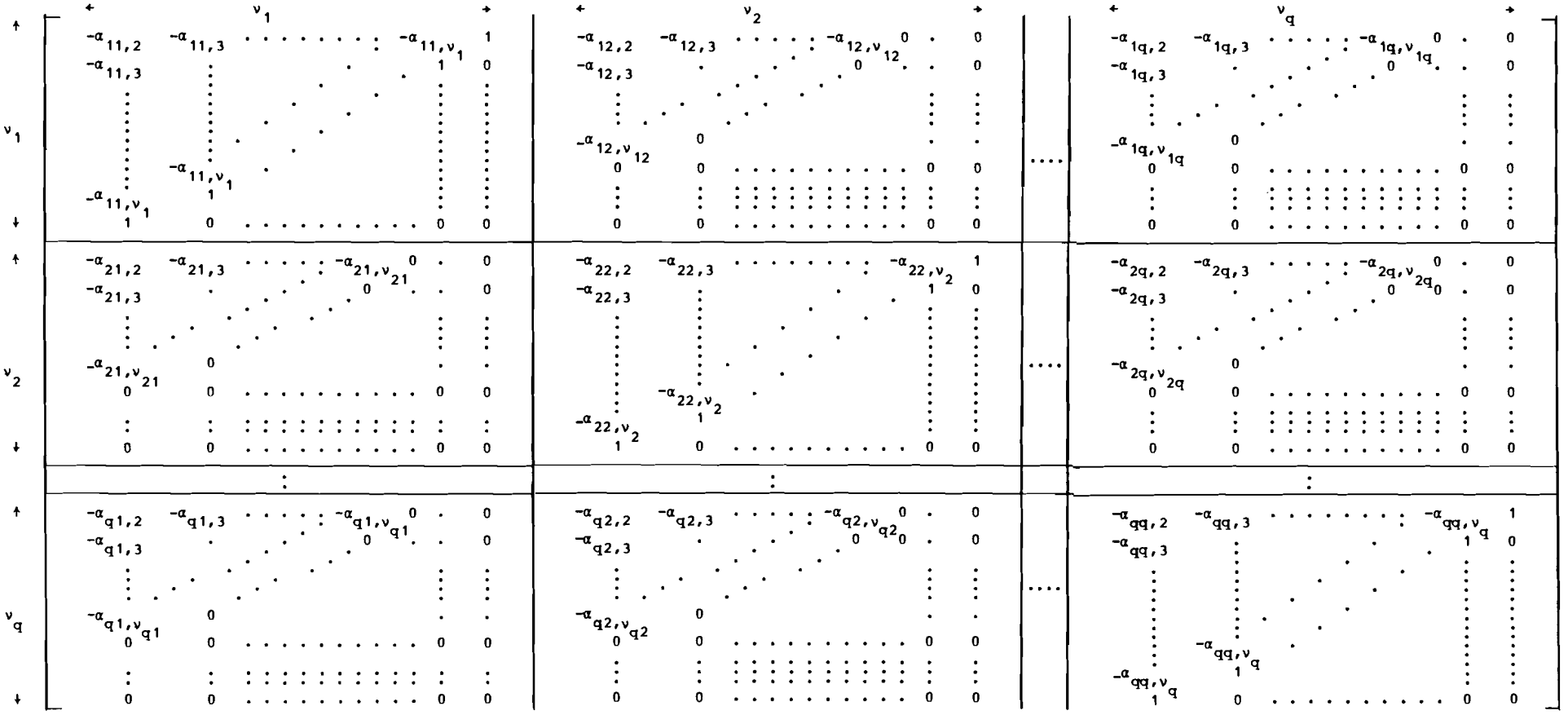
(3.26-a) Matrix $O(\alpha)$.



(3.24) Matrix $\overline{R}(\alpha)$. (Compare with (2.16-b): Canonical form of $\overline{B}(\beta)$)

$$
\begin{array}{c}
\leftarrow \quad q \quad \rightarrow \\
\left[\begin{array}{llll}
\ell_{11} & \ell_{12} & \ell_{13} & \ldots \ell_{1q} \\
k_{1,11} & k_{1,12} & k_{1,13} & \ldots k_{1,1q} \\
k_{1,21} & k_{1,22} & k_{1,23} & \ldots k_{1,2q} \\
\vdots & \vdots & \vdots & \vdots \\
k_{1,\mu_1 1} & k_{1,\mu_1 2} & k_{1,\mu_1 3} & \ldots k_{1,\mu_1 q} \\
\hline
\ell_{21} & \ell_{22} & \ell_{23} & \ldots \ell_{2q} \\
k_{2,11} & k_{2,12} & k_{2,13} & \ldots k_{2,1q} \\
k_{2,21} & k_{2,22} & k_{2,23} & \ldots k_{2,2q} \\
\vdots & \vdots & \vdots & \vdots \\
k_{2,\mu_2 1} & k_{2,\mu_2 2} & k_{2,\mu_2 3} & \ldots k_{2,\mu_2 q} \\
\hline
 & & \vdots & \\
\hline
\ell_{i1} & \ell_{i2} & \ell_{i3} & \ldots \ell_{iq} \\
k_{i,11} & k_{i,12} & k_{i,13} & \ldots k_{i,1q} \\
k_{i,21} & k_{i,22} & k_{i,23} & \ldots k_{i,2q} \\
\vdots & \vdots & \vdots & \vdots \\
k_{i,\mu_i 1} & k_{i,\mu_i 2} & k_{i,\mu_i 3} & \ldots k_{i,\mu_i q} \\
\hline
 & & \vdots & \\
\hline
\ell_{q1} & \ell_{q2} & \ell_{q3} & \ldots \ell_{qp} \\
k_{q,11} & k_{q,12} & k_{q,13} & \ldots k_{q,1q} \\
k_{q,21} & k_{q,22} & k_{q,23} & \ldots k_{q,2q} \\
\vdots & \vdots & \vdots & \vdots \\
k_{q,\mu_q 1} & k_{q,\mu_q 2} & k_{q,\mu_q 3} & \ldots k_{q,\mu_q q}
\end{array}\right]
\begin{array}{l}
\uparrow \\
 \\
 \\
\mu_1+1 \\
\downarrow \\
\uparrow \\
 \\
 \\
\mu_2+1 \\
\downarrow \\
 \\
\uparrow \\
 \\
 \\
\mu_i+1 \\
\downarrow \\
 \\
\uparrow \\
 \\
 \\
\mu_q+1 \\
\downarrow
\end{array}
\end{array}
$$

(3.25) Matrix K(k,ℓ). (Compare with (2.10): B(b,d))



(3.26-b) Matrix N(α). (Compare with (2.17-b): Canonical form of M(α))

SISO-case to a stochastic canonical MFD is

$$\boxed{\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K(\alpha)\underline{e}_p(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{e}_p(k) \end{aligned}} \qquad (3.27)$$

with : $K(\alpha) = N^{-1}(\alpha)O(\alpha)$

The prediction error model could not be found easily for a given pseudo-canonical MFD. However if we transform model (3.27) to a pseudo-canonical form then the one-step-ahead prediction error model for pseudo-canonical SSM's can be found.

3.3 Equation Error Model (EEM)

Also in this paragraph we will use different notations for the MFD in the SISO- (ARMA) and in the MIMO-case. It may be helpful to use eq.(3.4) and (3.5) in case of a generalisation from SISO- to MIMO-models

A deterministic model can be extended with a stochastic variable, the equation error. This equation error is defined as the model misfit that can be expressed as a linear expression of the model parameters. For the SISO-case:

$$y(k) = \underline{\varphi}^T(k)\underline{\theta} + e_e(k) \qquad (3.29)$$

with : $\underline{\varphi}(k)$: Vector with in- and output samples.
$\underline{\theta}$: Vector with model parameters.

For ARMA representations of processes the model extended with this stochastic variable becomes:

$$y(k) = -A(z^{-1})y(k) + \left[\, b_0 + B(z^{-1}) \,\right] u(k) + e_e(k) \qquad (3.30)$$

From this we see that for SISO-models the equation error is the same stochastic variable as the prediction error of paragraph 3.2. For the generalisation of (3.30) to MIMO-models two conditions have to be

fulfilled: - The error vector must be linear in the parameters
- If only one output is non-zero the expression must not violate eq.(3.29).

So for the MIMO-case:

$$\underline{y}(k) = \Omega^T(k)\underline{\theta} + \underline{e}_e(k) \tag{3.31}$$

with : $\Omega^T(k)$: Matrix with in- and output samples.
$\underline{\theta}$: Parameter vector.

To be able to introduce an equation error into the deterministic MFD we have to rewrite the MFD in the next form (see par.3.2),

$$\underline{y}(k) = -\Sigma_\mu^{-1}(z)P'(z)\underline{y}(k) + \Sigma_\mu^{-1}(z)Q(z)\underline{u}(k) \tag{3.15}$$

with $\Sigma_\mu(z)$ and $P'(z)$ given by (3.16) and (3.15).
Because $\Sigma(z)$ only contains time shifts, and $P'(z)\underline{y}(k)$ and $Q(z)\underline{u}(k)$ are linear expressions of the model parameters, an equation error satisfying (3.31) can be introduced. If also the original notation for the deterministic part of the MFD is chosen, the generalisation of the equation error model (3.30) to MIMO-models is

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + \Sigma_\mu(z)\underline{e}_e(k) \tag{3.32}$$

From this we see that for MIMO-models the equation error is not equal to the one-step-ahead prediction error. If we compare (3.32) with (3.21) we see that in the canonical form they relate to each other as

$$R(\alpha)\underline{e}_p(k) = \underline{e}_e(k) \tag{3.33}$$

In trying to find the equivalent SSM of (3.32) we have to distinguish between the canonical and pseudo-canonical forms

3.3.1 Canonical Form

Our starting point in this chapter was a SSM. So the MFD- and SSM-parameters relate to each other as

$$\overline{B}(\beta) = M(\alpha)B(b,d) \tag{3.3}$$

For canonical forms we have found that $M(\alpha)$ is square and regular, so that any set of MFD-parameters can be transferred to a set of SSM-parameters.
If we consider the equation error in (3.32) as an extra additive input to our system the equivalent SSM must look like

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K'\underline{e}_e(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + L'\underline{e}_e(k) \end{aligned} \tag{3.34}$$

The parameters in K' and L' can be found by solving an equivalent expression (eq.(3.35)) as for the b- and d-parameters (eq.(3.3))

$$\bar{R}(0) = M_v(\alpha)K(k',\ell') \tag{3.35}$$

with : $\bar{R}(0) = \bar{R}(\alpha)\Big|_{\alpha=0}$ (see eq.(3.24)).
$K(k',\ell')$ and $M_v(\alpha)$ given by (3.25) and (2.17-b).

Using the nice properties of R(0), (3.35) can be solved directly by splitting it up into

$$\begin{aligned} I &= R(\alpha)L' \\ 0 &= N(\alpha)K' - O(\alpha)L' \end{aligned} \tag{3.36}$$

Because $R(\alpha)$ and $N(\alpha)$ are regular (they have to be because $M(\alpha)$ is regular), the equivalent SSM of 3.32 is

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K'(\alpha)\underline{e}_e(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + R^{-1}(\alpha)\underline{e}_e(k) \end{aligned} \tag{3.37}$$

with : $K'(\alpha) = N^{-1}(\alpha)O(\alpha)R^{-1}(\alpha) = K(\alpha)R^{-1}(\alpha)$.
$R(\alpha)$ given by (3.19).

Notice that if we compare (3.37) and (3.27), relation (3.33) is confirmed in state space representation.

3.3.2 Pseudo-Canonical Form

Again we have to refer to our starting point in this chapter. For pseudo-canonical SSM's and equivalent MFD's we know that the transformation between the two parameter sets is not invertible. So an equivalent SSM of (3.32) can only be found if we can find a (k',ℓ')-parameter set such that

$$\bar{R}^{*}(0) = M(\alpha)K(k',\ell') \tag{3.38}$$

$K(k',\ell')$ and $M(\alpha)$ are given by (3.25) and (3.17-a). Matices with a * are pseudo-canonical forms of the matrices without a *. They will not be used here and therefore not given explicitly.
Eq.(3.38) can be split up into

$$\begin{aligned} I &= R^{*}(\alpha)L' \\ 0 &= N^{*}(\alpha)K' - O^{*}(\alpha)L' \end{aligned} \tag{3.39}$$

So only if R^{*} and N^{*} can be inverted, an equivalent SSM can be found being

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K'(\alpha)\underline{e}_e(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + R^{*-1}(\alpha)\underline{e}_e(k) \end{aligned} \tag{3.40}$$

with : $K'(\alpha) = N^{*-1}(\alpha)O^{*}(\alpha)R^{*-1}(\alpha)$

This will generally not be true because $M(\alpha)$ can not be inverted. So an equivalent of the general pseudo-canonical MFD can not be found.

We assumed that the deterministic MFD is causal. So the deterministic part can be transformed to an equivalent deterministic SSM. The stochastic pseudo-canonical MFD (eq.(3.32)) could not be transformed to an equivalent SSM in the form of eq.(3.34). To find the deterministic SSM extended with an equation error, we will first try to find the equivalent SSM of a model with time shifted equation errors.

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + \underline{e}(k) \tag{3.41}$$

Notice that also this error is linear in the parameters, but it is no generalisation of the SISO-model (3.30).

Eq.(3.3) is valid, so we have to find a set of (k',ℓ')-parameters such that

$$\bar{R}(\rho) = M(\alpha)K(k',\ell') \tag{3.42}$$

with : $\rho_{i,1i} = 1$

$\rho_{i,j\ell} = 0$ if $i \neq \ell$ or $j \neq 1$

$\bar{R}(\rho)$ is equivalent to $\bar{B}(\beta)$ except for its column dimension.

(see eq.(3.42-a) and (2.20)).

It appears that (3.42) can be satisfied for all pseudo-canonical forms if

$$L = 0 \tag{3.43}$$

$$k'_{i,\mu_i i} = 1$$

$$k'_{i,j\ell} = 0 \quad \text{if } j \neq \mu_i \text{ or } \ell \neq i$$

So the equivalent model of (3.41) is

$$\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k) + K'\underline{e}(k) \tag{3.44}$$

$$\underline{y}(k) = C\underline{x}(k) + D\underline{u}(k)$$

with : $K' =$ (3.45)

$$\left[\begin{array}{cccccc}
0 & 0 & 0 & \cdots & 0 & \uparrow \\
\vdots & \vdots & \vdots & & \vdots & \mu_1 \\
0 & 0 & 0 & \cdots & 0 & \\
1 & 0 & 0 & \cdots & 0 & \downarrow \\
\hline
0 & 0 & 0 & \cdots & 0 & \uparrow \\
\vdots & \vdots & \vdots & & \vdots & \mu_2 \\
0 & 0 & 0 & \cdots & 0 & \\
0 & 1 & 0 & \cdots & 0 & \downarrow \\
\hline
 & & & \vdots & & \\
\hline
0 & 0 & \cdots & 0 & 0 & \uparrow \\
\vdots & \vdots & & \vdots & \vdots & \mu_q \\
0 & 0 & \cdots & 0 & 0 & \\
0 & 0 & \cdots & 0 & 1 & \downarrow
\end{array}\right]$$

Now that we found an equivalent of (3.41), we can find the equivalent model of the equation error model (3.32) by a revers shift in time of the

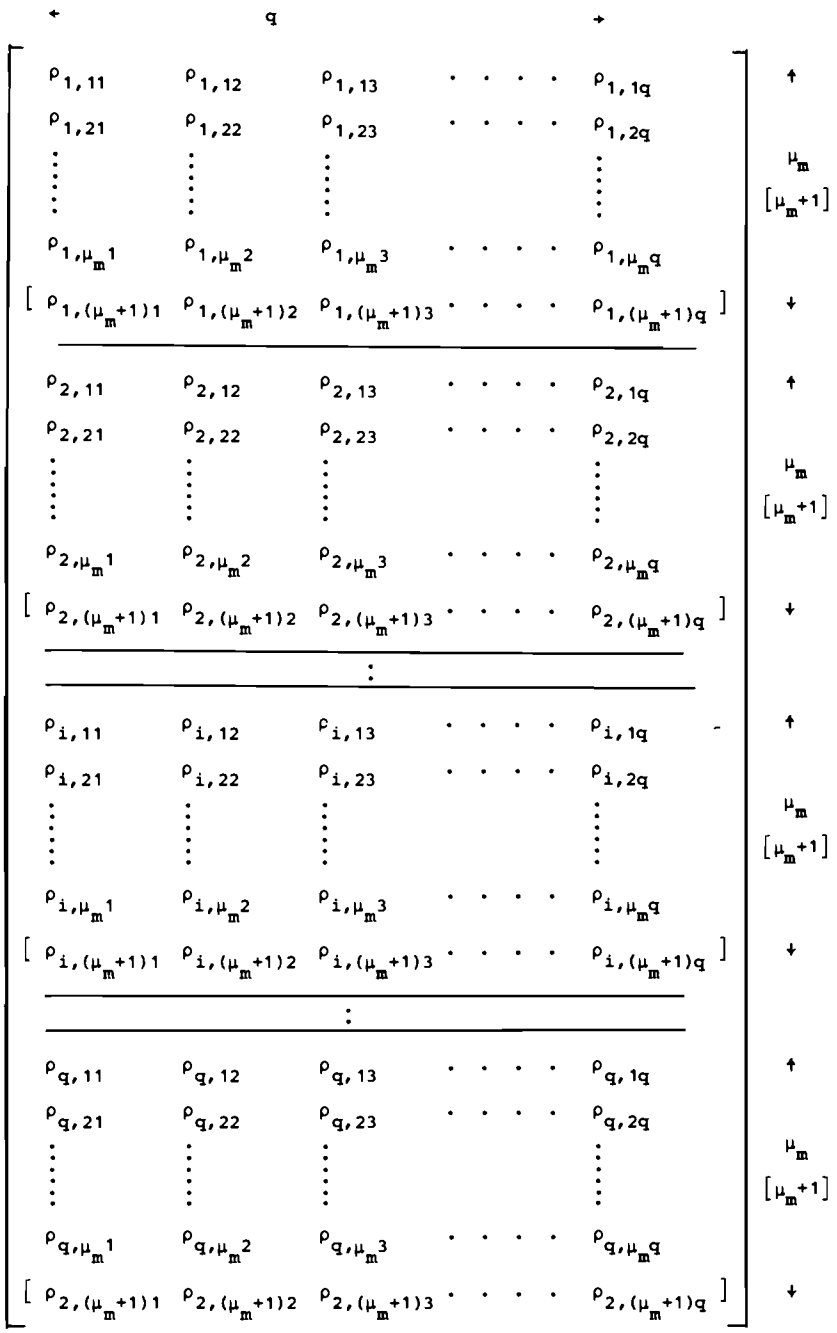


Fig. 3.42-a Matrix $\overline{R}(\rho)$

equation errors.

$$\boxed{\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K'\Sigma_{\mu}(z)\underline{e}_e(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) \end{aligned}} \qquad (3.46)$$

Of course this is not an ordinary SSM.

3.4 Innovation Error Model (IEM)

In the previous sections stochastic variables were introduced in a SSM by a transformation from the equivalent MFD. The only extra parameters of the resulting stochastic models are those who describe the stochastical behaviour of the introduced errors.
If we define the innovation error as the error in the estimated output, and if we introduce (nxq) parameters in a matrix (K) that distributes the innovation error over the next state elements, our model becomes:

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K\underline{e}_i(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{e}_i(k) \end{aligned} \qquad (3.47)$$

Given the parameter values, we now transform this model to the equivalent MFD. We have seen before (chapter 2) that the parameters of B and D are transformed according to

$$\overline{B}(\beta) = M(\alpha)B(b,d) \qquad (2.15)$$

With $\overline{B}(\beta)$, $M(\alpha)$ and $B(b,d)$ given by (2.16-a), (2.17-a) and (2.10). Similarly the 'parameters' of K and I are transformed according to

$$\overline{R}(\rho) = M(\alpha)K(k,I) \qquad (3.48)$$

with : $K(k,I) = K(k,\ell)\Big|_{L=I}$

With $R(\rho)$ and $K(k,\ell)$ given by (3.42-a) and (3.25). The equivalent MFD of (3.47) is

$$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + R(z)\underline{e}_i(k) \qquad (3.49)$$

With : $R(z) = [\ r_{i\ell}(z)\ ]$ $(i,\ell=1, \ldots ,q)$ (3.50)

$$r_{i\ell}(z) = \rho_{i,\mu_m\ell} z^{\mu_m-1} + \ldots + \rho_{i,1\ell}$$

$$r_{i\ell}(z) = \rho_{i,(\mu_m+1)\ell} z^{\mu_m} + \rho_{i,\mu_m\ell} z^{\mu_m-1} + \ldots + \rho_{i,1\ell}$$

for $\mu_i<\mu_m$

for $\mu_i=\mu_m$

From this we see that for the pseudo-canonical form R(z) contains more then (qxn) parameters, and the same conclusions drawn in sect.2.2 from the transformation from β- to (b,d)-parameters can be drawn from the transformation from ρ- to k-parameters.
For the canonical form we get

$$r_{i\ell}(z) = \rho_{i,\nu_i\ell} z^{\nu_i-1} + \ldots + \rho_{i,1\ell} \qquad (i\neq\ell)$$

$$r_{ii}(z) = z^{\nu_i} + \rho_{i,\nu_i i} z^{\nu_i-1} + \ldots + \rho_{i,1i} \qquad (3.51)$$

so that for the canonical form we have (qxn) parameters in R(z), and a transformation from (3,49) to (3.47) is always possible (see sect.2.3).

For SISO-models only canonical forms exists.

$$\underline{x}(k+1) = A\underline{x}(k) + \underline{b}u(k) + \underline{k}e_i(k)$$
$$y(k) = \underline{c}^T\underline{x}(k) + du(k) + e_i(k)$$

Its equivalent I/O-representation is the ARMAX model (see Åström and Bohlin 1965)

$$[\ 1+A(z^{-1})\ ]y(k) = [\ b_0+B(z^{-1})\ ]u(k) + [\ 1+C(z^{-1})\ ]e_i(k)$$

with : $C(z^{-1}) = c_1 z^{-1} + \ldots + c_{n'} z^{-n'}$ (see sect. 3.1)

and the relation between $\underline{c}$ and $\underline{k}$ parameters is given by

$$\begin{bmatrix} c_{n'} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} = \begin{bmatrix} -\alpha_1 & -\alpha_2 & \ldots & -\alpha_{n'} & 1 \\ -\alpha_2 & -\alpha_3 & & 1 & 0 \\ \vdots & & \iddots & & \vdots \\ -\alpha_{n'} & 1 & & 0 & 0 \\ 1 & 0 & \ldots & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ \hline \underline{k} \end{bmatrix}$$

$\alpha_i = -a_{n+1-i}$

$n' = n$

For pseudo-canonical forms the equivalent SSM of (3.49) exists if eq.(3.48) is fulfilled.

3.5 Review

In section 3.1 to 3.4 four different stochastic models are introduced. Fig. 3.52-a, 3.52-b and 3.52-c give an overview.

MFD	SSM
OEM :	
$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + P(z)\underline{e}_o(k)$	$\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k)$ $\underline{y}(k) = C\underline{x}(k) + D\underline{u}(k) + \underline{e}_o(k)$
PEM : (only canonical)	
$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + \Sigma_\nu(z)R(\alpha)\underline{e}_p(k)$	$\underline{x}'(k+1) = A\underline{x}'(k) + B\underline{u}(k) + K(\alpha)\underline{e}_p(k)$ $\underline{y}(k) = C\underline{x}'(k) + D\underline{u}(k) + \underline{e}_p(k)$
EEM :	
$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + \Sigma_\nu(z)\underline{e}_e(k)$	$\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k) + K(\alpha)R^{-1}(\alpha)\underline{e}_e(k)$ $\underline{y}(k) = C\underline{x}(k) + D\underline{u}(k) + R^{-1}(\alpha)\underline{e}_e(k)$
$P(z)y(k) = Q(z)\underline{u}(k) + \Sigma_\mu(z)\underline{e}_e(k)$	$\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k) + K'\Sigma_\mu(z)\underline{e}_e(k)$ $\underline{y}(k) = C\underline{x}(k) + D\underline{u}(k)$
IEM : (only canonical)	
$P(z)\underline{y}(k) = Q(z)\underline{u}(k) + R(z)\underline{e}_i(k)$	$\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k) + K\underline{e}_i(k)$ $\underline{y}(k) = C\underline{x}(k) + D\underline{u}(k) + \underline{e}_i(k)$
With : $K(\alpha) = N^{-1}(\alpha)O(\alpha)$ $N(\alpha)$, $O(\alpha)$, $R(\alpha)$, K' and $\Sigma_\mu(z)$ given by (3.26-a),(3.26-b),(3.19), (3.45) and (3.16).	

Fig. 3.52-a Overview of stochastic models in state space and Matrix Fraction Description

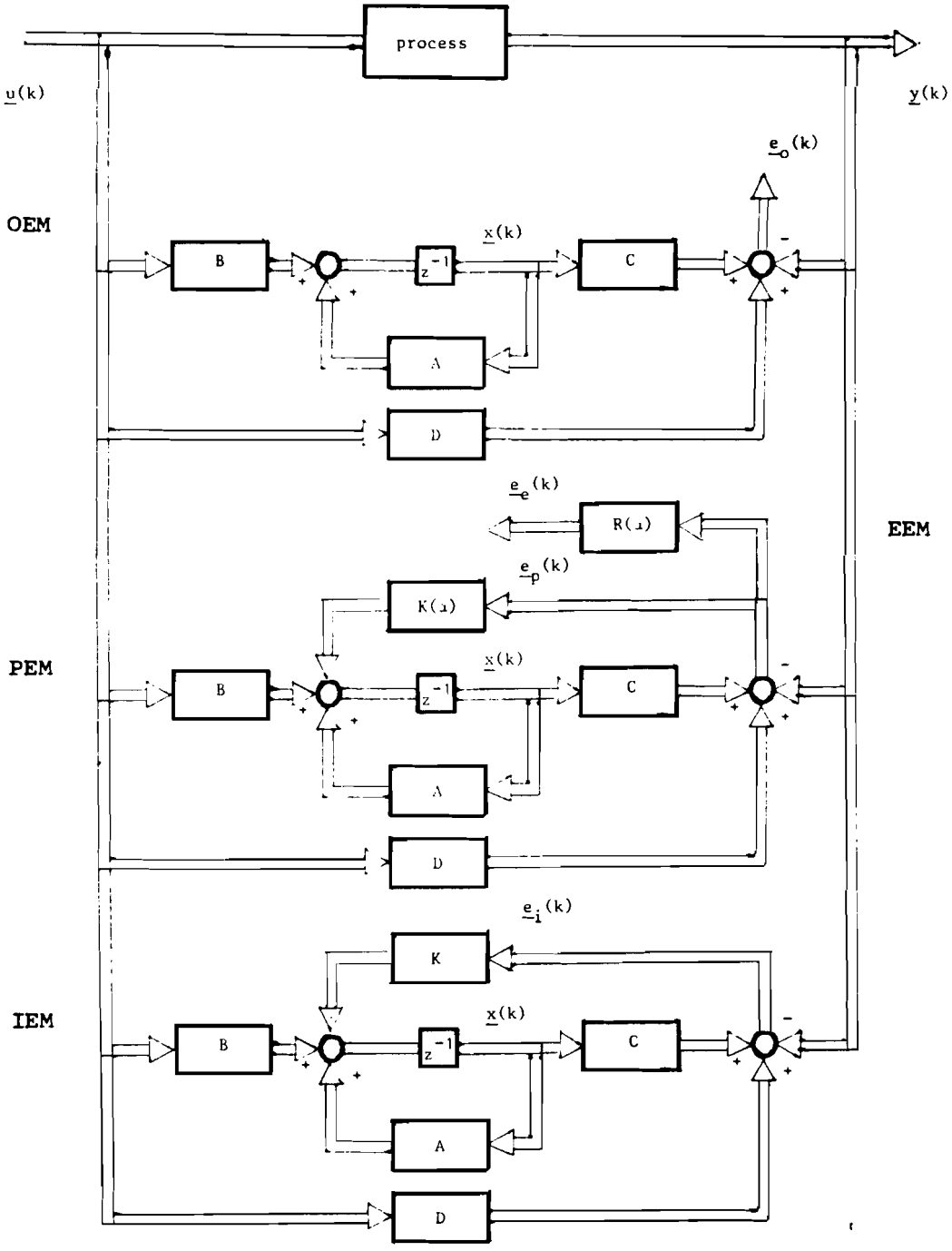


Fig. 3.52-b SSM's.

Canonical forms of PEM and EEM.

Pseudo-canonical form of OEM and IEM.

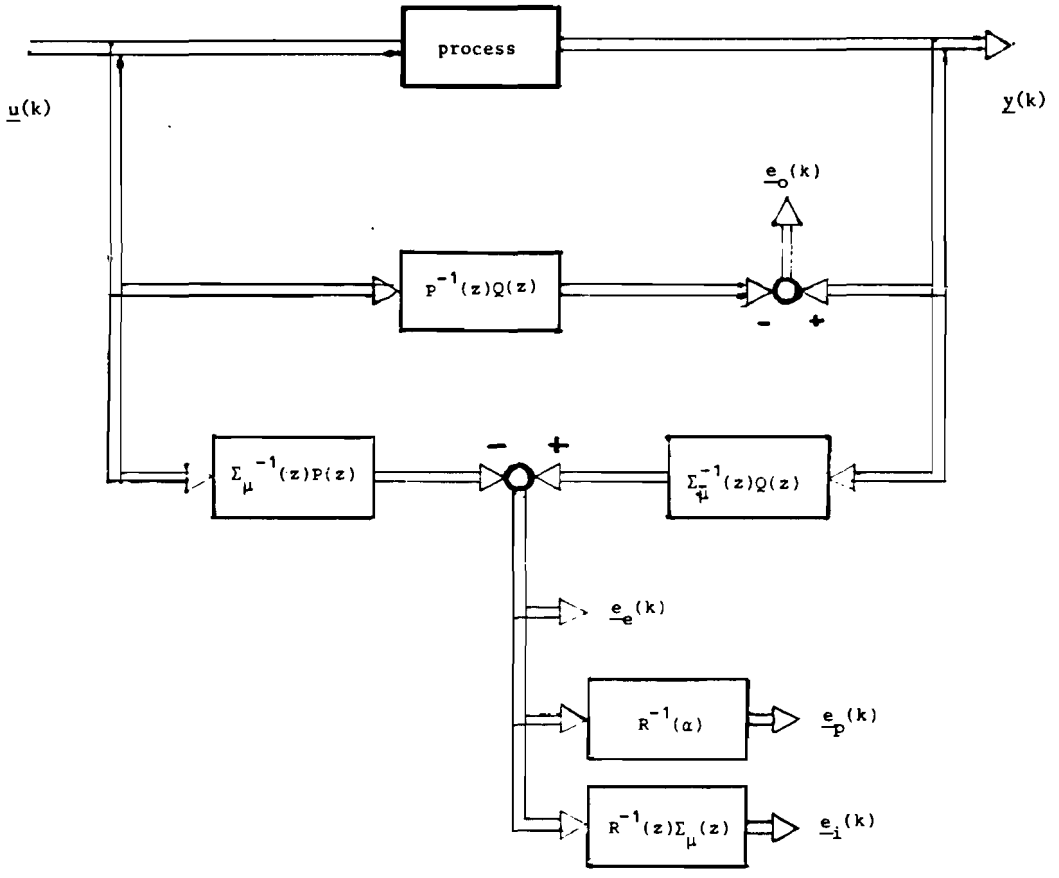


Fig. 3.52-c MFD's.

Canonical form of PEM.

Pseudo-canonical form of EEM, OEM and IEM.

The output error was introduced according to

$$\underline{y}(k) = \text{Func}\{\alpha,b,d,\underline{u}(k),\underline{u}(k-1), \ldots\} + \underline{e}_o(k) \qquad (3.9)$$

The one-step-ahead prediction error was introduced in MFD according to

$$\underline{y}(k) = \text{Func}\{\alpha,b,d,\underline{u}(k),\underline{u}(k-1), \ldots \ldots ,\underline{y}(k-1),\underline{y}(k-2), \ldots\} + \underline{e}_p(k) \qquad (3.14)$$

The equation error was introduced so that it was linear in the parameters. But similar to (3.9) and (3.14) the equation error could have been introduced conformable to

$$\begin{aligned}\Sigma_\mu(z)\underline{y}(k) = \text{Func}\{&\alpha,b,d,\underline{u}(k+\mu_m),\underline{u}(k+\mu_m-1), \ldots \\ &\ldots ,y_1(k+\mu_1-1),y_1(k+\mu_1-2), \ldots \\ &\ldots ,y_2(k+\mu_2-1),y_2(k+\mu_2-2), \ldots \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &\ldots ,y_q(k+\mu_q-1),y_q(k+\mu_q-2), \ldots\} + \Sigma_\mu(z)\underline{e}_e(k)\end{aligned} \qquad (3.53)$$

with : $\mu_m = \max(\mu_1,\ldots,\mu_q)$

The meaning of this expression is explained in fig. 3.53-a.

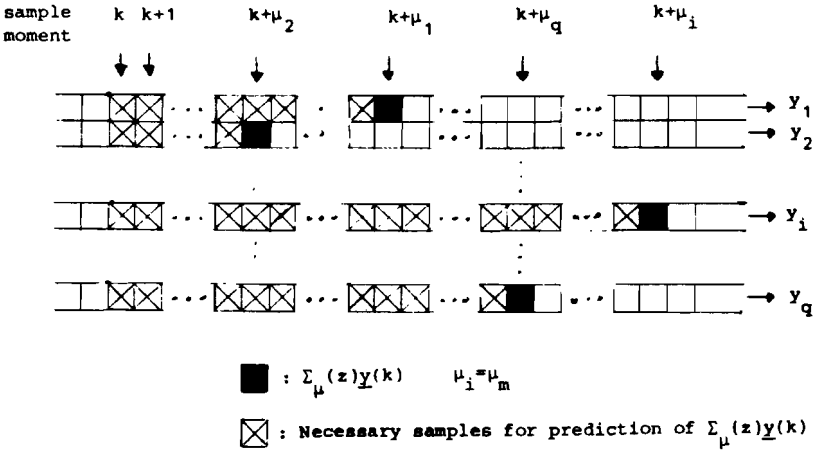


Fig. 3.53-a Explanation by eq.(3.53).

The equation error vector $\Sigma(z)\underline{e}(k)$ can be seen as a one-step-ahead prediction error of $\Sigma(z)\underline{y}(k)$.

The innovation error was introduced in SSM conformable to

$$\underline{y}(k) = \text{Func}\{\alpha,b,d,k,\underline{u}(k),\underline{u}(k-1), \ldots \ldots ,\underline{y}(k-1),\underline{y}(k-2), \ldots\} + \underline{e}_i(k) \tag{3.54}$$

Except for the pseudo-canonical MFD extended with an equation error, all stochastic MFD's in section 3.1 to 3.3 have an equivalent stochastic SSM that can be written as

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K(\alpha)\underline{e}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + L(\alpha)\underline{e}(k) \end{aligned} \tag{3.55}$$

For the PEM and OEM $L(\alpha)$ equals I so that the model sets defined by a PEM and OEM are part of the model set defined by a IEM. Only for specific cases the model set defined by a canonical EEM is part of the IEM model set. The model set defined by the stochastic MFD in its pseudo-canonical form is probably not a specific model set of the model set given by (3.53).

Because the deterministic part of the MFD's and SSM's in fig. 3.50.a are equal and no assumptions have been made on their stochastic properties of the noises, we may conclude that for given parameter values and a given data set

$$\underline{e}_e(k) = R(\alpha)\underline{e}_p(k) \tag{3.33}$$

$$\underline{e}_e(k) = \Sigma_\mu^{-1}(z)R(z)\underline{e}_i(k) \tag{3.56}$$

$$\underline{e}_e(k) = \Sigma_\mu^{-1}(z)P(z)\underline{e}_o(k)$$

All errors were introduced without any connection to an estimation algorithm. The innovation error was introduced with an extension of the parameter set, but also no assumptions were made on the stochastic behaviour of the innovation error.

The result of an estimation depends on this stochastic behaviour of the introduced noises, and the noise parameters are part of the model set. This will be treated in the next chapter.

4. Parameter Estimation

4.0 Introduction

In the previous chapter, 4 different stochastic models have been derived for the same observability pseudo-canonical forms.
A stochastic variable has been introduced that was assumed to be the cause of the model misfit.
In practice this model misfit could have been caused by system noises or observation noises uncorrelated to in- or outputs, or because the deterministic part of the process is not in our deterministic model set.
By defining four different stochastic variables we have created 4 different stochastic model sets, though all contained in the innovation representation (EEM only under specific restrictions).
The stochastic properties of the variables are also part of these model sets.

If we now have to estimate a model (with given structural indices) that fits our data as good as possible according to some criterion, we have to estimate the model parameters such that the errors are minimised according to this minimisation criterion.
In sect. 4.1 two criteria are discussed briefly. The LS-criterium and its impact on the different stochastic model representations is treated comprehensively in sect.4.2. In sect.4.3 hill-climbing methods, necessary for non-linear estimators, are described briefly.
After the correction of the data and the estimated model is discussed as a necessity after signal processing in sect.4.4, the implemented algorithm is described in sect.4.5.

4.1 Minimisation Criteria

For a Least Squares (LS) criterion the loss function that has to be minimised is

$$V(\underline{\theta}) = \sum_{k}^{N} \underline{e}^{T}(k)\underline{e}(k) \qquad (4.1)$$

with $\underline{e}(k)$ the model error discussed in chapter 3.
This LS-criterium is minimised if the power in this error variable is minimised. The advantage of this criterium is its simplicity and its great applicability in all kinds of representations. Many other minimisation criteria are based on this LS-criterium or can be seen as generalisations of it.

One of these generalisations is the Maximum likelihood (ML) criterium. This criterium looks for the parameter value that minimises

$$V(\underline{\theta}) = - \text{Log } L(\underline{\theta}) \qquad (4.2)$$

with $L(\underline{\theta})$ the likelihood function (see Åström 1980, Eykhoff et al 1985) This likelihood function is the probability density function of the parameters for a given I/O-data set. To be able to maximise this likelihood function with respect to $\underline{\theta}$, it must be known or assumed to be known. If the error sequences are white Gaussian noise sequences with covariance matrix equal to the identity matrix, so if

$$E\left\{e_i(k)e_i(\ell)\right\} = 0 \qquad \text{for } \ell \neq k \qquad (4.3)$$
$$\text{cov}\left\{\underline{e}(k)\underline{e}^{T}(k)\right\} = I$$

Then this ML-criterium is equal to the LS-criterion

Because of its simplicity and because the LS- and ML-criterion are equal if the process is in our model set defined by a stochastic model with error sequences that are Gaussian, white and therefore uncorrelated, the LS-criterium is chosen to be implemented. A possible next step is to implement a ML-criterion for stochastic model sets with Gaussian white noise sequences but with unknown covariance matrix.

4.2 LS-Estimator

If we want to find a model that fits the I/O-data best for a LS-criterium, we have to look for that parameter vector that minimises the power in the error variables

$$\min_{\underline{\theta}=\underline{\theta}_m} V(\underline{\theta}) = \min \left\{ \sum_k \underline{e}^T(k)\underline{e}(k) \right\} \tag{4.5}$$

In the minimum the derivatives must equal zero

$$\left. \frac{\partial V(\underline{\theta})}{\partial \underline{\theta}} \right|_{\underline{\theta}=\underline{\theta}_m} = 0 \tag{4.6}$$

If the error variable is linear in the parameters, the parameter set that satisfies (4.5) can be given as a simple mathematical expression. That is why we have to distinguish between linear and non-linear LS-estimators.

4.2.1 (Non-)Linear Estimators

Linearity in the parameters

If the error is a linear function of the parameters, the model can be written as

$$\underline{y}(k) = \Omega(k)\underline{\theta} + \underline{e}(k) \tag{4.7}$$

with : $\underline{\theta}$: Parameter vector

$\Omega(k)$: Matrix with observations (input and/or output samples)

The derivative of the loss function is

$$\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}} = \sum_k \frac{\partial}{\partial \underline{\theta}} \left(\underline{e}^T(k)\underline{e}(k) \right) \tag{4.8}$$

$$= -2 \sum_k \Omega^T(k)\{ \underline{y}(k) - \Omega(k)\underline{\theta} \} \tag{4.9}$$

For single output models the linear expresion becomes

$$y(k) = \underline{\Omega}^T(k)\underline{\theta} + e(k) \qquad (4.10)$$

The solution of (4.10) for $\underline{\theta}$ is a simple mathematical expression (see v/d Boom 1982). By considering our MIMO-model as q MISO- (Multiple-Input Single-Output) models, also $\underline{\theta}$ in (4.9) can be given as a simple mathematical expression (see Carriere 1984).

Non-linear Estimators

If the error in our model is not linear in the parameters, a nice mathematical expression for the wanted parameter vector $\underline{\theta}$ can not be given. Now we have to search for a parameter vector that satisfies (4.6).

The parameters found in this way do not necessarily have to be the parameters of the global minimum of the LS loss function. Also local minima, saddle points or even maxima satisfy (4.6). That is way we directly search for minima of the error function.

If nothing is known about the value of the absolute minimum we do not know for sure if the found minimum is local or global.

The methods we use for searching a minimum are known as hill-climbing methods although we will descent on a hill.

4.2.2 Stochastic Representations

We only considered linear (input-output behaviour) models. All linear stochastic models can be written as

$$\underline{y}(k) = \Omega^T(k)\underline{\theta}' + \underline{e}(k) \qquad (4.11)$$

Dependent on the model representation (see chapter 3) $\Omega(k)$ is filled with in- and/or outputs, and $\underline{\theta}'$ is some (non-)linear function of the wanted parameter vector $\underline{\theta}$. If the model is estimated by a LS-estimator, in the

minimum the derivative of the loss function equals zero

$$\left.\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}_{LS}} = -2\sum_{k} \underline{\Omega}^{T}(k)\underline{e}(k) = 0 \qquad (4.12)$$

So for an infinite number of samples a necessary condition is that the error is uncorrelated with the process variables in $\underline{\Omega}(k)$. It can be proved (see Eykhoff 1970) that this condition is also sufficient for a set of parameters to satisfy (4.5). So the best LS-estimate of $\underline{y}(k)$ for some specific $\underline{\Omega}(k)$ and $\underline{\theta}'$ is obtained if $\underline{e}(k)$ is uncorrelated to elements in $\underline{\Omega}(k)$.

If an OEM is estimated, the power in the output error is minimised. From (3.9) and (4.11) we see that for an OEM the errors are uncorrelated to inputs. This model is specifically suitable for simulation purposes.

For the estimated PEM the power in the prediction error is minimised. For a PEM $\underline{\Omega}(k)$ contains all data available at sample moment k (3.14). Because of this the PEM is well suited for prediction purposes.

The equation error whose power is minimised if an EEM is estimated, has no direct fysical interpretation. Its linearity in the parameters makes it suitable for fast estimation algorithms.
From (3.31) and (4.11) we see that the best LS-estimate is also uncorrelated to all available data at sample moment k.

Estimation results for OEM's, EEM's, and PEM's may differ extremely, dependent on the dynamic process behaviour.

From (3.47) and (4.11) we may conclude that what has been said about the PEM above, is also valid for the IEM, except that the prediction is called the innovation and the parameter set is extended. Because the IEM is defined for a SSM it is less practical to estimate an IEM in Matrix Fraction representation (eq.(3.49)).
Dependent on the model set of the pocess, the estimation of an IEM may give the same results as an OEM, EEM or PEM estimation, but generally better because the modelset of IEM covers all others (EEM only under specific restrictions).

Degree of non-linearity

For a MFD the equation error is linear in the parameters. The transformation from MFD- to SSM-parameters is given by the non-linear transformation (2.15), so that the equation error for a SSM is not linear in the parameters and a 'measure' of non-linearity can be obtained from (2.15).

The prediction error is related to the equation error by (3.33), so this prediction error is 'less' linear and a 'measure' of its non-linearity can be obtained by combination of (2.15) and (3.33).
For the OEM (K=0) and the IEM we may write

$$\underline{y}(k) = CA^k\underline{x}(0) + CA^k B\underline{u}(0) + \ldots + CAB\underline{u}(k-1) + D\underline{u}(k) + CA^k K\underline{e}(0) + \ldots + CAK\underline{e}(k-1) + \underline{e}(k) \qquad (4.13)$$

From this we see that although (4.13) is satisfying (4.11) by proper choice of $\underline{\theta}'$ (namely the $\underline{\theta}'$=Markov parameters), the output error is highly non-linear and the innovation error even less linear. Due to these non-linearities, difficulties may arise during estimation (see par. 4.5), and different hill-climbing methods may be desirable to deal with different degrees of linearity.

4.3 Hill Climbing methods.

If we use the LS-criterion for the parameter estimation and if the model error is a non-linear function of the parameters, we have to search for the optimal model that satisfies (4.15).

$$\min \sum_k \underline{e}^T(k)\underline{e}(k) = V(\underline{\theta}_m) \qquad (4.14)$$

$$\left.\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}_m} = 0 \qquad (4.15)$$

If for a given set of parameters the derivatives of the loss function can not be calculated (see appendix), direct search methods have to be used.

If the first derivatives can be calculated, these can be used to search in the downwards direction. The Conjugate Gradient method is one of these methods.
If also the second derivatives are available, even more powerful methods (in final rate of convergence) can be used. For instance the Newton Rapson Method. Of course the second derivatives can be approximated, resulting in Modified or Quasi Newton methods. This approximation can be made in different ways. (For a comprehensive overview of hill-climbing methods see Eykhoff 1974).
It depends on the mathematical expression of the chosen stochastic model error variable which method will be best, fast or most accurate.

4.3.1 Conjugate Gradient Method

If the loss function is quadratic in the parameters and if the number of parameters equals N, then this conjugate gradient method is garanteed to find the minimum in N steps (see Powell and Fletcher 1964). Every step explores a direction conjugate to the one before with respect to the Hessian of the error function.

The method is best described if we assume a quadratic loss function (higher term in Taylor expansion may be neglected around the minimum). In the minimum the derivatives are equal to zero and the loss function becomes:

$$V(\underline{\theta}) = V(\underline{\theta}_m) + \tfrac{1}{2}(\underline{\theta}-\underline{\theta}_m)^T A(\underline{\theta}-\underline{\theta}_m) \qquad (4.16)$$
$$= V^*(\underline{\theta}_m) - \underline{\theta}_m^T A\underline{\theta} + \tfrac{1}{2}\underline{\theta}^T A\underline{\theta}$$

with : $A = \dfrac{\partial^2 V(\underline{\theta}_m)}{\partial\underline{\theta}\partial\underline{\theta}^T}$ and $V(\underline{\theta}_m)$ and $V^*(\underline{\theta}_m)$ fixed values dependent on the minimum.

Because A is symmetric and possitive definite A always has N conjugate directions.

Suppose our initial guess of the parameter vector $\underline{\theta}$ is $\underline{\theta}(0)$. The first direction ($\underline{d}(1)$) can be chosen as

$$\left.\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}(0)} = \underline{d}(1) \qquad (4.17)$$

Now the minimum in this direction must be found. One way to do this line minimisation is given by Fletcher and Reeves (1964). Dependent on the given estimate of the minimum, the loss function value and its derivatives, a step size (h) is calculated. In the direction $\underline{d}(1)$ the loss function is calculated for

$$\underline{\theta}(0)-h\underline{d}(1)\ ,\ \underline{\theta}(0)-2h\underline{d}(1)\ ,\ \underline{\theta}(0)-4h\underline{d}(1)\ ,\ \ldots.$$

This sequence is stopped if the loss function does not decrease anymore. Then based on the function values and derivatives of the last 2 referenced parameter values the minimum is approximated between these last 2 referenced points (if the approximated minimum is not less then the last 2 referenced points, a new interval is chosen (see Fletcher and Reeves 1964) and the approximation is done again over this new interval). Because of the presumed quadratic loss function, in the found minimum ($\underline{\theta}(1)$) the gradient must be perpendicular to $\underline{d}(1)$ (see fletcher and Reeves). So the next conjugate direction is conjugated if it contains this derivative

$$\underline{d}(2) = -\left.\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}}\right|_{\underline{\theta}=\underline{\theta}(1)} + \text{const.}\underline{d}(1) \qquad (4.18)$$

The constant is dependent on the size of the present and previous derivative.

If the loss function is quadratic in the parameters we are sure to find the minimum in N steps (dimension of $\underline{\theta}$) because every next search direction is perpendicular to all previous explored directions.

4.3.2 Modified/ Quasi Newton Methods

If the second derivatives can be calculated and if the loss function is quadratic in the parameters, Newton's algorithm is guaranteed to find the minimum in 1 step.
The modified and quasi Newton algorithm use approximations of the second derivatives. In these approximations the structure of the loss function may be used.
These methods take the minimum of a hyperparaboloid through a given point ($\underline{\theta}(0)$) as the next estimate of the minimum ($\underline{\theta}(1)$). This minimum is dependent on the loss function value in $\underline{\theta}(0)$ and its first and second derivative.

Consider the quadratic loss function

$$V(\underline{\theta}) = V(\underline{\theta}_m) + \tfrac{1}{2}(\underline{\theta}-\underline{\theta}_m)^T A(\underline{\theta}-\underline{\theta}_m) \qquad (4.19)$$

with $V(\underline{\theta}_m)$ the minimum of the loss function.
Because A is symmetric and positive definite it can be written as

$$A = S\Lambda S^T \qquad (4.20)$$

with : Λ : Diagonal matrix with eigenvalues.
S : Matrix with orthonormal eigenvectors.

If we transform $\underline{\theta}$ to $\underline{z}$ so that

$$\underline{z} = \Lambda^{\frac{1}{2}} S^T(\underline{\theta}-\underline{\theta}_m) \qquad (4.22)$$

eq.(4.19) becomes:

$$V(\underline{z}) = V(\underline{z}_m) + \tfrac{1}{2}\underline{z}^T\underline{z} \qquad (4.23)$$

with : $\underline{z}_m = \underline{0}$

Because the derivative of $V(\underline{z})$ is given by

$$\frac{\partial V(\underline{z})}{\partial \underline{z}} = \underline{z} \tag{4.24}$$

$\underline{z}_m$ is given by

$$\underline{z}_m = 0 = \underline{z} - \frac{\partial V(\underline{z})}{\partial \underline{z}} \tag{4.25}$$

If the loss function is not quadratic, eq.(4.25) will not be zero. If $\underline{z}(i)$ is the present value of the parameter vector, then $\underline{z}(i+1)$ given by

$$\underline{z}(i+1) = \underline{z}(i) - \frac{\delta V(\underline{z})}{\delta \underline{z}}\bigg|_{\underline{z}=\underline{z}(i)} \tag{4.26}$$

will be a better estimate of $\underline{z}_m$.
The transformation from $\underline{z}$ to $\underline{\theta}$ yields

$$\underline{\theta} = \underline{\theta}_m + S\Lambda^{-\frac{1}{2}}\underline{z}$$

$$\frac{\partial V}{\partial \underline{z}} = \frac{\partial \underline{\theta}^T}{\partial \underline{z}} \frac{\partial V}{\partial \underline{\theta}} \tag{4.27}$$

With help of

$$\frac{\partial \underline{\theta}^T}{\partial \underline{z}} = \Lambda^{-\frac{1}{2}} S^T \tag{4.28}$$

this can be written as

$$\underline{\theta}(i+1) = \underline{\theta}(i) - S\Lambda^{-1}S^T \left[\frac{\partial V(\underline{\theta})}{\partial \underline{\theta}^T} \right]^T \Bigg|_{\underline{\theta}=\underline{\theta}(i)}$$

or

$$\underline{\theta}(i+1) = \underline{\theta}(i) - A^{-1} \frac{\partial V(\underline{\theta})}{\partial \underline{\theta}}\bigg|_{\underline{\theta}=\underline{\theta}(i)}$$

So the minimum of a paraboloid, dependent on the first and second derivative of the loss function in $\underline{\theta}(i)$, is taken as the next estimate of the parameter vector $\underline{\theta}$ that minimises the loss function. And in case of a quadratic loss function this estimate is the loss function.

4.4 Signal Processing

If the data for the estimation contains non-zero mean values, correction of this data with these mean values can avoid numerical problems in the estimation algorithm. Also scaling of the I/O-data can avoid numerical problems.
Because the offsets or zero levels of the signals and the initial state of the process are changed by this preprocessing, first the offset and initial state will be described in detail. Next the influence of mean value correction on the offset/zero level and the initial state will be dealt with.
Finally the influence of data scaling on the estimated parameters is examined.

4.4.1 Offset and Zero Levels.

Because the practical implication of a change in chosen zero-level (static balance) or a change in offset is nihil, they are treated together.
Suppose that the data set under consideration can be described exactly by the n-th order stable (also no poles in 1) model (4.1).

$$\begin{aligned} \underline{x}''(k+1) &= A\underline{x}''(k) + B\underline{u}''(k) \\ \underline{y}''(k) &= C\underline{x}''(k) + D\underline{u}''(k) \end{aligned} \qquad (*)\ (4.30)$$

(*: Symbols with an apostrophe will be corrected with a constant later on.)
However if we change the zero level (static balance) or introduce an offset on an output the simulated data will have a misfit that is constant in time. The n-th order model extended with a q-dimensional offset vector will be able to fit the I/O-data exactly. So the extended state space model is

$$\begin{aligned} \underline{x}''(k+1) &= A\underline{x}''(k) + B\underline{u}''(k) \\ \underline{y}'(k) &= C\underline{x}''(k) + D\underline{u}''(k) + \underline{y}''_{off} \end{aligned} \qquad (4.31)$$

with: $\underline{y}'(k) = \underline{y}''(k) + \underline{y}''_{off}$

If we change the zero-level of an input or introduce an offset on an input the simulated data will also have a misfit that is constant in time.
The process can now be described by

$$\underline{x}'(k+1) = A\underline{x}'(k) + B\underline{u}'(k) \qquad (4.32)$$
$$\underline{y}'(k) = C\underline{x}'(k) + D\underline{u}'(k) + \underline{y}''_{off}$$

with $\underline{u}'(k) = \underline{u}''(k) + \underline{u}_{off}$

$$\underline{y}''_{off} = \underline{y}'''_{off} - C\left[(I-A)^{-1}B + D\right]\underline{u}_{off}$$

Because the zero-levels of all variables are arbitrarily chosen, we always have to consider/model this static part of the process. All zero-level shifts and all introduced offsets can be modeled by (4.32).
Notice that a higher order model without offset vector would also be able to fit the data of the process. Only this higher order model uses more extra parameters, and would probably not fit to an other data set of the same process.

4.4.2 Initial State of the Process

The state performance of the process at the first sample moment (k=0) due to time moments previous to the available data, that has an influence on the process behaviour at future sample moments is referred to as the initial state of the process.
For state space descriptions of a process these conditions are described uniquely by the state vector at k=0, the initial state of the process/model. Depending on our knowledge of this initial state five situations can be distinguished.

1. The process is at rest for time moments previous to the available data (k<0) and the value of the constant in- and outputs at those moments based on the arbitrary chosen zero levels are known.
2. The same as 1. except that only the values of the constant inputs are known.

3. The same as 1. except that only the values of the constant outputs are known.
4. The same as 1. except that the constant values of in- and outputs are unknown.
5. Nothing about the process is known at time moments previous to the available data.

Situations 1,4 and 5 mostly occur in practical situations. Cases 2 and 3 are also mentioned because they show the effect of signal processing very clear.

Situation 1:

Most data sets are based on zero levels chosen such that in- and output signals are zero for time moments previous to the available data.
For preprocessing purposes we have to treat the general case. So for $k<0$

$$\underline{x}'^{0} = A\underline{x}'^{0} + B\underline{u}'^{0} \qquad (*) \quad (4.33)$$
$$\underline{y}'^{0} = C\underline{x}'^{0} + D\underline{u}'^{0}$$

The process is at rest so the initial state $\underline{x}'(0)$ is equal to $\underline{x}'^{0}$.
(*: Notice that the initial state only contains q independent variables due to the constraint of a process at rest. Because of the pseudo-canonical form those q independent variables can be distinghised very clear.)
The real measurable outputs can be written as

$$\underline{y}'^{00} = \underline{y}'^{0} + \underline{y}''_{off} \qquad (*) \quad (4.34)$$

Because $\underline{u}'^{0}$ and $\underline{y}'^{00}$ are known, for given parameter values we also know $\underline{x}'^{0}$ and $\underline{y}''_{off}$.

$$\underline{x}'(0) = (I-A)^{-1}B\underline{u}'^{0}$$
$$\underline{y}''_{off} = y'^{00} - C\underline{x}'^{0} - D\underline{u}'^{0} \qquad (4.35)$$

Consequently

$$S = \{A,B,C,D\} \tag{4.37}$$

is the parameter set that has to be estimated

(*): We presume the offset to be equal for all time moments k. If not, we are dealing with situation 2.

Situation 2 :

Also eq.(4.33) and (4.34) hold. Only $\underline{y}'^{00}$ is unknown.

$$\underline{x}'^{0} = (I-A)^{-1}B\underline{u}'^{0} \tag{4.35}$$

The parameter set that has to be estimated is

$$S = \{A,\ B,\ C,\ D,\ \underline{y}_{off}\} \tag{4.38}$$

Situation 3 :

Here also eq.(4.33) and (4.34) hold but we do not know the initial state.

$$\underline{y}''_{off} = y^{00} - C\underline{x}'^{0} - D\underline{u}'^{0} \tag{4.36}$$

The parameter set is

$$S = \{A,\ B,\ C,\ D,\ \underline{x}'^{0}\} \tag{4.39}$$

Only if (4.35) can be solved or if the model is strictly proper (D=0),

$$(I-A)^{-1}\underline{x}'^{0} = B\underline{u}'^{0} \tag{4.35}$$

we can calculate the offset with the estimated state vector

Situation 4 :

Again the same relations hold for k<0 but we do not know $\underline{u}'^{0}$ and $\underline{y}'^{00}$. The parameter set that has to be estimated is

$$S = \{A,\ B,\ C,\ D,\ \underline{y}_{off},\ \underline{x}'^{0}\} \tag{4.39-a}$$

Situation 5 :

This situation is different compared to the previous situations but the estimation is the same as situation 4, except that n parameters of $\underline{x}'(0)$ have to be estimated. The estimated $\underline{x}'(0)$ represents all conditions of the process at k=0. Also the offset has to be estimated because of the arbitrary chosen zero levels of input and output variables.

Now that we know which parameters have to be estimated and which already have a parameter dependent value defined by the problem at hand, we want to know the changes of these parameter values if we correct the signals with their means.

4.4.3 Mean Value Correction

As mentioned before the correction of input and output signals with their mean values has some great advantages. We will again consider the process whose dynamic and static part can be described by eq.(4.32).
For estimation purposes we have to split up the processing into two parts. The input and output mean value correction.

Output mean value correction

If we introduce a mean value and a corrected output vector, the mean value correction can be written as

$$\underline{y}(k) = \underline{y}'(k) - \overline{\underline{y}} \qquad (\text{for } k=-\infty, \ldots, \infty) \tag{4.40}$$

Now the process can be described by

$$\begin{aligned} \underline{x}'(k+1) &= A\underline{x}'(k) + B\underline{u}'(k) \\ \underline{y}(k) &= C\underline{x}'(k) + D\underline{u}'(k) + \underline{y}'_{off} \end{aligned} \tag{4.41}$$

with: $$\underline{y}'_{off} = \underline{y}''_{off} - \overline{\underline{y}} \qquad (\text{for } k=-\infty, \ldots, \infty) \tag{4.42}$$

So even if we do consider the initial condition of the process the correction of the output data has no influence on the estimation scheme. Only after estimation of the model (4.41) the true offset can be found by eq.(4.42).

Input Mean Value Correction

If we write the mean value correction as we did before for the output mean value correction, we get

$$\underline{u}(k) = \underline{u}'(k) - \overline{\underline{u}} \qquad (\text{for } k=-\infty, \ldots, \infty) \tag{4.43}$$

Because state space models are linear the next model will fit to the

input corrected data

$$\boxed{\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{y}_{off} \end{aligned}} \qquad (4.44)$$

with: $\underline{x}(k) = \underline{x}'(k) - \bar{\underline{x}}$ (for $k=-\infty, \ldots, \infty$) (4.45)

$$\bar{\underline{x}} = (I-A)^{-1}B\bar{\underline{u}} \qquad (4.46)$$

$$\underline{y}_{off} = \underline{y}'_{off} + C\bar{\underline{x}} + D\bar{\underline{u}} \qquad \text{(for } k=-\infty, \ldots, \infty) \qquad (4.47)$$

Because the mean value correction is also valid for $k<0$ the initial state will be changed by this correction.

$$\underline{x}(0) = \underline{x}'(0) - \bar{\underline{x}} \qquad (4.48)$$

So after mean value correction and estimation of the model we have to correct the initial state and the offset vector by eq.(4.48), (4.42) and (4.47).

4.4.4 Scaling

Parameters in our model set that are related to relatively small in- or output variables might cause numerical anomalies.
If we multiply every in- and output data sequence by a separate factor such that their powers are equal, the estimation is not influenced anymore by weak or strong signals. By this multiplication we also change the covariance matrix of the noise. Care should be taken about this if we estimate with a least of squares estimator. Of course after signal scaling the estimated parameters are not the true parameters.
To see what happens if we scale in- and output variables we introduce scaling factors and matrices

su_i : Scaling factor for input number i.
sy_i : Scaling factor for output number i

$$
\begin{aligned}
S_u &= \text{diag}\,(su_1, \ldots, su_p) \\
S_y &= \text{diag}\,(sy_1, \ldots, sy_q) \\
S_x &= \text{diag}\,(\underbrace{sy_1, \ldots, sy_1}_{\mu_1}, \underbrace{sy_2, \ldots\ldots, sy_q}_{n})
\end{aligned}
\tag{4.49}
$$

Suppose that before scaling a n-th order process can be described exactly by

$$
\begin{aligned}
z\underline{x}'(k) &= A\underline{x}'(k) + B\underline{u}'(k) \\
\underline{y}'(k) &= C\underline{x}'(k) + D\underline{u}'(k) + \underline{y}_{off} \\
\underline{x}(0) &= \underline{x}_0
\end{aligned}
\tag{4.50}
$$

The parameter set of this model is $\{A, B, C, D, \underline{y}_{off}, \underline{x}_0\}$
If we now scale the input like

$$\underline{u}(k) = S_u\underline{u}'(k) \tag{4.51}$$

The scaled process can be described by

$$\begin{aligned} \underline{x}'(k+1) &= A\underline{x}'(k) + BS_u^{-1}\underline{u}(k) \\ \underline{y}'(k) &= C\underline{x}'(k) + DS_u^{-1}\underline{u}(k) + \underline{y}_{off} \\ \underline{x}(0) &= \underline{x}_0 \end{aligned} \tag{4.52}$$

The parameter set of this model is $\{A,\ BS_u^{-1},\ C,\ DS_u^{-1},\ \underline{y}_{off},\ \underline{x}_0\}$
If we now scale the output variables like

$$\underline{y}(k) = S_y\underline{y}'(k) \tag{4.53}$$

The scaled process can be described by

$$\begin{aligned} \underline{x}'(k+1) &= A\underline{x}'(k) + BS_u^{-1}\underline{u}(k) \\ \underline{y}(k) &= S_yC\underline{x}'(k) + S_yDS_u^{-1}\underline{u}(k) + S_y\underline{y}_{off} \\ \underline{x}(0) &= \underline{x}_0 \end{aligned} \tag{4.54}$$

Because S_yC has not the pseudo-canonical form of an output matrix the parameter set $\{A,\ BS_u^{-1},\ S_yC,\ S_yDS_u^{-1},\ S_y\underline{y}_{off},\ \underline{x}_0\}$ does not represent the unique form we want to estimate.
If we transfer the state vector like

$$\underline{x}(k) = S_x\,\underline{x}'(k) \tag{4.55}$$

and multiply the system equation by S_x we get the wanted pseudo-canonical form

$$\begin{aligned} \underline{x}(k+1) &= S_xAS_x^{-1}\underline{x}(k) + S_xBS_u^{-1}\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + S_yDS_u^{-1}\underline{u}(k) + S_y\underline{y}_{off} \\ \underline{x}(0) &= S_x^{-1}\underline{x}_0 \end{aligned} \tag{4.56}$$

So after scaling of the data the parameters that fit the scaled data are

$$\boxed{\{\ S_x^{-1}AS_x\ ,\ S_x^{-1}BS_u^{-1}\ ,\ S_yDS_u^{-1}\ ,\ S_y\underline{y}_{off}\ ,\ S_x^{-1}\underline{x}_0\ \}} \tag{4.57}$$

Notice that $S_x^{-1}AS_x$ has a pseudo-canonical structure and $S_yCS_x^{-1}$ equals C.

This parameter set could also be found by considering the equivalent MFD

$$P(z)\underline{y}'(k) = Q(z)\underline{u}'(k) + P(z)\underline{y}_{off} \tag{4.58}$$

The initial state condition is translated to values of $\underline{y}'(k)$ and $\underline{u}'(k)$ for $k<0$. After scaling (4.58) becomes

$$P(z)S_y^{-1}\underline{y}(k) = Q(z)S_u^{-1}\underline{u}(k) + P(z)S_y^{-1}\underline{y}_{off}$$

and the caused change in α- and β-parameters can be transferred to an equivalent change in α-,b- and d-parameters through

$$\overline{B}(\beta) = M(\alpha)B(b.d) \tag{2.15}$$

The original parameter set that had to be estimated (4.50) can easily be found out of the estimated parameter set (4.57).

4.5 Implemented Estimation Algorithm

The implemented estimation algorithm estimates the (pseudo-)canonical state space parameters, as given in eq.(1.9), according to the LS-criterion.
Because all the stochastic models treated in chapter 3 have errors which are not linear in the parameters, the algorithm that tries to minimise the LS loss function uses hill-climbing methods to find the minimum.
To prevent the possibility of arriving at a local minimum, one should have a good (close to the global minimum) guess of the parameter values, and an appropriate hill-climbing method.
Because the equation error in a MFD is a linear function of the MFD-parameters, a LS estimated EEM in MFD, transformed to a SSM, will give a fast estimated EEM in state space representation (see Carierre 1984).
Because of introduced errors in the transformation in the pseudo-canonical case (in the canonical case, or if the number of parameters in MFD is equal to the number of parameters in SSM, only numerical errors will be made) a reduction of the loss function in the SSM might be possible to get a better EEM. Dependent on the system dynamics this EEM might be a good start for the estimation of a PEM, OEM and IEM.
Because of the difficulties that may arrise during estimation (see previous sections), several options are implemented. Also options that are not necessary for practical use but that do allow extra testing to get a better feeling for the estimation/minimisation algorithm.

Options

- The initial guess of the parameters can be read from a file or given inter-actively.

- The I/O-data can be read from a file or can be given by the user interactively (also standard functions can be chosen (sin,cos,step, impulse,..))

- An EEM, PEM, OEM or IEM can be executed (default: no estimation)

- The offset can be estimated, set to a fixed value (default=0) or set to a parameter dependent value (system at rest and known in- and output values for k<0).

- The initial state can be estimated fully, estimated with the assumption of a system at rest for k<0, set to a fixed point (default=0) or set to a parameter dependent value (system at rest and known in- and output values for k<0).

- To overcome difficulties if the initial state is not estimated, the first sample moment for the calculation of the errors and the first sample moment that contributes to the loss function can be chosen separately.
  (default: both moments are equal)

- It is possible to choose between the estimation of a canonical or pseudo-canonical form (default). Estimation of a PEM implies canonical estimation.
- A proper (default) or strictly proper (D=0) model can be estimated.

- If wanted the input and/or output data can be correted with its mean value (default: no correction).

- The possible signal scalings are:
  Output scaling such that all outputs have equal power.
  In- and output scaling such that the first calculated value of the loss function has a specified value. (default: no scaling)

- Possible hill-climbing methods are the conjugate gradient (default), modified Newton and quasi Newton method.

- A stop criterion is built in that stops the program if the percentual decrease of the loss function is less then a specific value (default: 1.0E-5). This value can be changed inter-actively.

- To prevent the program from stopping by an overflow, a maximum value for the (scaled) error function (not loss function) is implemented (default: 1.0E+15). This value can be changed inter-actively.

- The maximum number of loss function calculations can be set inter-actively (default: 100x(number of parameters))

- After estimation one can decide to continue with a different stop criterion and/or overflow value and/or number of function calculations.

- To show the estimation results the (initial and/or estimated) model can be stored, a plotable file with all estimation conditions and results can be made, and various files suitable for a plotprogram (GRA, see Ploemen 1982) can be made, i.e. impulse-, step- and initial state-response, simulated data, predicted data, Innovated data (only for IEM), I/O-data, output-, equation-, prediction- and innovation error. All possibilities for the initial and/or estimated model.

For further details see program manual.

5 Results

5.0 Introduction

To show most of the derived relations between the introduced errors and to show the implication of different structures on the estimation, estimations are executed with simulated data. The data is generated with a test model and white zero mean gaussian (output) noise is added (section 5.2).
To show the practical applicability of the program and to show some implemented options, practical data is used for several estimation runs. Because this real data is obtained from a SISO system, the results show some model choice implications very clear (only canonical form and no complex interaction between the variables). Because of this, these runs for this SISO system are shown first (sect.5.1).

Before the estimation results are given, first some conceptions have to be stated (clarified) to prevent misunderstandings:

- The Output Error Model (OEM) is the result of a minimisation of the output error loss function (OE-LF). Of course this model may be used to predict the next output, using all available data at the present time moment, resulting in the prediction with this OEM. Consequently it is possible to talk about the prediction error loss function (PE-LF) of the OEM.
A Prediction Error Model (PEM) is the result of the minimisation of the prediction error loss function. We may also use this PEM for simulations. The error made by the simulation is called the output error of the PEM. And also we have the output error loss function of the PEM.
An Equation Error Model (EEM) is the result of the minimisation of the equation error loss function (EE-LF). Similar as for previously mentioned models we have a prediction error and output error loss function of the EEM, and an equation error loss function of the OEM and PEM.

The innovation Error Model (IEM) is obtained by minimisation of the innovation error loss function (IE-LF) with respect to the α-, b-, d- and k-parameters. If we do not consider the estimated k-parameters, we may consider a output-, equation- and prediction error loss function of the IEM.

- The impulse response is a good measure for the simulation and dynamic behaviour (output error) of models. For MIMO-models we have qxp impulse responses. If the impulse responses of two models are equal, their simulation performances will be equal. If the impulse responses of an estimated model are equal to those of the test system, then no cross-correlation is necessary anymore for simulation comparison. Numerical comparison of simulation performances may show small but clear differences that can not be seen from plots of the impulse responses.

- A good numerical measure for the simulation performances of a model is the recontruction error (RE) of one output over a given time interval

$$RE(i) = \frac{\sum_{k=1}^{N} e_{oi}^2(k)}{\sum_{k=1}^{N} y_i^2(k)} \qquad (i=1,\ldots,q) \tag{5.1}$$

with: $e_{oi}(k)$ the output error i at sample moment k.

If we assume that the errors are uncorrelated to the outputs and the model simulation error is almost equal to the real (system) output noise, it is easy to see that

$$\frac{1}{RE(i)} = \left(S/N \right)_i + 1 \tag{5.2}$$

with: $\left(S/N\right)_i$ the Signal-to-Noise ratio on output i.

So for processes with 0dB white output noise the best reconstruction error (for an infinite number of samples) we can get is ½.

All given loss functions are normalised (similar as the energy in the model errors) such that the value given is the total loss function devided by the number of samples used for the estimation.

5.1 Retina-Reflex

The Retina-reflex is some response of the human eye on a light impulse. Because we are only interested in the measured I/O-data as such from which a model has to be estimated, we will not discuss how the data is obtained.
The input is an impuls on sample moment k=0. The response is given in fig.5.1.1 (in all the other plots this response is drawn as a broken line). The first sample moment used for all estimations is 0, and the number of samples used for the estimation is 89.

Because the input-data does not contain enough information to solve the LS-function for an EEM in MFD directly (see v.d.Boom 1982), all models are estimated with the implemented SSM estimation algorithm.
OEM's and IEM's are estimated Newton methods because the conjugate gradient method failed (OEM of fig.5.4.4/5 estimated with Modified Newton other OEM's and IEM's with Quasi Newton. Afterwards it appeared that the OE-LF of the OEM could be decreased a little by using the Quasi Newton method).

For the OEM's (fig.5.1.4, 5.1.5, 5.1.7 and 5.1.8) a randomly chosen stable system is taken as the initial guess of the parameter values.
For the EEM (fig.5.1.2 and 5.1.3) the estimated OEM (5-th order) was taken as the initial guess of the parameter values, just as was done for the OEM with initial state estimation (fig.5.1.6-a/b) and the OEM with offset and initial state estimation (fig.5.1.6-c/d) and the IEM of fig.5.1.9, 5.1.10 and 5.1.11.
For the IEM of fig.5.1.12, 5.1.13 and 5.1.14 the 5-th order EEM (with $K(\alpha)$) was taken as initial parameter guess.

From the figures 5.1.1 till 5.1.14 we mention the next obvious distinctions:

- Simulation with the EEM (=PEM) is not good (see fig.5.1.2).

- The prediction with the OEM is also not good (see fig.5.1.5). Notice the pikes at about k=0-5, 30-35 and 40-45, and compare them with the prediction of the 3-th order OEM with pikes at about k=0-3, 30-33 and 40-43 (see fig.5.1.7). Also mention the bends of the output data at those places (see fig.5.1.1).
- The number of pikes can be explained by the difference of use of estimated and real outputs (back to sample moment k-n) between the prediction and output error.
- Prediction of the 3-th order OEM is substantially better then the prediction of the 5-th order OEM.
- The simulation with the OEM, if the offset and initial state are estimated also, is very good, The practical implication of such a model is not considered (impuls responses are defined without initial state and offset) (see fig.5.1.6-c).
  The initial state response (u(0)=0) is very large, although the dynamic behaviour is 'similar' to that of the OEM without offset and initial state (see fig.5.1.6-d).
- The simulation with the 3-th order OEM looks like the simulation of the 5-th order EEM. Also the prediction does, except for 3 pikes at k=0-3 (see fig.5.1.3 and 5.1.8).
- Good simulation results with the IEM if the OEM is taken as initial guess for the parameters (see fig.5.1.9). Compare the prediction with this IEM with the prediction of the OEM. Pikes at k=0-5 have disappeared, but the rest of the prediction is exactly the same (see fig.5.1.10).
  The innovation, the prediction of the next output with use of the estimated k-parameters, is very good (see fig.5.1.11).
- Good prediction (K=K(α)) with IEM, if EEM is taken as initial guess of the parameters (also practical not of any interest). Poor simulation. Also good innovation (see fig.5.1.12/13/14).
  Compared to the innovation with the previous IEM (OEM as initial guess), errors have disappeared in the first samples but occur at about k=30 and k=40 (same as prediction with PEM).

The numerical estimation results are given in table 5.1.15 and 5.1.16.
(continue at page 95)

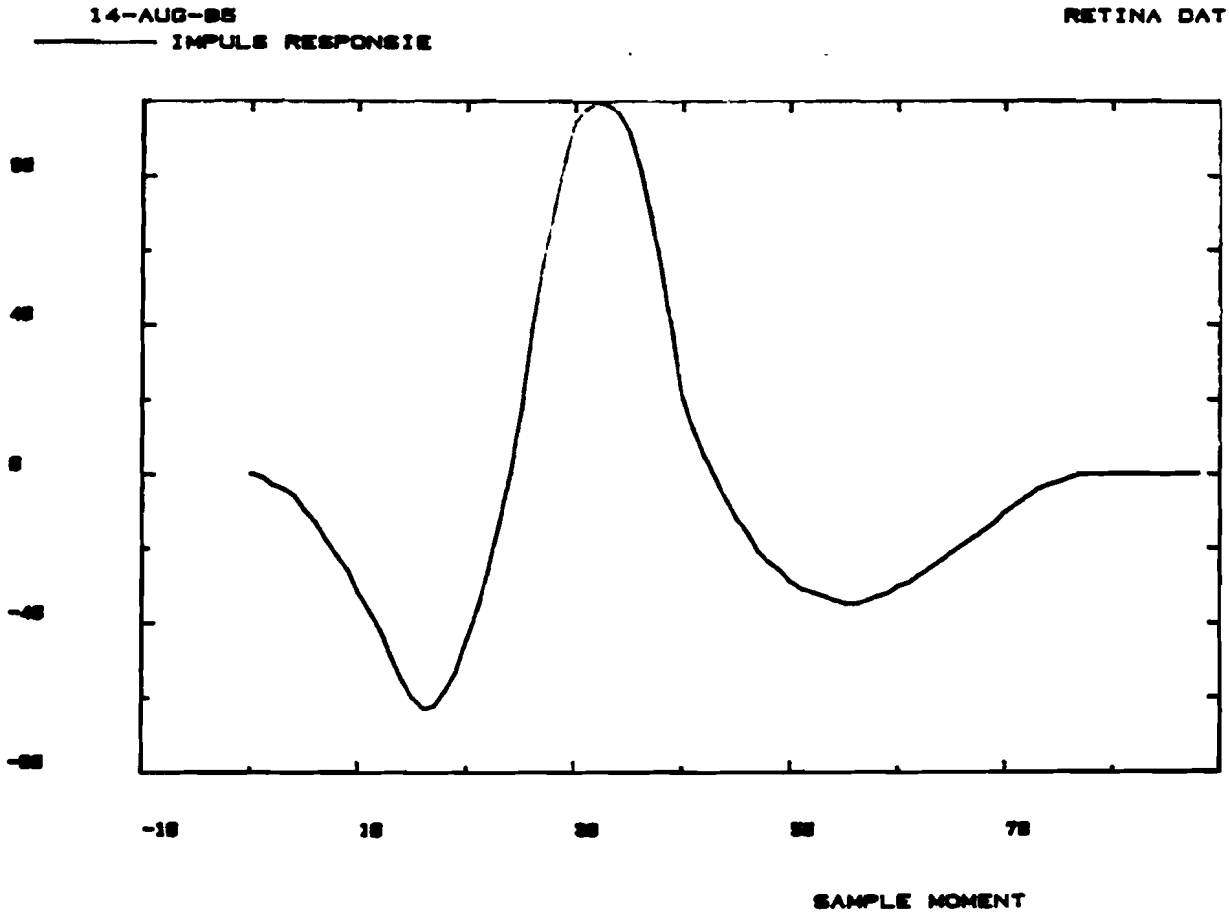


Fig. 5.1.1 Retina data (imp.resp.) used for estimation.

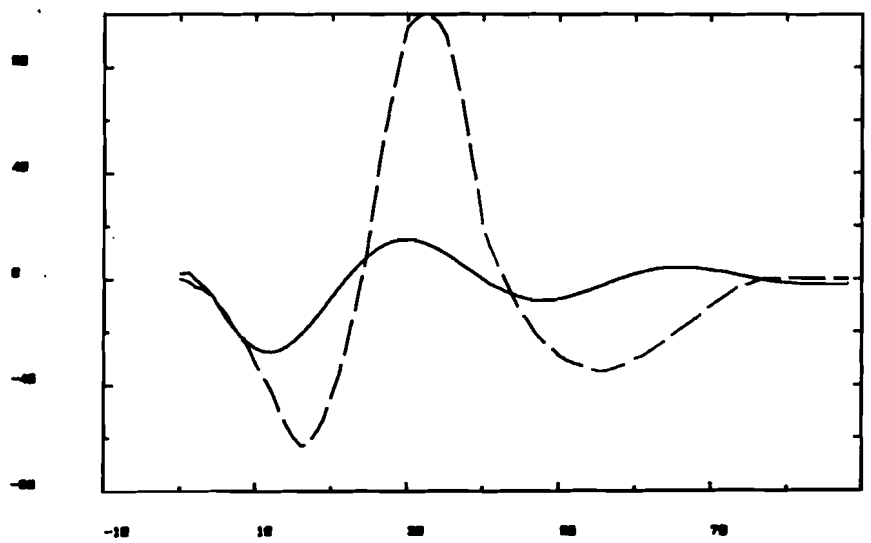


Fig. 5.1.2 Simulation of 5-th order EEM (=PEM).

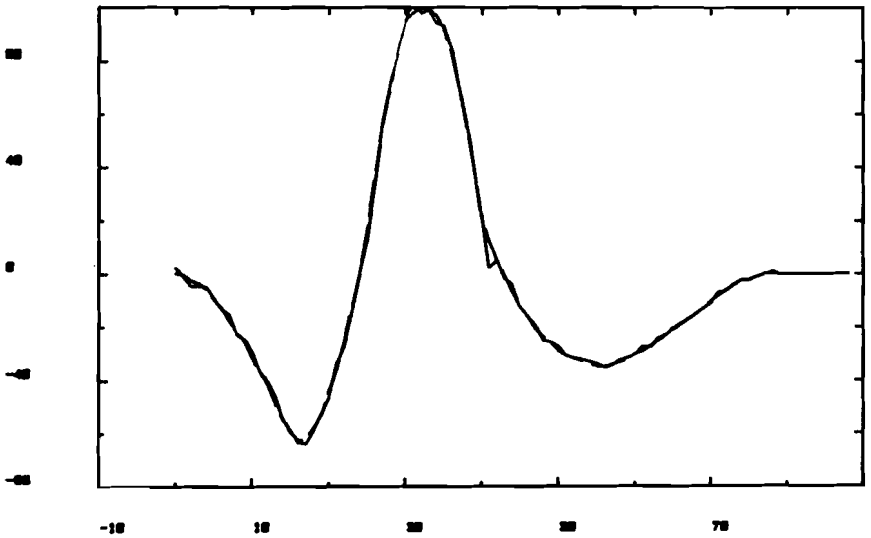


Fig. 5.1.3 Prediction of 5-th order EEM (=PEM).

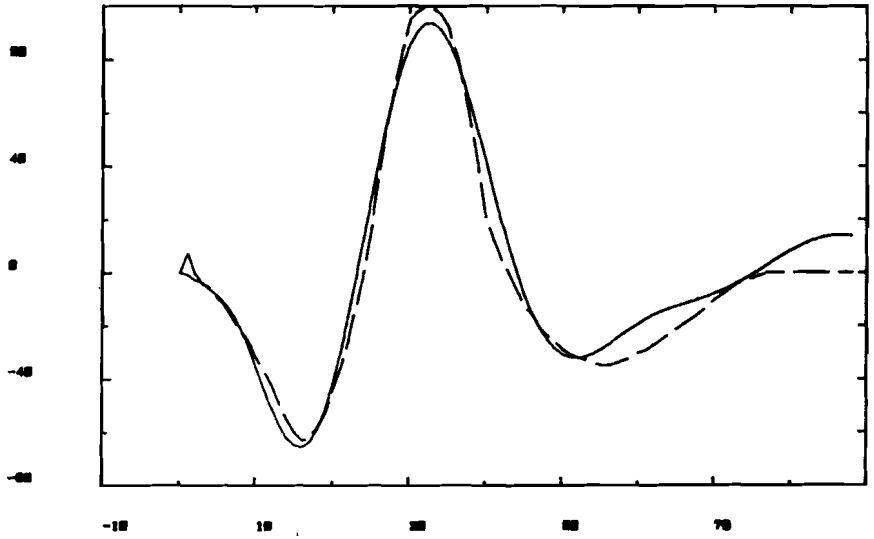


Fig. 5.1.4 Simulation with 5-th order OEM

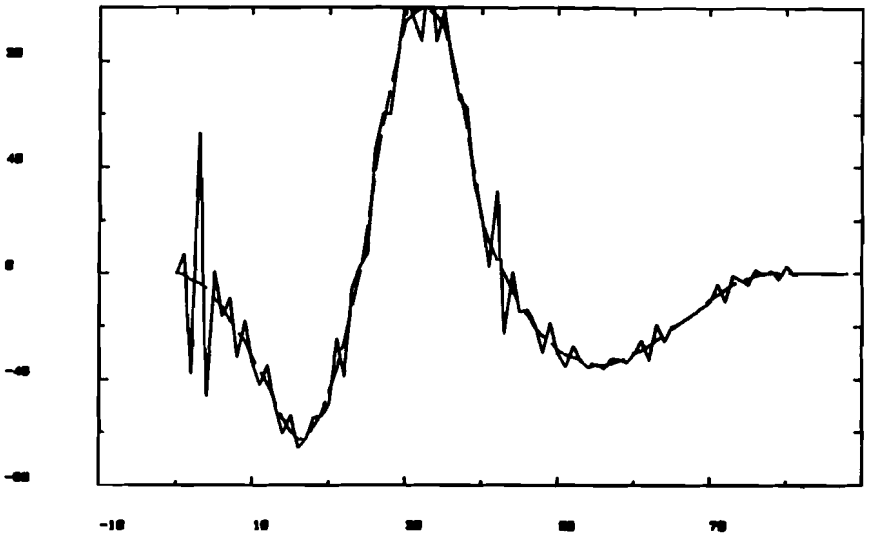


Fig. 5.1.5 Prediction of 5-th order OEM.

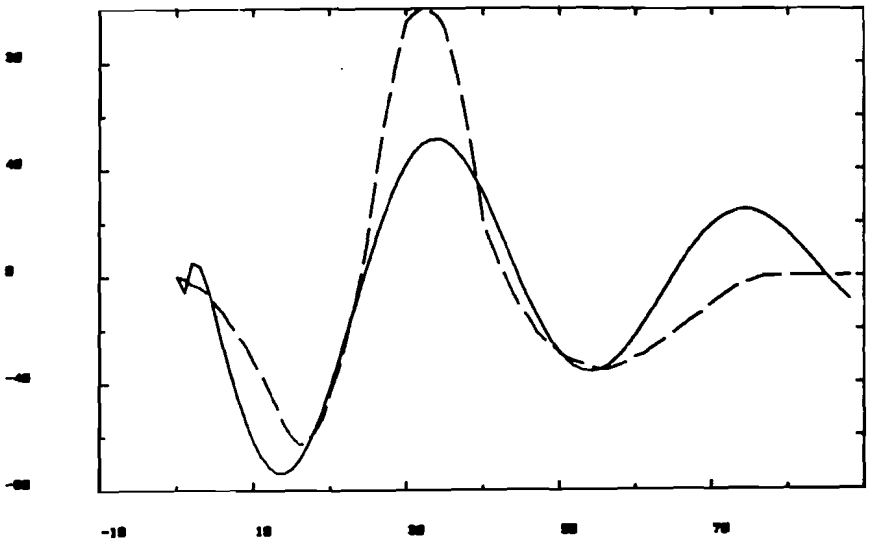


Fig. 5.1.7 Simulation with 3-th order OEM-

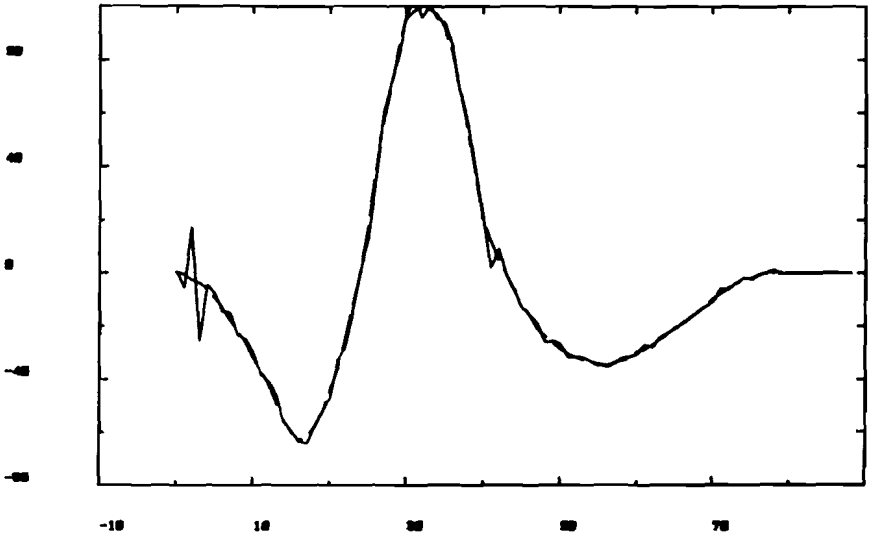


Fig. 5.1.8 Prediction of 3-th order OEM.

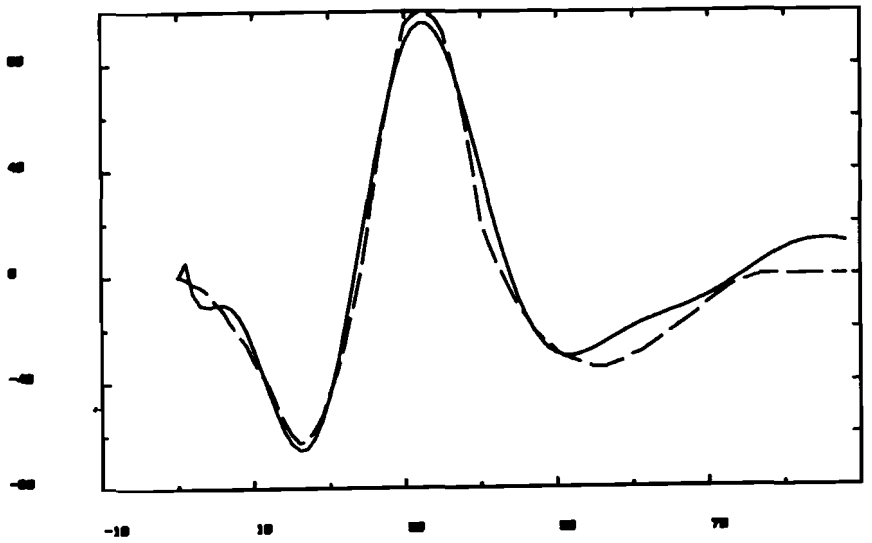


Fig. 5.1.6-a Simulation of 5-th order OEM with initial stated estimated.

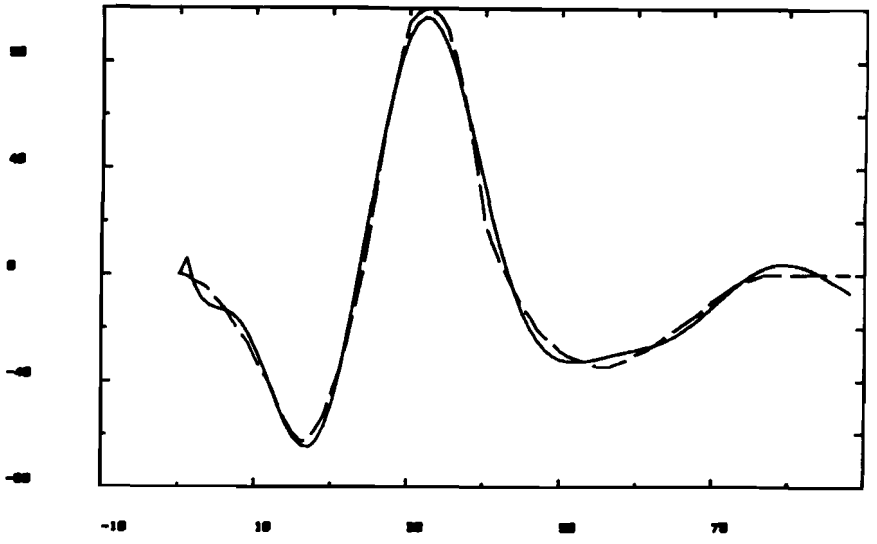


Fig. 5.1.6c Simulation of 5-th order OEM with offset and initial state estimated.

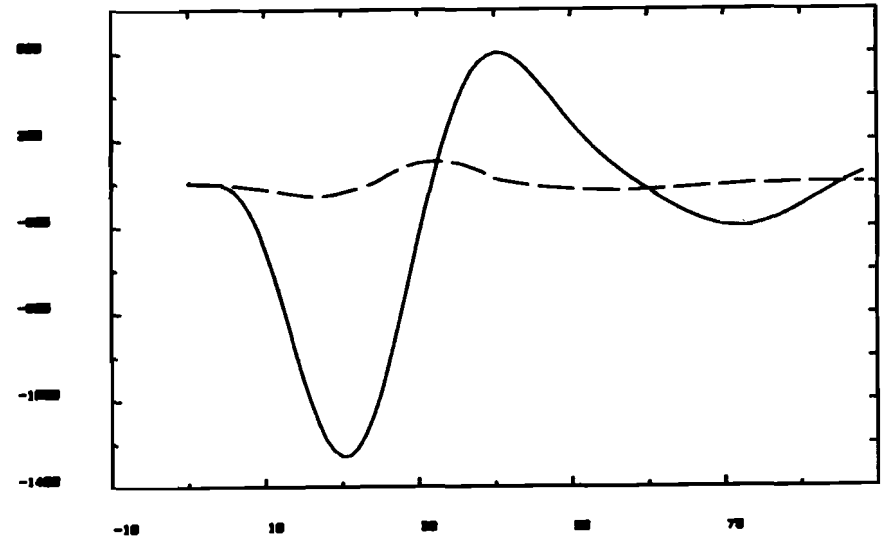


Fig. 5.1.6-b Initial state response of 5-th order OEM with initial state estimated.

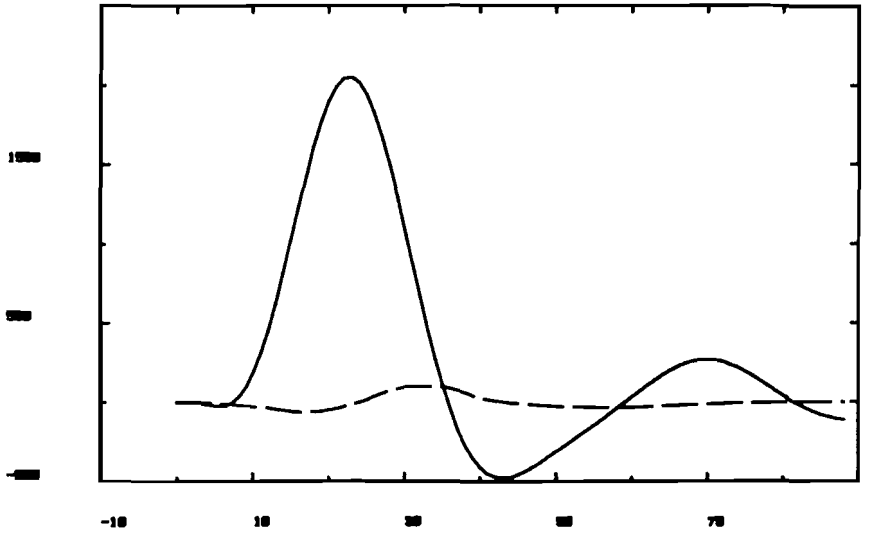


Fig. 5.1.6-d Initial state response of 5-th order OEM estimated with offset and initial state.

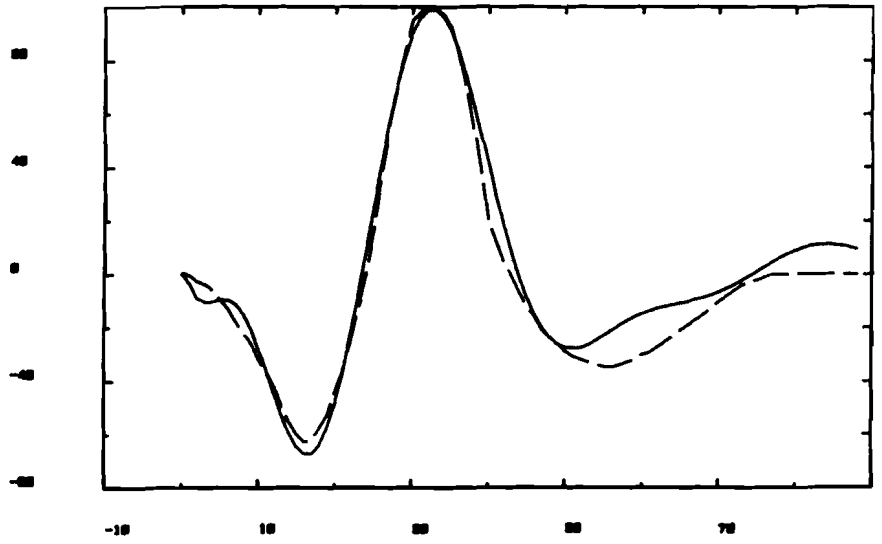


fig. 5.1.9 Simulation of 5-th order IEM (K=0).
(Initial parameter values of OEM).

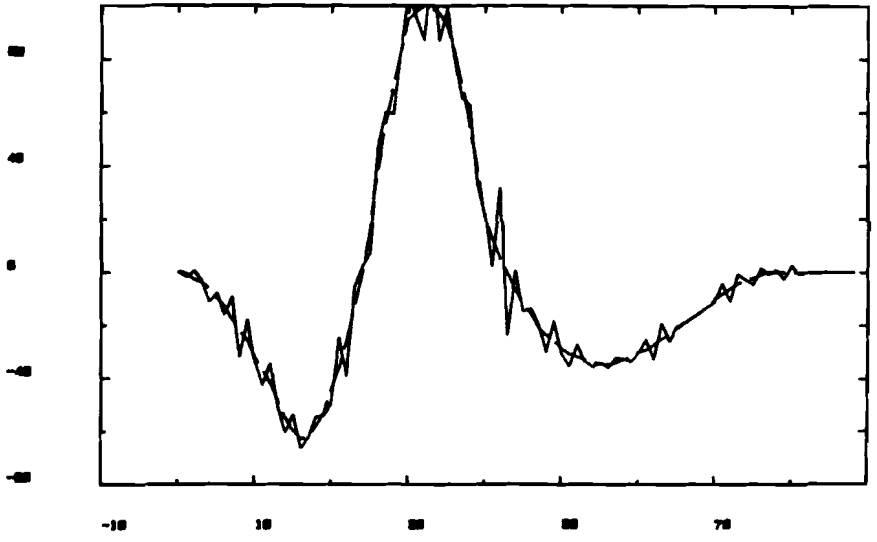


Fig. 5.1.10 Prediction of 5-th order IEM (K=0).
(Initial parameter values of OEM).

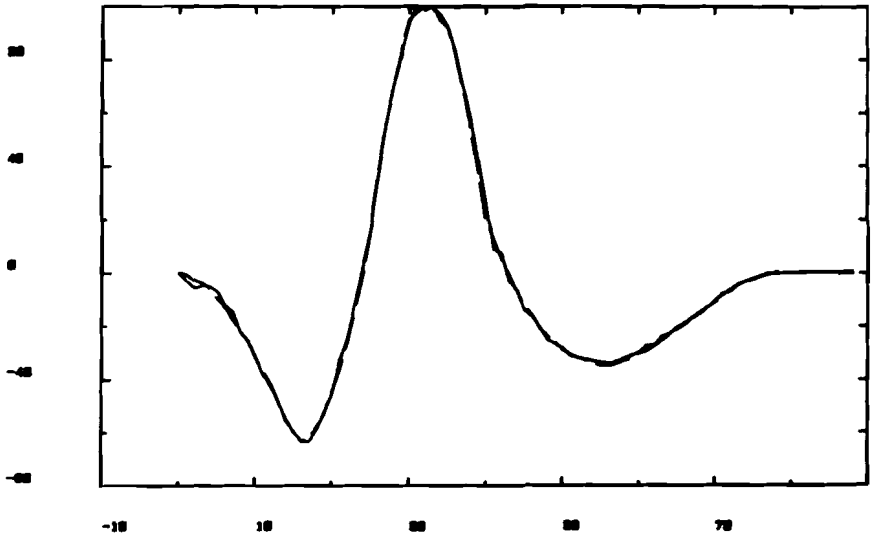


Fig. 5.1.11 Innovation of 5-th order IEM.
(Initial parameter values of OEM).

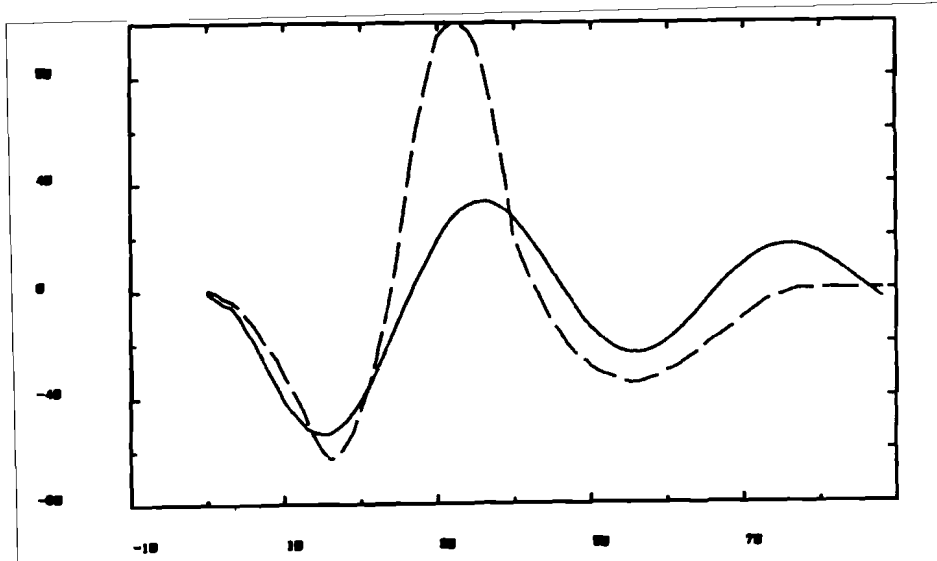


fig. 5.1.12 Simulation of 5-th order IEM (K=0).
(Initial parameter values of EEM (K=0))

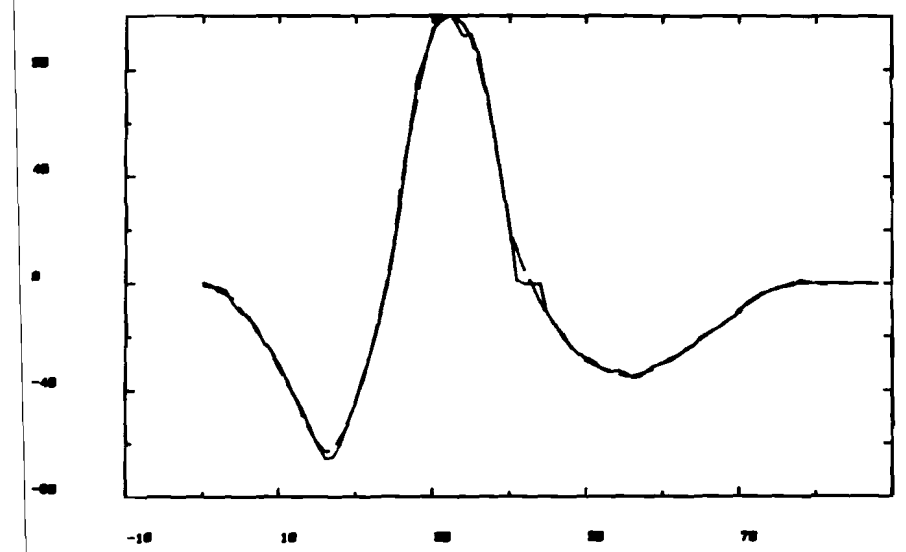


Fig. 5.1.13 Prediction of 5-th order IEM (K=0).
(Initial parameter values of EEM (K=0))

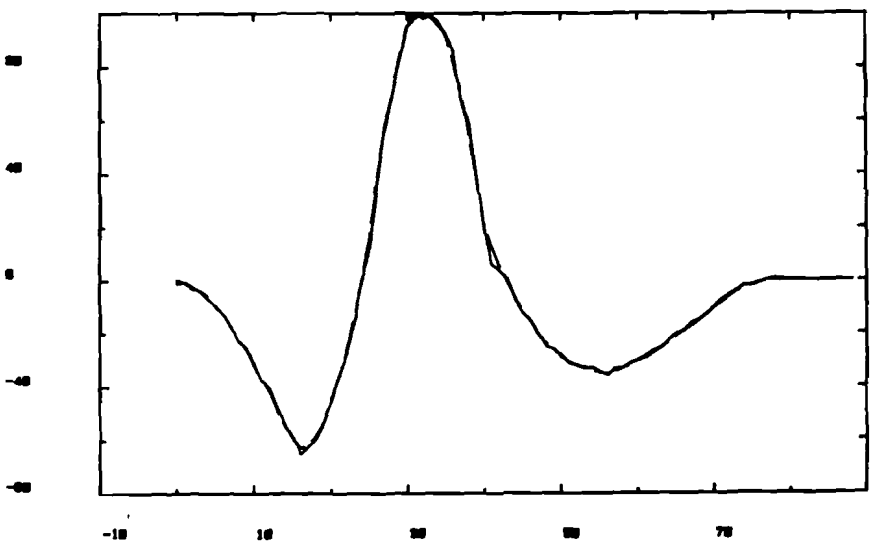


Fig. 5.1.14 Innovation of 5-th order IEM.
(Initial parameter values of EEM (K=0)).

Parameters		Model						
		OEM n=5	EEM n=5	OEM n=5 x(0)	OEM n=5 x(0), Off.	OEM order 3	IEM order 5 (OEM)	IEM order 5 (EEM)
A	α_1	6.839E-01	2.210E-01	6.411E-01	7.845E-01	5.292E-01	7.277E-01	6.575E-01
	α_2	-3.654E+00	-4.761E-01	-3.474E+00	-4.047E+00	-2.025E+00	-3.825E+00	-1.094E+00
	α_3	7.843E+00	4.204E-01	7.556E+00	8.423E+00	2.484E+00	8.095E+00	-1.028E-01
	α_4	-8.457E+00	-1.230E+00	-8.252E+00	-8.841E+00	--	-8.625E+00	-7.597E-02
	α_5	4.585E+00	2.044E+00	4.529E+00	4.681E+00	--	4.627E+00	1.605E+00
D	d	0.000E+00	2.038E+00	-4.993E+00	6.286E+00	0.000E+00	-4.026E-01	-1.093E+00
B	b_1	7.200E+00	2.511E+00	1.021E+01	8.109E+00	-7.783E+00	-3.815E+00	-3.233E+00
	b_2	-2.893E-01	-1.910E-01	1.046E+00	-1.639E+00	5.723E+00	-9.091E+00	-5.215E+00
	b_3	-4.318E+00	-2.661E+00	-3.662E+00	-6.949E-01	6.031E+00	-1.054E+01	-6.106E+00
	b_4	-6.786E+00	-5.550E+00	-2.185E+00	6.964E+00	--	-1.014E+01	-9.738E+00
	b_5	-9.060E+00	-1.024E+01	8.228E+00	1.625E+01	--	-9.452E+00	-1.498E+01
K	k_1	--	--	--	--	--	1.379E+00	2.395E+00
	k_2	--	--	--	--	--	9.653E-01	3.608E+00
	k_3	--	--	--	--	--	6.218E-01	4.549E+00
	k_4	--	--	--	--	--	3.453E-01	5.670E+00
	k_5	--	--	--	--	--	1.333E-01	6.168E+00
$\underline{x}(0)$	x_1	--	--	4.924E+00	5.244E+00	--	--	--
	x_2	--	--	-4.974E+00	1.027E+01	--	--	--
	x_3	--	--	-7.529E+00	9.160E+00	--	--	--
	x_4	--	--	-7.151E+00	2.323E+00	--	--	--
	x_5	--	--	-9.146E+00	-7.994E+00	--	--	--
Off.	y_{off}	--	--	--	-1.146E+01	--	--	--
Eigv.1	Re	9.328E-01	9.519E-01	9.250E-01	9.210E-01	9.650E-01	9.266E-01	-5.981E-01
	Im	2.160E-01	1.678E-01	2.206E-01	2.475E-01	1.540E-01	2.271E-01	6.590E-01
2	Re	9.328E-01	9.519E-01	9.250E-01	9.210E-01	9.650E-01	9.266E-01	-5.981E-01
	Im	-2.160E-01	-1.678E-01	-2.206E-01	-2.475E-01	-1.540E-01	-2.271E-01	-6.590E-01
3	Re	9.621E-01	-2.562E-01	9.606E-01	9.549E-01	5.541E-01	9.541E-01	9.705E-01
	Im	1.131E-01	5.446E-01	1.136E-01	1.293E-01	0.000E+00	1.179E-01	1.549E-01
4	Re	9.621E-01	-2.562E-01	9.606E-01	9.549E-01	--	9.541E-01	9.705E-01
	Im	-1.131E-01	-5.446E-01	-1.136E-01	-1.293E-01	--	-1.179E-01	-1.549E-01
5	Re	7.949E-01	6.529E-01	7.577E-01	9.291E-01	--	8.651E-01	8.957E-01
	Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00	--	0.000E+00	0.000E+00

Table 5.1.15 Retina-reflex; model parameters

Model	n	$\underline{x}(0)$	y_{off}	LF $\{(\sum e^2)/89\}$			RE
				PE	OE	IE	
OEM	5	N	N	1.095E+02	6.002E+01	--	3.785E-02
EEM	5	N	N	3.067E+00	1.102E+03	--	6.954E-01
OEM	5	Y	N	--	5.326E+01	--	3.359E-02
OEM	5	Y	Y	--	1.512E+01	--	9.536E-03
OEM	3	N	N	1.899E+01	4.677E+02	--	2.950E-01
IEM (OEM)	5	N	N	4.170E+01	6.452E+01	2.232E+00	4.069E-02
IEM (EEM)	5	N	N	4.602E+00	6.670E+02	1.877E+00	4.206E-01

Table 5.1.16 Retina-reflex; Model error functions

Attention should be given to the next results.

- Most eigenvalues are near the unit circle, indicating a relatively large sampling rate and a good performance of the estimation routine for systems with eigenvalues near the unit circle.
  The α-parameters of OEM, OEM with initial state, OEM with initial state and offset, and IEM (OEM) do not differ much (compare dynamic behaviour of error in simulations).
- The reconstruction error and OE-LF of the OEM with offset and initial state are obviously the best.
- OE-LF of OEM with initial state is better then the OE-LF of the OEM without initial state.
- OE-LF of IEM (OEM) is just slightly worse then the OE-LF of the OEM, because the OEM is taken as initial guess of the parameter values.
- The IE-LF of the IEM with the EEM as initial parameter guess is slightly better then the IE-LF of the IEM with the OEM as initial parameter guess. Both IE-LF's are less then the PE-LF of the PEM (=EEM).

5.2 Test system

To show the impact of most derived relations between introduced errors and to show the implication of (pseudo-)canonical forms on the estimation of MIMO-models, data is generated with a MIMO test system (model). Requirements for the test system are:

- The exact model (of the system) should not have too many parameters because of the possible lenght in time of the estimation. This resulted in a 3-th order system with 2 inputs and 2 outputs (n=3, p=q=2).
- Most estimations have to have clear results, so the eigenvalues do not have to be too close to the unit circle to avoid possible failures due to large eigenvalues (the previous example showed that the program also runs for systems with large eigenvalues). Because of this the system matrix F is chosen to be

$$F = \mathrm{Diag}\ (-0.2,\ 0.8,\ 0.5)$$

- All modes of the impuls responses should have equal energy so that they can be estimated equally good. The energy in one mode (if F is a diagonal matrix) can be given by (see Carriere 1984)

$$EN(i) = \frac{\sum_{j=1}^{q} h_{ji} \sum_{j=1}^{p} g_{ij}}{1-\lambda_i^2} \qquad (5.3)$$

with: λ_i eigenvalue i, and h_{ji}, g_{ij} entries of H and G.

- The energy in the outputs should not be the same, so that additive white output noise sequence with equal S/N-ratio on every output do not have the same auto-correlation and consequently the OE minimisation with the LS-criterion does not necessarily lead to the real system.

Last 2 restrictions lead to

$$G = \begin{bmatrix} -3.0 & 4.0 \\ -1.1 & 0.5 \\ 2.0 & 0.7 \end{bmatrix} ; \quad H = \begin{bmatrix} 3.4 & 9.8 & 7.8 \\ 1.95 & 0.85 & 2.45 \end{bmatrix} ; \quad D = \begin{bmatrix} 1.0 & 0.0 \\ 0.0 & 0.25 \end{bmatrix}$$

$\underline{x}(0)=\underline{0}$; EN(1) = 400.1 ; EN(2) = 392.4 ; EN(3) = 400.2 ;

In fig.5.2.1-a/b the impulse responses are given. In all the other impulse response plots these impulse responses are given as broken lines

- The initial state is chosen to be zero so that a wrong assumption on the initial state does not influence the results.

With this system output data is generated with zero mean white Gaussian noise as input. On every output white gaussian noise is added such that the S/N-ratio is 20dB for every output. Table 5.2.1 gives numerical values of the signals used for the estimation (500 sample moments).

Signal	Mean	St.Dev.(σ)	Power
Input 1	-1.048E-01	9.775E-01	9.665E-01
Input 2	-8.501E-05	9.075E-01	8.235E-01
Output 1	3.028E+00	2.361E+01	5.667E+02
Output 2	-1.291E-01	9.629E+00	9.273E+01

In fig.5.3.1-a/b the first 50 samples that are used for the estimation are plotted. In the following plots of outputs of estimated models this output data is drawn in broken lines.
In fig.5.3.2-a/b the simulated data of the test model is drawn, so without 20dB noise.
Notice the different vertical scaling of output 1 and output 2.

From the test model we see that both pseudo-canonical forms are possible ((1,2) and (2,1)) and that the canonical structure equals (2,1), so that the canonical form is identical to the pseudo-canonical form (structure (2,1)) because no extra structural zeros are included in the canonical structure (2,1) (compare the pseudo-canonical form (2,1)). Consequently the prediction error can be calculated for the pseudo-canonical forms (2,1) (only canonical forms are implemented in the program).
The test model is also not representable in the canonical form (1,2), which contains 1 parameters less then the pseudo-canonical form (1,2).
Notice that R(α) (see sect.3.2) is not equal to the identity matrix for the canonical structure (2,1).

The estimation results are shown in fig.5.2.2 till 5.3.6 and tables 5.3.7 and 5.3.8. The estimations are done over 500 samples with correct initial state assumption.

Because we want to compare the estimated model with the test model, in table 5.3.7 also the (pseudo-)canonical forms of the test model are included (they are obtained from an EEM estimation in MFD with undisturbed I/O-samples).

The initial guess for the EEM's are obtained from a EEM estimation in MFD, transformed to SSM (EE-LF of EEM estimated in MFD between brackets in table 5.3.8, EEM's again estimated with conjugate gradient method because it is faster than the Newton Methods). The initial guess for the OEM's were the estimated EEM's, and the initial guess for the IEM's were the estimated OEM's (OEM's and IEM's estimated with Quasi Newton method because conjugate gradient failed).

From the plots (fig.5.2.1 untill 5.3.6) we see that:

- The estimated EEM's do not have the same impuls responses as the original system (fig.5.2.2/3). The fact that fig.5.2.2-b shows good results is probably caused by the fact that the second MISO model has order 2.
  From the equation errors (fig.5.3.3) we see that both errors have 'similar stochastic properties. For structure (2,1) the equation error is not equal to the prediction error. This is why the prediction of output 2 with the EEM (fig.5.3.4) is not as good as the prediction of output 1.
- The OEM with structure (1,2) gives a very positive result as expected (fig.5.2.4-a/b). For structure (2,1) the algorithm probably found a local minimum (fig.5.2.4-c/d).
- If for the structure (2,1) we look at the IEM (fig.5.2.6) we see that extra parameters helped the algorithm to find a minimum much closer to the global minimum.
- The canonical estimations in structure (1,2) gives wrong results as expected (fig.5.2.7/8).
- From the innovation with an IEM (fig.5.3.6) we see that the innovation of output 2 is much better then the prediction with an EEM (fig.5.3.4). It is still worse than the innovation of output 1, because 20dB noise on a weak signal (output 2) does not increase the loss function as much as 20dB on a strong signal (output 1).

From table 5.3.8 we see that:

- Direct estimation of the SSM parameters of an EEM gives slightly better results (notice that the equivalent MFD has the same number of parameters) than in input-output description.
- Compared to the test models the EEM and OEM have a smaller EE-LF and OE-LF. This because the number of samples used for the estimation is limited and consequently the noise can be modelled for a small part.
- Estimation of IEM's includes a further decrease of the OE-LF (remember that the OEM's are taken as initial guess of the parameter values).
- All estimations give a slightly better estimation of output element 1 because output 1 has much more power or more specificly:
  The 1-st element of EE's for the EEM's are less than the 1-st element of the EE's of the test systems. The 1-st element of the OE for the OEM with structure (1,2) is less than the OE of the testsystem.

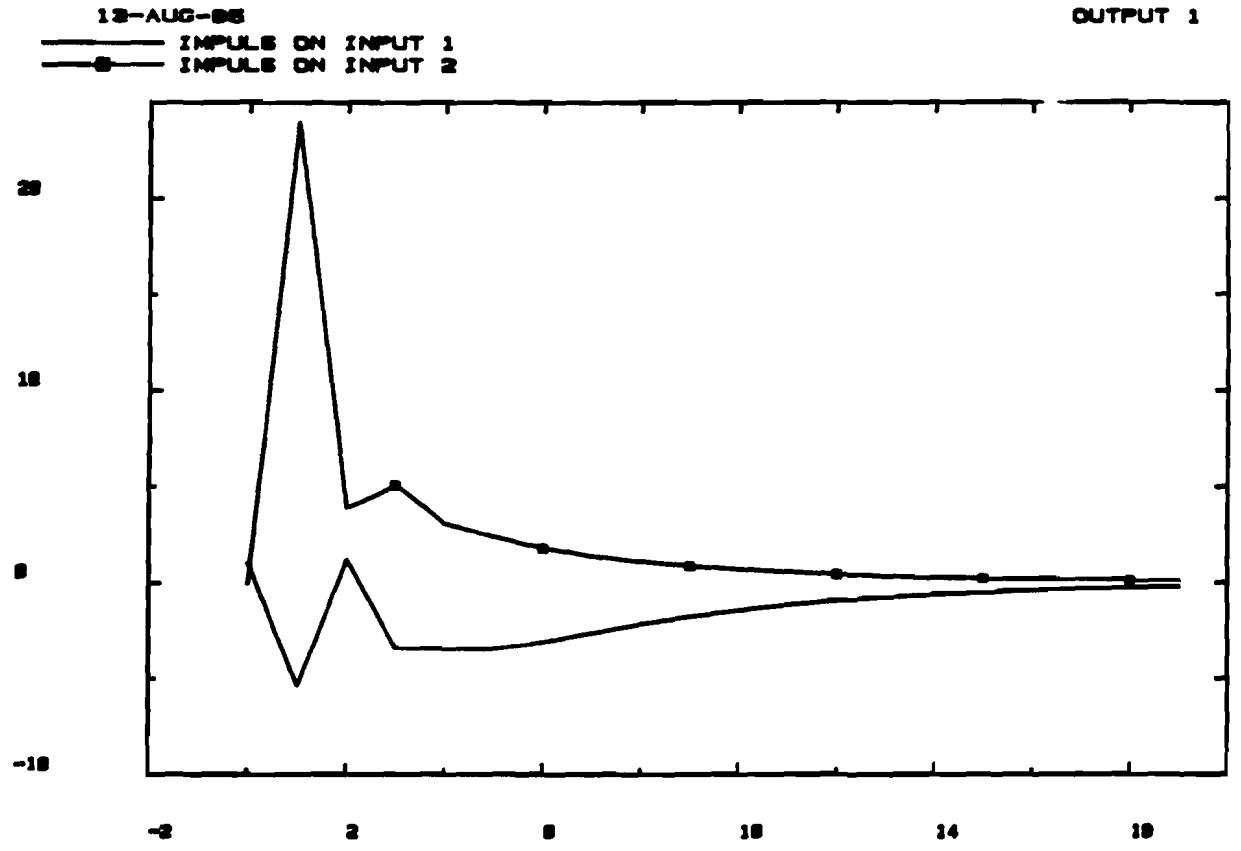


Fig. 5.2.1-a Imp.resp. of test model (so without noise). Output 1.

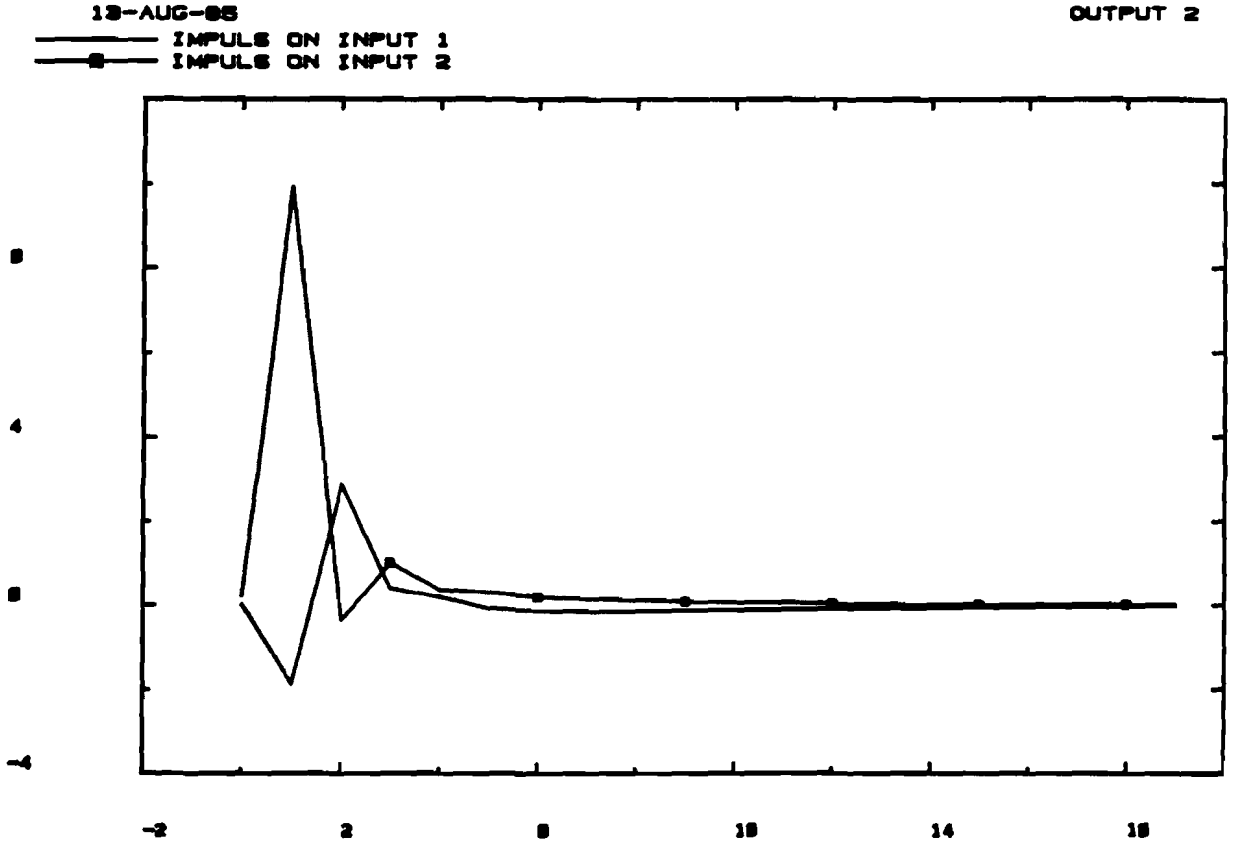


Fig. 5.2.1-b Imp.resp. of test model (so without noise). Output 2.

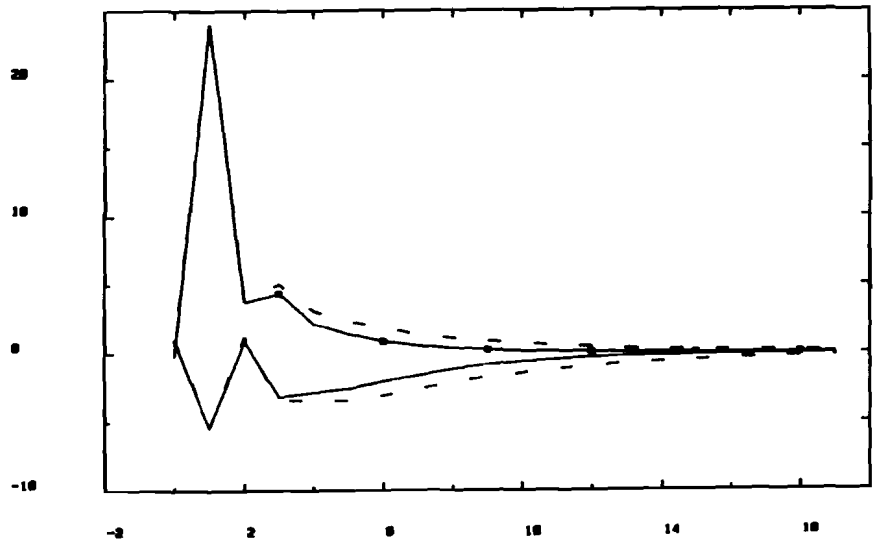


Fig. 5.2.2-a Imp.resp. of EEM structure (1,2). Output 1. (P_{Δ} CAN)

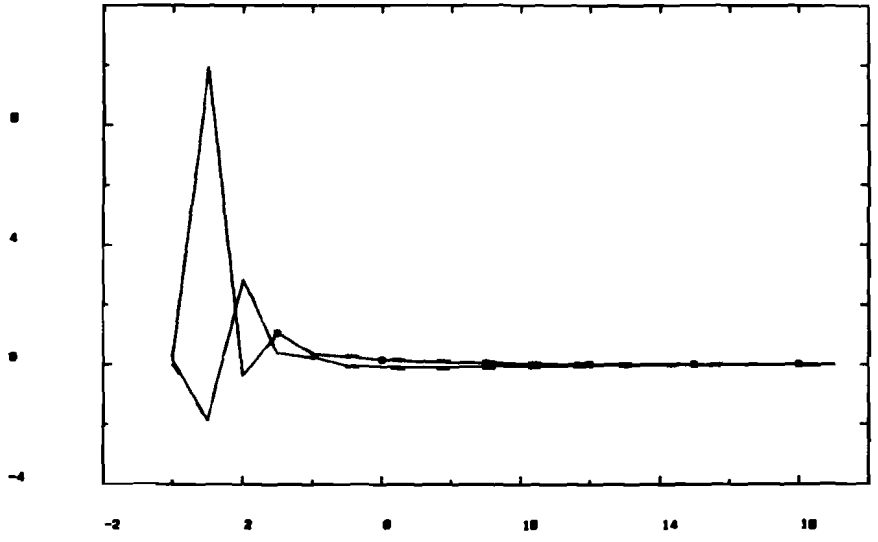


Fig. 5.2.2-b Imp. resp. of EEM structure (1,2). Output 2.

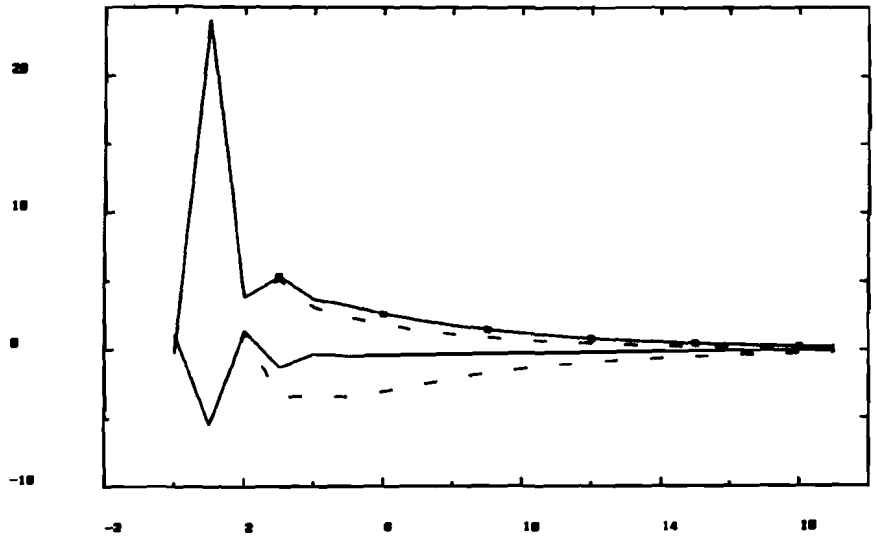


Fig. 5.2.3-a Imp.resp. of EEM structure (2,1). Output 1. CAN. / P_{Δ} (IN)

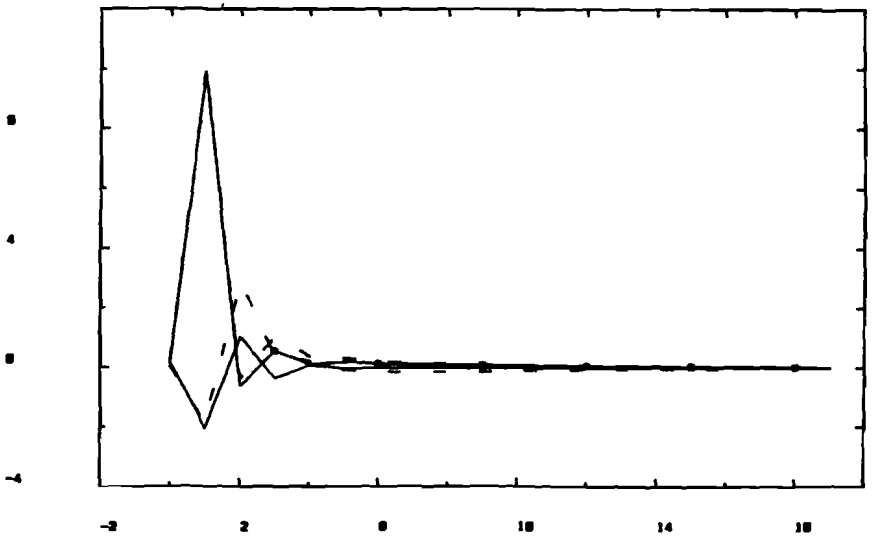


Fig. 5.2.3-b Imp.resp. of EEM structure (2,1). Output 2.

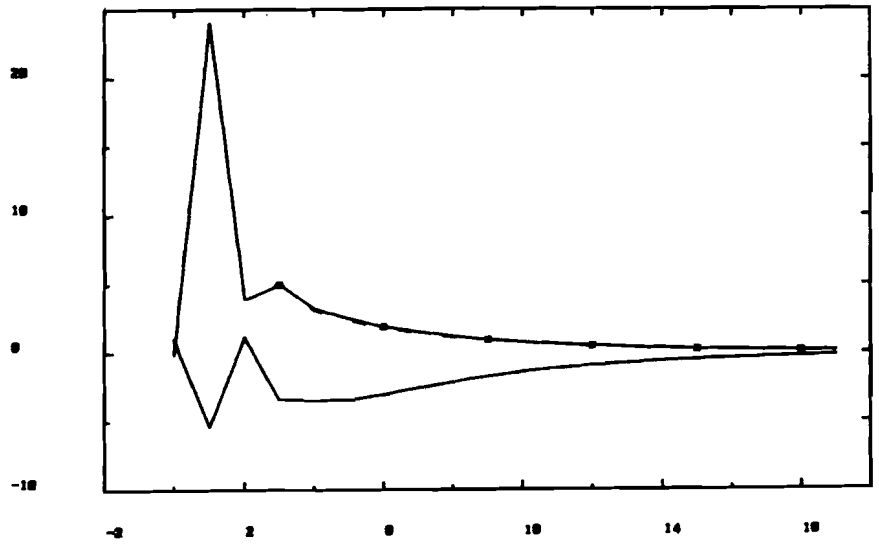


Fig. 5.2.4-a Imp.resp. of OEM structure (1,2). Output 1.

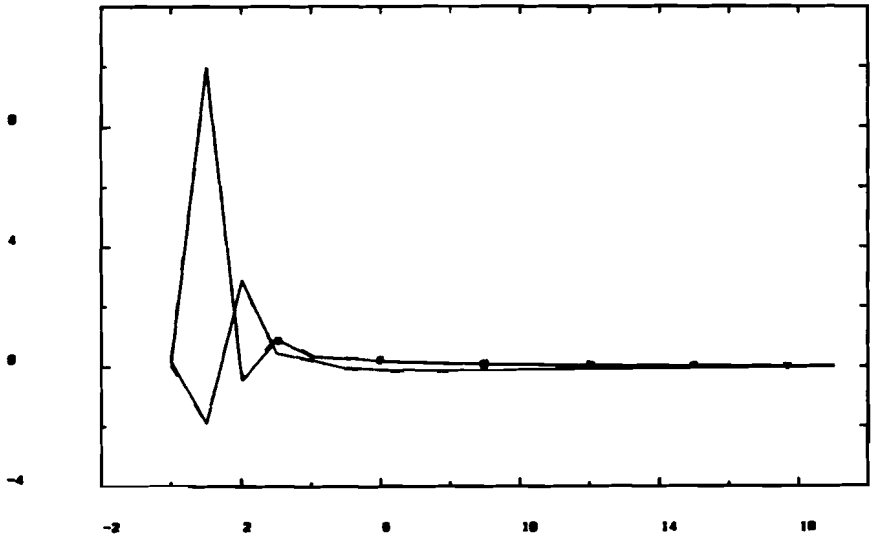


Fig. 5.2.4-b Imp.resp. of OEM structure (1,2). Output 2.

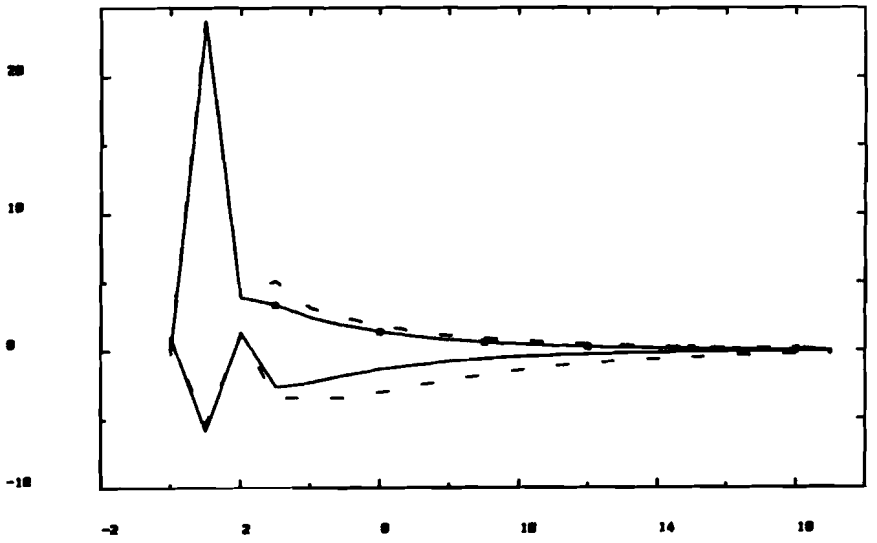


Fig. 5.2.4-c Imp.resp. of OEM structure (2,1). Output 1.

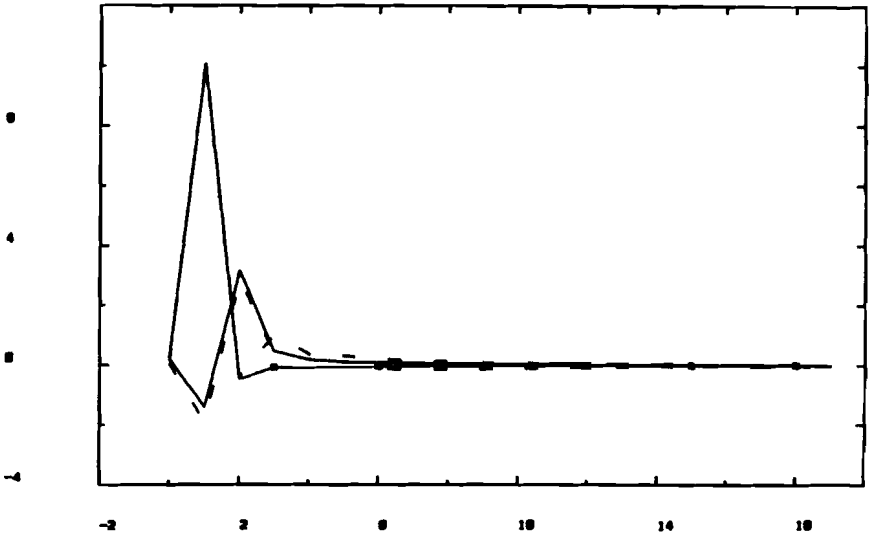


Fig. 5.2.4-d Imp.resp. of OEM structure (2,1). Output 2.

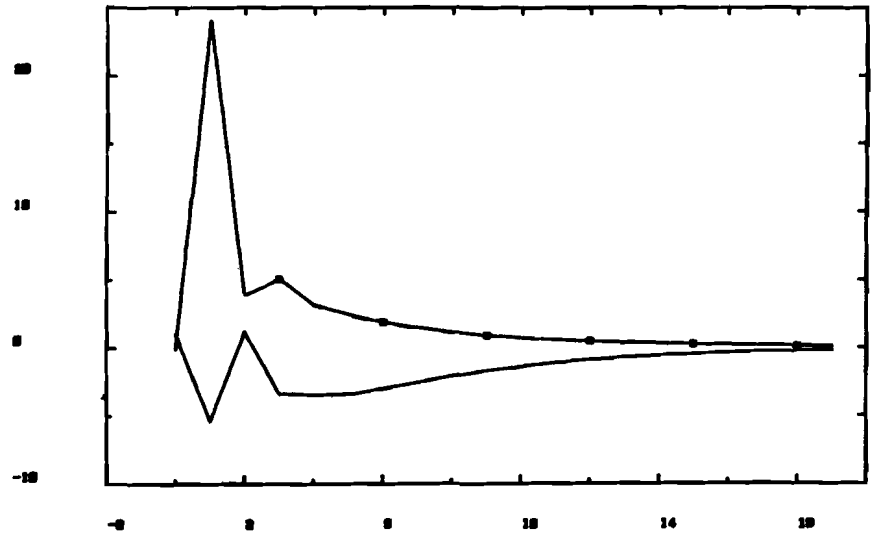


Fig. 5.2.5-a Imp.resp.of IEM (K=0) structure (1,2). Output1.

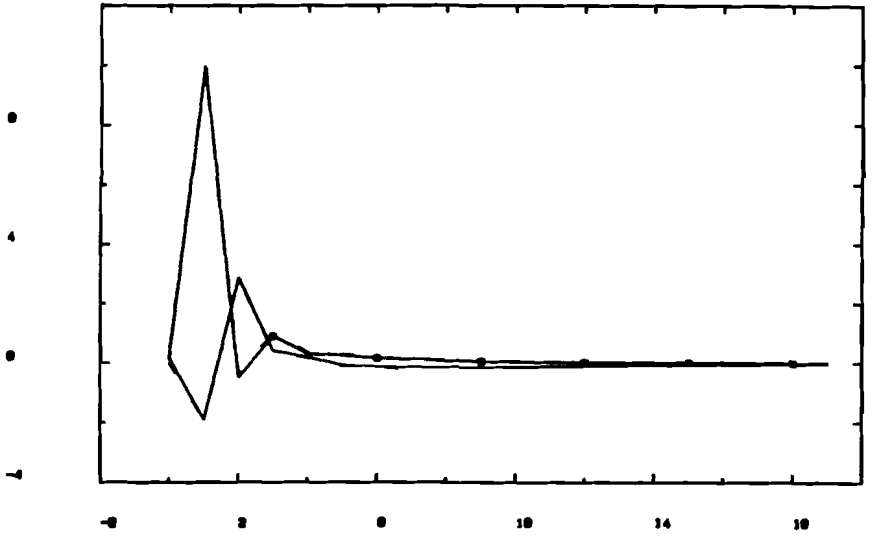


Fig. 5.2.5-b Imp.resp. of IEM (K=0) structure (1,2). Output 2.

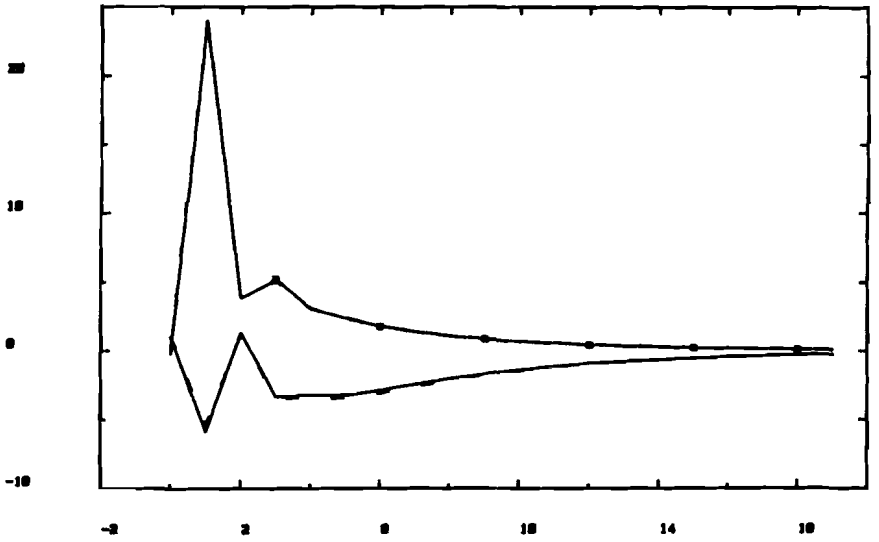


Fig. 5.2.6-a Imp.resp. of IEM (K=0) structure (2,1). Output 1.

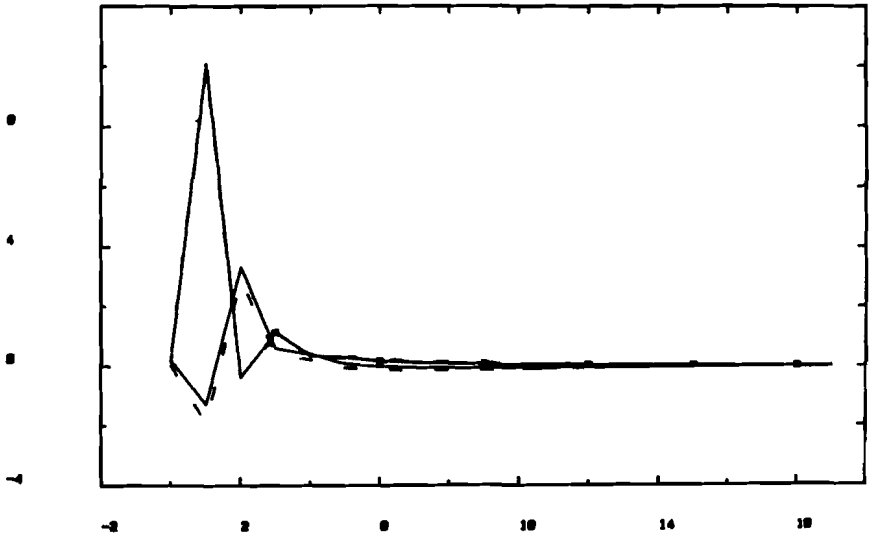


Fig. 5.2.6-b Imp.resp. of IEM (K=0) structure (2,1). Output 2.

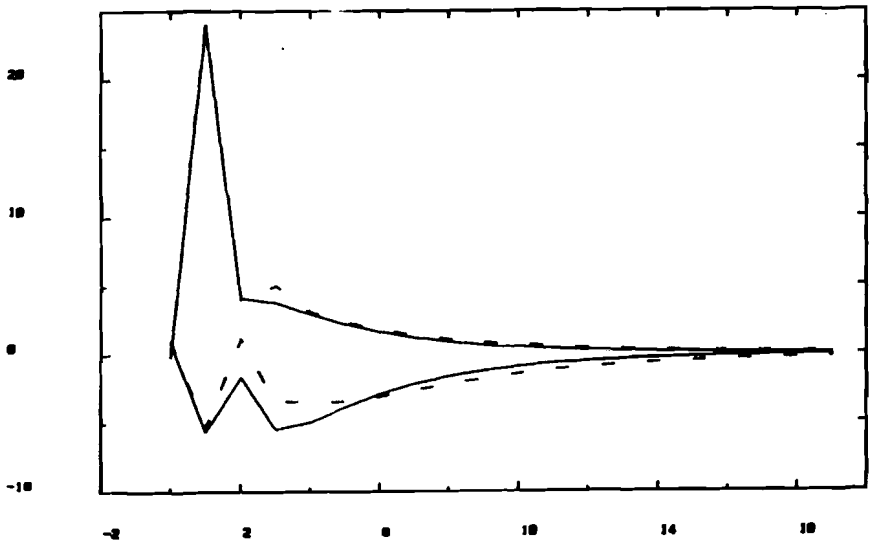


Fig. 5.2.7-a Imp.resp. of OEM estimated in canonical structure (1,2). Output 1.

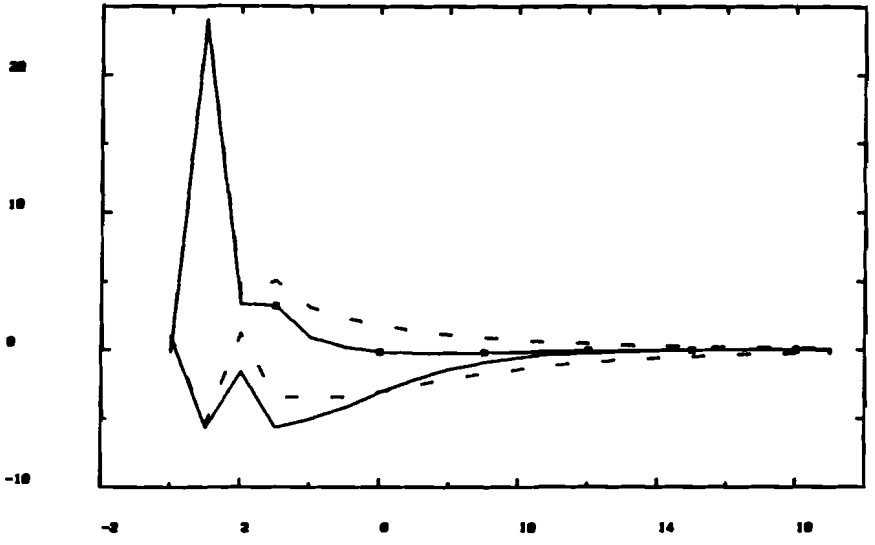


Fig. 5.2.8-a Imp. resp. of EEM estimated in canonical structure (1,2). Output 1.

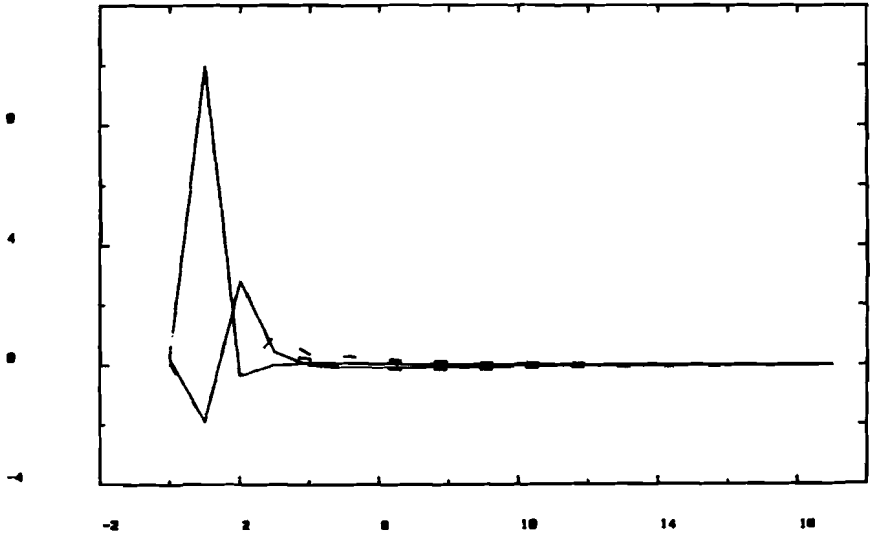


Fig. 5.2.7-b Imp.resp. of OEM estimated in canonical structure (1,2). Output 2.

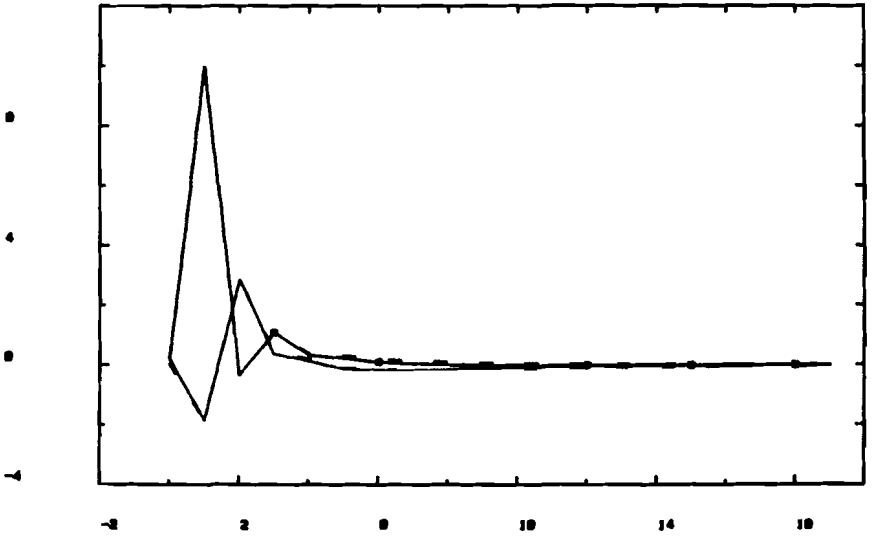


Fig. 5.2.8-b Imp.resp. of EEM estimated in canonical structure (1,2). Output 2.

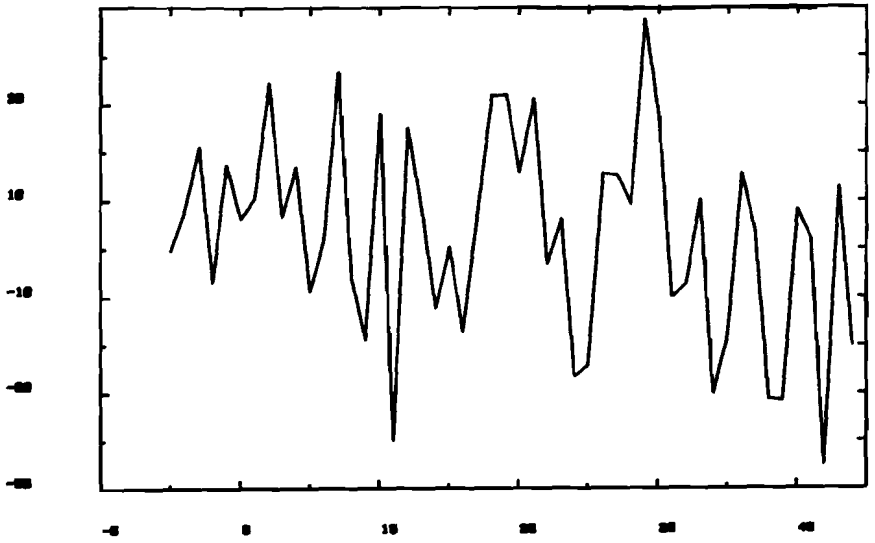


Fig. 5.3.1-a Used output data for estimation (first 50 samples) inclusive 20dB noise. Output 1.

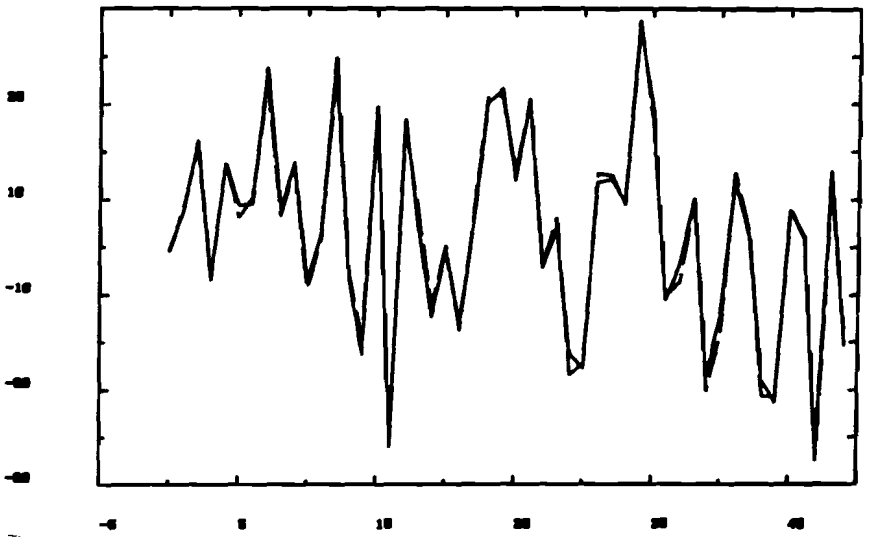


Fig. 5.3.2-a Simulated data (with test model) without noise. Output 1.

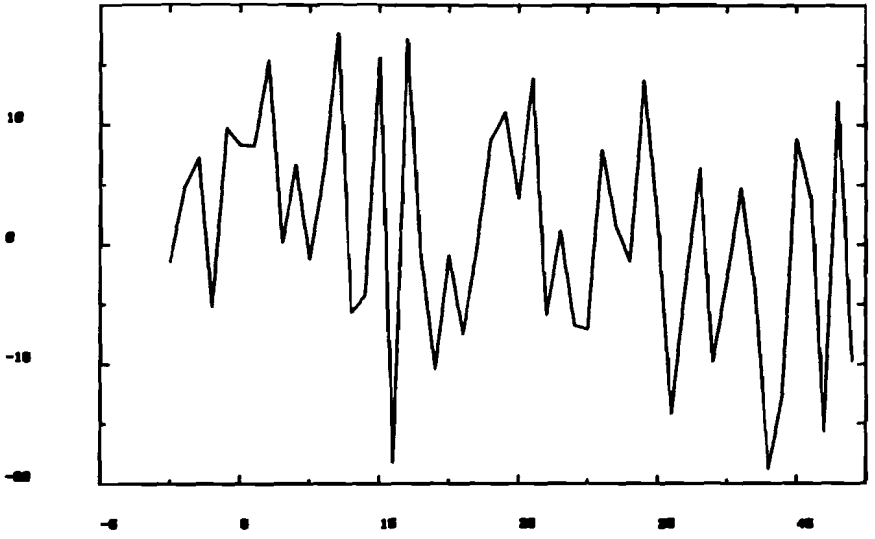


Fig. 5.3.1-b Used output data for estimation (first 50 samples) inclusive 20dB noise. Output 2.

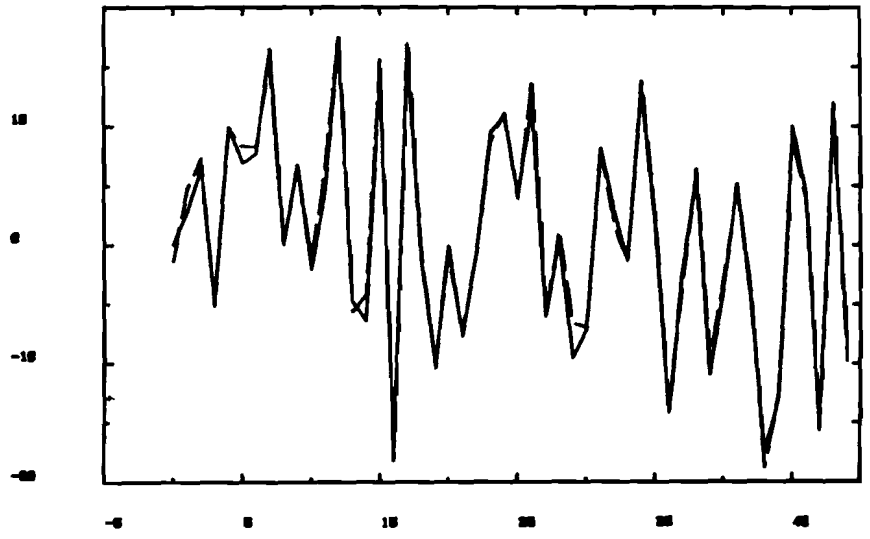


Fig. 5.3.2-b Simulated data (with test model) without noise. Output 2.

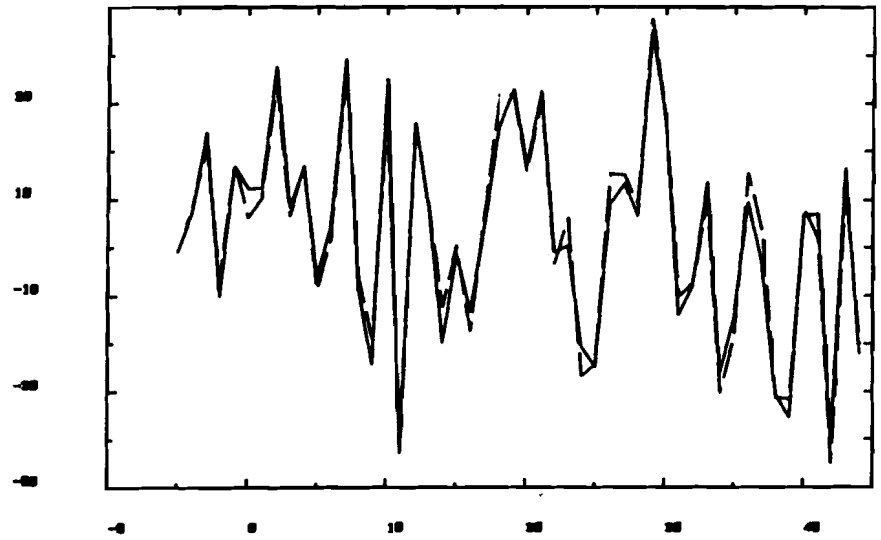


Fig. 5.3.4-a Prediction of EEM with structure (2,1). Output 1.

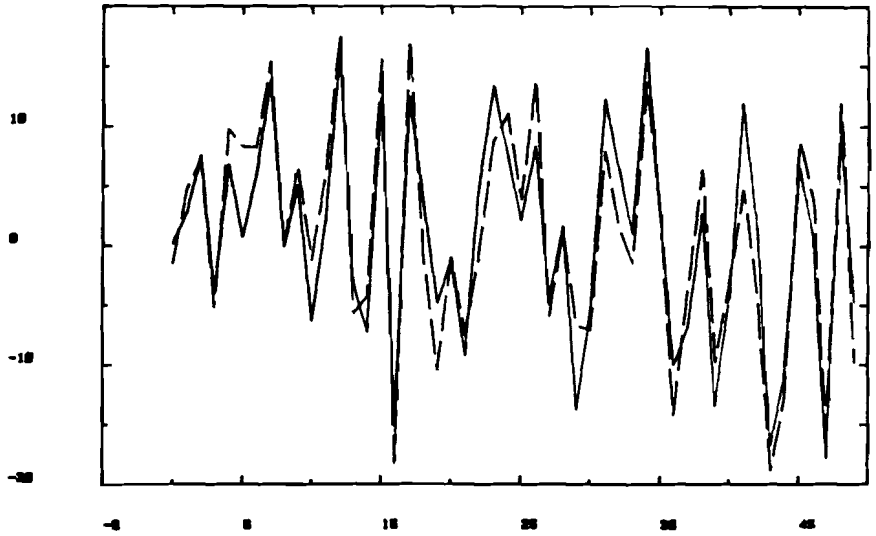


Fig. 5.3.4-b Prediction of EEM with structure (2,1). Output 2

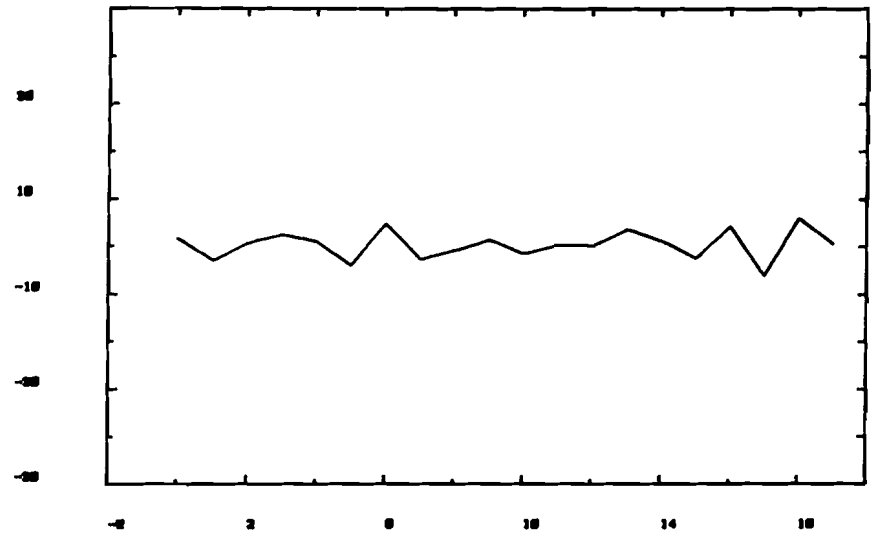


Fig. 5.3.3-a Equation error of EEM structure (1,2). Element 1.

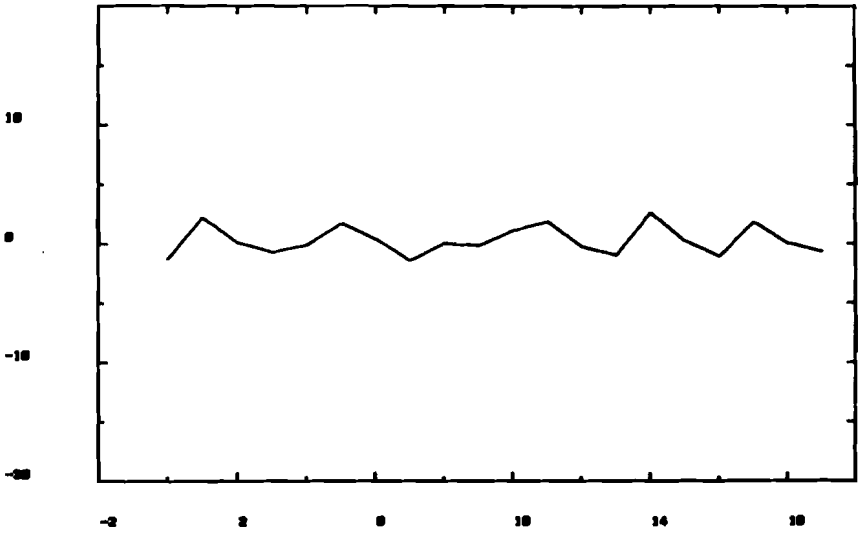


Fig. 5.3.3-b Equation error of EEM structure (1,2). Element 2.

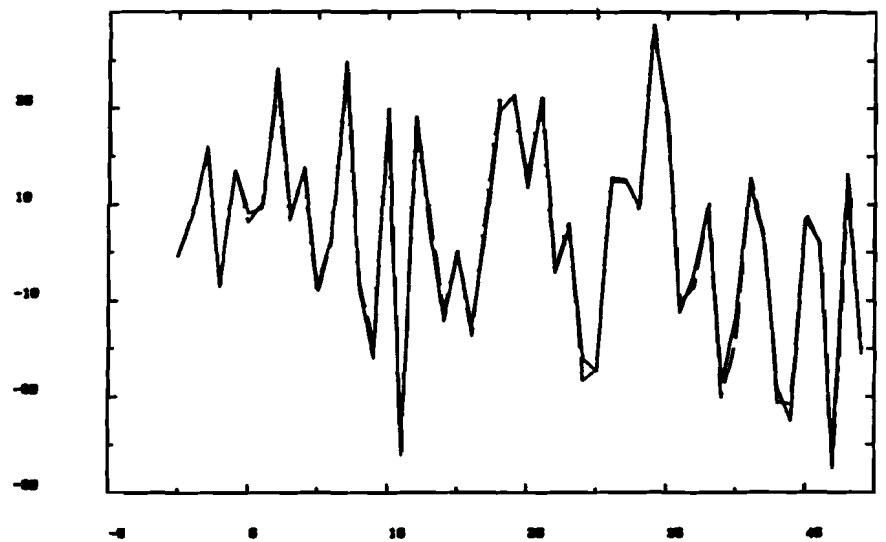


Fig. 5.3.6-a Innovation of IEM with structure (2,1). Output 1.

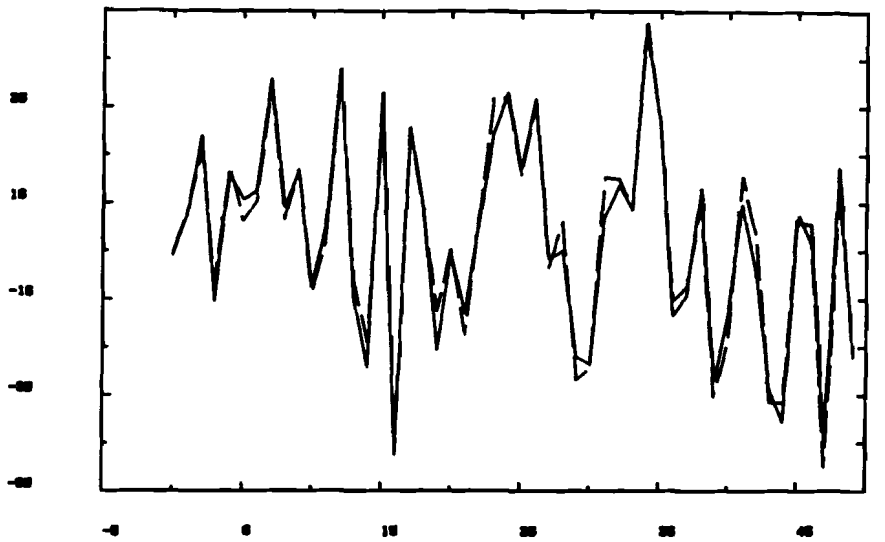


Fig. 5.3.5-a Prediction (with IEM with structure (2,1). Output 1.

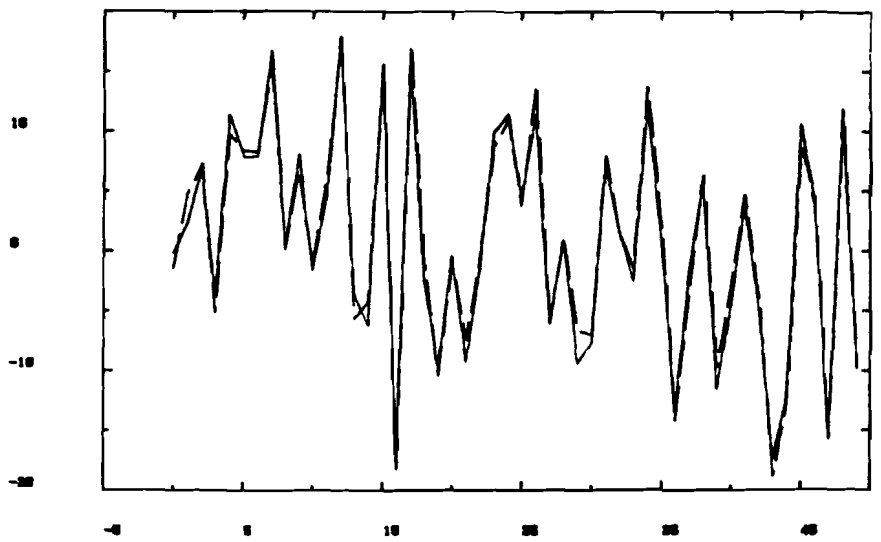


Fig. 5.3.6-b Innovation of IEM with structure (2,1). Output 2.

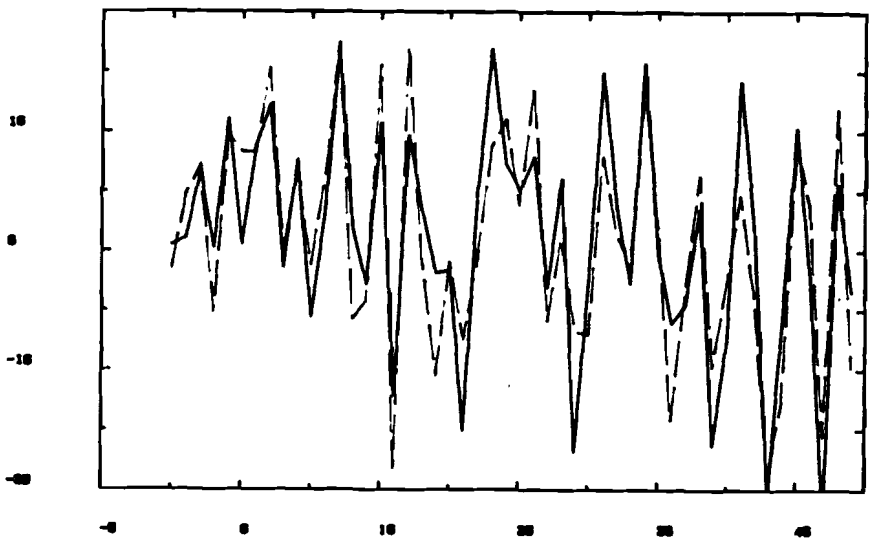


Fig. 5.3.5-b Prediction (K=0) with IEM with structure (2,1). Output 2.

Para-meters	Model; Structure (1,2), Pseudo-canonical.				Canonical		Para-meters	Model; structure (2,1)			
	Test	EEM	OEM	IEM	EEM	OEM		Test	EEM	OEM	IEM
A $\alpha_{11,1}$	8.799E-01	7.841E-01	8,711E-01	8.704E-01	7.925E-01	7.890E-01	A $\alpha_{11,1}$	1.390E+00	6.630E-01	6.825E-01	8.549E-01
$\alpha_{12,1}$	-1.690E+00	-1.475E+00	-1.667E+00	-1.667E+00	-1.556E+00	-1.475E+00	$\alpha_{11,2}$	-6.570E-01	7.719E-02	-2.324E-01	-1.435E-01
$\alpha_{12,2}$	9.621E-01	8.545E-01	9.290E-01	9.290E-01	0	0	$\alpha_{12,1}$	-2.581E+00	-1.103E+00	-1.204E+00	-1.455E+00
$\alpha_{21,1}$	3.882E-02	3.149E-02	3.326E-02	3.412E-02	3.233E-02	1.516E-02	$\alpha_{21,1}$	-9.145E-01	-2.658E-01	-6.109E-01	-6.559E-01
$\alpha_{22,1}$	1.635E-02	3.851E-02	2.046E-02	2.058E-02	3.696E-02	-2.759E-02	$\alpha_{21,2}$	1.039E+00	3.468E-01	8.343E-01	7.813E-01
$\alpha_{22,2}$	2.201E-01	2.123E-01	2.278E-01	2.278E-01	2.122E-01	1.618E-01	$\alpha_{22,1}$	1.757E+00	4.472E-01	1.088E+00	1.222E+00
B $b_{1,11}$	-5.380E+00	-5.437E+00	-5.409E+00	-5.409E+00	-5.665E+00	-5.648E+00	B $b_{1,11}$	-5.380E+00	-5.463E+00	-5.858E+00	-5.888E+00
$b_{1,12}$	2.396E+01	2.393E+01	2.401E+01	2.401E+01	2.381E+01	2.382E+01	$b_{1,12}$	2.396E+01	2.402E+01	2.390E+01	2.389E+01
$b_{2,11}$	-1.885E+00	-1.859E+00	-1.900E+00	-1.899E+00	-1.872E+00	-1.905E+00	$b_{1,21}$	1.216E+00	1.370E+00	1.339E+00	1.321E+00
$b_{2,12}$	9.940E+00	9.960E+00	9.947E+00	9.947E+00	9.968E+00	9.956E+00	$b_{1,22}$	3.930E+00	3.845E+00	3.844E+00	3.846E+00
$b_{2,21}$	2.872E+00	2.832E+00	2.886E+00	2.886E+00	2.829E+00	2.803E+00	$b_{2,11}$	-1.885E+00	-2.037E+00	-1.408E+00	-1.299E+00
$b_{2,22}$	-3.625E-01	-3.440E-01	-4.602E-01	-4.602E-01	-3.441E-01	-3.598E-01	$b_{2,12}$	9.940E+00	9.905E+00	1.005E+01	1.008E+01
D $d_{1,1}$	1.000E+00	9.124E-01	9.108E-01	9.108E-01	8.944E-01	8.945E-01	D $d_{1,1}$	1.000E+00	9.608E-01	9.797E-01	9.650E-01
$d_{1,2}$	-3.275E-15	-1.211E-01	-8.402E-02	-8.401E-02	-1.390E-01	-1.389E-01	$d_{1,2}$	-3.539E-15	-2.194E-01	-2.087E-01	-1.955E-01
$d_{2,1}$	2.776E-16	1.587E-01	1.596E-01	1.596E-01	1.622E-01	1.621E-01	$d_{2,1}$	-2.970E-15	1.560E-01	1.744E-01	1.763E-01
$d_{2,2}$	2.500E-01	2.042E-01	2.028E-01	2.028E-01	2.000E-01	2.000E-01	$d_{2,2}$	2.500E-01	1.629E-01	1.610E-01	1.660E-01
K $k_{1,11}$	--	--	--	-1.302E-04	--	--	K $k_{1,11}$	--	--	--	8.922E-02
$k_{1,12}$	--	--	--	-3.041E-05	--	--	$k_{1,12}$	--	--	--	9.635E-02
$k_{2,11}$	--	--	--	1.698E-04	--	--	$k_{1,21}$	--	--	--	4.833E-02
$k_{2,12}$	--	--	--	5.412E-05	--	--	$k_{1,22}$	--	--	--	7.823E-02
$k_{2,21}$	--	--	--	8.734E-05	--	--	$k_{2,11}$	--	--	--	6.238E-03
$k_{2,22}$	--	--	--	2.839E-05	--	--	$k_{2,12}$	--	--	--	3.826E-02
1. Re	-2.000E-01	-2.134E-01	-1.882E-01	-1.908E-01	6.062E-01	7.399E-01	1. Re	-2.000E-01	-3.089E-01	-5.724E-02	-2.256E-01
Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00	1.248E-01	0.000E+00	Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00
2. Re	8.000E-01	6.868E-01	8.012E-01	7.976E-01	6.062E-01	2.146E-01	2. Re	8.000E-01	8.196E-01	7.510E-01	8.088E-01
Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00	-1.248E-01	0.000E+00	Im	0.000E+00	0.000E-01	0.000E+00	0.000E+00
3. Re	5.000E-01	5.230E-01	4.858E-01	4.914E-01	-2.077E-01	-3.700E-03	3. Re	5.000E-01	1.362E-02	1.620E-01	4.953E-01
Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	0.000E+00	Im	0.000E+00	0.000E+00	0.000E+00	0.000E+00

Table 5.3.7 Model parameters estimated with test data.

Structure/ Model	EE			PE			OE			IE		
	LF (/500)	St.Dev.	Power	LF	St.Dev.	Power	LF	St.Dev.	Power	LF	ST.Dev.	Power
1 Test	1.887E+01	4.207E+00	1.770E+01	--	--	--	7.629E+00	2.560E+00	6.554E+00	--	--	--
2		1.080E+00	1.169E+00		--	--		1.034E+00	1.075E+00		--	--
1 EEM	1.687E+01	4.036E+00	1.635E+01	--	--	--	1.646E+01	3.776E+00	1.538E+01	--	--	--
2	(1.749E+01)	1.061E+00	1.126E+00		--	--		1.036E+00	1.072E+00		--	--
1 OEM	1.845E+01	4.158E+00	1.729E+01	--	--	--	7.567E+00	2.547E+00	6.488E+00	--	--	--
2		1.078E+00	1.163E+00		--	--		1.038E+00	1.079E+00		--	--
1 IEM	1.844E+01	4.157E+00	1.728E+01	--	--	--	7.558E+00	2.546E+00	6.484E+00	7.557E+00	2.546E+00	6.484E+00
2		1.076E+00	1.159E+00		--	--		1.035E+00	1.073E+00		1.035E+00	1.073E+00
1 EEM	2.439E+01	4.823E+00	2.326E+01	2.439E+01	4.823E+00	2.326E+01	3.821E+01	6.091E+00	3.710E+01	--	--	--
2 Can.		1.061E+00	1.125E+01		1.061E+00	1.125E+01		1.051E+00	1.105E+01		--	--
1 OEM	2.584E+01	4.874E+00	2.376E+01	2.584E+01	4.874E+00	2.376E+01	2.287E+01	4.552E+00	2.074E+01	--	--	--
2 Can.		1.443E+00	2.081E+00		1.443E+00	2.081E+00		1.457E+00	2.122E+00		--	--
2 Test	4,940E+01	5.500E+00	3.027E+01	8.484E+01	5.500E+00	3.027E+01	7.629E+00	2.560E+00	6.554E+00	--	--	--
1		4.373E+00	1.912E+01		7.383E+00	5.456E+01		1.034E+00	1.075E+00		--	--
2 EEM	2.167E+01	3.853E+00	1.500E+01	2.628E+01	3.853E+00	1.590E+01	5.298E+01	6.445E+00	4.795E+01	--	--	--
1	(2.171E+01)	2.560E+00	6.668E+00		3.324E+00	1.128E+01		2.230E+00	5.028E+00		--	--
2 OEM	3.771E+01	4.831E+00	2.447E+01	6.966E+01	4.831E+00	2.447E+01	2.524E+01	4.490E+00	2.245E+01	--	--	--
1		3.621E+00	1.324E+01		6.605E+00	4.519E+01		1.658E+00	2.780E+00		--	--
2 IEM	2.910E+00	4.042E+00	1.642E+01	4.685E+01	4.042E+00	1.642E+01	8.590E+00	2.644E+00	7.004E+00	8.744E+00	2.681E+00	7.194E+00
1		3.506E+00	1.230E+01		5.509E+00	1.587E+00		1.249E+00	1.587E+00		1.236E+00	1.550E+00

Table 5.3.8 Models estimated with test data; Error functions
Standard deviation and power for each error element
(PE not implemented for pseudo-canonical forms)

Conclusions

From the derived stochastic SSM's and for the shown estimation results we conclude that:

- For MIMO-models the equation error is not equal to the prediction error. For the canonical form they are related by a lower triangular matrix filled with specific auto-regressive parameters of the model and diagonal elements equal to 1.

- The equation and prediction error introduction are not very elegant introductions for the state space representation.
  The innovation and output error introduction are more straight forward introductions in SSM's.

- An EEM estimated in MFD parameters is preferable (if available and possible) above an EEM estimated in SSM, because it can be obtained much faster and is not sufficiently different if the number of parameters is equal in MFD and SSM.

- For predictive purposes the PEM is preferable if as less parameters as possible should be estimated. For initial parameter values one can choose the EEM estimated in SSM or MFD parameters.

- For predictive purposes the IEM is preferable if more accurate predictions (innovations) of the outputs are desired. If the IEM should also have good simulation results, the IEM should be estimated with the OEM as initial guess of the parameter values, because then the probability of ariving at a local minimumwith good simulation performances is larger then if an EEM is taken as initial guess of the parameter values. If the innovative demands are very strong, it might be better to choose the initial parameter values equal to the EEM.

- For simulation purposes the OEM is best, but if the simulation demand are not very strong and if one also wants to predict (innovate) with the model, one should at least try to estimate a IEM with initial parameter values equal to the OEM.

- The equation error in SSM is derived from the equation error in MFD. Because of this the EE-LF is 'not very' non-linear in the parameters. Possibly because of this property the conjugate gradient method is well suited for minimisation of the EE-LF in SSM.

- Because the output error and innovation error are far from linear, the Newton hill-climbing methods will find lower minima of the OE- and IE-LF than the conjugate gradient method. The conjugate gradient method can be used for the first iterations because it is much faster.
  The quasi-Newton method gave slightly better results than the modified Newton method (in our examples).

Recommendations for further research

- One limitation of the implemented routine is that the initial guess for the parameters has to be given in pseudo-canonical form. Implementation of a routine that transforms every SSM into its (pseudo-)canonical forms will solve this problem.
- Estimation in a specific pseudo-canonical form might fail because of local minima in the loss function. A routine that transforms the pseudo-canonical form into another pseudo-canonical form could possibly solve this problem.
- The IEM includes nxq extra parameters. If we do not want to limit ourselves to those nxq parameters how will the extended SSM (with F-matrix extended with noise dynamics) look like.
- The LS-estimation can be extended to MLE-estimation. The first step might be the introduction of q parameters, the auto-correlations of the noises.
- Comparisons can be drawn between OEM's estimated in SSM and OEM's estimated in MFD.
- The interpretation of differences in the impulse responses of EEM, PEM, OEM and IEM can be examined thoroughly.
- It may be desired to know before hand which hill-climbing method to use for the estimation. Perhaps there can be found some rules that can point out the best hill-climbing method for a perticular form of the expression of the model error.

Appendix A

Numerical Expression for Model Errors and their Derivatives

To be able to calculate the model error for given values of the parameters, we need a suitable expression to use in our estimation algorithm.
Because a non-linear LS-estimator is used we also want to have a numerical expression for the derivatives of the model error to the parameters. These derivatives enable us to use more accurate and faster hill-climbing methods.

The general state space representation is given by

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K\underline{e}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{e}(k) \end{aligned} \qquad (A.1)$$

For an EEM and PEM the K matrix is a function of the α-parameters in A. Because this function is quite complex, we will start from the I/O-representation in State Space parameters (eq.(2.8)) to find the expression for the error and the derivatives.

$$\bar{A}(\alpha)V_{\mu}(z)\underline{y}(k) = \bar{A}(\alpha)W_{\mu}(z)B(b,d)\underline{u}(k) + \Sigma_{\mu}(z)\underline{e}(k) \qquad (2.8)$$

In this way also the pseudo-canonical forms of EEM's may be considered although a proper SSM equivalent does not exist (see sect. 3.3) section (A.1)

For the output error (K=0) also the state vector has to be eliminated from (A.1) to get an expresion for the error as a function of the parameters and in- and output samples, see section (A.2). The derivatives of this expression are not given here, because this method is not implemented in the final version of the estimation routine. The calculation of this expresion (of sect.A.2) can be quite time consuming and is difficult to implement.
Because of these drawbacks, an algorithm is given in section A.3 that calculates the output error and its derivatives in a recursive

way.

Just as the state vector is a temporary storage in the model, its derivative is a temporary storage of the derivatives to the model variables. This algorithm is less complex then the one in sect. A.2. The algorithm in sect. A.2 is given also because it does not use state variables or their derivatives, and may be useful for recursive parameter estimation.

A.1 Equation Error and Prediction Error

To facilitate further derivations we introduce the next notations (see also (2.9), (2.12) and (2.10)).

$[\ A\ ]_{iv,w}$: Element of $\bar{A}(\alpha)$ with
row number i
column number $(\mu_1+1+ \ldots + \mu_{v-1}+1+w)$

$= \alpha_{iv,w}$ if $w<\mu_v$
$= 1$ if $i=v$ and $w=\mu_v+1$
$= 0$ if $i\neq v$ and $w\neq\mu_v+1$

$[\ W\ ]_{v\ell,ws}$: Element of $W_\mu(z)$ with
row number $(\mu_1+1+ \ldots +\mu_{v-1}+1+w)$
column number $(\mu_1+1+ \ldots +\mu_{\ell-1}+1+s)$

$= z^{w-s}$ if $v=\ell$ and $s<w$
$= 0$ if $v\neq\ell$ or $s>w$

$[\ \bar{A}W\ ]_{i\ell,s}$: Element of $\bar{A}(\alpha)W_\mu(z)$ with
row number i
column number $(\mu_1+1+ \ldots +\mu_{\ell-1}+s)$

$[\ B\ ]_{\ell,sj}$: Element of B(b,d) with
row number $(\mu_1+1+ \ldots + \mu_{\ell-1}+1+s)$
columnnumber j

$= d_{\ell,j}$ if $s=1$
$= b_{\ell,(s-1)j}$ if $s\neq 1$

Now $\bar{A}(\alpha)W_\mu(z)$ can be written as

$$\left[\ \bar{A}W\ \right]_{i\ell,s} = \sum_{v=1}^{q}\sum_{w=1}^{(\mu_v+1)} \left[\ \bar{A}\ \right]_{iv,w}\left[\ W\ \right]_{v\ell,ws}$$

$$= \sum_{v=1}^{q}\sum_{w=1}^{\mu_v} \left[\ \bar{A}\ \right]_{iv,w}\left[\ W\ \right]_{v\ell,ws} + \left[\ W\ \right]_{i\ell,(\mu_i+1)s}\Bigg|_{\ell=i}$$

$$= \sum_{v=1}^{q}\sum_{w=1}^{\mu_v} (-\alpha_{iv,w}z^{w-s})\Big|_{\substack{v=1\\ w-s\geq 0}} + z^{\mu_i+1-s}\Big|_{\substack{\ell=i\\ \mu_i+1-s\geq 0}}$$

$$= \sum_{w=1}^{\mu_v}(-\alpha_{i\ell,w}z^{w-s})\Big|_{w-s\geq 0} + z^{\mu_i+1-s}\Big|_{\substack{\ell=i\\ \mu_i+1-s\geq 0}}$$

The equation error can be written as

$$\underline{e}(k) = \Sigma_\mu^{-1}(z)\bar{A}(\alpha)V_\mu(z)\underline{y}(k) - \Sigma_\mu^{-1}(z)\bar{A}(\alpha)W_\mu(z)B(b,d)\underline{u}(k) \qquad (A.2)$$

The i-th element of this vector becomes:

$$e_i(k) = e_{iy}(k) + e_{iu}(k) \qquad (A.3)$$

with : $$e_{iy}(k) = z^{-\mu_i}\sum_{j=1}^{q}\left[\ \bar{A}V\ \right]_{ij}\ y_j(k) \qquad (a.4)$$

$$= z^{-\mu_i}\sum_{j=1}^{q}\sum_{s=1}^{\mu_j}\left(-\alpha_{ij,s}z^{s-1}y_j(k)\right) + z^{-\mu_i}z^{\mu_i}y_i(k) \qquad (A.5)$$

$$= \sum_{j=1}^{q}\sum_{s=1}^{\mu_j}\left(-\alpha_{ij,s}y_j(k+s-1-\mu_i)\right) + y_i(k) \qquad (A.6)$$

and $$e_{iu}(k) = z^{-\mu_i} \sum_{j=1}^{p} \left[\bar{A}WB \right]_{ij} u_j(k) \tag{A.7}$$

$$= z^{-\mu_i} \sum_{j=1}^{p} \sum_{\ell=1}^{q} \sum_{s=1}^{\mu_\ell+1} \left[\bar{A}W \right]_{i\ell,s} \left[B \right]_{\ell,sj} u_j(k) \tag{A.8}$$

$$= z^{-\mu_i} \sum_{j=1}^{p} \sum_{\ell=1}^{q} \sum_{s=1}^{\mu_\ell+1} \left[\sum_{w=1}^{\mu_\ell} (-\alpha_{i\ell,w} z^{w-s}) \Big|_{w-s\geq 0} \right.$$

$$\left. + z^{\mu_i+1-s} \Big|_{\substack{\ell=i \\ \mu_i+1-s\geq 0}} \right] \left[B \right]_{\ell,sj} u_j(k) \tag{A.9}$$

$$= \sum_{j=1}^{p} \sum_{\ell=1}^{q} \sum_{s=1}^{\mu_\ell+1} \left[\sum (-\alpha_{i\ell,w} \left[B \right]_{\ell,sj} u_j(k-\mu_i+w-s)) \Big|_{w-s\geq 0} \right.$$

$$\left. + \left[B \right]_{\ell,sj} u_j(k+1-s) \Big|_{\substack{\ell=1 \\ \mu_i+1-s\geq 0}} \right] \tag{A.10}$$

The prediction error can be found by solving

$$\underline{e}_e(k) = R(\alpha)\underline{e}_p(k) \tag{3.33}$$

with $R(\alpha)$ a lower triangular matrix (eq.(3.19)).

To derive a suitable expression for the derivative of the equation error to the model parameters, we again use the same expression, eq.(A.2). The derivatives to the α-parameters become:

$$\frac{\partial \underline{e}(k)}{\partial \alpha_{ij,\ell}} = \frac{\partial \underline{e}(k)}{\partial [A]_{ij,\ell}} \tag{A.11}$$

$$= \Sigma_\mu^{-1}(z) \frac{\partial \bar{A}}{\partial \alpha_{ij,\ell}} V_\mu(z)\underline{y}(k) - \Sigma_\mu^{-1}(z) \frac{\partial \bar{A}}{\partial \alpha_{ij,\ell}} W_\mu(z)B(b,d)\underline{u}(k)$$

$$= -\Sigma_{\mu}^{-1}(z)\begin{bmatrix} & z^{\ell-1} & \end{bmatrix}\underline{y}(k) +$$

↑ row i
↑ column j

$$+\Sigma_{\mu}^{-1}(z)\begin{bmatrix} z^{\ell-1} & \dots & z & 1 \end{bmatrix}B(b,d)\underline{u}(k)$$

↑ row i
↑ column $\mu_1+..+\mu_{\ell-1}+\ell$

so $$\frac{\partial e_n(k)}{\partial \alpha_{ij,\ell}} = 0 \quad \text{for } n \neq i \tag{A.12}$$

and $$\frac{\partial e_i(k)}{\partial \alpha_{ij,\ell}} = -y_j(k-\mu_i+\ell-1) + \left(\underline{d}_j^T z^{-\mu_i+\ell-1} + \underline{b}_{j,1}^T z^{-\mu_i+\ell-2} + .. \right.$$

$$\left. \dots + \underline{b}_{j,\ell-1}^T \right)\underline{u}(k)$$

Derivation to the b- and d-parameters yields:

$$\frac{\partial \underline{e}(k)}{\partial [B]_{\ell,sj}} = -\Sigma_{\mu}^{-1}(z)\overline{A}(\alpha)W_{\mu}(z)\begin{bmatrix} & 1 & \end{bmatrix}\underline{u}(k) \tag{A.13}$$

↑ row $\mu_1+\dots+\mu_{\ell-1}+\ell-1+s$
↑ column j

$$= -\Sigma_{\mu}^{-1}(z)\overline{A}(\alpha)\begin{bmatrix} 1 \\ z \\ \vdots \\ z^{\mu_\ell+1-s} \end{bmatrix}\underline{u}(k) \tag{A.14}$$

← row $\mu_1+..+\mu_{\ell-1}+s$

↑ column j

So
$$\frac{\partial e_i(k)}{\partial [B]_{\ell,sj}} = \left(\alpha_{i\ell,s} z^{-\mu_i} + \ldots + \alpha_{i\ell,\mu_\ell} z^{-\mu_i+\mu_\ell-m} \right) u_j(k) \quad \text{for } i \neq \ell \tag{A.15}$$

$$\frac{\partial e_\ell(k)}{\partial [B]_{\ell,sj}} = \left(\alpha_{ii,s} z^{-\mu_i} + \ldots + \alpha_{ii,\mu_i} z^{-m} - z^{-m+1} \right) u_j(k) \tag{A.16}$$

The derivative of the prediction error can be found by solving

$$\frac{\partial \underline{e}(k)}{\partial \theta} = R(\alpha) \frac{\partial \underline{e}_p(k)}{\partial \theta} \qquad (\theta \text{ no } \alpha\text{-parameter}) \tag{A.17}$$

$$\frac{\partial \underline{e}(k)}{\partial \alpha} = \frac{\partial R(\alpha)}{\partial \alpha} \underline{e}_p(k) + R(\alpha) \frac{\partial \underline{e}_p(k)}{\partial \alpha} \tag{A.18}$$

These expressions can easily be implemented on a computer.

A.2 Output Error

To derive an expression for the output error as a function of the model parameters and in- and output samples, we have to eliminate the state vector in the output error model.

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{e}(k) \end{aligned} \tag{A.19}$$

This can be written as (see sect. 2.1)

$$x_{i,1} = y_i - e_i - \underline{d}_i^T \underline{u} \tag{A.19-a}$$

$$x_{i,2} = zy_i - ze_i - z\underline{d}_i^T \underline{u} - \underline{b}_{i,1}^T \underline{u}$$

$$x_{i,j} = z^{j-1} y_i - z^{j-1} e_i - z^{j-1} \underline{d}_i^T \underline{u} - z^{j-2} \underline{b}_{i,1}^T + \ldots + \underline{b}_{i,j-1}^T \underline{u}$$

$$x_{i,\mu_i} = z^{\mu_i-1} y_i - z^{\mu_i-1} e_i - z^{\mu_i-1} \underline{d}_i^T \underline{u} - z^{\mu_i-2} \underline{b}_{i,1}^T \underline{u} + \ldots + \underline{b}_{i,\mu_i-1}^T \underline{u}$$

$$zx_{i,\mu_i} = \underline{\alpha}_i^T \underline{x} + \underline{b}_{i,\mu_i}^T \underline{u} \qquad (i=1,\ldots,q) \tag{A.19-b}$$

Similar to sect.2.1 we insert eq.(A.19-a) into eq.(A.19-b) and introduce the matrices $V_{\mu-1}(z)$, $W_{\mu-1}(z)$ and $B'(b,d)$ (see eq.(2.4), (2.5) and (2.7) and (A.20-a)).

If only the error is shifted to the left hand side, eq.(A.19-b) becomes

$$z^{\mu_i} e_i(k) = \underline{\alpha}^T \left[V_{\mu-1}(z)\underline{e} + W_{\mu-1}(z)B'(b,d)\underline{u} - V_{\mu-1}(z)\underline{y} \right] + z^{\mu_i} y_i - \left[z^{\mu_i}, \ldots, z, 1 \right] \begin{bmatrix} \underline{d}_i^T \\ \underline{b}_{i,1}^T \\ \vdots \\ \underline{b}_{i,\mu_i}^T \end{bmatrix} \qquad (A.20)$$

From this expression we see that the best way to calculate the error recursively is to calculate

$$\Sigma_\mu(z)\underline{e}(k) = \left[e_1(k+\mu_1), e_2(k+\mu_2), \ldots, e_q(k+\mu_q) \right]^T \qquad (A.21)$$

for every sample moment. In this way the error ($\Sigma(z)e\_(k)$) can be calculated recursively.

Using all previously used notations, eq.(A.20) becomes:

$$e_i(k+\mu_i) = \sum_{j=1}^{q} \sum_{\ell=1}^{\mu_q} \left[\alpha_{ij,\ell} \left(e_j(k+\ell-1) - y_j(k+\ell-1) + \sum_{v=1}^{\ell} \sum_{w=1}^{p} [B]_{j,vw} u_w(k+\ell-v) \right) \right] + y_i(k+\mu_i) - \sum_{v=1}^{\mu_i+1} \sum_{w=1}^{p} [B]_{i,vw} u_w(k+\mu_i-v+1) \qquad (A.22)$$

This is a suitable expression to implement. Because this calculation is recursive, one should be aware of mistakes if the initial conditions can not be disgarded.

A.3 Output Error and Innovation Error

To calculate the error and its derivatives for the OEM and IEM, for given values of the parameters, we do not have to eliminate the state vector. By calculating the state vector and its derivative recursively, the error

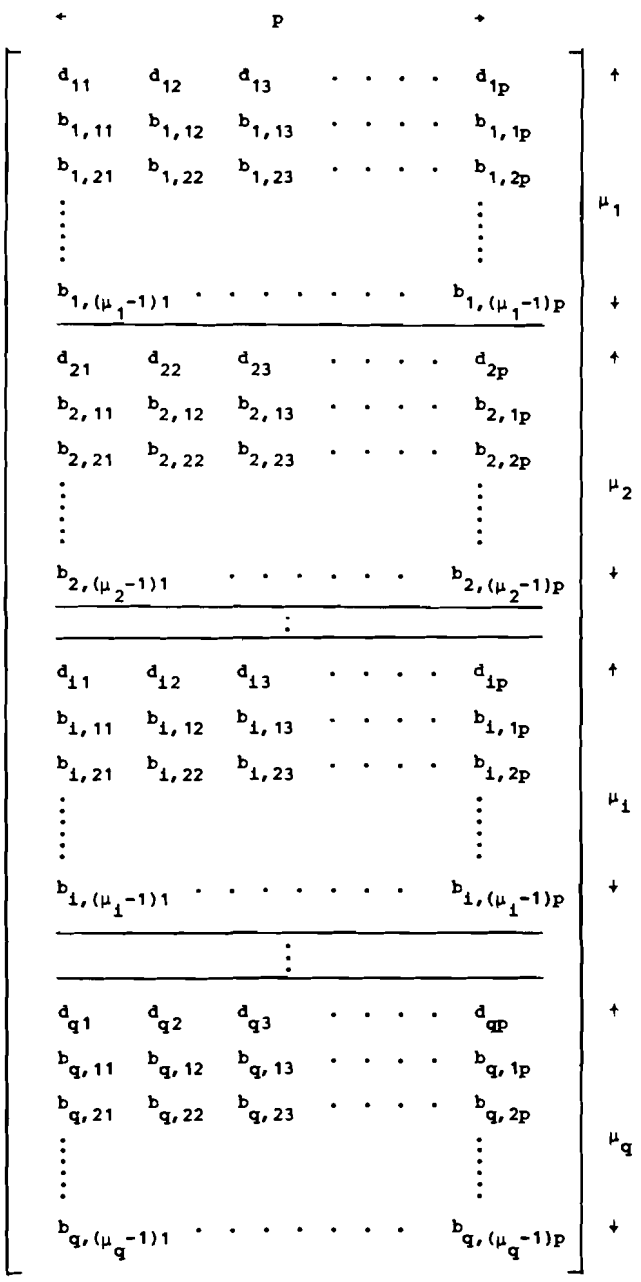


Fig. A.20-a Matrix B'(b,d).

and its derivative can be calculated recursively. This method is less complex then the one shown in sect. A.2.
Also the initial state and the offset vector on the output will be considered below as parameters.
The model becomes:

$$\begin{aligned} \underline{x}(k+1) &= A\underline{x}(k) + B\underline{u}(k) + K\underline{e}(k) \\ \underline{y}(k) &= C\underline{x}(k) + D\underline{u}(k) + \underline{y}_{off} + \underline{e}(k) \\ \underline{x}(0) &= \underline{x}_0 \end{aligned} \tag{A.23}$$

For the OEM K equals zero.
For given values of the parameters and a given data set, $\underline{e}(k)$ and $\underline{x}(k)$ can be calculated recursively by solving (A.23) for every sample moment.
The derivatives can also be calculated recursively by solving

$$\frac{\partial \underline{x}(0)}{\partial \theta} = \underline{0} \qquad (\theta \text{ not an initial state element}) \tag{A.24}$$

$$= [0, \ldots, 0, \underset{\uparrow \text{ row i}}{1}, 0, \ldots, 0] \quad (\text{if } \theta \text{ is a state element})$$

$$\frac{\partial \underline{e}(k)}{\partial \theta} = -C \frac{\partial \underline{x}(k)}{\partial \theta} - \frac{\partial D}{\partial \theta}\underline{u}(k) - \frac{\partial \underline{y}_{off}}{\partial \theta} \tag{A.25}$$

$$\frac{\partial \underline{x}(k+1)}{\partial \theta} = \frac{\partial A}{\partial \theta}\underline{x}(k) + A\frac{\partial \underline{x}(k)}{\partial \theta} + \frac{\partial B}{\partial \theta}\underline{u}(k) + \frac{\partial K}{\partial \theta}\underline{e}(k) + K\frac{\partial \underline{e}(k)}{\partial \theta} \tag{A.26}$$

So given a parameter set and a $\underline{x}(k)$, we have to calculate every sample moment

$$\underline{e}(k)\ ,\ \partial \underline{e}(k)/\partial\theta\ ,\ \underline{x}(k+1) \text{ and } \partial \underline{x}(k+1)/\partial\theta.$$

This is a very straight forward way to calculate the error and its derivative for a SSM.

REFERENCES

Åström K.J., T. Bohlin, 1965, NUMERICAL IDENTIFICATION OF LINEAR DYNAMICS SYSTEMS FROM NORMAL OPERATING RECORDS, Proc. IFAC SymP. on Self Adaptive Control Systems, Teddington, U.K., Ed. P.H. Hammond, p96-111.

Åström K.J., 1980, MAXIMUM LIKELIHOOD AND PREDICTION ERROR METHODS, Automatica, vol.16, p551-574.

Beckers F., 1985, STRUCTURAL AND PARAMETRIC IDENTIFICATION OF MULTI-VARIABLE SYSTEMS REPRESENTED BY MATRIX FRACTION DESCRIPTION, Masters Thesis, Eindhoven University of Technology.

Boom A.J.W. van den, 1982, SYSTEM IDENTIFICATION: On the variety and coherence in parameter- and order estimation methods, Dissertation. THE .

Boom A.J.W. van den, R. Bollen, 1983, THE INTER-ACTIVE PROGRAM PACKAGE SATER part 1: Description of the SATER system, Internal Report.

Damen A.H., Y. Tomita, P.M.J. van de Hof, 1985, EQUATION ERROR VERSUS OUTPUT ERROR METHOD IN SYSTEM IDENTIFICATION, Internal Report ER 85/06.

Elgerd O.I., 1967, CONTROL SYSTEMS THEORY, McGraw-Hill, Tokyo,Ltd.

El-Sherief H.E., 1984, STATE AND PARAMETER ESTIMATION OF LINEAR STOCHASTIC MULTIVARIABLE SAMPLED DATA SYSTEMS, Trans. on SMC,vol.14,no.6, p911-919. IEEE

Eykhoff P. et al, 1970, THEORY AND APPLICATION OF THE KALMAN FILTER (Dutch), Internal course.

Eykhoff P., 1974, SYSTEM IDENTIFICATION: Parameter and State Estimation, J.Wiley en Sons,N.Y.

Eykhoff P, A.J.W. van den Boom, A.A.H. Damen, P.M.J. Van den Hof, P.H.M. Janssen, 1985, STOCHASTIC SYSTEM THEORY; Identification, Internal course.

Fletcher R., C.M. Reeves, 1964, FUNCTION MINIMISATION BY CONJUGATE GRADIENTS.

Guidorzi R.P., 1975, CANONICAL STRUCTURES IN THE IDENTIFICATION OF MULTI-VARIABLE SYSTEMS, Automatica, vol.11, p361-374.

Guidorzi R.P., 1981, INVARIANTS AND CANONICAL FORMS FOR SYSTEMS STRUCTURAL AND PARAMETRIC IDENTIFICATION, Automatica, vol.17, no.1, p117-133.

Guidorzi R.P., 1982, MULTI-STRUCTURAL MODEL SELECTION, Proc. 4-th European Meeting on Cybernetics and System Research, Wien.

Guidorzi R.P., S. Beghelli, 1982, INPUT-OUTPUT MULTI-STRUCTURAL MODELS IN MULTIVARIABLE SYSTEMS IDENTIFICATION, Proc. 4-th IFAC Symp., p359-543.

Hajdasinski A.K., 1980, LINEAR MULTIVARIABLE SYSTEMS: Preliminary Problems in Mathematical Descriptions, Modelling and Identification, TH-Report 80-E-106.

Hajdasinski A.K., 1983, DESCRIPTION OF MULTI-VARIABLE DYNAMIC SYSTEMS BY MEANS OF CANONICAL FORMS; Unique Parametrisation problems, Internal course.

Ljung L., Söderström, 1983, THEORY AND PRACTICE OF RECURSIVE IDENTIFICATION.

Meertens I.B., 1983, STRUCTURAL AND PARAMETRIC IDENTIFICATION OF MIMO-SYSTEMS ACCORDING TO GUIDORZI. Further developments and applications, Master thesis, Dept. of Elect. Eng, Eindhoven University of technology.

Overbeek A.J.M. van, L. Ljung, 1982, ON-LINE STRUCTURE SELECTION FOR MULTIVARIABLE STATE SPACE MODELS, Automatica, vol.18, no.5, p52-543.

Pfennings J., 1983, MULTISTRUCTURAL IDENTIFICATION (overlapping parametrisations), Master thesis, Dept. of Elect. Eng., Eindhoven University of Technology.

Ploemen E.M.M., 1982, USERS MANUAL PLOTPROGRAM GRAFOI (Dutch), Internal report.

Strmcnik S., F. Bremsak, 1985, AN EFFICIENT ALGORITHM FOR THE TRANSFORMATION OF THE INPUT-OUTPUT MODEL INTO STATE SPACE MODEL, Int. J. Control, vol.41, no.1, p145-154.

Weast R.C., S.M. Selly, 1967, HANDBOOK OF TABLES FOR MATHEMATICS, Cleveland (Ohio), The Chemical Rubber Co.

Zee G.A. van, O.H. Bosgra, 1982, GRADIENT COMPUTATION IN PREDICTION IDENTIFICATION OF LINEAR DISCRETE-TIME SYSTEMS, IEEE Trans. ~~on AC~~, vol.27, no.3, june 1982. Autom. Control

4964

DEPARTMENT OF ELECTROTECHNICAL ENGINEERING
EINDHOVEN UNIVERSITY OF TECHNOLOGY
Group Measurement and Control

PARAMETER ESTIMATION OF MULTI-VARIABLE PROCESSES REPRESENTED BY STOCHASTIC MODELS IN PSEUDO-CANONICAL FORM; Manual Program LS_SSM

by A.J.M. Veltmeijer

This report is submitted in partial fulfillment of the requirements for the degree of electrotechnical engineer (M.Sc.) at the Eindhoven University of Technology.
The work was carried out from sept.1984 untill aug.1985 in charge of Prof.dr.ir. P. Eykhoff and under supervision of Dr.ir. A.A.H. Damen and Ir. P.M.J. Van den Hof.

Contents

Introduction

This manual describes the program LS_SSM. This program estimates a state space model (SSM) in (pseudo-)canonical form by means of the least squares (LS) criterion (this LS-criterion should not be mixed up with the ordinary LS estimation of I/O-models).
This program minimises the sum of squares of the simulation error or several different prediction errors.
Because 4 different stochastic models can be estimated and many options are added to choose the best possible estimation condition and to show the results in the best possible way, the program is a very flexible and powerful tool for practical parameter estimation as well as for educative aspects of parameter estimation.
For the theoretical aspects the reader is referred to the report 'parameter estimation of multi-variable processes represented by stochastic models in pseudo-canonical form'.
In the first chapter of this manual aspects are treated which only concern those users that are interested in practical estimation. Those users that are interested in the details of the implemented estimation program are referred to chapter 2.

1.0 The estimation program LS_SSM.

The main parts of the program LS_SSM are given in the next flowdiagram.

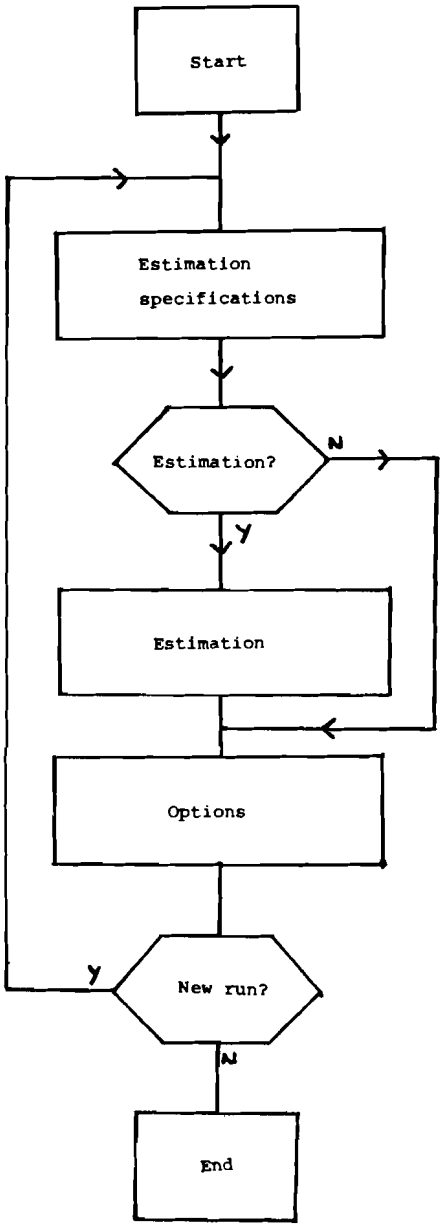


Figure 1.1

In the next 3 sections the blocks 'estimation specifications', 'estimation' and 'options' will be described in detail.

1.1 Estimation specification

After the program is started the first question will be:

```
*************************************************************
*                                                           *
* This program enables you to estimate an EEM, OEM, PEM     *
* or IEM in (pseudo-)canonical state space representa-      *
* tion. Several hill-climbing methods can be used.          *
* If only the option are wanted for the inserted model      *
* and data, the estimation can be skipped.                  *   (1.2)
*                                                           *
*************************************************************


Do you want all implemented  DEFAULT  ANSWERS ?
(All defaults will be  shown) [Y/N] :
```

Dependent on the answer many other questions are asked (N) or only 4 questions are asked ((Y), (1.3), (1.5-a) and (1.6)), before all inserted answers and/or default answers are shown. Then specific questions can be generated again by the user, and the answers can be changed.
The fastest way to pass this 'estimation specification' block is to answer Y(es) on the question above and to change afterwards the answers you do not want to be default.
All questions will be described in detail. The program is protected against impossible answers.

```
Enter the FILENAME for the start/initial  values of the PARAMETERS
or **** if you  want to enter the initial values
                                                              (1.3)
FILENAME : ****
```

To be able to choose a random model as initial guess of the parameters the model can be inserted inter-actively (be aware to insert a pseudo-canonical structure). Default names for the matrices are A, B, C, D, K and OFF for the system-, distribution-, output-, input-output-, noise distribution and offset matrices.

```
Enter structure of the file :
   [1] : THE-structure
   [2] : MATLAB-structure (default)
                                                              (1.4)
Answer : 2
```

MATLAB-structure is implemented because matrices put in a file with

MATLAB-structure can be used by the inter-active program package MATLAB, which is mainly made for matrix calculations.
The THE-structure is the standard structure in the group 'Measurement and Control' and other groups at the 'Technische Hogeschool Eindhoven'.

```
Enter the FILENAME of I/O-DATA or **** if you want
to enter the data.                                          (1.5-a)

FILENAME : ****
```

If the I/O-data (see previous question for structure choice) is read inter-actively, then standard functions can be given as sin, cos, step, impulse, etc.

```
Any special function for input no.:  1

  [0] : NO
  [1] : Zero
  [2] : Impuls (unit weight)                                (1.5-b)
  [3] : Step   (unit weight)
  [4] : Sinus(K/5 rad.)
  [5] : Cosinus(K/5 rad.)
  [6] : Arctan(K/5)

Answer :
```

If no initial state is estimated then the error calculation can be done from an earlier sample moment then the first sample moment that contributes to the loss function (1.6 ι). Default both sample moments are equal. The first possible sample moment is zero.

```
Enter start sample for the minimisation : 0
                                                            (1.6)
Enter start sample for the error calculations : 0

Enter number of samples for the minimisation :
```

If the answers to the questions above are STS,STSFC and NSA then the sample moments read are STSFC till STSFC+NSA+IN-1, because the equation error may be a function of future I/O-samples (see report).

```
Which MODEL do you want to estimate ?
   [0] : No estimation (default)
   [1] : Equation error model
   [2] : Output error model                                 (1.7)
   [3] : Prediction error model (Can. form)
   [4] : Innovation error model

Answer : 0
```

'No estimation' is implemented (1.7) to be able to make plotable and

graphic files with the initial model (see options). Only for canonical forms the prediction error minimisation is implemented (see report).

```
Which UNIQUE FORM do you want to estimate ?
   [0] : Pseudo-canonical form (default
   [1] : Canonical form                                    (1.8)

Answer : 0
```

Canonical estimation may include estimation of less parameters but with a bigger chance to estimate the wrong form.

```
Do you want to estimate a
   [0] : PROPER model (default)
   [1] : STRICTLY PROPER model (D=0)                       (1.9)

Answer : 0
```

Proper estimation includes esimation of qxp extra parameters for direct coupling of in- and outputs.

```
Enter choice of INITIAL STATE estimation.
   [0] : No estimation. (Default value =0)
   [1] : Full estimation
   [2] : Estimation of a parameters
         (for the a subsystems)
   [3] : No estimation. Values dependent on  constant
         signal value for k<0 (will be asked).
   [4] : No estimation. Values will be asked.
                                                           (1.10)
Answer : 0

Enter choice in OFFSET estimation.
   [0] : No estimation. Values already entered. (default)
   [1] : estimation
   [2] : No estimation. Values dependent on  constant
         signal values for k<0 (will be asked)
   [3] : No estimation. Values are asked.

Answer : 0
```

If the system is at rest for sample moments previous to those used for the estimation, then only q independent parameters have to be estimated for the initial state.

Initial state and offset (fixed values) can be inserted. The initial values for the initial state parameters will be asked if the initial state is estimated (not read from a file).
If the system is at rest for sample moments previous to those used for the estimation, and the initial state and offset will not be estimated, the extra information (system at rest) can be used for calculation of the initial state and the offset if the constant in- and outputs are known. These constant in- and outputs will be asked.

```
Enter wanted MEAN VALUE CORRECTION
   [0] : No Correction (default)
   [1] : Input Mean Value Correction
   [2] : Output Mean Value Correction
   [3] : In- and Output Mean Value Correction                (1.11-a)

Answer : 0
```

```
Enter wanted signal SCALING FACTOR.
   [0] : Automatic scaling of initial loss function.
         All scaling factors equal.
   [1] : No scaling (default).
   [2] : Full automatic scaling.
         All powers of in- and output equal.                 (1.11-b)
   [3] : Own scaling of initial loss function.
   [4] : Own full scaling.

Answer : 1
```

Not all combination of initial state estimation, offset estimation, mean value correction and signal scaling are implemented and/or possible (program will give warnings, and will not permit forbidden combinations).
Input mean value correction may include correction of the initial values of the offset and initial state, and inserted constant values of in- and outputs for k<0. Scaling may also include scaling of initial values of the parameters and fixed variables.
The preprocessing and signal/model recovery (mean value correction and scaling) will be done in the block 'estimation' without knowledge of the user.
The initial loss function is the value of the loss function of the inserted initial guess of the model. This value must be given if the

answer to question (1.11-b) is 3. All scaling factors have to be given if 'own full scaling' is chosen.
Full automatic scaling includes scaling such that the initial loss function has a specific value and all outputs have equal energy (the covariance matrix of the noise will be changed by scaling).

```
Enter wanted HILL-CLIMBING method
   [1] : Conjugate Gradient (default).
   [2] : Modified Newton                                    (1.12)
   [3] : Quasi-Newton

Answer : 1
```

Conjugate gradient method is recommended for EEM and PEM estimation ('not very non-linear'). Modified Newton and quasi Newton are much slower algorithms but must be used for the OEM and IEM estimation. If one of the two Newton methods stops, the other one should be used in trying to find a possible lower minimum.

After all these questions answered (default or not) an overview over all given answers is given. For understanding of the abbreviations between brackets, an example is given below in case of default answers and initial model and I/O-data inserted inter-actively.

```
Enter the number if you want to CHANGE the ENTRY.
   [1] : Filename initial spec          : ****
       : File structure                 : 0
   [2] : Filename I/O-sample            : ****
       : File structure                 : 2
   [3] : Start sample calculations      :     0
       : Start sample minimisation      :     0
       : Number of samples for estimation :   10        (1.13)
   [4] : Wanted model. (No/Equ/Out/Pre/Inn): 0
   [5] : Canonical estimation          (N/Y): 1
   [6] : Strictly proper model         (N/Y): 0
   [7] : X0. (DefX0/Full/0-par/OwnI0/OwnX0): 0
       : Offset. (DefOFF/Yes/OwnI0/OwnOFF): 0
   [8] : Mean Value Corr.        (No/I/O/IO): 0
   [9] : Scaling.(DefLF,No,FAut,OwnLF,FOwn): 1
  [10] : Hill-climbing.          (CG/MN/QN): 1
   [0] : NO CHANGES

Answer :
```

If an answer must be corrected, the number has to be inserted and the specific question will appear again. If no changes are wanted he program continues with the 'estimation' or 'option' block (see fig.1.1).

1.2 Estimation

The stimation is split up in a preprocessing part, the estimation part and a signal/model recovery part (see fig.1.14).

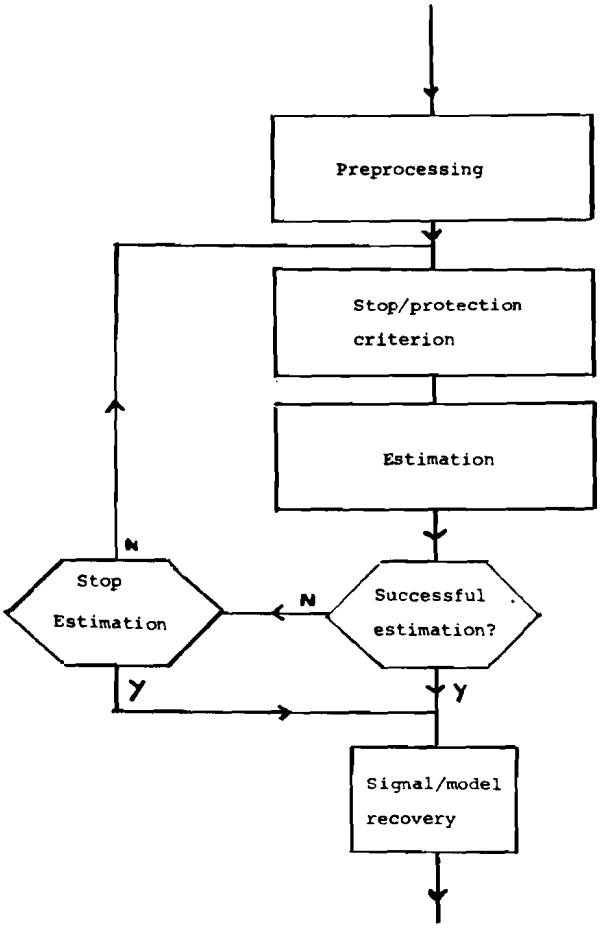


Fig. 1.14.

1.2.1 Preprocessing / Signal and model recovery

Before the estimation is started the I/O-data and model has to be scaled as given by the 'estimation specification' (section 1.1). Also the I/O-data and possibly the initial state and offset have to be corrected with mean values derived from the I/O-data.
All scaling and mean value correction is done such that the corrected initial model has the same I/O behaviour (for corrected I/O-data) as the inserted model. After estimation the signals and estimated model have to be rescaled and recorrected with the mean values.

1.2.2 Stop/protection criteria

After preprocessing the next question will appear:

```
Stop criterium is :  0.10000D-04
Overflow value is :  0.10000D+16
Maximum number of calls is :  1100

Type new value or ZERO in case of NO CHANGE
Stop criterium :
Overflow value for errors :
Number of calls :
```
(1.15)

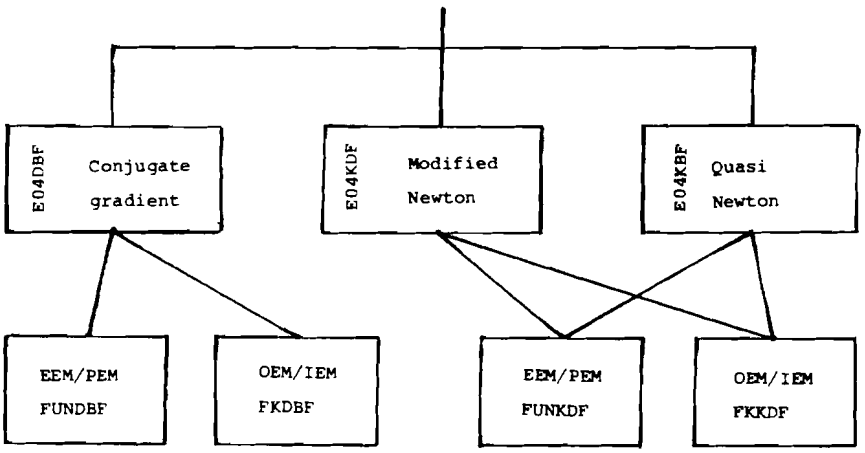
The maximum number of calls is the maximum number of times that the NAG-routine (see next section) calls the subroutine that calculates the loss function and/or the derivative of the loss function to the parameters. Default value is 100x(number of parameters).

The stopcriterion value (default 1.0E-5) is a lower bound for the decrease of the loss function. If the relative decrease of the loss function is less than this value, the NAG-routine immediatly stops running and returns to the main program (for the Newton methods). For the conjugate gradient method the loss function and the derivatives are set to zero so that the NAG-routines detects a minimum and stops the estimation. For both cases the loss function and parameter values are saved in a common block.
A maximum value for the model error is built in to prevent the program from stopping by an overflow without saving the results. During estimation of the error and its derivatives, every time a multiplication is done, the values are compared with the overflow value (default 1.0E+15) . If the absolute value of the calculation is larger than the overflow value, then the result of the multiplication is set to the overflow value (sign not changed). Consequently the biggest possible loss function (and derivative) is (1.0E+30x(number of samples)x(number of outputs)).

In spite of these protections the NAG-routine may stop at an overflow in NAG-subroutines called by the main NAG-routine. Dependent at the problem at hand the user may change the 3 mentioned default values. If no changes are wanted 3x <CR> (read as 0.0E+00) is the fastest way to pass this block.

1.2.3 Estimation

Dependent on the wanted hill climbing method one of the 3 implemented NAG-rouitnes is called. For every method a distinction is made between the EEM, PEM and OEM, IEM estimation, because of the totally different ways of error and derivative calculations in the two cases. The subroutines that calculate the loss function and/or derivative for the Newton methods are the same for both hill-climbing methods. This can be draw in a flow diagram as shown in the next diagram:



(see chapter 2 for subroutine names and descriptions)

Figure 1.16.

If the estimation is not finished, so due to some criterion it stopped, then the next question is asked

(1.17)

and the estimation can be done again with new stop criteria.

1.3 Options.

If the estimation is done succesfully, is stopped due to some protection criterion without continuation afterwards, or if the estimation is skipped, then the next question will appear:

```
Enter wanted OPTION.
   [1] : Store model.
   [2] : Store entries and results in  plotable file.
         (prediction error calculations only valid
          for canonical forms)
   [3] : Generate graphic files.                          (1.18)
         (prediction error calculations only valid
          for canonical forms)
   [0] : EXIT.

Answer :
```

All filenames of files that are generated will be given. Filenames are generated automatically based on the name of the filename with the initial model, filename of the I/O-data or a new name (FOWNET, for a 3-th order SISO model) based on a lenght of 9 and the structural indices. For OEM, EEM, PEM and IEM the first variable will be an O, E, P and I. The first variable for a new model will be F.

If the anser to the question above is 1, the next question will appear:

```
Which MODEL.
   [1] : Initial model.
   [2] : Estimated model.
   [3] : Initial model without K but inclusive          (1.19)
         distribution filter for equation model

Answer : 1
```

The initial model has those parameters as given for the initial guess of the model parameters before the estimation. Also for the estimated model matrices are stored even if they are not estimated (OFF and K).

If a EEM or PEM is estimated then K(a) (see section 3.2 of report) is stored. The reader should be carefull with the use of this matrix. It might have no validity for pseudo-canonical forms. If K(a) could not be calculated K is set to 0. The program MATLAB might be able to calculate K(a) nevertheless (see report for calculation).

If the initial model does not contain K(a) but a stored model is wanted with K(a), then the answer to the question must be 3.

If the answer to question (1.19) is 2, then a plotable file (can be read by $TYPE <filename>) is made with all estimation conditions, all model

errors for the initial and estimated model (see example). This file ('... .PLO") is the only way to store the esimated initial state value.

If files are wanted that are suitable for the plotprogram GRA then again a choice must be made between the initial or estimated model and all possibilities will appear.

```
Enter wanted graphic file :
   [1] : Initial state response (200 samp.).
   [2] : Impuls response (200 samp.).
   [3] : Step response (200 samp.).
   [4] : Simulated output over minimisation  interval
   [5] : Predicted output over minimisation  interval          (1.20)
   [6] : Innovated output over minimisation  interval
   [7] : Equation, output, prediction error  and innovation error
         over minimisation interval
   [8] : I/O-data used for minimisation interval
   [0] : EXIT

Answer :
```

The first variable in all graphic files is the sample moment that starts at 0.
For the initial state response, the inputs are ignored and the responses on the initial state are stored as next variable (in file '... .GIS', 200 samples).
Also the impulse or step responses can be stored (in '... .GIM' and '... .GST', 200 samples). The first q variables are the are the responses to an impulse/step on input 1. The next q variables (q+2 to 2xq+1) are the responses to an impulse on input 2, etc.
For file number 4 ('... .GOS') variables 2 to q+1 are the simulated outputs. For file number 5 ('... .GOP') variables 2 to q+1 are the predicted outputs (K=K(α) is only valid for canonical forms), and for file number 6 ('... .GOI') variables 2 to q+1 are the innovated outputs with K equal to the k-parameters of the model. If file number 7 ('... .GER') variable 2 to q+1 are the equation errors, variable q+2 to 2xq+1 are the output errors, variable 2xq+2 to 3xq+1 are the prediction errors (only valid for canonical forms) and variable 3xq+2 to 4xq+1 are the innovation errors (equal to output errors if K=0). For file number 8 ('... .GIO') variable 2 to p+1 are the inputs and variable p+2 to p+q+1 are the outputs used for the estimation. If 0 is answered, question (1.19) will appear again.

2.0 Implementation of program LS_SSM

The program LS_SSM is implemented on a VAX computer system. Except for the 'include' statement, every routine is written in standard fortran 77. If in a fortran file the next statement is written:

```
INCLUDE '<filename>'
```

then after compilation the file <filename> is included in the program at that specific place.
Common blocks are used because it is impossible to transfer all information (I/O-data and estimation specifications) through the NAG-routines to subroutines (called by a NAG-routine) that calculates the loss function and derivatives (for the structure of the implementation of NAG-routines and subroutines called by the NAG-routines, we refer to figure 1.14)

In the fortran files of the program every step is explained, so that in this manual we will only explain all variable names (sect.2.1), file structures (sect.2.2) and briefly all subroutines.

2.1 Used variables

Between brackets: P = Parameter (of program)
I = Integer
R = Real
D = Double precision
Cn = Character string with lenght n.
NAG = see NAG-library routine description

A(MINP,MINP) : Sytem matrix (D)
AHULP(MINP,MINP): System matrix (D)
B(MINP,MIPP) : Distribution matrix (D)

C(MIQP,MINP) : Output matrix (D)

C1ANS : Answer to question, Y or N (C1)

D(MIQP,MIPP) : Input-output matrix (D)

DDUMV1/2(MIPP+MIQP) : Dummy matrices (D)

DDUMV3/4/5(MINP) : Dummy matrices (D)

DELTA : Difference interval (NAG) (D)

DERIVA(MINP,MIQP,MDPARP) : Derivatives of model errors (D)

DOVERFL : Overflow value for model errors an derivatives (D)

DPAR : NUmber of parameters (I)

DSTCRI : Relative lower bound for dercrease of loss function (D)

FC(N) : Parameter vector (NAG) (D)

FCHLP(MDPARP) : Parameter vector stored in common block (D)

FDBF : Loss function value (NAG (F)) (D)

FILINI : filename initial model (C40)

FILIO : Filename I/O-data (C40)

FILMOD : Filename estimated model (C40)

GC(N) : Derivatives of loss function (NAG) (D)

GDFB(MDPARP) : Derivatives of loss function (NAG (G)) (D)

HDPAR : Hulp integer for temporarily valid number of parameters (I)

HESL(LH) : (NAG) (D)

IBOUND(N) : (NAG) (I)

ICORRV(MIPP+MIQP) : Vector flag for I/O-data and model mean value correction (I)

ICAN : Canonical flag (I)

IDEF : Default flag (I)

IERR : MOdel flag (I)

IFAIL : Failure flag (I)

IFINI : Filstructure for file FILINI (I)

IFIO FILIO (I)

IFMOD FILMOD (I)

IMP : Special function for THE-file (P)

IN : Dimension of model (I)

IP : Number of inputs (I)

IPRINT : Flag for routines MONDBF and MONKDF (NAG) (I)

IQ : Number of outputs (I)

ISTATE : State estimation flag (I)

ISTATV(MDPARP) : (NAG (ISTATE) (D)

ISTRPR : Strictly proper estimation flag (I)

IW(2) : (NAG) (I)

KF : STS+1-STSFC or 1 dependent on used routine (I)

KL : STS+NSA-STSFC (I)

KMAT(MINP,MIQP) : NOise distribution matrix (D)

LH : Lenght of HESL (I)

LIW : Lenght of IW (I)

LOCSCH : (NAG) (C1)

LOSSF : Value of loss function before estimation (D)

LW : Lenght of W (I)

MAXCAL : MAximum number of loss function calculations (I)

MDPARP : Maximum number of parameters (P)

MINP : MAximum dimension of model (P)

MIPP : MAximum number of inputs (P)

MIQP : Maximum number of outputs (P)

MEAN(MIPP+MIQP) : Mean values (D)

MEASURE : Indication of I/O-data file for THE structure (P)

MODNAM : Model name (THE-file) (C8)

MUHLP(MIQP) : Structure indices of model in PARHLP (I)

MUINI : PARINI (I)

MUMOD : PARMOD (I)

MSAMPP : Maximum number of samples (P)

N : Number of parameters (NAG) (I)

NIN : Unit number for reading from terminal (P)

NITYPE : Number of possible filetypes (THE-file) (P)

NOI : Noise indicator for special function (THE-file) (P)

NOUT : Unit number for writing to terminal (P)

NSA : Number of samples for estimation (I)

NSPFO : Number of special forms defined (THE-file) (P)

NSPFU : Number of special functions defined (THE-file) (P)

NSUBT : Number of special models defined (THE-file) (P)

```
OFF(MIQP)          : Offset vector (D)
OFFMOD(MIQP)       : Offset vector stored in a common block (D)
PARHLP(MDPARP)     : Parameter vector for temporaly storage (D)
PARINI(MDPARP)     :                       initial model (D)
PARMOD(MDPARP)     :                       estimated model (D)
RAM                : Special function (THE-file): ramp (P)
PARSYS(MDPARP)     : Parameter vector for temporaly storage (D)
RESIDU(MSAMPP,MIQP) : MOdel errors. (sometimes also model simulation and
                      prediction) (D)
SCALE              : Scaling factor that scales loss function (D)
SCALEV(MIPP+MIQP)  : Scaling factor for I/O-data and model scaling (D)
SFDM               : Special function (THE-file): diagonal matrix (P)
SFFM               :                              ; full matrix (P)
SFLTM              :                              ; lower triangular matrix
                                                    (P)
SFNIA              :                              ; no information available
                                                    (P)
SPFORM             : Special form (THE-file) (I)
SS                 : Type of model (THE-file); State Space (P)
SSCAN              :                          ; SS canonical (P)
SSCANP             :                          ; SSCAN strictly Proper (P)
SSP                :                          ; SS strictly Proper (P)
SSPCAN             :                          ; SS Pseudo-canonical (P)
STA                : Special function (THE-file); step (P)
STDEV(MIPP+MIQP)   : Standard deviations (D)
STEPMX             : Maximal step between iterations (NAG) (D)
STS                : First sample moment that contributes to the loss
                                                          function (I)
STSFC              : First sample moment for error calculations (I)
SUBT               : Subtype of model (THE-file) (I)
SYSIO(MSAMPP,MIPP+MIQP) : I/O-data, sample moment STSFC to STS+NSA+IN
U0(MIPP)           : Fixed values for inputs for k<0 (D)
UN1/ ... /UN6      : Unit numbers (P)
W(LW)              : (NAG) (D)
WR                 : flag for writing model (I)
```

XCHLP(MDPARP) : Parameter vector stored in common block (D)

X0(MINP) : Initial state (D)

X0MOD(MINP) : Initial state stored in common block (D)

XTOL : Tolerance in estimation of parameters (NAG) (D)

XTOLV(MDPARP) : Tolerance vector for estimation of parameters (NAG)(D)

Y0(MIQP) : Fixed values for outputs for k<0 (D)

ZERO : Special function (THE-file); zero (P)

2.2 File structures

To come to a standard file structure a first step is made in standardisation in the 'Measurement and Control' groups of the physical and electrotechnical departments of the University of Technology Eindhoven. Because this standard is not common use yet, and because it is not fully described, the standard description (agreed sofar) is given in section 2.2.2.
Because the MATLAB-structure is a standard for the MATLAB program package, and because it is a very simple way of storing data, this structure is tested thoroughly the for the implementation in LS_SSM (section 2.2.1)

2.2.1 MATLAB-structure

Sample files (I/O-data file)

A I/O-file in MATLAB-structure is a sequential formatted file with the next sequence of variables and the next formats:

NSAMP,IP,IQ	FORMAT(I5,2I1)	
IP inputs	FORMAT(4Z18)	(1-st sample moment)
IQ outputs	FORMAT(4Z18)	(1-st sample moment)
IP inputs	FORMAT(4Z18)	(2-nd sample moment)
.....		
IQ outputs	FORMAT(4Z18)	(NSAMP-st sample moment)

Model files (Matrices files)

Also these files in MATLAB-structure are sequential and formatted. Every matrix is stored as:

NAME,IROW,ICOL	FORMAT(A4,3I4)	(matrix name, row and column dimension)
Element(1,1)...		

```
.....Element(IROW,1)          FORMAT(4Z18)
.........................................
Element(1,ICOL).....
......Element(IROW,ICOL)      FORMAT(4Z18)
```

2.2.2 THE-structure

All files with THE-structure have direct access and are unformatted. The standardisation accomplished sofar is:

```
C
C
C  SYSTEM MATRIX STORAGE
C  --------------------
C
C*****************************************************************************
C
C
C  BASE SECTION
C  ------------
C
C  RECORD
C  ------
C

C  1        NREC           (MAX) NUMBER OF DATARECORDS          (I2)
C                          IN ONE DATA SECTION
C           LREC           LENGTH OF EACH RECORD IN             (I2)
C                          UNITS OF 4 BYTES (LONGWORDS)
C           IREC           POINTER TO THE BEGIN RECORD OF       (I2)
C                          THE DATA SECTION
C
C  2        MREC           THE NUMBER OF VARIABLES IN ONE       (I2)
C                          RECORD
C           INFO           POINTER TO THE BEGIN RECORD OF       (I2)
C                          THE INFORMATION SECTION
C           DTYPE          DATATYPE OF THE VARIABLES IN EACH    (CHAR*2)
C                          DATA SECTION.
C                    I2    INTEGER*2
C                    I4    INTEGER*4
C                    R4    REAL*4
C                    D8    DOUBLE PRECISION
C                    C8    COMPLEX
C
C  3        NVAL           NOT OF INTEREST
C           EVTC           NOT OF INTEREST
C           FILMOD         NOT OF INTEREST
C           UPDATE         NOT OF INTEREST
C
C
```

```
C
C
C
C  INFO SECTION ( System Model File )
C  ----------------------------------
C
C  RECORD
C  ------
C
C  INFO       ITYPE      **  SYSTEM MODEL                              (I2)
C             SUBTYPE1   10  S.S  CANONICAL         (A,B,C,D)          (I2)
C                        11  S.S  PSEUDO CANONICAL  (A,B,C,D)
C                        12  S.S  CANONICAL         (A,B,C)
C                        13  S.S  PSEUDO CANONICAL  (A,B,C)
C                        14  S.S                    (F,G,H,K)
C                        15  S.S                    (F,G,H)
C                        20  M.F.D.                 (P,Q)
C                        21  M.F.D.                 (P,Q,C)
C                        30  IMPULS RESPONSE MODEL  (A1,A2,A3...)
C                        40  TRANSFER FUNCTION      (A,B)
C                        41  FACTORIZED TRANSFER    (a,b)
C                            FUNCTION
C             SUBTYPE2   **  FURTHER SPECIFICATION                     (I2)
C                            IS POSSIBLE
C
C  INFO+1     INEP           NOT OF INTEREST
C
C  INFO+2     TITLE                                                    (CHAR*8)
C
C  INFO+3     DATUM          (DDMMMJJ)                                 (CHAR*8)
C
C  INFO+4     TIJD           (HHMMSS)                                  (CHAR*8)
C
C  INFO+5     INUM           NUMBER OF N*M*K MATRICES                  (I2)
C
C
C
C  INFO+6     NAME           NAME OF MATRIX 1                          (CHAR*8)
C
C  INFO+7                    SIZE OF MATRIX 1
C             N              NUMBER OF ROWS OF MATRIX 1                (I2)
C             M              NUMBER OF COLUMNS OF MATRIX 1             (I2)
C             K              DEPTH OF MATRIX 1                         (I2)
C
C  INFO+8                    STORAGE METHOD OF MATRIX 1
C             METH           NUMBER OF VARIABLES                       (I2)
C                            STORED IN ONE RECORD
C             SPEC1          SPECIAL FORM                              (I2)
C                       0    NO INFORMATION AVAIBLE
C                       1    FULL MATRIX
C                       2    DIAGONAL MATRIX
C                       3    LOWER TRIANGULAR MATRIX
C             SPEC2          SPECIAL STORAGE METHOD                    (I2)
C                       1    YES
C                       0    NO
C
C  INFO+9     BEG1           NUMBER OF THE BEGIN RECORD                (I2)

C                             OF MATRIX 1
C                        .
C                        .
C                        .
C                        .
C                        .
C                        .
C
C   EXTRA INFO SECTIONS, LIKE THE INFO SECTIONS INFO+6 UNTIL
C   INFO+10, SHOULD BE INSERTED FOR EVERY MATRIX
C
C
C*************************************************************************
```

```
C
C
C   INFO SECTION ( I/O-Sample File )
C   --------------------------------
C
C   RECORD
C   ------
C
C   INFO        ITYPE   **      I/O Samples              (I2)
C
C   INFO+1      INFF            Not of interest          (I2)
C
C   INFO+2      TITLE                                    (CHAR*8)
C
C   INFO+3      DATUM           (DDMMMJJ)                (CHAR*8)
C
C   INFO+4      TIJD            (HHMMSS)                 (CHAR*8)
C
C   INFO+5      INUM            Number of sample moments (I2)
C               NIN             Number of inputs.        (I2)
C               NOUT            Number of outputs        (I2)
C
C   INFO+6      SPECFO          Special form input no.1 (I2)
C                       0       Zero
C                       1       Impuls
C                       2       Step
C                       3       Ramp
C                       4       White gaussian noise
C
C   INFO+7      ISPFO1          Information on special form of
C                               input no.1                   (DP)
C                               Impuls  : Hight
C                               Step    : Hight
C                               Ramp    : Derivative
C                               Noise   : Mean
C               ISPFO2          Information on special form of
C                               input no.1                   (DP)
C                               Noise   : Standard deviation
C                       .
C                       .
C                       .
C                       .
C
C
C   INFO+6      OUTNOI          Output noise on output no.1      (I2)
C   +2*NIN              0       No
C                       1       Yes
C
C   INFO+7      IOUTN1          Mean of output noise on


C   +2*NIN                      output no.1                  (DP)
C               IOUTN2          Standard deviation of output
C                               noise on output no.1         (DP)
C
C   INFO+8      SYSNOI          System noise on output no.1      (I2)
C   +2*NIN              0       No
C                       1       Yes
C
C   INFO+9      ISYSN1          Mean of system noise on
C                               output no.1                  (DP)
C               ISYSN2          Standard deviation of system
C                               noise on output no.1         (DP)
C                       .
C                       .
C                       .
C                       .
C*****************************************************************************
```

```
C
C   DATA SECTION (System Model File)
C   ---------------------------------
C
C
C   (BEGIN RECORD NUMBER MATRIX 1)          DATA
C                                             .
C                                             .
C                                             .
C   (BEGIN RECORD NUMBER MATRIX 2)          DATA
C                                             .
C                                             .
C                                             .
C                .
C                .
C                .
C                .
C                .
C   (BEGIN RECORD NUMBER MATRIX I)          DATA
C                                             .
C                                             .
C                                             .
C
C
C*****************************************************************************
C
C   DATA SECTION (I/O Sample File)
C   ------------------------------
C
C   RECORD
C   ------
C
C   IREC                : Inputs and outputs of sample moment 0.
C
C   IREC+1              : Inputs and outputs of sample moment 1.
C
C                .
C                .
C                .
C
C   IREC+NREC-1         : Inputs and outputs of sample moment NREC-1.
C
C
C*****************************************************************************
C
C
C  STORAGE METHOD AND RECORD LENGTH DEFINITION
C  -------------------------------------------
C
C
C  STORAGE METHOD                    DIRECT, UNFORMATTED, FIXED RECORDSIZE
C
C  READ AND WRITE OPERATIONS         READ(unit,REC=record) list
C                                    WRITE(unit,REC=record) list
C
C  USED MEMORY SPACE                 INTEGER*2          2 BYTES (PDP DEFAULT)
C                                    INTEGER*4          4 BYTES (VAX DEFAULT)
C                                    REAL*4             4 BYTES
C                                    DOUBLE PRECISION   8 BYTES
C                                    CHARACTER*N        N BYTES
C
C  RECORD LENGTH                     DEPENDEND OF METH (INFO+8)
C
C  IF YOU USE THE DIRECT, UNFORMATTED STORAGE METHOD, THEN THE
C  RECORD LENGTH HAS TO BE GIVEN IN LONGWORDS. (1 LONGWORD IS 4 BYTES)
C  WENN YOU WANT TO STORE ONE PARAMETER VALUE OF THE SYSTEM
C  MATRIX IN JUST ONE RECORD, THEN THE RECORD LENGTH IS 2
C  ONE DOUBLE PRECISION CONSTANT WILL USE 2 LONGWORDS IN MEMORY.
C  IF YOU WANT TO STORE A COMPLEX CONSTANT, THEN YOU MUST
C  STORE TWO REAL*4 CONSTANTS. (THE REAL AND THE IMAGINAIR PART).
C  SO THE RECORD LENGTH IS ALSO 2 LONGWORDS FOR ONE COMPLEX CONSTANT.
C
```

```
C   STORAGE OF DIFFERENT SYSTEM MATRICES
C
C   SUBTYPE 10 S.S. CANONICAL          (A,B,C,D)   4 2-DIMENSIONAL MATRICES
C           11 S.S. PSEUDO CANONICAL   (A,B,C,D)   4 2-DIMENSIONAL MATRICES
C           12 S.S. CANONICAL          (A,B,C)     3 2-DIMENSIONAL MATRICES
C           13 S.S  PSEUDO CANONICAL   (A,B,C)     3 2-DIMENSIONAL MATRICES
C           14 S.S.                    (F,G,H,K)   4 2-DIMENSIONAL MATRICES
C           15 S.S.                    (F,G,H)     3 2-DIMENSIONAL MATRICES
C           20 M.F.D                   (P,Q)       2 3-DIMENSIONAL MATRICES
C           21 M.F.D                   (P,Q,C)     3 3-DIMENSIONAL MATRICES
C           30 IMPULS RESPONSE         (A1,A2,...) 1 3-DIMENSIONAL MATRIX
C           40 TRANSFER FUNCTION       (A,B)       2 3-DIMENSIONAL MATRICES
C           41 FACTORIZED TRANSFER     (a,b)       2 3-DIMENSIONAL MATRICES
C              FUNCTION
C
C   WENN YOU WANT TO STORE A SPECIAL MATRIX (SPEC1) IN A SPECIAL
C   WAY, THEN YOU MUST WRITE A '1' IN SPEC2
C   FOR INSTANCE, YOU WISH TO STORE JUST THE DIAGONAL ELEMENTS OF THE
C   DIAGONAL MATRIX FORM.
C   A FURTHER SPECIFICATION OF THESE METHODS CAN BE DEFINED IN
C   THE FUTURE
C
C   EVERY MATRIX IS STORED BY FIRST CHANGING THE ROW NUMBER, THEN
C   THE COLUMN NUMBER, AND AS LAST ONE THE DEPTH NUMBER.
C
C
C*****************************************************************************
```

```
************************************************************************
C
      PROGRAM LS_SSM
C
************************************************************************
C
C This program enables the user to estimate inter-actively a EEM, OEM,
C PEM or IEM in state space representation with help of Hill-climbing
C methods of the NAG-library.
C Many options are included to facilitate and/or improve the estimation
C to help the user in understanding and to put the results  in files for
C several diffrent ways of result presentation (all graphic or plotable
C files. Also the initial model can be chosen for the options.
C
C Subroutines : CHPCAN,FSCALE,GRAFIL,MATVEC,MINFIP,MODPRO,PLOFIL,
C PRFILT,RESEQU,RESPRE,RESOUT,RWIO,RWSSM,SIGPRO,UTOX,VECMAT,
C E04DBF,E04KBF,E04KDF
C
C Author : A.J.M. Veltmeijer
C Datum  : Feb. 1985
C
************************************************************************
C
C Parameters for THE file structure (FILSTR), maximum dimensions of the
C model to be estiomated (MAXDIM)and units (UNIT), Overflow value (OVERFL)
c stop criterion (STCRIT) and necessary parameters for subroutine MINFIP
c
C Parameters
C
      INCLUDE 'FILSTR.PAR'
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      INCLUDE 'OVERFL.PAR'
      INCLUDE 'STCRIT.PAR'
      PARAMETER (MACHEP=0.12D-15)
      PARAMETER (TRESH=0.1D-6)
C
C The state space matrices and the parameter vectors
c
C A           : System matrix
c B           : Distribution matrix
c C           : Output matrix
c D           : Input-output matrix
c KMAT        : Distribution matrix for the model error
C OFF         : Offset vector
C X0          : Initial state
c
C PARINI      : The initial/start values for the model parameters.
C PARMOD      : The estimated model parameters.
C PARSYS      : Temporaly storage for the parameter values.
C PARHLP      : Temporaly storage for the parameter values.
C Storage of A,B,C and D into PAR... :
C
C ( n (=IN)          : dimension of the model (and state vector))
C (  p (=IP)          : number of inputs   )
C (  q (=IQ)          : number of outputs  )
C (  mu(i) (=MU(I))   : structural index i )
C (  mu(1)+mu(2)+ ... +mu(q)=n             )
C
C PAR(1)                 = A(mu(1),1).............PAR(n) = A(mu(1),n)
C PAR(n+1)               = A(mu(1)+mu(2),1).................
```

```
C                              ..... PAR(n*q) = A(n,n)
C
C PAR(n*q+1)              = D(1,1).............PAR(n*q+p) = D(1,p)
C PAR(n*q+p+1)            = B(1,1).............PAR(n*q+2*p) = B(1,p)
C PAR(n*q+2*p+1)          = B(2,1)............................
C                              ...........PAR(n*q+(mu(1)+1)*p) = B(mu(1),p)
C PAR(n*q+(mu(1)+1)*p+1) = D(2,1)........................
C                             ..................................
C                              ............. PAR(n*q+(n+q)*p) = B(n,p)
C
C KMAT, OFF and X0 only stored for estimation or temporarily storage
C
C PAR(n*q+q*p+p*n+1)      = KMAT(1,1).....PAR(n*q+n*p+q*p+q) = KMAT(1,q)
C                              ...........PAR(n*q+n*p+q*p*q*n) = KMAT(n,q)
C
C PAR(2*n*q+(n+q)*p+1)    = OFF(1).....PAR(2*n*q+(n+q)*p+q) = OFF(q)
C PAR(2*n*q+(n+q)*p+q+1) = X0(1).....PAR(2*n*q+(n+q)*p+q+n) = X0(n)
C
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     2          D(MIQP,MIPP),KMAT(MINP,MIQP),OFF(MIQP),
     1          PARMOD(MDPARP),PARINI(MDPARP),
     1          PARSYS(MDPARP),PARHLP(MDPARP),X0(MINP)
C
C Signal processing, (first IP values for inputs, then IQ values
C for outputs)
C
      DOUBLE PRECISION MEAN(MIPP+MIQP),STDEV(MIPP+MIQP),
     1          SCALEV(MIPP+MIQP),SCALE,X0MEAN(MINP),LOSSF
      INTEGER ICORRV(MIPP+MIQP)
C
C Dummy variables, For own use (DUM) or used by NAG-library.
C
      DOUBLE PRECISION DDUMV1(MIPP+MIQP),DDUMV2(MIPP+MIQP),
     1          DDUMV3(MINP),DDUMV4(MINP),DDUMV5(MINP),
     1          DDUM1,DDUM2,DDUM3
      INTEGER LIW,LW,LH
C
C Used by NAG hill-climbing routines, Conjugate Gradient,
C Modified Newton and Quasi-Newton,
C
      DOUBLE PRECISION XTOLV(MDPARP),FDBF,GDBF(MDPARP),
     1          FEST
      DOUBLE PRECISION XTOL,ETA,DELTA,STEPMX,FC,GC(MDPARP),
     1          HESL(MDPARP*((MDPARP-1)/2)+1),
     1          W(MDPARP*(7+(MDPARP-1)/2)+1)
      INTEGER IPRINT,MAXCAL,IBOUND,ISTATV(MDPARP),IW(2),INTYPE
      LOGICAL LOCSCH
C
C The structural indices of described models MU?(i),
C and the file structure of the model ?, IF?,
C
      INTEGER MUSYS(MIQP),MUINI(MIQP),MUHLP(MIQP),
     1          IFSYS,IFMOD,IFINI,IFIO,IFHLP
C
C File names,
C
      CHARACTER*40 FILMOD,FILINI,FILHLP,FILSYS,
     1          FILIO
C
C File storage
C
      INTEGER SUBT,IDEF,SPFORM,WR
```

```
      CHARACTER*8 MODNAM
C
C Common block variables not mentioned before
C
      INTEGER IN,IP,IQ,IERR,ICAN,ISTEPR,IOFF,ISTATE,STSFC,STS,NSA,
     1        MUMOD(MIQP),IXQ,INIT
      DOUBLE PRECISION SYSIO(MSAMPP,MIQP+MIPP),
     1        RESIDU(MSAMPP,MIQP),UQ(MIPP),YQ(MIQP),
     1        XOMOD(MINP),OFFMOD(MIQP),FCHLP,XCHLP(MDPARP),
     1        DOVERF,DSTCRI
C
C Detected mistakes.
C
      INTEGER IFAIL,IFAIL7
C
C Dimensions of the model at hand,
C values of parameters and wanted specific estimations.
C M=Maximum number of; H=Help variable.
C
      INTEGER HDPAR,DPAR,MDPAR,MIQ,MIN,MIP,MSAMP
C
C Answers
C
      CHARACTER*1  C1ANS
C
C External declarations for NAG routines.
C
      EXTERNAL FUNKDF,FKKDF,MONKDF,FUNDBF,FKDBF,MONDBF,E04JBS
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /ESTIMA/IERR,ICAN,ISTEPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MUMOD
      COMMON /UQYQ/UQ,YQ
      COMMON /OFFXQ/OFF,XQ
      COMMON /STOPIT/FCHLP,XCHLP,INIT
      COMMON /MINIMI/DOVERF,DSTCRI
C
C Assign values of parameters to variables to be able to use these
C in common blocks,
C NOT USED BACAUSE OF INCLUDE STATEMENTS IN SUBROUTINES.
C
C       MIN=MINP
C       MIP=MIPP
C       MIQ=MIQP
C       MSAMP=MSAMPP
C       MDPAR=MDPARP
C
```

MAXDIM.PAR

```
      PARAMETER (MINP=14)
      PARAMETER (MIPP=5)
      PARAMETER (MIQP=5)
      PARAMETER (MSAMPP=1000)
      PARAMETER (MDPARP=
     1          ((2*MINP*MIQP)+(MINP*MIPP)+(MIPP*MIQP)+MIQP+MINP))
```

UNITS.PAR

```
      PARAMETER UN1=1
      PARAMETER UN2=2
      PARAMETER UN3=3
      PARAMETER UN4=4
      PARAMETER UN5=5
      PARAMETER UN6=6
      PARAMETER NIN=5
      PARAMETER NOUT=6
```

STCRIT.PAR

```
      PARAMETER (STCRIT=1.0D-5)
```

OVERFL.PAR

```
      PARAMETER (OVERFL=1.0D15)
      PARAMETER (OVERF2=1.0D15)
```

FILSTR.PAR

```
      PARAMETER NITYPE=3
      PARAMETER SYSMOD=1
      PARAMETER MEASUR=2
      PARAMETER SAMPLE=3
C
      PARAMETER NSUBT=11
      PARAMETER SSCAN=10      ! State Space Canonical
      PARAMETER SSPCAN=11     ! State Space Pseudo-Canonical
      PARAMETER SSCANP=12     ! SS Can. Strictly Proper
      PARAMETER SSPCAP=13     ! SS Pseudo-can. Strictly Proper
      PARAMETER SS=14         ! SS normal
      PARAMETER SSP=15        ! SS strictly proper
      PARAMETER SSO=16        ! SS offset
      PARAMETER SSPO=17       ! SS strictly proper with offset
      PARAMETER SSCANO=18     ! SS Canonical with offset
      PARAMETER SSCAPO=19     ! SS Canonical Strictly proper with offset
      PARAMETER SSPCAO=20     ! SS Pseudo-can. with offset
      PARAMETER SSPCPO=21     ! SS Pseudo-can. Stricly proper with offset
C
      PARAMETER NSPFO=4
      PARAMETER SFNIA=0
      PARAMETER SFFM=1
      PARAMETER SFDM=2
      PARAMETER SFLTM=3
C
      PARAMETER NSPFU=4
      PARAMETER ZERO=0
      PARAMETER IMP=1
      PARAMETER STA=2
      PARAMETER RAM=3
      PARAMETER NOI=4
```

```
************************************************************************
C
      SUBROUTINE CHPCAN(IN,IQ,A,C,MU,IFAIL)
C
************************************************************************
C
C This subroutine checks the canonical form of matrices A and C.
C A(MINP,MINP) : System matrix. Dimension IN*IN
C C(MIQP,MIPP) : Output matrix. Dimension IQ*IN
C
C IFAIL : Mistake
C ---------------
C 1     : In a row of C an element is non-zero in a colom with number
C         higher then previous detected structural zero's in the row
C         with an higher number.
C 2     : A value not equal to 1.0 or 0.0 is found in a row of C.
C 3     : An element of A that should be 1.0 according to the calculated
C         structural indices from C, is not 1.0.
C 4     : A structural zero in A is not zero.
C
C Subroutines : None
C
C Author : A.J.M. Veltmeijer
C Date   : May 1985
C
************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      DOUBLE PRECISION A(MINP,IN),C(MIQP,IN)
      INTEGER MU(IQ),IFAIL,IN,IQ,
     1        MUTOT
C


************************************************************************
C
      SUBROUTINE DEREQU(DPAR,IN,IP,IQ,MU,PARVEC,
     1          DERIVA,N,OVERFL,IFAIL)
C
C***********************************************************************
C
C       This subroutine calculatesthe derivatives of the equation-
C       error e(k) and puts it in array DERIVA
C
C       Author : A.J.M. Veltmeijer
C       Date   : Dec. '84
C
C***********************************************************************
C
      INCLUDE 'MAXDIM.PAR'
C
      INTEGER ACOL,FIBLOK,IN,IP,IQ,DPAR,IFAIL
      INTEGER MU(IQ),K,ROWAI,COLAJ,IERR,ISTATE,ICAN,ISTRPR,IOFF
      DOUBLE PRECISION PARVEC(DPAR),SUM,UO(MIPP),YO(MIQP),OVERFL,
     1          XO(MINP),OFF(MIQP)
      DOUBLE PRECISION SYSIO(MSAMPP,MIQP+MIPP),DERIVA(MINP+1,MIQP,DPAR)
C
      COMMON /IODATA/SYSIO
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /UOYO/UO,YO
      COMMON /XOOFF/XO,OFF
C
```

```
*************************************************************************
C
      SUBROUTINE DERNUL(IN,IP,IQ,MU,DMU,DPAR,ICAN,ISTRPR,DERIVA)
C
*************************************************************************
C
C Dependent on ICAN and ISTRPR the derivatives to the canonical
C structural zero a-parameters and the parameters in the input-output
C matrix are set to zero.
C
C ICAN=1 : DERIVA(IN+1-DMU(I),I,can.a-par.)=0.0
C ISTRPR=1 : DERIVA(IN-DMU(I),I,d-par.) =0.0    (I=1,...,IQ)
C
C Author : A.J.M. Veltmeijer
C Date :   June 1985
C
*************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
C
      INTEGER IN,IP,IQ,MU(IQ),DMU(IQ),DPAR,ICAN,ISTRPR
      DOUBLE PRECISION DERIVA(MINP+1,MIQP,DPAR)
C



*************************************************************************
C



      SUBROUTINE DEROUT(DPAR,IN,IP,IQ,MU,PARVEC,
     1          DERIVA,K,DMU,OVERFL,IFAIL)
C
*************************************************************************
C
C       This subroutine calcculates the derivative of the equation
C       error I at sample moment K-DMU(I),
C       And puts it in DERIVA(IN+1-DMU(I))
C       IH(I) must be MAXMU-MU(I) at the input, Unchanged.
C
C       Author : A.J.M. Veltmeijer
C       Date   : Feb. 1985
C
*************************************************************************
C
      INCLUDE 'DELERAMAND]MAXDIM.PAR'
C
      DOUBLE PRECISION DERIVA(MINP+1,MIQP,DPAR),
     1          PARVEC(DPAR),UA0(MIPP),YA0(MIQP),
     2          SYSIO(MSAMPP,MIQP+MIPP),RESIDU(MSAMPP,MIQP),SUM
      INTEGER IN,IP,IQ,MU(IQ),DMU(IQ),K,BLOKJ,BLOKS,IERR,IOFF,ISTATE,
     1          COLAJ,COLAV,COLAS,I,J,S,L,V,W,ROWAI,ICAN,ISTRPR
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /UA0YA0/UA0,YA0
C
```

```
************************************************************************
C
      SUBROUTINE DVTEST(N,IN,IP,IQ,MU,A,B,C,D,K,OFF,X0,IMAT,IX0,
     1          STS,ENS,FC,GC)
C
************************************************************************
C
C TEST PROGRAMMA VOOR FKDBF OM AFGELEIDEN TE TESTEN
C PARAMETERS ZITTEN IN A,B,C,D,K,OFF EN X0, FUNCTIE WAARDE IN FC EN
C AFGELEIDEN MET DELTA = 10E-8 IN GC
C
************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      INCLUDE 'FILSTR.PAR'
C
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),




     1          D(MIQP,MIPP),K(MINP,IQ),OFF(IQ),X0(IN),FC,GC(MDPARP),
     1          RESIDU(MSAMPP,MIQP),SYSIO(MSAMPP,MIQP+MIPP),
     1          PARVEC(MDPARP),HFC
      INTEGER IN,IP,IQ,N,MU(IQ),STS,ENS,IMAT,IX0,IERR,ICAN,ISTRPR,
     1          IOFF,ISTATE,DPAR
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
      COMMON /ESTIMA/ IERR,ICAN,ISTRPR,IOFF,ISTATE
C
```

```
C***********************************************************************
C                                                                      *
      SUBROUTINE EIGV(A,IA,N,WR,WI,INTGER,IFAIL)
C                                                                      *
C       This subroutine determines the eigenvalues of the              *
C     N-dimensional, unsymmetric, square, real matrix A.               *
C     To perform this operation the routine calls the pro-             *
C     grams F01ATF, F01AKF and F02AFF from the NAG-library.            *
C                                                                      *
C                                                                      *
C  Parameters                                                          *
C  A(IA,N)  doub.precision [IO] : contains matrix A.On exit is         *
C                                 overwritten.                         *
C  IA         integer      [I]  : number of rows of A.                 *
C  N          integer      [I]  : number of colomns of A.              *
C  WR(N)    doub.precision [O]  : contains on successfull exit        *
C                                 the real part of the singular        *
C                                 values of matrix A.                  *
C  WI(N)    doub.precision [O]  : contains on successfull exit        *
C                                 the imaginary part of the sin-       *
C                                 gular values of matrix A.            *
C  INTGER(N)  integer      [W]    : work array.                       *
```

```
C  IFAIL      integer      [O]  : contains on exit an error code:  *
C                                - IFAIL=0 on successfull exit      *
C                                - IFAIL=ERRCOD if the eisenvalues  *
C                                  has not been found.              *
C                                                                   *
C     Routines                                                      *



C     F01ATF,F01AKF,F02APF (from the NAG-library).                  *
C                                                                   *
C     Files                                                         *
C     NONE.                                                         *
C                                                                   *
C         Author : P.M. Carriere                                    *
C         Date   : July '84                                         *
C         Version: 1                                                *
C                                                                   *
C*******************************************************************
C
      INTEGER IA,N,INTGER(N),IFAIL,IT,IK,IL
C
      DOUBLE PRECISION A(IA,N),WR(N),WI(N)
C



*********************************************************************
C
        SUBROUTINE EQUINN(IN,IP,IQ,MU,PARVEC,STS,NSA)
C
*********************************************************************
C
C       This subroutine calculates the innovation error. The model given
C       by the parameters in PARVEC has to be given in the canonucal
C       form. The equation errors have to be given by RESIDU.
C        Ther new calculated errors will be stored in RESIDU at the same
C       place as the equation errors.
C
C       PARVEC : Parameters given in array form. first the a-parameters
C                row by row ,then the b,d-parameters subsystem by sub-
C                subsystem.(block by block.
C
C       Author : A.J.M.Veltmeijer
C       Date   : Feb 1985
C
*********************************************************************
C
        INCLUDE 'MAXDIM.PAR'
C



        INTEGER IN,IP,IQ,MU(MIOP),STS,NSA
        DOUBLE PRECISION PARVEC(MDPARP),RESIDU(MSAMPP,MIOP)
C
        COMMON /RES/RESIDU
C
```

```
      SUBROUTINE ERROR(NAME,IFAIL,NOUT)
C*************************************************************
C                                                            *
C     This SUBROUTINE ERROR types the error messages         *
C     when an error occurs in reading a matrix from          *
C     the data file that contains the system.                *
C                                                            *
C     Author  : J.H. Vaessen                                 *
C     Date    : 04-10-'83                                    *
C     Revised : A.J.M. Veltmeijer                            *
C     Date    : 10-04-'85                                    *
C     Version : 2                                            *
C                                                            *
C*************************************************************
      CHARACTER*4 NAME


*******************************************************************************
C
      SUBROUTINE FKDBF(N,XC,FC,GC)
C
*******************************************************************************
C
C This subroutine is called by the NAG-library routine E04DBF
C E04DBF : A hill climbing algorithm according to the conjugate gradient
C          method.
C N : Number of parameters (must not be changed)
C XC : Parameter set (must not be changed)
C FC : Must contain loss function on exit
C GC : Must contain the derivative of the lossfunction in given
C      parameter set.
C
C Canonical structural zero's are included in XC. Also d-parameters
C of input-output matrix if model is estimated strictly proper. Both
C have to be corrected in GC
C
C ICAN=1 : Canonical estimation
C ISTRPR=1 : Strictly proper estimation
C IERR=1 : -
C      2 : Equation error minimalisation
C      3 : Output error minimilisation
C      4 : Pediction error minimalisation
C      5 : Innovation error minimalisation
C IOFF=1 : Offseet vector included in XC
C ISTATE=1 : Initial state (n-parameters) included in XC
C        2 : Initial state (a-parameters) included in XC
C
C STSFC : Not used.
C STS : First sample moment in SYSIO for calculations SYSIO(STS_STSFC+1,-)
C NSA : Number of sample moments
C IKF : STS_STSFC+1    SYSIO(IKF,-)
C IKL : STS+NSA-STSFC  SYSIO(IKL,-)
C
C FC and GC are calculated by means of simulation with parameter set XC
C
C       X(k+1) = Ax(k) + Bu(k) + Ke(k)
C       y(k) = Cx(k) + Du(k) + e(k) + OFF
C
C       Dx/Dpar = (DA/Dpar)*x + ADx/Dpar + (DB/Dpar)*u +
C                 (DK/Dpar)*e + KDe/Dpar
C       De/Dpar = -(Dx/Dpar) - (DD/Dpar)*u - DOFF/Dpar
C
```

```
C
C IMAT=0  : A,B,C and D have to be transferred
C IMAT=1  : A,B,C,D AND KMAT HAVE TO BE TRANSFERRED
C IMAT=2  : A,B,C,D AND OFF HAVE TO BE TRANSFERRED
C IMAT=3  ; A,B,C,D K AND OFF HAVE TO BE TRANSFERRED
C IX0=0   : X0 does not has to be transferred
C IX0=1   : Full state vector has to be transferred
C IX0=2   : Q parameters of state vector have to be transferred
C Storage into PARVEC : first A row MU(1),MU(1)+MU(2), ...
C                       then D row 1 B row 1 to MU(1)
C                       then D row 2 B row MU(1)+1 to MU(1)+MU(2)
C                       then ..........(D,B)....
C                       then K row by row
C                       then OFF
C                       finally from PARVEC(DPAR-IN) or PARVEC(DPAR-IQ)
C                               to PARVEC(DPAR)
C
C
C
C Author : A.J.M. Veltmeijer
C Date   : June 1985
C
*******************************************************************************
C
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      INTEGER N
      DOUBLE PRECISION XC(N),FC,GC(N),HGC(MDPARP)
      INTEGER STSFC,STS,NSA,IERR,ICAN,ISTRPR,IOFF,ISTATE,IN,IP,IQ,
     1        MU(MIQP),DMU(MIQP),MMU,IMAT,IX0,DPAR,INIT
      DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP),U0(MIPP),Y0(MIQP),
     1        X0(MINP),OFF(MIQP),XNEW(MINP),XOLD(MINP),
     1        DXNEW(MINP,MDPARP),DXOLD(MINP,MDPARP),ERR(MIQP),
     1        DERERR(MIQP,MDPARP),FCHLP,XCHLP(MDPARP)
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1        D(MIQP,MIPP),KMAT(MINP,MIQP),DOVERF,DSTCRI
C
      COMMON /IODATA/SYSIO
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MU,DMU,MMU
      COMMON /U0Y0/U0,Y0
      COMMON /OFFX0/OFF,X0
      COMMON /STOPIT/FCHLP,XCHLP,INIT
      COMMON /MINIMI/DOVERF,DSTCRI
C
```

```
************************************************************************
C
      SUBROUTINE FKKDF(IFLAG,N,XC,FC,GC,IW,LIW,W,LW)
C
************************************************************************
C
C This subroutine is called by the NAG-library routine E04KDF
C E04KDF : A hill climbing algorithm according to the modified Newton
C          method.
C N : Number of parameters (must not be changed)
C XC : Parameter set (must not be changed)
C FC : Must contain loss function on exit
C GC : Must contain the derivative of the lossfunction in given
C      parameter set.
c
c IFLAG=1 : Only residuals are required
C IFLAG=2 : Also Derivatives are required
C If IFLAG is set to a negative number E04KDF stops.
C
C Canonical structural zero's are included in XC. Also d-parameters
C of input-output matrix if model is estimated strictly proper. Both
C have to be corrected in GC
C
C ICAN=1 : Canonical estimation
C ISTRPR=1 : Strictly proper estimation
C IERR=1 : -
C      2 : Equation error minimalisation
C      3 : Output error minimilisation
C      4 : Prediction error minimalisation
C      5 : Innovation error minimalisation
C IOFF=1 : Offsaet vector included in XC
C ISTATE=1 : Initial state (n-parameters) included in XC
C        2 : Initial state (a-parameters) included in XC
C
C STSFC : Not used.
C STS : First sample moment in SYSID for calculations SYSID(STS_STSFC+1,-)
C NSA : Number of sample moments
C IKF : STS_STSFC+1    SYSID(IKF,-)
C IKL : STS+NSA-STSFC  SYSID(IKL,-)
C
C FC and GC are calculated by means of simulation with parameter set XC
C
C       X(k+1) = Ax(k) + Bu(k) + Ke(k)
C       y(k) = Cx(k) + Du(k) + e(k) + OFF
C
C       Dx/Dpar = (DA/Dpar)*y + ADx/Dpar + (DB/Dpar)*u +
C                 (DK/Dpar)*e + KDe/Dpar
C       De/Dpar = -(Dx/Dpar) - (DD/Dpar)*u - DOFF/Dpar
C
C
C IMAT=0  : A,B,C and D have to be transferred
C IMAT=1  : A,B,C,D AND KMAT HAVE TO BE TRANSFERRED
C IMAT=2  : A,B,C,D AND OFF HAVE TO BE TRANSFERRED
C IMAT=3  ; A,B,C,D K AND OFF HAVE TO BE TRANSFERRED
```

```
C IX0=0   : X0 does not has to be transferred
C IX0=1   : Full state vector has to be transferred
C IX0=2   : Q parameters of state vector have to be transferred
C Storage into PARVEC : first A row MU(1),MU(1)+MU(2), ...
C                       then D row 1 B row 1 to MU(1)
C                       then D row 2 B row MU(1)+1 to MU(1)+MU(2)
C                       then ..........(D,B)....
C                       then K row by row
C                       then OFF
C                       finally from PARVEC(DPAR-IN) or PARVEC(DPAR-IQ)
C                               to PARVEC(DPAR)
C
C
C
C Author : A.J.M. Veltmeijer
C Date   : June 1985
C
*******************************************************************************
C
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      INTEGER N
      INTEGER IFLAG,IW(LIW),LIW,LW
      DOUBLE PRECISION W(LW)
      DOUBLE PRECISION XC(N),FC,GC(N),HGC(MDPARP)
      INTEGER STSFC,STS,NSA,IERR,ICAN,ISTRPR,IOFF,ISTATE,IN,IP,IQ,
     1        MU(MIQP),DMU(MIQP),MMU,IMAT,IX0,DPAR,INIT
      DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP),U0(MIPP),Y0(MIQP),
     1        X0(MINP),OFF(MIQP),YNEW(MINP),XOLD(MINP),
     1        DXNEW(MINP,MDPARP),DXOLD(MINP,MDPARP),ERR(MIQP),
     1        DERERR(MIQP,MDPARP),FCHLP,XCHLP(MDPARP)
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1        D(MIQP,MIPP),KMAT(MINP,MIQP),DOVERF,DSTCRI
C
      COMMON /IODATA/SYSIO
      COMMON /SAMPMD/STSFC,STS,NSA
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MU,DMU,MMU
      COMMON /U0Y0/U0,Y0
      COMMON /OFFX0/OFF,X0
      COMMON /STOFIT/FCHLP,XCHLP,INIT
      COMMON /MINIMI/DOVERF,DSTCRI
C
```

```
************************************************************************
C
      SUBROUTINE FSCALE(IN,IP,IQ,PARVEC,MU,IERR,KI,KF,KL,KP,
     1          LOSSF,SCALE,IMAT,IXO,DPAR)
C
************************************************************************
C
C       This subroutine calculates a double precision SCALE.
C       This SCALE is th scaling factor for the calculations done
C       in RESEQU,DEREQU if IERR=1 or RESOUT,DEROUT if IERR=2.
c LOSSF : The wanted value for the loss function
C SCALE : Scaling factor for all in- and outputs to make the loss
C         function equal to LOSSF
C KI : The sample moment to start the calculations of the errors.
C       SYSIO(KI,-)
C KF : First sample moment for the calculation of the loss function.
C       SYSIO(KF,-)
C KL : Last sample moment for the calculation of the loss function.
C       SYSIO(KL,-)
C KP : Sample moment to end the calculations of the errors.
C
C Subroutines : RESEQU,VECMAT,SIMULA,EQUINN
C
C Author      : A.J.M. Veltmeijer
C Datum       : Feb. 1985
C
************************************************************************
C
      PARAMETER (OVERFL=1.0D30)
C
      INCLUDE 'MAXDIM.PAR'
C
      DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP),SCALE,HSCALE,
     1  RESIDU(MSAMPP,MIQP),PARVEC(MDPARP),LOSSF,
     1          A(MINP,MINP),B(MINP,MIPP),
     1          C(MIQP,MINP),D(MIQP,MIPP),KMAT(MINP,MIQP),
     1          OFF(MIQP),XO(MINP)
      INTEGER IN,IP,IQ,KI,KF,KL,KP,MU(MIQP),IERR,IFAIL,
     1          IMAT,IXO,DPAR
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
```

```
************************************************************************
C
      SUBROUTINE FUNDBF(N,XC,FC,GC)
C
************************************************************************
C
C This subroutine is made to be used in the NAG-library routine
C E04DBF. It calculates the equation error and its derivates and
C presents them in a suitable form (FC,GC) to the calling
C program.
C
C XC : Parameter vector supplied by the NAG-routine
C
C Author : A.J.M. Veltmeijer
C Date   : Feb. 1985
C
************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      INTEGER N
      DOUBLE PRECISION XC(N),FC,GC(N),HGC(MDPARP)
      INTEGER STSFC,STS,NSA,IERR,ICAN,ISTRPR,IOFF,ISTATE,IN,IP,IQ,
     1        MU(MIQP),DMU(MIQP),MMU,DPAR,INIT
      DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP),U0(MIPP),Y0(MIQP),
     1        RESIDU(MSAMPP,MIQP),DERIVA(MINP+1,MIQP,MDPARP),DHULP,
     1        X0(MINP),OFF(MIQP),XCHLP(MDPARP),FCHLP,DOVERF,DSTCRI
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1        D(MIQP,MIPP),KMAT(MINP,MIQP),DUMV1(MIQP),
     1        DERPRE(MIQP,MDPARP),RESPRE(MIQP)
C
      COMMON /IODATA/SYSIO



      COMMON /RES/RESIDU
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MU,DMU,MMU
      COMMON /U0Y0/U0,Y0
      COMMON /X0OFF/X0,OFF
      COMMON /STOPIT/FCHLP,XCHLP,INIT
      COMMON /MINIMI/DOVERF,DSTCRI
C
```

```
************************************************************************
C
      SUBROUTINE FUNKDF(IFLAG,N,XC,FC,GC,IW,LIW,W,LW)
C
************************************************************************
C
C This subroutine is called by Nag-routine E04KDF if the equation or
C prediction error has to be calculated
C




C Author : A.J.M. Veltmeijer
C Date   : June 1985
C
************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
C
      INTEGER IFLAG,N,IW(LIW),LIW,LW
      DOUBLE PRECISION XC(N),FC,GC(N),W(LW),HGC(MDPARP)
      INTEGER STSFC,STS,NSA,IERR,ICAN,ISTRPR,IOFF,ISTATE,IN,IP,IQ,
     1        MU(MIQP),DMU(MIQP),MMU,DPAR,INIT
      DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP),UO(MIPP),YO(MIQP),
     1        RESIDU(MSAMPP,MIQP),DERIVA(MINP+1,MIQP,MDPARP),DHULP,
     1        XO(MINP),OFF(MIQP),XCHLP(MDPARP),FCHLP,DOVERF,DSTCRI
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1        D(MIQP,MIPP),KMAT(MINP,MIQP),DDUMV1(MIQP)
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MU,DMU,MMU
      COMMON /UOYO/UO,YO
      COMMON /XOOFF/XO,OFF
      COMMON /STOPIT/FCHLP,XCHLP,INIT
      COMMON /MINIM1/DOVERF,DSTCRI
C
```

```
*******************************************************************************
C
      SUBROUTINE GRAFIL(IN,IP,IQ,MU,A,B,C,D,KMAT,OFF,X0,FILNAM,UN)
C
*******************************************************************************
C
C This subroutine enables the user to write several variables (optional)
C in a file suitable for the graphic program GRA to read.
C On entry :
C model : A,B,C,D,KMAT,OFF
C initial state : X0
C Hulp state because SIMULA changes state : XH
C dimensions : IN,IP,IQ
C structural indices : MU(-)
C file name : FILNAM
C Unit number : UN
C Subroutines : SIMULA,PRFILT
C
C Author : A.J.M. Veltmeijer
C Date   : 19-6-85
C
*******************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
      PARAMETER (OVERFL=1.0D30)
C
      INTEGER IN,IP,IQ,MU(IQ),UN,STSFC,STS,NSA,DPAR
      DOUBLE PRECISION SYSIO(MSAMPP,MIQP+MIPP),RESIDU(MSAMPP,MIQP),
     1          A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),D(MIQP,MIPP),
     1          KMAT(MINP,MIQP),OFF(IQ),
     1          X0(IN),XH(MINP),KH(MINP,MIQP),
     1          PARVEC(MDPARP)
      REAL RHELP(MIPP*MIQP+1)
      CHARACTER*40 FILNAM
C
      COMMON /IODATA/SYSIO
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /RES/RESIDU



*******************************************************************************
C
      SUBROUTINE MATVEC(IN,IP,IQ,MU,A,B,D,KMAT,OFF,X0,
     1          IMAT,IX0,DPAR,PARVEC)
C
C*******************************************************************************
C
C This subroutine writes a system from the matrices A,B,C,D,OFF into
C the vector PARVEC.
C First a parameter rows of A row by row then block by block (subsystem
C by subsystem)
C The rows of D and Blocks of B
C IN, IP, IQ : Real dimensions of matrices.
C MU(MIQP) :Structural indices
```

```
C IMAT=0  : A,B,C and D have to be transferred
C IMAT=1  : A,B,C,D AND KMAT HAVE TO BE TRANSFERRED
C IMAT=2  : A,B,C,D AND OFF HAVE TO BE TRANSFERRED
C IMAT=3  ; A,B,C,D K AND OFF HAVE TO BE TRANSFERRED
C IX0=0   : X0 does not has to be transferred
C IX0=1   ; Full state vector has to be transferred
C IX0=2   : State vector only consists of a parameters.
C Subroutines :None
C
C Author      :A.J.M.Veltmeijer
C Date        :Dec. 1984
C
C*************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
C
      DOUBLE PRECISION PARVEC(MDPARP)
      DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),D(MIOP,MIPP),
     1          KMAT(MINP,MIOP),OFF(MIOP),X0(MINP)
      INTEGER IP,IQ,IN,AROW,ACOL,BROW,DROW,IMAT,IX0
      INTEGER MU(MIOP)
      CHARACTER*1  ANSWER
C


C*************************************************************************
C
      SUBROUTINE MINFIP(NM,M,N,A,W,IP,B,IERR,RV1,XHULP,
     1                  TETA,TRESH,MACHEP)
C
      INTEGER I,J,K,L,M,N,II,IP,I1,KK,K1,LL,L1,M1,NM,ITS,IERR
      DOUBLE PRECISION A(NM,N),W(N),B(NM,IP),RV1(N),XHULP(NM,IP),
     1                 TETA(N),C,F,G,H,S,X,Y,Z,EPS,SCALE,MACHEP,
     2                 DSQRT,DMAX1,DABS,DSIGN,TRESH
C
      DATA ERRCOD/4/
C
C
C     THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE MINFIT,
C     NUM. MATH. 14, 403-420(1970) BY GOLUB AND REINSCH.
C
C       THIS PROGRAM  HAS BEEN  EXTENDED BY P.CARRIERE (MARCH 1984) IN
C     ORDER TO PROVIDE DIRECTLY THE SOLUTION OF THE EQUATION A * X = B
C     IN THE CASES THAT B IS A VECTOR (SOLUTION IS WRITTEN IN TETA AND
C     IN FIRST COLOMN OF XHULP) AND B IS A MATRIX (SOLUTION IS WRITTEN
C     IN MATRIX XHULP).
C
C
C     HANDBOOK FOR AUTO. COMP., VOL II-LINEAR ALGEBRA, 134-151(1971).
C
C     THIS SUBROUTINE DETERMINES  THE SOLUTION OF THE LINEAR
C
C     SYSTEM AX=B BY PERFORMING THE SINGULAR VALUE DECOMPOSITION
C          T                                                  T
C     A=USV  OF A REAL  M BY N RECTANGULAR MATRIX, FORMING U B
C
C     RATHER THAN U.  HOUSEHOLDER BIDIAGONALIZATION AND A VARIANT OF
C
C     THE QR ALGORITHM ARE USED.
```

```
C
C     ON INPUT:
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT.  NOTE THAT NM MUST BE AT LEAST
C          AS LARGE AS THE MAXIMUM OF M AND N;
C
C        M IS THE NUMBER OF ROWS OF A AND B;
C
C        N IS THE NUMBER OF COLUMNS OF A AND THE ORDER OF V;
C
C        A CONTAINS THE RECTANGULAR COEFFICIENT MATRIX OF THE SYSTEM;
C
C        IP IS THE NUMBER OF COLUMNS OF B.  IP CAN BE ZERO;
C
C        B CONTAINS THE CONSTANT COLUMN MATRIX OF THE SYSTEM
C          IF IP IS NOT ZERO.  OTHERWISE B IS NOT REFERENCED.
C
C
C     ON OUTPUT:
C
C        A HAS BEEN OVERWRITTEN BY THE MATRIX V (ORTHOGONAL) OF THE
C          DECOMPOSITION IN ITS FIRST N ROWS AND COLUMNS.  IF AN
C          ERROR EXIT IS MADE, THE COLUMNS OF V CORRESPONDING TO
C          INDICES OF CORRECT SINGULAR VALUES SHOULD BE CORRECT;
C
C        W CONTAINS THE N (NON-NEGATIVE) SINGULAR VALUES OF A (THE
C          DIAGONAL ELEMENTS OF S).  THEY ARE UNORDERED.  IF AN
C          ERROR EXIT IS MADE, THE SINGULAR VALUES SHOULD BE CORRECT
C          FOR INDICES IERR+1,IERR+2,...,N;
C
C                                  T
C        B HAS BEEN OVERWRITTEN BY U B.  IF AN ERROR EXIT IS MADE,
C                     T
C          THE ROWS OF U B CORRESPONDING TO INDICES OF CORRECT
C          SINGULAR VALUES SHOULD BE CORRECT;
C
C     XHULP CONTAINS ON SUCCESSFULL EXIT THE GENERAL SOLUTION ( B IS
C          MATRIX ) OF THE EQUATION :
C                              A * X = B
C
C      TETA CONTAINS ON SUCCESSFULL EXIT THE FIRST COLOMN OF XHULP.
C          IT IS DE SOLUTION OF EQUATION A * X = B IN THE CASE
C          THAT B IS A VECTOR.
C
C        IERR IS SET TO
C          ZERO       FOR NORMAL RETURN,
C          ERRCOD     IF THE SOLUTION OF EQUATION A * X = B HAS NOT
C                     BEEN FOUND.
C
C        RV1 IS A TEMPORARY STORAGE ARRAY.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO R. S. GARBOW,
C
C
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
```

```
*****************************************************************************
C
      SUBROUTINE MODPRO(IN,IP,IQ,A,B,D,KMAT,OFF,XO,MU,SCALE)
C
*****************************************************************************
C
C Subroutine to correct parameters in A(MINP,MINP), B(MINP,MIPP) and
C D(MIQP,MIPP),OFF(MIQP) if the in- and output
C samples are multiplied by
C input 1 : SCALE(1)
C input p : SCALE(IP)
C output 1: SCALE(IP+1)
C output q: SCALE(IP+IQ)
C IN, IP, IQ : Real dimensions of matrices.
C MU    : Structural indices
C
C Subroutines : None
C
C Author      : A.J.M. Veltmeijer



C Date        : 30 May 1985
C
*****************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INTEGER IN,IP,IQ,MU(MIQP),MUTOT,COLA
      DOUBLE PRECISION SCALE(IQ+IP),A(MINP,IN),
     1          B(MINP,IP),D(MIQP,IP),KMAT(MINP,IQ),OFF(IQ),YO(IN)
C


*****************************************************************************
C
      SUBROUTINE MONDBF(N,XC,FC,GC,NCALL)
C
C*****************************************************************************
C
C       Used by NAG-routine E04DBF.
C
C Author : A.J.M. Veltmeijer
C Date   : 1985
C
C*****************************************************************************
C
      INCLUDE 'UNITS.PAR'
      INCLUDE 'MAXDIM.PAR'
C
      INTEGER N,NCALL,MOCALL,FUCALL



      INTEGER IN,IP,IQ,MU(MIQP),STSERR,NSAERR
      DOUBLE PRECISION XC(N),FC,GC(N)
      DOUBLE PRECISION SYSIO(MSAMPP,MIQP+MIPP)
C
```

```
*******************************************************************************
C
      SUBROUTINE MONKDF(N,XC,FC,GC,ISTATE,GFJNRM,COND,POSDEF,NITER,
     1          NF,IW,LIW,W,LW)
C
*******************************************************************************
C
C Used by NAG-routine E04KDF.
C
C Author : A.J.M. Veltmeijer
C Date   : 1985
C
*******************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
C
      INTEGER N,ISTATE(N),NITER,NF,IW(LIW),LIW,LW
      LOGICAL POSDEF
      DOUBLE PRECISION XC(N),FC,GC(N),GFJNRM,COND,W(LW)
C


*******************************************************************************
C
      SUBROUTINE PLOFIL(UN,DPAR,IMAT,IXO,FILINI,PARINI,FILMOD,PARMOD,
     1          FILIO,IMVCR,INSCAL,SCALEV)
C
*******************************************************************************
C
C This subroutine makes a plotable file with the estimation conditions
C and estimation results (see LS_SSM for variable names)
C
C Author : A.J.M. Veltmeijer
C Date   : aug. 1985
C
*******************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INCLUDE 'UNITS.PAR'
C
      DOUBLE PRECISION PARINI(MDPARF),PARMOD(MDPARF),SCALEV(MIPF+MIQF),
     1          SYSIO(MSAMPF,MIQF+MIPF),RESIDU(MSAMPF,MIQF),UO(MIPF),
     1          YO(MIQF),XO(MINF),OFF(MIQF),A(MINF,MINF),B(MINF,MIPF),
     1          C(MIQF,MINF),D(MIQF,MIPF),KMAT(MINF,MIQF),
     1          AHULP(MINF,MINF),MEAN(MINF),HMEAN(MIQF),STDEV(MINF),
     1          HSTDEV(MIQF),POWER(MINF),HPOWER(MIQF),DPHLP1(MINF),
     1          DPHLP2(MINF)
      CHARACTER*40 FILMOD,FILIO,FILPLO,FILINI
      INTEGER IN,IP,IQ,UN,DPAR,IMAT,IXO,IMVCR,INSCAL,STSFC,STS,NSA,
     1          IERR,ICAN,ISTRPR,IOFF,ISTATE,MU(MIQF),DMU(MIQF),MMU
C
      COMMON /IODATA/SYSIO
      COMMON /SAMPMO/STSFC,STS,NSA
      COMMON /RES/RESIDU
      COMMON /ESTIMA/IERR,ICAN,ISTRPR,IOFF,ISTATE
      COMMON /MATDIM/IN,IP,IQ
      COMMON /STRIND/MU,DMU,MMU
      COMMON /UOYO/UO,YO
      COMMON /XOOFF/XO,OFF
C
```

```
*************************************************************************
C
      SUBROUTINE PRFILT(IN,IQ,MU,A,KMAT,IFAIL)
C
*************************************************************************
C
C This subroutine calculates the prediction filter of the state space
C representation of a equation error model. Only canonical parameters
C are used for calculations.
C
C A : System matrix with a-parameters
C mu : Structural indices
C KMAT : On exit the prediction filter
C
C N(a)KMAT=O(a) : See report
C
C Subroutines : MINFIP
C
C Author : A.J.M. Veltmeijer
C Date   : 19-6-85
C
*************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      PARAMETER (TRESH=0.1D-6)
      PARAMETER (MACHEP=0.12D-15)
C
      INTEGER IN,IQ,MU(MIQP),IFAIL
      DOUBLE PRECISION A(MINP,MINP),KMAT(MINP,MIQP),
     1          O(MINP,MIQP),N(MINP,MINP),DDUMV1(MINP),
     1          DDUMV2(MINP),DDUMV3(MINP)
C


*************************************************************************
C
      SUBROUTINE RDDATA(FILENA,NAME,VAR,NVAR,NROWS,NCOLS,ISTAT,IFAIL)
C*************************************************************************C
C                                                                        C
C This subroutine reads data from the MATLAB data file called 'FILENA',C
C It reads the data belonging to the blok called 'NAME' in the array   C
C named 'VAR'.                                                         C
C The index 'NVAR' gives the number of double precision reals available C
C in the array 'VAR'.                                                  C
C The variable 'NROWS' gives the number of rows of the array and the   C
C variable 'NCOLS' indicates the number of columns.                    C
C The variable 'ISTAT' indicates wether the variables are real         C
C (ISTAT=0) or complex (ISTAT=1). Entering the subroutine the variable C
C 'ISTAT' must have a value corresponding with the type of array 'VAR' C
C If ISTAT = 0 VAR wil have the structure VAR(NROWS,NCOLS).            C
C If ISTAT = 1 VAR will have the structure VAR(2,NR,NC) with VAR(1,*,*)C
C being the real part of the element (*,*) and VAR(2,*,*) being the    C
C imaginary part.                                                      C
C The variable 'IFAIL' indicates wether the reading is finished        C
C succesfull (IFAIL = 0) or not (IFAIL # 0).                           C
C                                                                      C
C If the array to be read from file is complex the structure of the    C
C array after reading will be:                                         C
C      A(1,*,*)  : real part of element *,*                            C
C      A(2,*,*)  : imaginary part of element *,*                       C
C                                                                      C
```

```
C Possible errors are:                                                  C
C    IFAIL=1      - The arrays indicated by  the string 'NAME' are not C
C                   of the same type.                                   C
C    IFAIL=2      - The array indicated by the string 'NAME' is not    C
C                   found in the data file.                             C
C    IFAIL=3      - The array 'VAR' is to small to contain all the     C
C                   elements                                            C
C                                                                       C
C Author  : T. Backx                                                    C
C Date    : 11-08-83                                                    C
C Update  :                                                             C
C                                                                       C
C***********************************************************************C
      INTEGER NROWS,NCOLS,ISTAT,IFAIL,NVAR
      DOUBLE PRECISION VAR(NVAR)
      CHARACTER*40 FILENA
      CHARACTER*4 NAME
C
      INTEGER I,J,IFILE,IST,I2,NELEM
      INTEGER IREC,I1,INDST,INDEND
      DOUBLE PRECISION DUMMY(4)
      CHARACTER*4 BLNAME
C

.

************************************************************************
C
        SUBROUTINE RDLAB(IN,IP,IQ,A,B,C,D,KMAT,OFF,NAMA,NAMB,NAMC,NAMD,
     1           NAMK,NAMOFF,FILNAM,IFAIL)
C
C***********************************************************************
C
C     This SUBROUTINE RDLAB reads the matrices of the system,
C     needed for simulation, from a file with MATLAB-structure.


.


C A,B,C,D,OFF
C NAMA,NAMB,NAMC,NAMD,NAMOFF : Names of matrices.
C
C     This subroutine needs the subroutines:
C             RDDATA, ERROR, RDWRMA and SCHRMA
C
C     Author  : J.H. Vaessen
C     Date    : 04-10-'83
C     Revised : A.J.M. Veltmeijer
C     Date    : 10-04-'85
C     Version : 2
C
C***********************************************************************
C
        INCLUDE 'MAXDIM.PAR'
      DOUBLE PRECISION
     %       A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),D(MIQP,MIPP),
     1          KMAT(MINP,MIQP),OFF(MIQP)
      CHARACTER*1 ANS
      CHARACTER*4 NAME,NAMA,NAMB,NAMC,NAMD,NAMK,NAMOFF
      CHARACTER*40 FILNAM
C
```

```
C*********************************************************************
C
      SUBROUTINE RDSYS(MRANK,MIP,MIQ,IRANK,IP,IQ,A,B,C,D,FILNAM,
     %                 IFAIL)
C*********************************************************************
C                                                                     *
C     This SUBROUTINE READSYS reads the matrices of the system,       *
C     needed for simulation, from a file created with MATLAB.         *



C     This subroutine needs the subroutines:                          *
C             RDDATA, ERROR, RDWRMA and SCHRMA                        *
C                                                                     *
C     Author  : J.H. Vaessen                                          *
C     Date    : 04-10-'83                                             *
C     Revised : A.J.M. Veltmeijer                                     *
C     Date    : 10-04-'85                                             *
C     Version : 2                                                     *
C                                                                     *
C*********************************************************************
      DOUBLE PRECISION
     %       A(MRANK,MRANK),B(MRANK,MIP),C(MIQ,MRANK),D(MIQ,MIP)
      CHARACTER*1 ANS
      CHARACTER*4 NAME
      CHARACTER*40 FILNAM
C



C*********************************************************************
C
      SUBROUTINE RDTHE(IN,IP,IQ,A,B,C,D,KMAT,OFF,NAMA,NAMB,NAMC,NAMI,
     1          NAMK,NAMOFF,FILNAM,IFAIL)
C
C*********************************************************************
C
C       This subroutine reads the matrices of an
C       State Space model from a file (FILNAM).
C       (See System file structure)
C
C       A       : System Matrix         (IN*IN)
C       B       : Distribution Matrix   (IN*IP)
C       C       : Output matrix         (IP*IN)
C       D       : Input/Output Matrix   (IQ*IP)
C       K       : NOISE FILTER          (IN*IQ)
C       OFF     : Offset vbector        (IQ)
C       IN      : Dimension of the MOdel
C       IP      : Number of inputs
C       IQ      : Number of Outputs
C       UN      : Unitnumber
C       SPFORM  : What kind of State Space model
```

```
C       MODNAM  : Nam of model
C       IFAIL   : 1 Matrix not of correct type
C                 2 Matrix not found
C                 3 Matrix to large
C                 4 Dimensions not correct
C                 5 Proper matrix presumed
C
C       Author  : A.J.M. Veltmeijer
C       Date    : Dec. '85
C
C*****************************************************************************
C
        INCLUDE 'FILSTR.PAR'
        INCLUDE 'UNITS.PAR'
        INCLUDE 'MAXDIM.PAR'
C
        CHARACTER*8 NAMA,NAMB,NAMC,NAMD,NAMK,NAMOFF,NAME,
     1              TITLE,DATUM,TIJD,MODNAM
        CHARACTER*2 DTYPE
        CHARACTER*40 FILNAM
        INTEGER*2 NREC,LREC,IREC,MREC,INFO,NVAL,EVTC,FILMOD,UPDATE,
     1            ITYPE,STYPE1,STYPE2,N,M,K,INFF,INUM,METH,SPEC1,SPEC2,
     2            BEGREC,DUM1,DUM2,DUM3
        INTEGER SFFORM,RD,IN,IF,IQ,UN,NA,NB,NC,EXIST(6),
     1          IFAIL,HLPREC
        DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1          D(MIQP,MIPP),KMAT(MINP,MIQP),OFF(MIQP)
C
```

```
      SUBROUTINE RDWRMA(DMAT,MROW,MCOL,IROW,ICOL)
C*****************************************************************
C                                                                *
C     This SUBROUTINE REWRIMAT rewrites a matrix that has        *
C     been filled with the SUBROUTINE RDDATA to a matrix         *
C     having an ordinairy block structure.                       *
C                                                                *
C     Author  : J.H. Vaessen                                     *
C     Date    : 05-10-'83                                        *

C     Version : 1                                                *
C                                                                *
C*****************************************************************
      DOUBLE PRECISION DMAT(MROW,MCOL)
C
```

```
C*************************************************************************
C
      SUBROUTINE RESEQU(IN,IP,IQ,MU,PARVEC,KF,KL,OVERFL,IFAIL)
C
C*************************************************************************
C
C This subroutine calculates the equation error I for sample moment
C KF-1 to KL-1 and writes it in RESIDU(KF,I) to RESIDU(KL,I).
C
C IN,IP,IQ : Dimensions of the state space matrices.
C MU : Structural indices
C PARVEC : Parameters written in an array
C OVERFL : Maximum value during overflow protection
C IFAIL : 0 No overflow
C         1 Overflow in RESIDU
C         2 Overflow in intermediate results
C
C Subroutines : None
C
C Author      : A.J.M. Veltmeijer
C Date        : Dec 1984
C
C*************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      DOUBLE PRECISION PARVEC(MDPARP),RESIDU(MSAMPP,MIQP),
     1          SYSIO(MSAMPP,MIPP+MIQP),SUM1,SUM,OVERFL,U0(MIPP),
     1          Y0(MIQP),X0(MINP),OFF(MIQP)
      INTEGER MU(MIQP),BLOKL,COLAL,IN,IP,IQ,K,MSAMP,I,J,L,S,T,IFAIL
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
      COMMON /U0Y0/U0,Y0
      COMMON /X0OFF/X0,OFF
C
C*************************************************************************
C                                                                       *
      SUBROUTINE RWIO(IP,IQ,FILNAM,IFIL,
     1          STSFC,NSAMP,UN,IREAD,IFAIL)
C
C*************************************************************************
C                                                                       *
C     This subroutine reads (if IREAD=1) NSAMP double precision       *
C     samples starting at sample STSFC+1 from a file into a           *
C     double precision matrix SYSIO or writes (if IREAD=2) NSAMP      *
C     or write from terminal (IREAD=3, STSFC not used) NSAMP
C     double precision  samples from matrix SYSIO to a file with      *
C     double precision samples.                                       *
```

```
C     FILNAM contains the name of the file .The file as a MATLAB   *
C     structure.                                                   *
C                                                                  *
C  Parameters                                                      *
C  SYSIO(MSAMPP,IP+IQ) doub.prec.[O] (IREAD=1) : contains the I/O  *
C                                        : samples to be read      *
C                                  [I] (IREAD=2) : or written.     *
C  STSFC          integer   [I] : contains the number of the       *
C                                  first sample moment to be read. *
C                                  Or first sample moment that has *
C                                  to be written in SYSIO(STSFC+1,-) *
C  FILNAM(40)     character [I] : contains the name of the file.   *
C  UN             integer   [I] : contains unit number.           *
C  IREAD          integer   [I] : indicates if it has to be read  *
C                                  (=1) or written (=2) to file.   *
C  IFAIL          integer   [O] : error indicator                 *
C                                  = 0  no error                   *
C                                  = ERRC01 on open/read error     *
C                                  = ERRC02 if parameters do       *
C                                    not correspond                *
C     Files                                                        *
C     FILNAM (UNIT=UN)                                             *
C                                                                  *
C     Routines                                                     *
C     None.                                                        *
C                                                                  *
C         Author : P.M. Carriere (RWIDFI)                          *
C         Date   : July '84                                        *
C         Version: 1                                               *
C         Modified : A.J.M. Veltmeijer                             *
C         Date     : may '85                                       *
C                                                                  *
C******************************************************************
C
        INCLUDE 'FILSTR.PAR'
        INCLUDE 'MAXDIM.PAR'
        INCLUDE 'UNITS.PAR'
C                                                                  *
        INTEGER IP,IQ,NSAMP,
     1          STSFC,IREAD,IFAIL,I,J,IWK1,IWK2,
     1          IWK3,UN
        CHARACTER*40 FILNAM
        CHARACTER*2 DTYPE
        CHARACTER*4 NAME
        INTEGER*2 SPFU(MIPP),I2H
        INTEGER*2 NREC,LREC,MREC,IREC,INFO,NVAL,EVTC,FILMOD,UPDATE,
     1            ITYPE,IP2,IQ2,INUM
C
        DOUBLE PRECISION SYSIO(MSAMPP,MIPP+MIQP)
C
        COMMON /IODATA/SYSIO
```

```
**************************************************************************
C
        SUBROUTINE RWSSM(IN,IP,IQ,A,B,C,D,KMAT,OFF,DEF,SURT,MODNAM,
     1           SPFORM,FILNAM,WR,ISTRUC,UN,IFAIL)
C
**************************************************************************
C
C This subroutine reads (WR=0) or writes (WR=1) a state space model
C named MODNAM from or to a file named FILNAM using unit number UN.
C The model is stored or has to be stored in the form SPFORM.
C The file can have a THE- or MATLAB-structure (ISTRUC=1 or 2).
C On succesful exit IFAIL=0
C
C A(MINP,MINP) : System matrix
C B(MINP,MIPP) : Input matrix
C C(MIQP,MINP) : Distribution matrix
C D(MIQP,MIPP) : Input -output matrix
C KMAT(MINP,MIQP): Noise filter
C OFF(MIQP) : Offset vector
C WR=0 : Read from file
C WR=1 : Write to file
C WR=2 : Read from terminal
C
C The matrices can be named by the user during run (DEF=0) or are named
C automatically (DEF=1) (A,B,C,D,OFF)
C If by reading D or OFF are not found the are supposed to be zero dependent
C on SURT. By writing SURT determines if D or OFF are written.
C
C Subroutines : RDTHE,RDLAB,WRLAB,WRTHE
C
c Author : A.J.M. Veltmeijer
C Date : June 1985
C
**************************************************************************
C
        INCLUDE 'MAXDIM.PAR'
        INCLUDE 'UNITS.PAR'
        INCLUDE 'FILSTR.PAR'
C
        DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1           D(MIQP,MIPP),KMAT(MINP,MIQP),OFF(MIQP)
        INTEGER IN,IP,IQ,DEF,SURT,SPFORM,WR,ISTRUC,IUN,IFAIL
        CHARACTER*40 FILNAM
        CHARACTER*8 MODNAM,NAMA8,NAMB8,NAMC8,NAMD8,NAMK8,NAMO8,
     1           TIME,DATE
        CHARACTER*4 NAMA4,NAMB4,NAMC4,NAMD4,NAMK4,NAMO4
C


      SUBROUTINE SCHRMA(A,JA,M,N,NAME,NOUT)
C**************************************************************************
C                                                                         *
C        SCHRMA is a SUBROUTINE that writes an arbitrary matrix          *
C        SCHRMA(A,JA,M,N,NAME,NOUT)                                      *
C                                                                         *
C**************************************************************************
C
 1000 INTEGER IA(5)
      INTEGER*2 NAME(3)
      DOUBLE PRECISION A(JA,N)
C
```

```
************************************************************************
C
      SUBROUTINE SIGPRO(IP,IQ,KF,KL,IMEAN,ISTDEV,ISCALE,
     1          MEAN,STDEV,SCALE)
C
************************************************************************
C
C This subroutine allows all kinds of signal processing.
C The signals are supposed to be in SYSIO.
C
C IMEAN=0       : No mean manipulation
C      =1       : Correct with mean values given by MEAN
C      =2       : Calculate mean values and put them in MEAN
C      =3       : Calculte mean values, put them in MEAN and correct
C                 signals.
C ISTDEV=0      : No manipulation
C       =1      : -
C       =2      : Calculate powers of signals and put them in STDEV
C       =3      : -
C       =4      : Calculate standard deviations of signals and put them
C                 in STDEV.
C ISCALE=0      : No manipulations
C       =1      : Correct signals with scale factors in SCALE
C       =2      : Calculate scaling factors and put the, in SCALE
C                 : (all powers equal to minimal power)
C       =3      : -
C
C MEAN(MIQP+MIPP) : Gemiddelde van SYSIO(-,MIQP+MIPP)
C STDEV(MIPP+MIQP): POwer or standard deviation of SYSIO(-,MIQP+MIPP)
C SCALE(MIPP+MIQP): Scaling factor for SYSIO(-,MIPP+MIQP)
C       (If only one vector is used the other two should be dummy
C       variables for security)
C KF : First sample that must be processed is SYSIO(KF,-)
C KL : Last sample that must be processed is SYSIO(KL,-)
C
C Subroutines : None
C Author      : A.J.M. Veltmeijer
C Date        : May 1985
C
************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INTEGER IP,IQ,KF,KL,IMEAN,ISTDEV,ISCALE




      DOUBLE PRECISION MEAN(MIPP+MIQP),STDEV(MIPP+MIQP),
     1          SCALE(MIPP+MIQP),SYSIO(MSAMPP,MIPP+MIQP),
     1          DUMMY
C
      COMMON /IODATA/SYSIO
C
```

```
*************************************************************************
C
      SUBROUTINE SIMULA(IN,IP,IQ,A,B,C,D,K,OFF,X0,STS,ENS,ISPSIM,INP)
C
*************************************************************************
C
C simulate output and calculate state vector (and prediction error
C from Y(STS) to Y(ENS).
C X0 : on entry : X(STS) (NOT CHANGED)
C Model : A,B,C,D,OFF
C K : Prediction filter. (Derived from I/O prediction error model
C ISPSIM=0 : Simulate initial output with only initial state.
C        1 : Simulate output with impuls on input INP (with X0)
C        2 : Simulate output  with step on input INP (with X0)
C        3 : Simulate output with input in SYSIO(STS+1,-) to SYSIO(ENS+1,-)
C        4 : Predict output with I/O-samples in SYSIO(STS+1,-) to ...
C
C STS : Start sample moment
C ENS : End sample moment
C




C RESIDU(STS+1,-) to RESIDU(ENS+1,-) ; Results
C
C Subroutines : None
C
C Author : A.J.M. Veltmeijer
C Date   : 19-6-85
C
*************************************************************************
C
      INCLUDE 'MAXDIM.PAR'
      INTEGER STS,ENS,ISPSIM,INP
      DOUBLE PRECISION A(MINP,IN),B(MINP,IP),C(MIQP,IN),D(MIQP,IP),
     1          K(MINP,IQ),OFF(IQ),X0(IN),XOLD(MINP),XN(MINP),
     1          SYSIO(MSAMPP,MIQP+MIPP),RESIDU(MSAMPP,MIQP),
     1          PRERR(MIQP)
C
      COMMON /IODATA/SYSIO
      COMMON /RES/RESIDU
C


*************************************************************************
C
      SUBROUTINE UTOX(IN,IP,IQ,DPAR,MU,PARVEC,U,X)
C
*************************************************************************
C
C This subroutine calculates the steady state response of a model in
C PARVEC . The input is U the calculated response is X.
C A and B matrices are known (in PARVEC)
C
C (I-A)X=BU
C
C Subroutines : MINFIP  (Solution of Ax=b)
C
C Author : A.J.M. Veltmeijer
```

```
C Date :   June 1985
C
*******************************************************************************
C
        INCLUDE 'MAXDIM.PAR'
        PARAMETER (TRESH=0.1D-6)
        PARAMETER (MACHEP=0.12D-15)
C
        INTEGER IN,IP,IQ,DPAR,MU(IQ),IDUM1
        DOUBLE PRECISION PARVEC(DPAR+1),U(MIPP),X(MINP),A(MINP,MINP),
     1          UHELP(MINP),DDUMV1(MINP),DDUMV2(MINP),DDUMV3(MINP),
     1          DDUM1,DDUM2
C

*******************************************************************************
C
        SUBROUTINE VECMAT(IN,IP,IQ,MU,DPAR,PARVEC,A,B,C,D,KMAT,OFF,X0,
     1          IMAT,IX0)
C
C*******************************************************************************
C
C This subroutine writes a model from the vector PARVEC into
C the matrices A(MINP,MINP), B(MINP,MIPP) and D(MIQP,MIPP).
C IN, IP, IQ : Real dimensions of matrices.
C MU(MIQP) :Structural indices
C IMAT=0   : A,B,C and D have to be transferred
C IMAT=1   : A,B,C,D AND KMAT HAVE TO BE TRANSFERRED
C IMAT=2   : A,B,C,D AND OFF HAVE TO BE TRANSFERRED
C IMAT=3   ; A,B,C,D K AND OFF HAVE TO BE TRANSFERRED
C IX0=0    : X0 does not has to be transferred
C IX0=1    : Full state vector has to be transferred



C IX0=2    : Q parameters of state vector have to be transferred
C Storage into PARVEC : first A row MU(1),MU(1)+MU(2), ...
C                       then D row 1 B row 1 to MU(1)
C                       then D row 2 B row MU(1)+1 to MU(1)+MU(2)
C                       then ..........(D,B)....
C                       then K row by row
C                       then OFF
C                       finally from PARVEC(DPAR-IN) or PARVEC(DPAR-IQ)
C                               to PARVEC(DPAR)
C
C Subroutines : None
C
C Author       : A.J.M. Veltmeijer
C Date         : Nov. 1984
C
C*******************************************************************************
C
        INCLUDE 'MAXDIM.PAR'
C
        DOUBLE PRECISION PARVEC(MDPARP)
        DOUBLE PRECISION A(MINP,MINP),B(MINP,MIPP),C(MIQP,MINP),
     1          D(MIQP,MIPP),KMAT(MINP,MIQP),OFF(MIQP),X0(MINP)
        INTEGER IP,IQ,IN,AROW,ACOL,BROW,DROW,IMAT,IX0
        INTEGER MU(MIQP)
        CHARACTER*1  ANSWER
C
```

```
C*************************************************************************
C
      SUBROUTINE WRLAB(A,B,C,D,KMAT,OFF,NAMA,NAMB,NAMC,
     1           NAMD,NAMK,NAMOFF,FILN3,UN,IN,IP,IQ)
C
*************************************************************************



C This subroutine writes under the names NAMA,NAMB,NAMC,NAMD and
C NAMOFF the system matrices  A, B, C, D and OFF into a file with a
C MATLAB structure  (see MATLAB documentation) in order to
C allow further operations. The name of this file is contai-
C ned into FILN3.
C
C Parameters:
C A(MINP,IN)      doub.prec. [I] : contains system matrix A.
C B(MINP,IP)      doub.prec. [I] : contains system matrix B.
C C(MIQP,IN)      doub.prec. [I] : contains system matrix C.
C D(MIQP,IP)      doub.prec. [I] : contains system matrix D.
C FILN3(40)          character  [I] : name of the file where
C                                     the system matrices will
C                                     be written.
C UN              inteser    [I] : unit number of file
C
C Files
C Unit = UN (file = FILN3)
C
C Routines: None
C
C Author : P.M. Carriere
C Revised: A.J.M. Veltmeijer
C Date   : July '84
C Version: 1
C
C*************************************************************************
C
        INCLUDE 'MAXDIM.PAR'
C
      INTEGER I,J,UN,IST
      INTEGER IP,IN,IQ
C
      DOUBLE PRECISION A(MINP,IN),B(MINP,IP),C(MIQP,IN),
     1                 D(MIQP,IP),KMAT(MINP,IQ),OFF(MIQP)
C
      CHARACTER*4 NAME,NAMA,NAMB,NAMC,NAMD,NAMK,NAMOFF
      CHARACTER*40 FILN3
C
```

```
*************************************************************************
C
      SUBROUTINE WRTHE(A,B,C,D,KMAT,OFF,NAMA,NAMB,NAMC,NAMD,NAMK,
     1          NAMOFF,FILNAM,UN,IN,IP,IQ,DATE,TIME,SUBT,
     1          MODNAM,SPFORM)
C
*************************************************************************
C
C       This subroutine reads (RD=1) or writes (RD=0) the matrices of an
C       State Space model from or into a file (FILNAM).
C       Depending on SPFORM D is read/written or not.
C       (See System file structure)
C
C       A       : System Matrix          (IN*IN)
C       B       : Distribution Matrix    (IN*IP)
C       C       : Output matrix          (IP*IN)
C       D       : Input/Output Matrix    (IQ*IP)
C       K       : Noise filter           (IN*IQ)
C       IN      : Dimension of the MOdel
C       IP      : Number of inputs
C       IQ      : Number of Outputs
C       UN      : Unitnumber
C       SPFORM  : What kind of State Space model
C       MODNAM  : Nam of model
C       IFAIL   : 1 Matrix not of correct type
C                 2 Matrix not found
C                 3 Matrix to large
C                 4 Dimensions not correct
C                 5 Proper matrix presumed
C
C       Author  : A.J.M. Veltmeijer
C       Date    : Dec. '85
C
C*************************************************************************
C
      INCLUDE 'FILSTR.PAR'
      INCLUDE 'UNITS.PAR'
      INCLUDE 'MAXDIM.PAR'
C
      CHARACTER*8 NAMA,NAMB,NAMC,NAMD,NAMK,NAMOFF,NAME,
     1            TITLE,DATE,TIME,MODNAM
      CHARACTER*2 DTYPE
      CHARACTER*40 FILNAM
      INTEGER*2 NREC,LREC,IREC,MREC,INFO,NVAL,EVTC,FILMOD,UPDATE,
     1          ITYPE,STYPE1,STYPE2,N,M,K,INFF,INUM,METH,SPEC1,SPEC2,
     2          REGREC,DUM1,DUM2,DUM3
      INTEGER SPFORM,RD,IN,IP,IQ,UN,NA,NB,NC,EXIST(5),
     1          NUMMAT,IFAIL,HLPREC
      DOUBLE PRECISION A(MINP,IN),B(MINP,IP),C(MIQP,IN),D(MIQP,IP),
     1          KMAT(MINP,IQ),OFF(MIQP)
C
```