Faculty of Electrical Engineering, Computer Science and Telecommunications
University of Zielona Góra

Lecture Notes in Control and Computer Science
Volume 1

Published in a cooperation with Lubusky Scientific Society in Zielona Góra, Poland

Marcin WITCZAK
Institute of Control and Computation Engineering
University of Zielona Góra
ul. Podgórna 50
65-246 Zielona Góra, Poland
e-mail: `M.Witczak@issi.uz.zgora.pl`

Referees:

- Ewaryst RAFAJŁOWICZ, Wrocław University of Technology
- Jan Maciej KOŚCIELNY, Warsaw University of Technology
- Zdzisław KOWALCZUK, Gdańsk University of Technology

The text of this book was prepared based on the author's Ph.D. dissertation

Partially supported by the State Committee for Scientific Research (KBN)
in Poland

ISBN 83-88317-65-2

Camera-ready copy prepared from the author's LaTeX2$_\varepsilon$ files.
Printed and bound by University of Zielona Góra Press, Poland.

CONTENTS

In memory of

Krystyna Witczak

1950-2000

Pamięci

Krystyny Witczak

1950-2000

Acknowledgments

It is a pleasure to express my sincere gratitude and thanks to:

- Professor Józef Korbicz for suggesting the problem, and for his continuous support and supervision of this work,
- Professor Ron J. Patton, whose works introduced me to the field of fault diagnosis for dynamic systems.

In addition, I wish to thank all my friends and colleagues at the Institute of Control and Computation Engineering, who helped me in many, many ways while I was preparing this work.

Notation

t	time
k	discrete time
$\mathcal{E}(\cdot)$	expectation operator
$\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k \ (\dot{\boldsymbol{x}}(t), \dot{\hat{\boldsymbol{x}}}(t)) \in \mathbb{R}^n$	state vector and its estimate
$\boldsymbol{y}_k, \hat{\boldsymbol{y}}_k \in \mathbb{R}^m$	output vector and its estimate
$\boldsymbol{e}_k \in \mathbb{R}^n$	state estimation error
$\boldsymbol{\varepsilon}_k \in \mathbb{R}^m$	output error (residual)
$\boldsymbol{u}_k \in \mathbb{R}^r$	input vector
$\boldsymbol{d}_k \in \mathbb{R}^q$	unknown input vector, $q \leq m$
$\boldsymbol{w}_k,\ \boldsymbol{v}_k$	process and measurement noise
$\boldsymbol{Q}_k,\ \boldsymbol{R}_k$	covariance matrices of $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$
$\boldsymbol{p}$	parameter vector
$\boldsymbol{f}_k \in \mathbb{R}^s$	fault vector
$\boldsymbol{g}(\cdot),\ \boldsymbol{h}(\cdot)$	non-linear functions
$\boldsymbol{E}_k \in \mathbb{R}^{n \times q}$	unknown input distribution matrix
$\boldsymbol{L}_{1,k},\ \boldsymbol{L}_{2,k}$	fault distribution matrices
$n_{1,y}, \ldots, n_{m,y},\ n_{1,u}, \ldots, n_{m,u}$	maximum lags in the outputs and inputs
$n_t,\ n_v$	number of input-output measurements for identification and validation
n_p	number of populations
n_m	population size
n_d	initial depth of the trees
n_s	tournament population size
$\mathbb{T},\ \mathbb{F}$	terminal and function sets

Abbreviations

ANN	Artificial Neural Network
BEUIO	Bounded-error EUIO
EA	Evolutionary Algorithm
EKF	Extended KF
EUIO	Extended UIO
FDI	Fault Detection and Isolation
GA	Genetic Algorithm
GP	Genetic Programming
GMDH	Group Method of Data Handling
KF	Kalman Filter
MISO	Multi-Input Single-Output
MIMO	Multi-Input Multi-Output
MLP	Multi-Layer Perceptron
RBF	Radial Basis Function
SISO	Single-Output Single-Input
UIF	Unknown Input Filter
UIO	Unknown Input Observer

INTRODUCTION

It is well known that there is an increasing demand for modern systems to become more effective and reliable. This real world's development pressure has transformed automatic control, initially perceived as the art of designing a satisfactory system, into the modern science that it is today. The observed increasing complexity of modern systems necessitates the development of new control techniques. To tackle this problem, it is obviously profitable to have all the knowledge concerning a system behaviour. Undoubtedly, an adequate model of a system can be a tool providing such knowledge. Models can be useful for system analysis, e.g. to predict or to simulate a system behaviour. Indeed, nowadays, advanced techniques for designing controllers are also based on models of systems. Application of models leads directly to the problem of *system identification*.

The main objective of system identification is to obtain a mathematical description of a real system of interest. In the case of *phenomenological models*, whose structures are built based on physical consideration, i.e. on physical laws governing the system that is being studied, the system identification problem reduces to the parameter estimation one. Given a structure of such a model and knowing that its parameters have a physical meaning, it is possible to predict their nominal values. This possibility extremely facilities parameter estimation, especially for model structures which are non-linear in their parameters. On the other hand, the high complexity of a large majority of real systems makes it impossible to perform physical consideration underlying phenomenological models. In such situations, the *behavioural models*, which merely approximate the system input-output behaviour, have to be employed.

Although the majority of industrial systems are non-linear in their nature, the most common approach to settle the model construction problem is to use the well-known tools for linear systems (Ljung 1987, Nelles 2001, Walter and Pronzato 1997). In spite of the simplicity of such an approach, the use of linear models will usually introduce a degree of approximation. While such an approximation may be fully acceptable in many cases, there are applications for which a detailed description of a system of interest is of great practical importance. This is the main reason for further development of *non-linear system identification* theory. Indeed, a few decades ago, non-linear system identification was a field of several ad-hoc approaches, each applicable only to a very restricted class of systems. With the advent of neural networks, fuzzy models, and modern structure optimisation techniques, a much wider class of systems can be handled.

The most popular classical non-linear identification methods usually employ various kinds of polynomials as a foundation for the model construction procedure, for example, the orthogonal regression estimator (Billings *et al.* 1989), which combines the structure determination with parameter estimation, as well as the Group Method of Data Handling (GMDH), introduced by Ivakhnenko (Farlow 1984, Ivakhnenko 1968). The main advantage of such approaches is that polynomial models are linear in their parameters and this extremely facilities parameter estimation. In spite of the considerable usefulness of such approaches, there are applications for which polynomial models do not give satisfactory results. To overcome this problem, the so-called Soft Computing methods can be employed.

The most popular approach is to use either neural networks (Duch *et al.* 2000, Hertz *et al.* 1991, Nelles 2001) or fuzzy neural networks (Nuck *et al.* 1997, Nelles 2001). Many works confirm their effectiveness and recommend their use. On the other hand, there is no efficient approach to selecting structures of such networks. Thus, many experiments have to be carried out to obtain an appropriate configuration. Another problem arises from dynamic systems modelling. In this context, a network of dynamic neurons or a recurrent network can be used. In both cases the process of training is usually relatively complex. An alternative approach, which seems to avoid these difficulties, is to employ Genetic Programming (GP) (Esparcia-Alcazar 1998, Gray *et al.* 1998, Koza 1992, Witczak and Korbicz 2000a, Witczak and Korbicz 2000b, Witczak and Korbicz 2002). GP is an extension of genetic algorithms (Michalewicz 1996), which are a broad class of stochastic optimisation algorithms inspired by some biological processes, which allow populations of organisms to adapt to their surrounding environment. The main difference between these two approaches is that in GP the evolving individuals are parse trees rather than fixed-length binary strings. The main advantage of GP over neural networks is that the models resulting from this approach are less sophisticated (from the point of view of the number of parameters). This means that those models, in spite of the fact that they are of the behavioural type, are more transparent and hence they provide more information about a system behaviour. Moreover, model structures resulting from this approach can further be reduced in a very intuitive way.

Unlike it has been done in the past, modern control techniques should take into account the system's safety. This requirement goes beyond the normally accepted safety-critical systems of nuclear reactors and aircraft, where safety is of paramount importance, to less advanced industrial systems. Therefore, it is clear that the problem of *fault diagnosis* constitutes an important subject in modern control theory. This is the main reason why the design and application of the model-based fault diagnosis has received considerable attention during the last few decades.

In a fault diagnosis task, the model of the real system of interest is utilised to provide estimates of certain measured and/or unmeasured signals. Then, in the most usual case, the estimates of the measured signals are compared with their originals, i.e. a difference between the original signal and its estimate is used to form a residual signal. This residual signal can then be employed for *Fault Detection and Isolation* (FDI). This means that the problems of system identification and fault diagnosis are closely related.

Irrespective of the identification method used, there is always the problem of model uncertainty, i.e. the model-reality mismatch. Thus, the better the model used to represent a system behaviour, the better the chance of improving the reliability and performance in diagnosing faults. Unfortunately, disturbances as well as model uncertainty are inevitable in industrial systems, and hence there exists a pressure creating the need for robustness in fault diagnosis systems. This robustness requirement is usually achieved in the fault detection stage.

In the context of robust fault detection, many approaches have been proposed ((Chen and Patton 1999, Patton *et al.* 2000) and the references therein). Undoubtedly, the most common one is to use robust observers, such as the Unknown Input Observer (UIO) (Alcorta *et al.* 1997, Chen *et al.* 1996, Chen and Patton 1999, Patton and Chen 1997, Patton *et al.* 2000), which can tolerate a degree of model uncertainty and hence increase the reliability of fault diagnosis. In such an approach, the model-reality mismatch is represented by the so-called unknown input and hence the state estimate and, consequently, the output estimate are obtained taking into account model uncertainty. As in system identification, much of the work in this subject is oriented towards linear systems. This is mainly because of the fact that the theory of observers (or filters in the stochastic case) is especially well-developed for linear systems.

Unfortunately, the existing non-linear extensions of the UIO (Alcorta *et al.* 1997, Chen *et al.* 1996, Chen and Patton 1999, Patton and Chen 1997, Seliger and Frank 2000) require a relatively complex design procedure, even for simple laboratory systems (Zolghardi *et al.* 1996). Moreover, they are usually limited to a very restricted class of systems. One way out of this problem is to employ linearisation-based approaches, similar to the Extended Kalman Filter (EKF) (Anderson and Moore 1979). In this case, the design procedure is as simple as that for linear systems. On the other hand, it is well known that such a solution works well only when there is no large mismatch between the model linearised around the current state estimate and the non-linear behaviour of the system. Thus, the idea is to improve the convergence of linearisation-based observers.

Another problem is that, even for linear systems, the research concerning UIOs is strongly oriented towards deterministic systems. Indeed, the question of detecting and isolating faults for systems with both modelling uncertainty and the noise has not attracted enough research attention, although most fault diagnosis systems suffer from both modelling uncertainty and the noise. The existing approaches (see (Chen and Patton 1999, Chen *et al.* 1996, Keller and Darouach 1999) and the references therein), which can be applied to linear stochastic systems, rely on a similar idea to that of the classical Kalman Filter (KF) (Anderson and Moore 1979). The main drawback to such techniques lies in their restrictive assumptions concerning the noise distribution, i.e. it is assumed that the process and measurement noises are zero-mean white noise sequences. However, in many practical situations it is more natural to assume that only bounds on the noise signals are available (Maksarow and Norton 1996a, Maksarow and Norton 1996b, Milanese *et al.* 1996, Walter and Pronzato 1997). This bounded-error approach describes a set of all states that are consistent with the model, the measured data and the error (or the noise) bounds. All members of this feasible set are then possible solutions to the state estimation problem. Unfortunately, the set obtained in such a way may become extremely complex. For the sake of computational complexity, this feasible set is usually characterised by the smallest (in some sense) ellipsoid that encloses it. Although, in the case of the observers of this type, the so-called unknown input can be treated in a similar way as the process noise, i.e. the only requirements are the bounds of the unknown input, it seems especially attractive to employ the bounded-error approach to design an UIO for linear stochastic systems. This is especially true from the point of view of fault isolation. Indeed, in order to design a fault diagnosis system, which is based on a bank of observers, each of the observers should be insensitive to one fault while sensitive to others. This can be achieved by combining the classical UIO with bounded-error techniques, which results in an observer for a wide class of linear stochastic systems.

Another problem arises from the application of fault diagnosis to non-linear stochastic systems. Unfortunately, the only existing approaches to this class of systems consist in the application of the EKF. Indeed, the non-linear extensions of the UIO (Alcorta *et al.* 1997, Chen *et al.* 1996, Chen and Patton 1999, Patton and Chen 1997, Seliger and Frank 2000) can only be applied to non-linear deterministic systems. Thus, it seems especially attractive to extend the combination of bounded-error techniques and the UIO so that it can be applied to non-linear stochastic systems.

It should also be pointed out that the application of robust observer-based fault diagnosis techniques is usually very limited due to the necessity of having a non-linear state-space model of a system. This is the main reason why this work is concerned with both system identification and robust fault detection.

The objective of this work is twofold. The first subject concerns the application of genetic programming to the design of models of non-linear discrete-time systems. In particular, the problem is to develop algorithms for designing both state-space and input-output models. The second subject focuses on designing robust state estimators for the purpose of fault detection. In particular , the problem is to develop algorithms which allow to design unknown input observers for both deterministic and stochastic non-linear discrete-time systems.

In particular, within the framework of this work, the following problems will be addressed:

Theoretical aspects:

- development of genetic programming-based non-linear system identification algorithms for both input-output and state-space models:
  - stability analysis of state-space models,
  - selection strategies of control parameters,

- development of an unknown input observer for a class of deterministic non-linear discrete-time systems:
    - convergence analysis,
- application of bounded-error techniques to the design of unknown input observers for:
    - discrete-time linear stochastic systems,
    - discrete-time non-linear stochastic systems.

Application aspects:

- system identification based on simulated as well as real-world data from the Lublin sugar factory,
- fault detection of simulated and real systems.

The book is divided into 4 chapters. Chapter 1 presents well-known and frequently applied model structures, from classical approaches to neural networks models. Both static and dynamic models are considered. A detailed analysis of these structures, as well as their drawbacks and advantages, is presented. Model selection techniques are discussed as well. Chapter 2 reviews the most popular model-based residual generation schemes. The main attention of this chapter focuses on the problem of fault detection, which constitutes the most important fault diagnosis stage (without fault detection it is impossible to perform fault isolation). In particular, various model-based fault detection schemes are presented, from simple model-based residual generators to more advanced, robust observer-based approaches. This review concerns fault detection schemes for both linear and non-linear systems. The advantages, drawbacks and possible application areas are discussed as well.

Genetic programming-based system identification techniques are proposed in Chapter 3. This chapter briefly reviews the well-known and frequently applied evolutionary algorithms, with special emphasis put on genetic algorithms and genetic programming. The proposed modification of the genetic programming approach is applied to the dynamic model design. In particular, input-output and state-space identification schemes for dynamic systems are proposed. The stability of models resulting from the state-space identification scheme is considered. The final part of this chapter contains experimental results which confirm the effectiveness of the proposed approach. Selection strategies of the genetic programming algorithm's control parameters are discussed as well.

In Chapter 4, the concept of an extended unknown input observer for non-linear discrete-time deterministic systems is introduced, and then the design algorithm is described in detail. A comprehensive convergence analysis is performed. The obtained results are then employed to increase the convergence rate of the observer. To tackle this problem, a genetic programming-based technique is proposed. The chapter contains numerical simulation results regarding state estimation as well as fault diagnosis of an induction motor.

The chapter also discusses the application of bounded-error state estimation techniques to the design of unknown input observers for both linear and non-linear stochastic systems. The problem of state estimation of linear systems with bounded system and measurement noises is formulated, and a suitable algorithm is given. It is shown how to employ the proposed algorithm to design an unknown input observer for a wide class of linear stochastic systems. Moreover, an extension of the approach being considered which can be applied to non-linear stochastic systems is proposed. The final part of this chapter contains experimental results concerning state estimation as well as fault diagnosis for both linear and non-linear stochastic systems.

The main original achievements of this work and final remarks are presented in Conclusions.

The book was developed partially within the grant *Identification and Fault Detection of Non-linear Dynamic Systems* (2001-2002) of the State Committee for Scientific Research in Poland and within the EU FP 5 Research Training Network project DAMADICS: *Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems* (2000-2003).

Chapter 1

MODEL STRUCTURES

This chapter reviews the most popular and frequently used model structures. Both static and dynamic and linear and non-linear models are considered. Although the work focuses on the identification of dynamic non-linear systems, to make this review self-contained, the well-known static and/or linear models are presented as well. Because dynamic models inherit various properties of their static counterparts, the following structure of the present chapter is justified. Section 1.1 briefly reviews both linear and non-linear model structures for static systems. In particular, the classical linear, look-up table and polynomial models are presented. Finally, as a relatively new identification tool, artificial neural networks are introduced.

Because of the fact that fuzzy logic-based approaches constitute qualitative modelling tools, they are beyond the scope of this chapter.

Section 1.2 is devoted to dynamic systems. Its structure is similar to that of Section 1.1. Well-known linear models open the list of tools for the identification of dynamic systems. In the sequel, various classical (including polynomials) and neural network models of non-linear dynamic systems are presented.

In all the cases drawbacks and advantages are discussed, with special emphasis put on the applicability to real-world identification problems. The stability of dynamic models is examined as well.

1.1. Static models

1.1.1. Linear models

Linear models constitute well-known and frequently used approximation tools. These model structures can be employed for the identification of non-linear static systems only when their non-linear characteristic is weak. The main advantage of linear models is their simplicity. Moreover, they are linear in their parameters. This facilities parameter estimation, which can be realised by the celebrated least-square algorithm (Ljung 1987, Nelles 2001, Walter and Pronzato 1997) or by the bounded-error techniques (Milanese *et al.* 1996, Walter and Pronzato 1997).

A linear model can be written as

$$\hat{y} = \sum_{i}^{r} p_i u_i = \boldsymbol{p}^T \boldsymbol{u}, \quad \text{with } u_0 = 1. \tag{1.1}$$

In spite of the simplicity of linear models, their application is usually very limited. This is the case, for example, in fault detection, where a model of good quality is of great importance.

1.1.2. Polynomial models

Polynomials form a natural extension of linear representation. Indeed, a linear model can be considered as a special case of a polynomial one.

An r-dimensional polynomial of the l-th degree can be written as

$$\begin{aligned}\hat{y} =& \underbrace{p_0 + \sum_{i}^{r} p_i u_i}_{\text{linear part}} + \sum_{i_1=1}^{r}\sum_{i_2=i_1}^{r} p_{i_1 i_2} u_{i_1 i_2} + \cdots + \\ &+ \sum_{i_1=1}^{r} \cdots \sum_{i_l=i_{l-1}}^{r} p_{i_1 \cdots_l} u_{i_1} u_{i_l}.\end{aligned} \tag{1.2}$$

The first two terms in (1.2) describe a linear model while the remaining ones have non-linear properties, e.g. u_1^2, $u_1 u_2$, u_1^l, $u_1^{l-1}u_2$, etc. As can be observed, the r-dimensional polynomial of the l-th degree possesses

$$\begin{aligned}&1 + \begin{pmatrix} r \\ 1 \end{pmatrix} + \begin{pmatrix} r+1 \\ 2 \end{pmatrix} + \cdots + \begin{pmatrix} r+l-1 \\ l \end{pmatrix} = \\ &= \sum_{i=1}^{l} \frac{(r+i-1)!}{i!(r-1)!} = \frac{(r+l)!}{r!l!}\end{aligned} \tag{1.3}$$

parameters, i.e. $\dim(\boldsymbol{p}) = (r+l)!/(r!l!)$. For example, the 2-dimensional polynomial of the 3-rd degree can be written as follows:

$$\begin{aligned}\hat{y} =& p_0 + p_1 u_1 + p_2 u_2 + p_{11} u_1^2 + p_{12} u_1 u_2 + p_{22} u_2^2 \\ &+ p_{111} u_1^3 + p_{112} u_1^2 u_2 + p_{122} u_1 u_2^2 + p_{222} u_2^3,\end{aligned} \tag{1.4}$$

or in a more convenient form:

$$\hat{y} = \boldsymbol{p}^T \boldsymbol{z}, \tag{1.5}$$

where $\boldsymbol{p} = (p_0, p_1, p_2, p_{11}, p_{12}, p_{22}, p_{111}, p_{112}, p_{122}, p_{222})$ and $\boldsymbol{z} = (1, u_1, u_2, u_1^2,$ $u_1 u_2, u_2^2, u_1^3,$ $u_1^2 u_2, u_1 u_2^2, u_2^3)$. Similarly, the polynomial (1.2) can be expressed in the form (1.5). Like linear models, polynomials are linear in their parameters, and hence parameter estimation can be realised with the least-squares (Ljung 1987, Nelles 2001, Walter and Pronzato 1997) or bounded-error (Milanese *et al.* 1996, Walter and Pronzato 1997) algorithms. On the other hand, the number of parameters (1.3) grows significantly with an increase in the number of system inputs r and/or in the degree of a polynomial. To overcome this problem, efficient algorithms such as the orthogonal least squares (Billings *et al.* 1989) or the GMDH (Farlow 1984, Ivakhnenko 1968) can be introduced. The orthogonal least squares method can automatically select the relevant terms from a full polynomial, and leads to a reduced polynomial model with significantly fewer parameters. The GMDH approach decomposes the model construction task into simple sub-tasks. Each sub-task consists in determining an elementary model, e.g. a 2-dimensional polynomial of the 2-nd degree. In the case of a polynomial of the 2-nd degree or other 2-dimensional models, the structure of the GMDH algorithm can be as follows (cf. Fig. 1.1):

Step 1 : Determine all elementary models whose inputs consist of all the possible couples of input variables, i.e. $(r-1)r/2$ couples (elementary models).

Step 2 : Using a new data set (not employed during the model determination phase), select several elementary-models which are best-fitted in terms of the criterion chosen.

Step 3 : If the termination condition is reached (one of the models fits the data with a desired accuracy, or the introduction of new elementary models did not cause a significant increase in the approximation abilities of the whole model in terms of the criterion chosen), then STOP, otherwise use the outputs of the best-fitted elementary models (selected in *Step 2*) to form the input vector, and then go to *Step 1*.

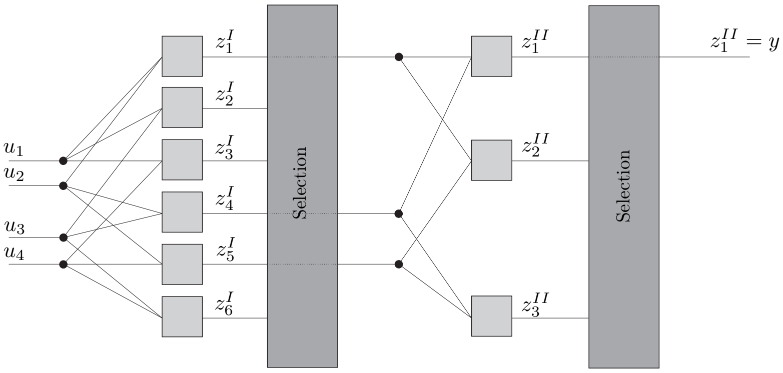

Fig. 1.1 . The principle of the GMDH algorithm.

Example 2.1. The problem is to model the following relation:

$$y = -\frac{1}{0.1 + u^2}, \tag{1.6}$$

with a polynomial (and a linear function as a special case), given a set of input-output measurements $\{u_i, y_i\}_{i=1}^{n_t}$, where u_i is generated according to the uniform distribution, i.e. $u_i \in \mathcal{U}(-2, 2)$, and $n_t = 100$. The next task is to test the effectiveness of the obtained model with a validation data set $\{u_i, y_i\}_{i=1}^{n_v}$, where $u_i \in \mathcal{U}(-4, 4)$ and $n_v = 100$.

To tackle the above problem, polynomials of a degree between 1 to 4 were derived using the least-squares method. Then they were tested with the validation data set, and the results are shown in Fig. 1.2.

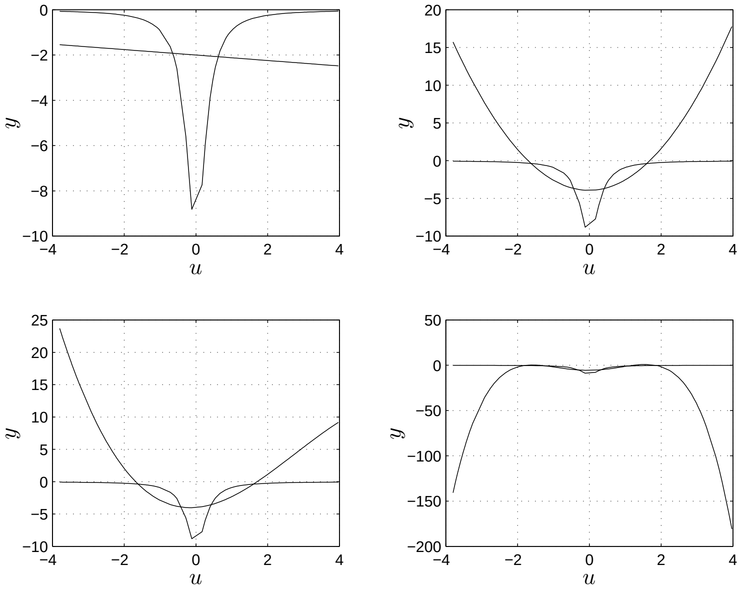

Fig. 1.2. $y = -1/(0.1 + u^2)$ and the corresponding polynomial models (1st degree (top-left), 2nd degree (top-right), 3rd degree (bottom-left), and 4th degree (bottom-right) polynomials).

In spite of the simplicity of linear and polynomial models, this example shows obviously that the accuracy of linear models is typically low, i.e. the stronger the non-linear characteristic of the system, the lower the accuracy. As the name "linear" suggests, their interpolation as well as extrapolation behaviour is linear. The accuracy of polynomial models is usually limited since high-degree polynomials are not practicable. As can be observed in Fig. 1.2, the interpolation behaviour is strongly related to the degree of a polynomial. The extrapolation behaviour depends on the highest-order terms, i.e. polynomials tend to $+\infty$ or $-\infty$ with a rate determined by them. Those drawbacks of polynomials have led to the development of *splines*, which are locally defined low-degree polynomials.

1.1.3. Look-up tables

Look-up tables (Nelles 2001) are the most common static non-linear models used in practical implementations. Although they are limited to problems with one or two-dimensional input spaces, their popularity comes from their simplicity and the fact that they require low computational burden. Moreover, in most applications where look-up tables are employed, the identification procedure boils down to storing the data, i.e. no “real” optimisation methods are required. On the other hand, the application of look-up tables is limited to low-dimensional problems. This is mainly because only one- and two-dimensional mappings can be visualised. Another reason is that, unlike low-dimensional mappings, high-dimensional mappings cannot be realised with grid-based look-up tables (*curse of dimensionality*). This leads directly to various decomposition techniques.

In this section, the consideration is limited to one-dimensional look-up tables, since two-dimensional look-up tables are their straightforward extensions.

Let us consider a one-dimensional look-up table shown in Fig. 1.3. It consists of eight points (u_i, y_i), i.e. for eight input values u_i, $i = 1, \ldots, 8$ the corresponding output values y_i, $i = 1, \ldots, 8$ are stored in this look-up table. The output of one-dimensional look-up table is given as:

$$\hat{y} = \frac{y_{\text{left}}(u_{\text{right}} - u) + y_{\text{right}}(u - u_{\text{left}})}{u_{\text{right}} - u_{\text{left}}}, \tag{1.7}$$

where $(u_{\text{left}}, y_{\text{left}})$ and $(u_{\text{right}}, y_{\text{right}})$ are the closest points to the left and right of u, respectively. If u has either no left or no right neighbour, the output $\hat{y}$ of a look-up table is not defined. However, any kind of extrapolation behaviour can be realised by keeping the output of a look-up table constant over the range $u_{\text{min}}, \ldots, u_{\text{max}}$. Look-up tables can also be expressed in an equivalent form using triangular basis

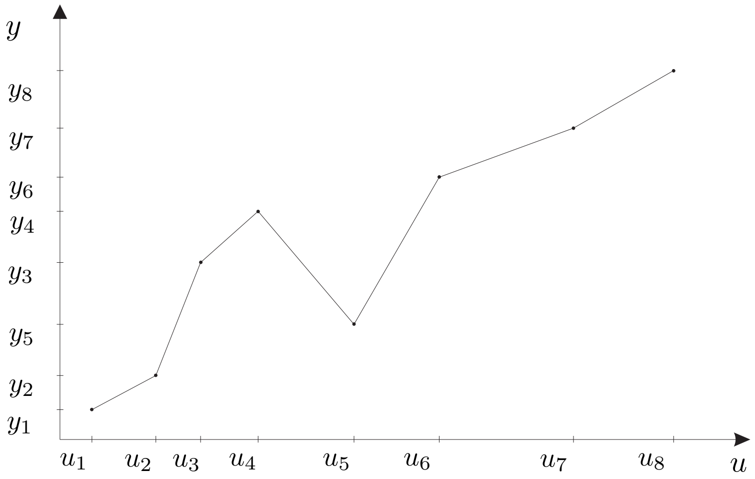

Fig. 1.3 . An exemplary one-dimensional look-up table.

functions (Fig. 1.4), i.e.

$$\hat{y} = \sum_{i=1}^{n_t} y_i \phi_i(u, \bar{\boldsymbol{u}}), \tag{1.8}$$

where $\bar{\boldsymbol{u}} = (u_1, \ldots, u_{n_t})$ is composed of n_t look-up table points. The u_i, $i = 1, \ldots, n_t$ represent centres of the basis functions. The basis functions can be written as

$$\phi_i(u, \bar{\boldsymbol{u}}) = \begin{cases} (u - u_{i-1})/(c_i - c_{i-1}) & \text{for } c_{i-1} \leq u \leq c_i, \\ (u - u_{i+1})/(c_i - c_{i+1}) & \text{for } c_i \leq u \leq c_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{1.9}$$

It should be pointed out that such a basis function, similarly to (1.7), realises the linear interpolation behaviour of the look-up table. Moreover, the expression (1.8) is linear with respect to y_i and non-linear with respect to u_i. This relationship does not matter as long as both u_i and y_i represent the input-output measurements. However, when look-up tables are to be optimised, this issue becomes important. Finally, it is worth noticing that basis functions may have more sophisticated shapes than the triangular ones, e.g. higher-order splines or Gaussian functions.

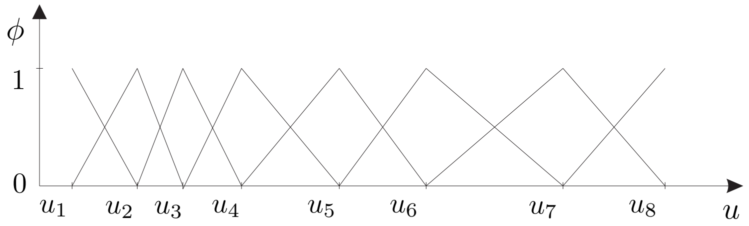

Fig. 1.4 . Basis functions corresponding to the look-up table.

1.1.4. Neural networks

This section reviews the well-known and frequently used Artificial Neural Networks (ANNs) which can be employed to the identification of static non-linear systems. In particular, the so-called feed-forward networks such as Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) networks are considered.

1.1.4.1. Multi-layer perceptron

Artificial neural networks consist of a number of sub-units called neurons. The classical neuron structure (Fig. 1.5) can be described by:

$$y = f\left(\sum_{i=0}^{r} p_i u_i\right) = f(\boldsymbol{p}^T \boldsymbol{u}), \text{ with } u_0 = 1, \tag{1.10}$$

where $f(\cdot)$ stands for the so-called activation function, and, as usual, $\boldsymbol{p}$ is the parameter (or weight) vector to be estimated. It is obvious that the behaviour of the neuron (1.10) depends on the activation function $f(\cdot)$. There are, of course, many different functions which can be employed to settle this problem. The simplest choice is to use a linear activation function resulting in the so-called *Adaline* neuron (Hertz *et al.* 1991). Nevertheless, real advantages of neural networks can be fully exploited when activation functions are non-linear. Typically, the activation function is chosen to be of the saturation type. The common choice is to use sigmoidal functions such as the logistic

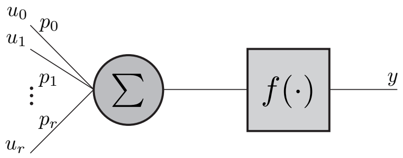

Fig. 1.5 . The classical neuron structure.

$$f(x) = \text{logistic}(x) = \frac{1}{1+\exp(-x)} \tag{1.11}$$

and the hyperbolic tangent

$$f(x) = \tanh(x) = \frac{1-\exp(-2x)}{1+\exp(-2x)} = 2\text{logistic}(2x) - 1 \tag{1.12}$$

functions. As can be seen from (1.12), the functions can be transformed into each other. Moreover, these two functions share an interesting property, namely, that their derivatives can be expressed as simple functions of the outputs. As long as a gradient-based algorithm is used to obtain parameter estimates, this property leads to a significant decrease in the computational burden, thus making the network synthesis process more effective.

The multi-layer perceptron is a network consisting of neurons divided into the so-called layers (Fig. 1.6). Such a network possesses an input layer, one or more hidden layers, and an output layer. The main tasks of the input layer are data preprocessing (e.g. scaling, filtering, etc.) and passing the input signals into the hidden layer. Therefore, only the hidden and output layers constitute a “true” model. The connections between neurons are designed in such a way that each neuron of the former layer is connected with each element of the succeeding one. The non-linear neural network model (cf. Fig. 1.6) can symbolically be expressed as follows:

$$y = \boldsymbol{g}_3\left(\boldsymbol{g}_2\left(\boldsymbol{g}_1\left(\boldsymbol{u}\right)\right)\right), \tag{1.13}$$

where $\boldsymbol{g}_1(\cdot)$, $\boldsymbol{g}_2(\cdot)$, and $\boldsymbol{g}_3(\cdot)$ stand for the operators defining signal transformation through the 1st, 2nd, and output layers, respectively.

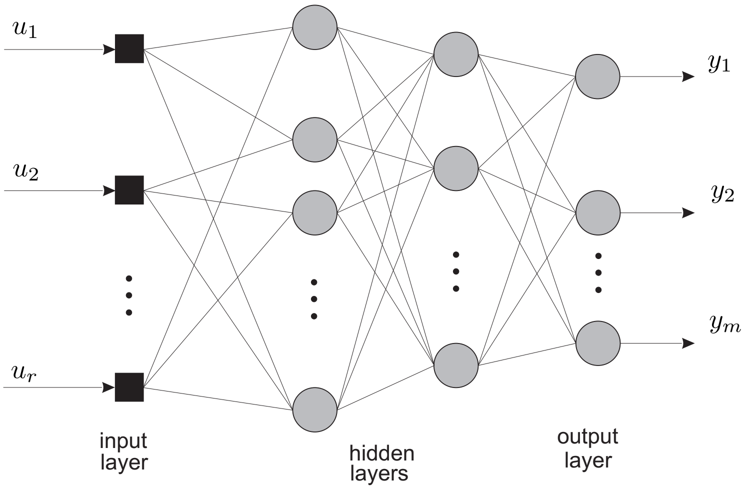

Fig. 1.6 . An exemplary multi-layer perceptron with 3 layers.

One of the fundamental advantages of neural networks is their learning and adaptational abilities. An MLP network is a *universal approximator* (Hornik *et al.* 1989). This means that the MLP can approximate any smooth function with an arbitrary degree of accuracy as the number of hidden layer neurons increases. From the technical point of view, the training of neural networks is nothing else but parameter estimation. Indeed, once the structure of a network is known, the remaining task is to obtain the parameter vector $\boldsymbol{p}$. To tackle this problem, the celebrated back-propagation algorithm (Duch *et al.* 2000, Hertz *et al.* 1991) can be employed. Other possibilities involve the application of various stochastic (Walter and Pronzato 1997) or evolutionary algorithms (Duch *et al.* 2000). These approaches should be adopted when classical gradient-based algorithms fail to converge to the satisfactory results. This is, however, a common situation, owing to the multimodal character of the optimisation index. Another problem may occur because of a large number of parameters to be estimated. This is especially true for neural networks with many hidden layers.

The main drawback to neural networks arises from a model structure selection. There are, of course, many more or less sophisticated approaches to this problem, and they can be divided into three classes:

1. Bottom-up approaches: starting with a relatively simple structure, the number of hidden neurons is increased,

2. Top-down approaches: starting from a relatively complex structure, which seems to be sufficient to solve an identification problem, the "excessive" neurons are removed,

3. Discrete optimisation methods: with each network structure an evaluation

value is associated, and then the network structure space is explored to find an appropriate configuration.

Unfortunately, the efficiency of those algorithms is usually very limited. As a result, neural networks with very poor generalisation abilities are obtained. Another drawback to neural networks is that models resulting from this approach are not in "a human readable" form. This means that the structure of such a network is not able to provide any practical knowledge about that of a real system. Indeed, according to the literature, neural networks are called "black boxes".

1.1.4.2. Radial basis function networks

Radial basis function networks as rivals of arduously learning multi-layer perceptrons have received considerable research attention in recent years (Duch *et al.* 2000, Nelles 2001). This kind of networks requires many nodes to achieve satisfactory results. The problem is somewhat similar to that of selecting an appropriate structure of a multi-layer perceptron. The RBF (Radial Basis Function) network (Fig. 1.7) consists of three layers, namely, the input layer, one hidden layer, and the output layer. The output ϕ_i of the i-th neuron of the hidden layer is a non-linear function of the Euclidean distance from the input vector $\boldsymbol{u}$ to the vector of centres $\boldsymbol{c}_i$ and can be expressed as follows:

$$\phi_i = f\left(\|\boldsymbol{u} - \boldsymbol{c}_i\|_2\right), \ i = 1, \ldots, n_h, \tag{1.14}$$

where $\|\cdot\|_2$ stands for the Euclidean norm, and n_h is the number of neurons in the hidden layer. The j-th network output is a weighted sum of the hidden neurons' output:

$$y_j = \sum_{i=1}^{n_h} p_{ji}\phi_i. \tag{1.15}$$

The activation function $f(x)$ is usually chosen to posses a local character and a maximum at $x = 0$. Typical choices for the activation function are the Gaussian function:

$$f(x) = \exp\left(-\frac{x^2}{\rho^2}\right), \tag{1.16}$$

and the inverse multi-quadratic function:

$$f(x) = \frac{1}{\sqrt{x^2 + \rho^2}}, \tag{1.17}$$

where ρ signifies an additional free parameter.

The fundamental task in designing RBF networks is the selection of the number of hidden neurons, and the activation function type. Then, the function centres and their positions should be chosen. In this context, it should be pointed out that too small a number of centres may result in poor approximation properties.

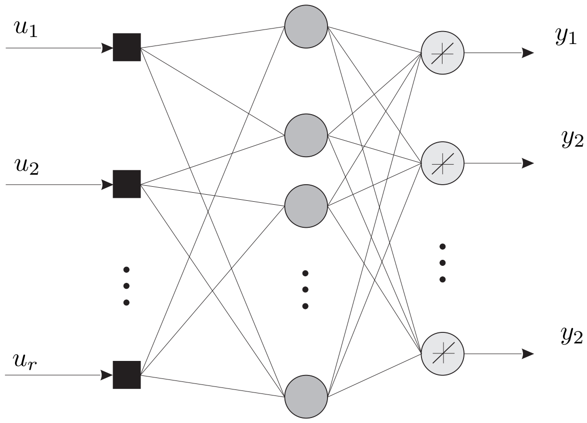

Fig. 1.7. An exemplary radial basis function networks.

On the other hand, the number of exact centres increases according to the dimension of the input space. For the sake of this, the application of RBF networks is rather restricted to low-dimensional problems.

The typical strategy for the RBF network training consists in exploiting the linearity of the output layer parameters (weights) and the geometric interpretability of the hidden layer parameters. The hidden layer parameters are determined first and, subsequently, the output layer parameters are obtained by means of some well-known approaches for linear parameter estimation, e.g. by the least-squares algorithm. There are, of course, many more or less sophisticated approaches to selecting the centres and the widths of the basis function. The simplest one consists in randomly selecting these parameters; however, this is not a really practical approach. More efficient and, of course, more sophisticated approaches rely on the application of clustering, grid-based and subset selection techniques, as well as non-linear optimisation (see (Nelles 2001) for a survey).

1.2. Dynamic models

Like static models, their dynamic counterparts are commonly employed for forecasting the behaviour of a system. Contrary to the case of static models, two situations have to be distinguished here, i.e. *simulation* and *prediction.* If the response of the model output has to be calculated while the system output is unknown (or is not considered as an additional source of information), we deal with *simulation.* On the other hand, if the system output is known up to some time instant, i.e. $k-1$, and the model output l steps is asked for in the future, we deal with *prediction.* The choice of a suitable configuration depends on the application.

The present section is organised in a similar way as Section 1.1. It starts from relatively simple linear models and goes to polynomials and other classical approaches to end up with advanced neural network architectures.

1.2.1. Linear models

In this section, various structures of linear dynamic models are briefly reviewed. In particular, a general model structure (in an input-output configuration) is presented. The remaining models are derived from this general framework. The state-space linear models are presented as well.

The general model describing deterministic (u_k) as well as stochastic (v_k) influences can be given as follows (Nelles 2001) (Fig. 1.8a):

$$y_k = \frac{B(q)}{F(q)A(q)}u_k + \frac{C(q)}{D(q)A(q)}v_k, \tag{1.18}$$

where (in this section) q stands for the forward shift operator, i.e. $q^{-1}u_k = u_{k-1}$. The terms $B(q)/(F(q)A(q))$ and $C(q)/(D(q)A(q))$ are called the input and noise transfer functions, respectively (since they relate the input u_k and the noise v_k to the output y_k, respectively). The notation of transfer functions and the assumption regarding the polynomial structure of $A(q), \ldots, F(q)$ have been accepted as standard since the publication of Ljung's book (Ljung 1987). Subsequently, by making special assumptions concerning $A(q), \ldots, F(q)$, the widely-known linear dynamic models are derived. For the sake of brevity, models without the output feedback and models discarding the input u_k are not considered.

One of the most frequently used system description is the ARX (AutoRegressive with eXogenous input) model (Fig. 1.8b). It can be obtained by setting $C(q) = D(q) = F(q) = 1$ in a general model structure (1.18), i.e.

$$y_k = \frac{B(q)}{A(q)}u_k + \frac{1}{A(q)}v_k. \tag{1.19}$$

In this case, deterministic and stochastic parts of the ARX model posses identical denominator dynamics. This fact can be interpreted in another way by considering $1/A(q)$ as a noise model. Such a situation may be fully acceptable if the noise enters the process early.

Another possible system description is the ARMAX (AutoRegressive Moving Average with eXogenous input) model (Fig. 1.8c):

$$y_k = \frac{B(q)}{A(q)}u_k + \frac{C(q)}{A(q)}v_k. \tag{1.20}$$

As with ARX models, in the case of ARMAX models it is assumed that there is identical denominator dynamics for the input and noise transfer functions. Unlike in the ARX model, here the noise transfer function is more flexible owing to the moving average polynomial $C(q)$.

The ARARX (AutoRegressive AutoRegressive with eXogenous input) model (Fig. 1.8d) is another simplification of the general structure (1.18), and can be expressed as:

$$y_k = \frac{B(q)}{A(q)}u_k + \frac{1}{D(q)A(q)}v_k. \tag{1.21}$$

Unlike in the previous structures, in this case an additional flexibility in the denominator of the noise transfer is introduced. Without this modification the ARARX model is simply the ARX model.

In order to complete the list of models possessing the same denominator dynamics in the input and noise transfer functions, the ARARMAX (AutoRegressive AutoRegressive Moving Average with eXogenous input) model can be defined as

$$y_k = \frac{B(q)}{A(q)} u_k + \frac{C(q)}{D(q)A(q)} v_k. \tag{1.22}$$

It should be pointed out that, owing to its complexity, this model structure is rarely used in practice.

Another class of models, which do not share a common denominator in the input and noise transfer functions, are OE (Output Error) models (Fig. 1.8e):

$$y_k = \frac{B(q)}{F(q)} u_k + v_k. \tag{1.23}$$

Unlike in the case of linear dynamic models presented above, here the noise disturbs the system additively at the output, not somewhere inside the system.

A modified version of the OE model possessing an additional degree of freedom for the noise model is called the BJ (Box-Jenkins) model (Fig. 1.8f):

$$y_k = \frac{B(q)}{F(q)} u_k + \frac{C(q)}{D(q)} v_k, \tag{1.24}$$

Out of all linear models discussed so far the BJ model is the most general and flexible one. It allows to estimate separate transfer functions with arbitrarily defined numerators and denominators. On the other hand, such a flexibility leads to a large number of parameters to be estimated. This is the main reason why the BJ model is rarely used in practice. Finally, it should be pointed out that the stability of linear models can easily be attained by setting the coefficients of the polynomial $A(q)$ (or $F(q)$) in such a way that its roots lie inside a unit circle.

1.2.1.1. State-space models

Instead of applying the input-output configuration, the so-called state-space model can be employed. The linear state space model takes the following form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{w}_k, \tag{1.25}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}\boldsymbol{x}_{k+1} + \boldsymbol{v}_{k+1}. \tag{1.26}$$

The most straightforward way to produce a state-space model is to obtain an input-output model, i.e. an OE model (1.23). Neglecting the influence of the process noise $\boldsymbol{w}_k$ ($\boldsymbol{w}_k = \boldsymbol{0}$), the OE model can be expressed in a canonical state-space

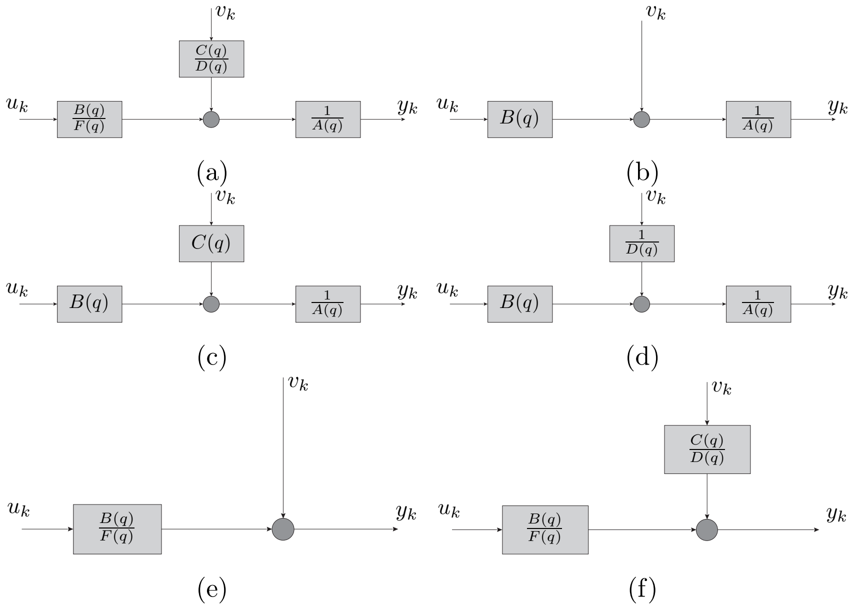

Fig. 1.8 . Elementary models for linear dynamic systems: General (a), ARX (b), ARMAX (c), ARARX (d), OE (e), and BJ (f).

form:

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 1 \\ -f_{n_y} & -f_{n_y-1} & \cdots & -f_1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u_k, \tag{1.27}$$

$$y_{k+1} = [b_{n_u}, b_{n_u-1}, \ldots, b_1]\boldsymbol{x}_{k+1} + v_{k+1}, \tag{1.28}$$

where f_i and b_i stand for coefficients of the polynomials $F(q)$ and $B(q)$, respectively. In the state-space form, the relationship between the input, noise, and the output is written as a system of first-order difference equations (or differential equations in the continuous case) using an auxiliary state vector $\boldsymbol{x}_k$. If all the states x_i, $i = 1, \ldots, n$ are measurable, then the parameter matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ can be easily estimated by well-known linear optimisation techniques. However, in most cases the state vector $\boldsymbol{x}_k$ is either unmeasurable or partially measurable, i.e. only few state variables can be measured. To obtain the linear state-space model, the effective sub-space algorithms can be applied, e.g. *N4SID*, described in (Van Overschee and De Moor 1994).

The state-space description became a dominating approach after the works of Kalman (1961) and Luenberger. This is mainly because of the observed increasing applicability of observers (or filters in the stochastic case) in modern control systems.

The physical relations underlying the system studied can usually be incorporated into the state-space models more easily than into the input-output models. The number of variables is usually smaller in the state-space representation than in the input-output one. For a system of the n-th order, a state-space model possesses $n + r$ ($\boldsymbol{u}_k \in \mathbb{R}^r$) variables, while an input-output model requires $r \times n_u + n_y \times m$. Another advantage of state-space models is that, instead of selecting the lags of input n_u and output n_y signals, the problem reduces to obtaining the dimension n of the state vector $\boldsymbol{x}_k$. This is especially important for non-linear MIMO systems. Unlike in input-output approaches, the identification of MIMO systems can be performed relatively easily. Indeed, as it seems quite unproblematic to obtain the MISO input-output model, the identification of MIMO systems usually requires a model decomposition, i.e. the MIMO model consists of various compositions of MISO models. Another difficulty arises from the dynamics order determination. The common approach is to obtain a set of candidate models of different orders and to select the best one (in terms of the criterion chosen). This is mainly due to the relatively small computational cost devoted to a linear dynamic model determination. There are, of course, many more sophisticated solutions to this problem, e.g. (Ljung 1987, Chapter 16, p. 413).

Like in the case of linear input-output models, when dealing with linear state-space models the stability can easily be attained by ensuring that the eigenvalues of the matrix $\boldsymbol{A}$ lie inside a unit circle.

1.2.2. Polynomial and other classical non-linear models

Non-linear state-space models can be defined analogously to their linear counterparts, described in the previous section. All non-linear input-output models can be written in the form:

$$\hat{y}_k = g(\boldsymbol{\phi}_k), \tag{1.29}$$

where $\boldsymbol{\phi}_k$ may contain the previous or current system input u_k, the previous system or model output (y or $\hat{y}$), and the previous prediction error. The model structure depends on the choice of $\boldsymbol{\phi}_k$, e.g. for the NOE (Nonlinear OE) model, the vector $\boldsymbol{\phi}_k$ takes the form $\boldsymbol{\phi_k} = (u_k, \ldots, u_{k-n_u}, \hat{y}_{k-1}, \ldots, \hat{y}_{k-n_y})$.

One drawback of non-linear dynamic input-output models arises while selecting the dynamics order. Indeed, the dynamics order is crucial for the identification performance, and no really efficient methods for its determination are available. Even if it is assumed that $n = n_y = n_u$, the user is left with a time-consuming trial-and-error approach. This becomes particularly bothersome when different input n_u and output n_y orders are considered. The situation grows complicated when MISO systems are considered.

Similarly to linear dynamic input-output models, their non-linear counterparts can be applied to the identification of MIMO systems. This usually requires a model decomposition, i.e. instead of an MIMO model, various compositions of MISO models are employed.

Another disadvantage of that kind of models is that, in general, their stability cannot be proven. The only way to check the stability is to use time-consuming Monte Carlo methods.

The most straightforward way to realise the non-linear relationship $f(\cdot)$ in (1.29) is to use polynomials. As has been analyzed in Section 1.1.2, polynomials suffer from undesirable properties concerning their interpolation and extrapolation behaviour as well as the curse of dimensionality. Hence, the same consequences arise for dynamic polynomial models.

Kolmogorov-Gabor models constitute the most general class of polynomial dynamic models. As an example, let us consider a second-order model ($n_y = n_u = 2$) derived from a polynomial of the second degree:

$$\begin{aligned} y_k =& p_1 + p_2 u_{k-1} + p_3 u_{k-2} + p_4 y_{k-1} + p_5 y_{k-2} \\ & + p_6 u_{k-1}^2 + p_7 u_{k-2}^2 + p_8 y_{k-1}^2 + p_9 y_{k-2}^2 \\ & + p_{10} u_{k-1} u_{k-2} + p_{11} u_{k-1} y_{k-1} + p_{12} u_{k-1} y_{k-2} \\ & + p_{13} u_{k-2} y_{k-1} + p_{14} u_{k-2} y_{k-2} + p_1 5 y_{k-1} y_{k-2}. \end{aligned} \tag{1.30}$$

As can be observed in (1.30), the complexity of the Kolmogorov-Gabor model increases strongly in accordance with an increase in the the dynamics order and/or in the degree of the polynomial. This results in a large number of parameters to be estimated. To overcome such a problem, techniques like orthogonal regression estimator (Billings *et al.* 1989) can be employed. They allow one to construct a reduced polynomial model that contains only the most relevant terms.

The parametric Volterra-series model constitutes a remedy for the complexity of the Kolmogorov-Gabor polynomial. It realises the following relation:

$$y_k = g(u_{k-1}, \dots, u_{k-n_u}) + p_1 y_{k-1} + \cdots + p_{n_y} y_{k-n_y}. \tag{1.31}$$

With this simplification, the model becomes linear with respect to y_{k-i} and hence its stability can be easily proven. A loss of generality is a consequence of such a reduction. As an example, let us consider a second-order model ($n_y = n_u = 2$) and a polynomial of the second degree:

$$\begin{aligned} y_k =& p_1 + p_2 u_{k-1} + p_3 u_{k-2} + p_4 y_{k-1} + p_5 y_{k-2} \\ & + p_6 u_{k-1}^2 + p_7 u_{k-2}^2 + p_8 u_{k-1} u_{k-2}. \end{aligned}$$

Another important class of non-linear input-output models are Hammerstein models. They constitute probably the most widely known and applied input-output description. Such a model is composed of two sub-models connected in a cascade (Fig. 1.9). The former is static and non-linear and the latter is dynamic and linear:

$$y_{n,k} = g(u_k), \tag{1.32}$$

$$y_k = b_1 y_{n,k-1} + \cdots + b_{n_u} y_{n,k-n_u} - a_1 y_{k-1} - \cdots - a_{n_y} y_{k-n_y}. \tag{1.33}$$

Such a structure is able to describe all systems for which the actuators nonlinearity (characteristics of a valve, saturation of an electro-magnetic motor, etc.) is dominant and other non-linear effects can be neglected. The stability of such a model can easily be proven. Indeed, since the linear sub-model is stable, the entire model is stable as well. On the other hand, the structural assumptions regarding the Hammerstein model are very restrictive and hence it can be applied to a very limited class of systems. The non-linearity $g(\cdot)$ in (1.33) can be realised with, for example, polynomials, neural networks, nonparametric approaches (Greblicki 1996), etc.

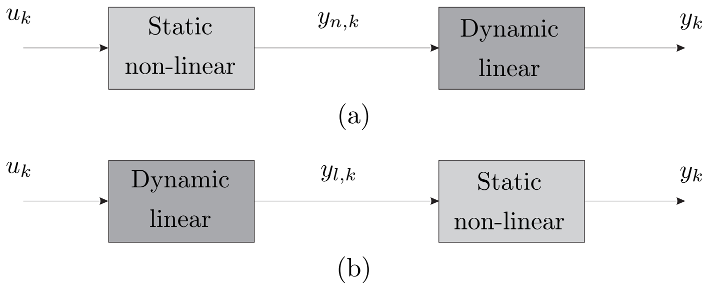

Fig. 1.9 . Hammerstein (a) and Wiener (b) models.

Reversed Hammerstein models, i.e. Wiener models, form another important class of models:

$$y_l = b_1 u_{k-1} + \cdots + b_{n_u} u_{k-n_u} - a_1 y_{l,k-1} - \cdots - a_{n_y} y_{l,k-n_y}, \qquad (1.34)$$

$$y_k = g(y_{l,k}). \qquad (1.35)$$

As in the case of Hammerstein models, here polynomials can be used for the approximation of $g(\cdot)$, but any other approximator or even non-parametric approaches (Greblicki 1994, Greblicki 2001) can be applied.

1.2.3. Neural networks

1.2.3.1. Recurrent networks

This section considers the application of neural networks to the identification of non-linear dynamic systems. Similarly to polynomials and other classical approaches, neural networks can be modified in such a way that they can be useful to identify a dynamic system. It can be achieved by introducing tapped delay lines into the model (Fig. 1.10). The multi-layer perceptron or the radial basis function network can be employed as the main part of the overall model.

A recurrent network developed by Williams and Zipser (1989) consists of n fully connected neurons, r inputs and m outputs (Fig .1.11). As can clearly be seen, such a network has no feed-forward architecture. The main disadvantages of such

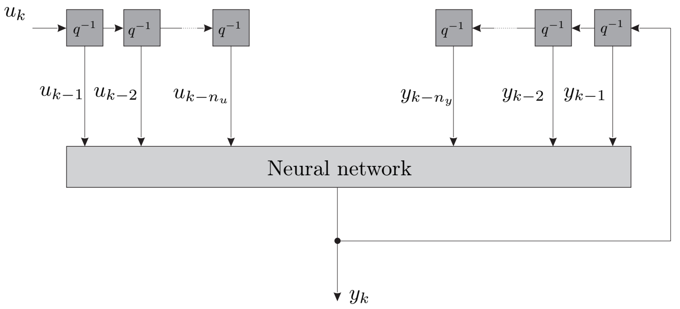

Fig. 1.10 . A neural network with tapped delay lines.

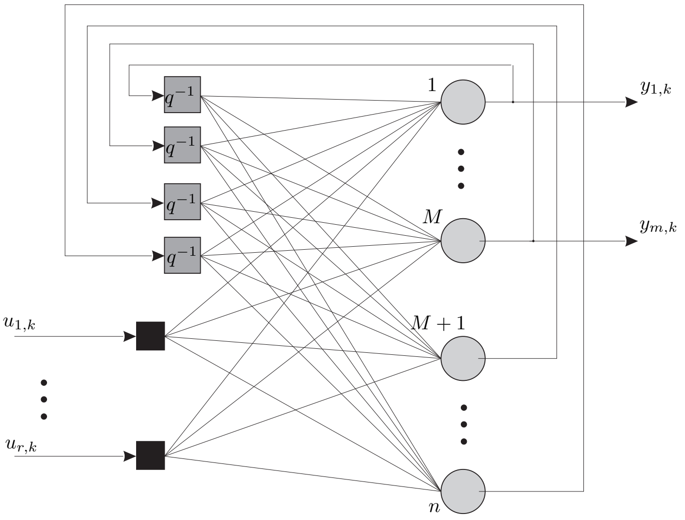

Fig. 1.11 . A recurrent neural network developed by Williams and Zipser.

networks are caused by the slow convergence of the existing training algorithms as well as stability problems.

Contrary to the above-mentioned fully recurrent structures, partially recurrent networks are based on feed-forward multi-layer perceptrons containing the so-called context layer, as in the case of the Elman neural network (Fig. 1.12) (Nelles 2001). In such a network, the feedback connections are realised from the hidden or output layers to the context neurons. The recurrency is more structured, which leads to the faster training. As in the case of recurrent networks and most of other approaches, the disadvantages of partially recurrent neural networks arise from the model order selection as well as stability.

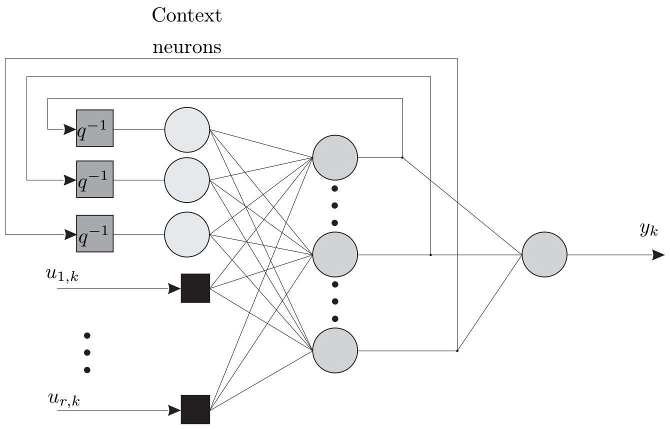

Fig. 1.12 . A partially recurrent Elman neural network.

1.2.3.2. Locally recurrent globally feed-forward networks

In the case of fully or partially recurrent neural networks, all or selected connections are allowable. All neurons are similar to those of static networks, i.e. they are static (no feedback within a neuron). Those global connections cause various disadvantages, e.g. the lack of stability. An alternative solution is to introduce dynamic neurons into the feed-forward network. There are, of course, many different neuron models which can be employed for that purpose. The best known architectures are:

- Neurons with local activation feedback (Fasconi *et al.* 1992)

$$y_k = g\left(y_{l,k}\right), \; y_{l,k} = \sum_{i=1}^{r} p_i u_{i,k} + \sum_{i=1}^{n_y} p_{r+i} y_{l,k-i}, \tag{1.36}$$

- Neurons with local synapse feedback (Back and Tsoi 1991)

$$y_k = g\left(\sum_{i=1}^{r} G_i(q) u_{i,k}\right), \; G_i(q) = \frac{\sum_{j=0}^{n_u} b_j q^{-1}}{\sum_{j=0}^{n_y} a_j q^{-1}}, \tag{1.37}$$

- Neurons with output feedback (Gori *et al.* 1989)

$$y_k = g\left(\sum_{i=1}^{r} p_i u_{i,k} + \sum_{i=1}^{n_y} p_{r+i} y_{k-i}\right), \tag{1.38}$$

- Neurons with an infinite impulse response (IIR) filter (Ayoubi 1994, Patan 2000, Patton and Korbicz 1999)

$$y_k = f\left(y_{l,k}\right), \; y_{l,k} = \sum_{i=0}^{n_u} b_i z_{k-1} + \sum_{i=1}^{n_y} a_i y_{l,k-1},$$

$$z_k = \sum_{i=1}^{r} p_i u_{i,k}. \tag{1.39}$$

The main advantage of locally recurrent globally feed-forward networks is that their stability can be proven relatively easily. As a matter of fact, the stability of the network depends only on the stability of the neurons. In most cases the stability conditions of a neuron boil down to checking the stability of a linear submodel. The feed-forward structure of such networks seems to make the training process easier. On the other hand, the introduction of dynamic neurons increases the parameter space significantly. This drawback together with the non-linear and multi-modal properties of an identification index implies that parameter estimation (or training) becomes relatively complex. To overcome this problem, Mrugalski and Witczak (2002) developed a new neural network synthesis algorithm. In particular, they employed the GMDH approach together with neurons with an infinite impulse response (IIR) filter (1.39). They assumed that the activation function $f(\cdot)$ is invertible (e.g. $\tanh(\cdot)$), which made it possible to use well-known parameter estimation tools for models which are linear in their parameters.

1.3. Model selection

1.3.1. Data acquisition and preparation

The problem of data acquisition constitutes an important preliminary part of any system identification procedure, and is closely related to the experiment design (see (Rafajłowicz 1996, Ljung 1987, Walter and Pronzato 1997) and the references therein). In the case of a known model structure, the problem reduces to an appropriate selection of the experimental conditions with respect to the parameters to be estimated (Rafajłowicz 1989, Rafajłowicz 1996, Uciński 1999, Uciński 2000, Walter and Pronzato 1997). In the present work, the model structure is assumed to be unknown and hence the experimental conditions should be chosen in such a way as to provide maximum information about the system's input-output behaviour. This is, of course, a very sophisticated problem and the reader is referred to (Ljung 1987, Walter and Pronzato 1997) and the references therein for further explanations. It should be also pointed out that the experiment designing procedure is usually limited owing to "reality" constrains, e.g. in process industry, it may be not allowed at all to manipulate a system in a production mode.

When the data have been collected in a physical plant, they are usually not in the form which seems to be appropriate for use in a model construction procedure. This is mainly because of high- and low-frequency disturbances, offset levels, outliers, etc. In order to overcome such problems, the approaches described in (Ljung 1987) can be employed. In the case of readers familiar with the *MATLAB System Identification Toolbox* (Ljung 1988), the problem reduces to simply calling appropriate procedures.

When the data have been collected and appropriately prepared for the model construction procedure, they are usually divided into two sets, namely, the identification (n_t input-output measurements) data set and the validation (n_v input-output measurements) data set. The former is used to obtain the model structure and to estimate its parameters, while the latter is employed to evaluate the goodness of fit of the identified model by making a comparison between the process output and the model predicted output either visually or by some formal distance measure. It should be pointed out that there exist many more or less sophisticated approaches to model validation. However, model testing by experimental data (*cross-validation*) seems to be a quite good technique, and so it is often used in practice.

1.3.2. Model selection criteria

Let $\mathbb{M}$ be a set of model structures that compete for the description of the same data:

$$\mathbb{M} = \{M_i, i = 1, \dots, n_m\}.$$

It corresponds to structures of different types and complexity. With each of these structures a parameter vector $\boldsymbol{p}^i$ is associated. It is assumed that the most complex structure is that containing the greatest number of parameters. Once the set of model structures has been selected, the problem is to choose the best possible model. The criterion for this task is usually based on some scalar measures (cost functions). Such cost functions are usually obtained using a difference between the system output measurement and the model output $\hat{y}$. The output measurement consists of the true system output y and the noise v, i.e. it is assumed that the output measurement equals $y + v$.

In a probabilistic framework the expectation of the squared difference between the system output measurement and the model output can be used as a cost function, i.e.

$$\mathcal{E}\left\{(y + v - \hat{y})^2\right\} = \mathcal{E}\left\{(y - \hat{y})^2\right\} + 2\mathcal{E}\left\{(y - \hat{y})v\right\} + \mathcal{E}\left\{v^2\right\}. \quad (1.40)$$

Assuming that the noise v is uncorrelated with the model and system outputs, equation (1.40) becomes:

$$\mathcal{E}\left\{(y + v - \hat{y})^2\right\} = \mathcal{E}\left\{(y - \hat{y})^2\right\} + \mathcal{E}\left\{v^2\right\}. \quad (1.41)$$

It is obvious that the term $\mathcal{E}\left\{v^2\right\}$ cannot be minimised as it constitutes the variance of the measurement noise. Thus the cost function reaches minimum if $y = \hat{y}$.

The model error $\mathcal{E}\left\{(y-\hat{y})^2\right\}$ can be further decomposed as:

$$\begin{aligned}\mathcal{E}\left\{(y-\hat{y})^2\right\} &= \mathcal{E}\left\{((y-\mathcal{E}\left\{\hat{y}\right\})-(\hat{y}-\mathcal{E}\left\{\hat{y}\right\}))^2\right\} \\ &= \left(y-\mathcal{E}\left\{\hat{y}\right\}\right)^2+\mathcal{E}\left\{(\hat{y}-\mathcal{E}\left\{\hat{y}\right\})^2\right\}. \end{aligned} \tag{1.42}$$

The two terms constituting the model error, i.e. $\left(y-\mathcal{E}\left\{\hat{y}\right\}\right)^2$ and $\mathrm{var}(\hat{y}) = \mathcal{E}\left\{(\hat{y}-\mathcal{E}\left\{\hat{y}\right\})^2\right\}$, are called the bias and variance errors, respectively.

The bias error is caused by the restricted flexibility of the model. In practice most systems are quite complex, and the class of models typically applied is not capable of representing the system correctly. The only exception occurs when the true structure of the system is known, i.e. it is obtained as a result of the physical consideration underlying the system being studied. Therefore, the bias error decreases as the model complexity increases. Since the model complexity is related to the number of parameters, the bias error depends qualitatively on it. From the above consideration it is clear that the bias error represents the systematic deviation between the system and the model that exists due to the model structure.

The variance error is caused by a deviation of the estimated parameters from their optimal values. Indeed, since the model parameters are estimated based on finite and noisy data, these parameters usually deviate from their optimal values. The variance error describes that part of the model error which comes from parameter uncertainty. Undoubtedly, the fewer parameters the model possesses, the more accurately they can be estimated using the identification data. Thus, the variance error increases accordingly to an increase in the number of parameters.

From the above discussion it is clear that a compromise between the bias and variance errors should be established (bias/variance trade-off). This can be achieved by an appropriate selection of the model complexity.

It can be observed during model determination that the error on the identification data (which is approximately equal to the bias error) decreases as the model complexity increases. On the other hand, the error on the validation data (which is equal to the bias error plus variance error) starts to rise again beyond the point of optimal complexity. If this effect is ignored, it may lead to a model that is either overly complex (*overfitting* (low bias, high variance)) or too simple (*underfitting* (high bias, low variance)). In order to avoid overfitting, a complexity penalty term should be introduced. This results in information criteria that reflect the value of a cost function and the complexity. There are, of course, many different information criteria, e.g. Akaike's, Bayesian, Khinchin's law of iterated logarithm criterion, final prediction error criterion, structural risk minimisation. However, they all in one way or another implement the following structure:

$$\textsc{information criteria} = \mathrm{IC}(\textsc{cost function}, \textsc{model complexity}).$$

The introduction of such criteria makes it possible to avoid data splitting, i.e. dividing the data set into the identification and validation data sets. In this case, the entire data set can be used for parameter estimation. This is especially important when only a small data set is disposable. On the other hand, it seems profitable to use, if possible, such criteria together with the validation data sets.

The Akaike Information Criterion (AIC) (Walter and Pronzato 1997) is one of the best known criteria which can be employed to select the model structure and to estimate its parameters:

$$\left(\hat{M}, \hat{\boldsymbol{p}}\right) = \arg \min_{M_i \in \mathbb{M}} \min_{\boldsymbol{p}^i \in \mathbb{P}^i} J_{\text{AIC}}(M_i(\boldsymbol{p}^i)). \tag{1.43}$$

In order to formulate a detailed description of (1.43), it is convenient to assume that the data satisfy:

$$\boldsymbol{y}_k = \hat{\boldsymbol{y}}_k(M^*(\boldsymbol{p}^*)) + \boldsymbol{v}_k, \ k = 1, \ldots, n_t,$$

where M^* denotes the correct model structure, $\boldsymbol{p}^*$ is the true value of its parameters, and $\boldsymbol{v}_k$ is a sequence of random independent variables assumed to be normally distributed. Determination of $\hat{M}$ can thus be realised, similarly to the way it was done in (Walter and Pronzato 1997), as follows:
Using the validation data set $\{(\boldsymbol{y}_k, \boldsymbol{u}_k)\}_{k=0}^{n_v}$, obtain:

$$\hat{M} = \arg \min_{M_i \in \mathbb{M}} J_m(M_i), \tag{1.44}$$

$$J_m(M_i) = \frac{1}{2} j(M_i(\hat{\boldsymbol{p}}^i)) + \frac{1}{n_t} \dim \boldsymbol{p}^i, \tag{1.45}$$

where:

$$\hat{\boldsymbol{p}}^i = \arg \min_{\boldsymbol{p}_i \in \mathbb{P}^i} j(M_i(\boldsymbol{p}^i)), \quad i = 1, \ldots, n_m, \tag{1.46}$$

are obtained using the identification data set $\{(\boldsymbol{y}_k, \boldsymbol{u}_k)\}_{k=0}^{n_t}$:

$$\begin{gathered} j(M_i(\boldsymbol{p}^i)) = \ln \det \textstyle\sum_{k=0}^{n_t-1} \boldsymbol{\varepsilon}_k(M_i(\boldsymbol{p}^i)) \boldsymbol{\varepsilon}_k^T(M_i(\boldsymbol{p}^i)), \\ \boldsymbol{\varepsilon}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k(M_i(\boldsymbol{p}^i)), \end{gathered} \tag{1.47}$$

and, consequently, $\hat{\boldsymbol{p}}^i$, which corresponds to the best model structure $\hat{M} = M_i$, is chosen as $\hat{\boldsymbol{p}}$.
The model determination process can then be realised as follows:

Step 0: Select the set of possible model structures $\mathbb{M}$.

Step 1: Estimate parameters of each of the models $M_i, \ i = 1, \ldots, n_m$, according to (1.47).

Step 2: Select the model which is best suited in terms of the criterion (1.44).

Step 3: If the selected model does not satisfy the prespecified requirements, then go to *Step 0.*

1.4. Conclusions

The characterisation of a suitable model structure is not an exact science, it is rather an art. This is mainly because it involves arbitrary decisions resulting in various, more or less important consequences. Indeed, if the class of models is not appropriately selected, then we cannot expect to do more than to find the best model in the class thus defined. It is vain to expect that such a model will be a good description of the system of interest. Thus, the characterisation cannot be considered as an initial step of an identification procedure. In other words, it cannot be performed only once but should be carried out alternately with a model quality determination (parameter estimation and model validation).

Usually, linear model structures are selected first as a potential solution to the identification problem. In the case of dynamic systems, OE and ARX models are preferred as they are relatively simple and there are efficient parameter estimation techniques for such structures. More complex structures such as ARARX, ARMAX, and BJ are rarely used. Another important class are linear state-space models. They became popular after the works regarding observers and Kalman filters were published. Nowadays, observers and filters constitute an important part of modern control systems. State-space models are better suited to MIMO systems than input-output models. This superiority can also be seen during the model order selection stage. Unlike input-output models, where lags of input and output signals should appropriately be chosen, state-state model merely require the dimension of the state vector.

If the linear model is not suitable for a given identification task, then non-linear models should be employed. Polynomials constitute a natural extension of linear models. In both the dynamic and static cases, in spite of the fact that polynomials are linear in their parameters, which simplifies parameter estimation, there are many serious drawbacks regarding extrapolation, stability, the curse of dimensionality, etc. That is why the application of polynomials should be limited to very specific tasks.

Artificial neural networks as relatively new system identification tools have received considerable research attention. Many works confirm their effectiveness and recommend their use. Undoubtedly, the most popular neural architectures for static systems are MLP and RBF networks. Their applicability can be extended to dynamic systems by introducing tapped delay lines. This approach is rather limited to very simple systems as the number of inputs to a neural network increases significantly with the dynamics order.

Recurrent and partially recurrent neural architectures seem to be a more natural way of describing dynamic systems. They resemble non-linear state-space models. On the other hand, the lack of stability (in a general case) as well as various problems regarding training algorithms make the applicability of such networks rather limited.

Locally recurrent and globally feed-forward networks seem to be a very interesting identification tool. Their stability depends completely on the stability of dynamic neurons (which is relatively easy to be proven) and their structure resembles that of the MLP. On the other hand, a more sophisticated neuron structure results in a large number of parameters to be estimated. This problem becomes especially important owing to the non-linear characteristic of the cost function. To tackle this difficulty, a GMDH network with dynamic neurons has been proposed.

Unlike in the case of linear systems, there is no general approach which can be applied to obtain non-linear state-space models. Such a general framework should ensure stability as well as transparent (to a certain degree) interpretability of models. In practice, the spectrum of available identification techniques for non-linear state-space models reduces to the physical consideration underlying the system being studied from which such models can be obtained. Unfortunately, industrial systems are usually very complex and hence it is rather impossible to perform such consideration. Thus, the development of a non-linear state-space model construction framework seems to be a challenging task.

In the present chapter, the importance of data preparation and selection of an identification criterion were discussed as well.

Chapter 2

RESIDUAL GENERATION TECHNIQUES

A fault can generally be defined as an unexpected change in a system of interest, e.g. a sensor malfunction. All the unexpected variations that tend to degrade the overall performance of a system can also be interpreted as faults. Contrary to the term *failure*, which suggests a complete breakdown of the system, the term *fault* is used to denote a malfunction rather than a catastrophe.

Since a system can be split into three parts (Fig. 2.1), i.e. actuators, system dynamics, and sensors, such a decomposition leads directly to three classes of faults:

- *actuators faults*, which can be viewed as any malfunction of the equipment that actuates the system, e.g. a malfunction of an electro-mechanical actuator for a diesel engine (Blanke *et al.* 1994),
- *system dynamics faults* (or component faults), which occur when some changes in the system make the dynamic relation invalid, e.g. a leak in a tank in a two-tank system,
- *sensors faults*, which can be viewed as serious measurements variations.

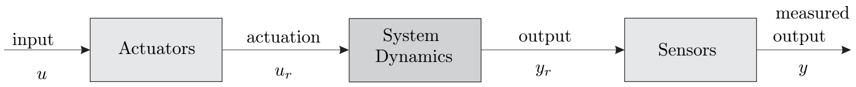

Fig. 2.1 . A control system.

Since faults and their possible forms are defined, it is possible to describe the methodology of fault diagnosis.

There are two general classes of model-based approaches to FDI. The first class employs quantitative system descriptions (e.g. difference or differential equations, state-space models, transfer functions, neural networks, etc.) as a source of knowledge regarding a system behaviour. These approaches utilise results from widely-understood control theory, i.e. state observers or filters, parameter estimation, parity relation concepts, etc. In one way or another, they are used to generate fault symptoms resulting in, e.g., a deviation of model parameters, a difference between the model and system responses, etc. By monitoring those quantities, it is possible to detect and isolate each fault. The essential requirement is to possess *a priori* knowledge regarding the relationship between the system and the faults.

The second class is based on the application of qualitative reasoning and qualitative modelling (Lunze and Schiller 2000, Zhang and Roberts 2000). Qualitative models of a system are used to estimate a system behaviour under the normal and faulty operating conditions. In this work, research attention is focused on the quantitative model-based fault diagnosis.

In the sequel, quantitative model-based fault diagnosis is simply called model-based fault diagnosis since no qualitative techniques are considered.

Model-based fault diagnosis can be defined as a three-task process consisting of the detection, isolation and identification of system faults (Fig. 2.2). These tasks can be summarised as follows (Chen and Patton 1999):

Fault detection: to make a decision regarding the system stage - either that something is wrong or that everything works under the normal conditions,

Fault isolation: to determine the location of the fault, e.g. which sensor or actuator is faulty,

Fault identification: to determine the size and type or nature of the fault.

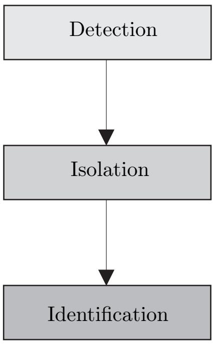

Fig. 2.2 . The three-stage process of fault diagnosis.

On the other hand, the fault diagnosis process can be viewed as a two-stage process, i.e. residual generation and decision making based on this residual (Fig. 2.3).

If the residuals are properly generated, then fault detection becomes a relatively easy task. Since without fault detection it is impossible to perform fault isolation and, consequently, fault identification, all efforts regarding the improvement of residual generation seem to be justified. This is the main reason why the research effort of this work is oriented towards fault detection and especially towards residual generation.

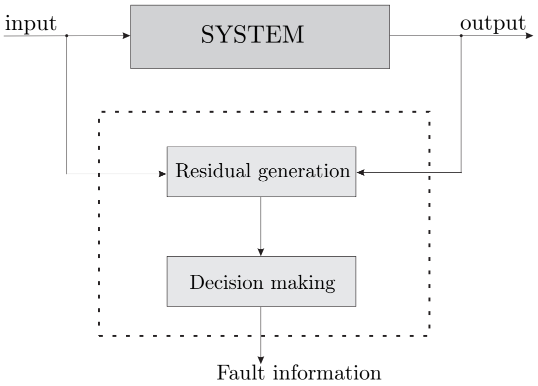

Fig. 2.3 . The two-stage process of fault diagnosis.

There have been many developments in model-based fault detection since the beginning of the 1970s, regarding both the theoretical context and the applicability to real systems (see (Chen and Patton 1999, Patton *et al.* 2000) for a survey). Generally, the most popular approaches can be split into four categories, i.e.

- parameter estimation,
- factorisation,
- parity relation,
- observer-based.

All of them, in one way or another, employ a mathematical system description to generate the residual signal. In almost all cases, the residual signal is obtained as a difference between system outputs and model outputs, i.e. $\boldsymbol{r}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k$.

The simplest model-based residual generation scheme can be realised in a way similar to that shown in Fig. 2.4. In this case, the design procedure reduces to system identification, and fault detection boils down to checking the norm of the residual signal $\|\boldsymbol{r}_k\|$. This means that it is sufficient to apply one of the approaches presented in Chapter 2. In such a simple residual generation scheme, neural networks seem to be especially popular (Patton and Korbicz 1999).

Irrespective of the identification metod used, there is always the problem of model uncertainty, i.e. the model-reality mismatch. Thus, the better the model used to represent a system behaviour, the better the chance of improving the reliability and performance in diagnosing faults. This is the main reason why the fault detection scheme shown in Fig. 2.4 is rarely used for maintaining fault diagnosis of high-safety systems. Indeed, disturbances as well as model uncertainty are inevitable in industrial systems, and hence there exists a pressure creating the need for robustness in fault diagnosis systems. This robustness requirement is usually achieved in the fault detection stage, i.e. the problem is to develop residual

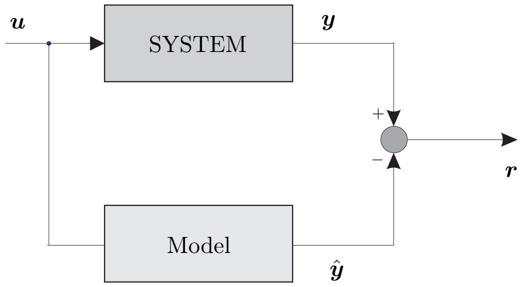

Fig. 2.4 . A simple residual generation scheme.

generators which should be insensitive (as far as possible) to model uncertainty and real disturbances acting on a system while remaining sensitive to faults. In one way or another, all of the most popular approaches mentioned above can realise this requirement for linear systems.

Other problems arise from fault detection of non-linear systems. Indeed, the available non-linear system identification techniques limit the application of fault detection. For example, in the case of observer-based FDI, state-space models cannot be usually obtained using physical consideration (physical laws governing the system being studied). This means that a model which merely approximates the system-input behaviour (no physical interpretation of the state vector or parameters) should be employed. On the other hand, there is no disposable approach which can be applied to obtain such a model automatically. Thus, the FDI system designer is left with time consuming trial-and-error procedure.

One objective of this work is to provide such an efficient system identification technique. This, however, will be the subject of the subsequent chapter.

In the present chapter, the most popular residual generation approaches are briefly reviewed. In particular, their applicability to both linear and non-linear systems together with robustness issues is discussed. The main attention is focused on designing robust observers. This is mainly because many of them can be extended in such a way that they can be useful for the residual generation of non-linear systems.

Although it is discrete-time systems that are considered in this work, some of the techniques are described in a continuous-time form. This is due to the fact that they were originally presented in such a form. However, most of them can relatively easily be applied to discrete-time systems.

The chapter is organised as follows. First, the most popular residual generation techniques are briefly reviewed, and then the attention is focused on observer-based approaches.

2.1. A review of the most popular approaches

2.1.1. Parameter estimation

The task consists in detecting faults in a system by measuring the input $\boldsymbol{u}_k$ and the output $\boldsymbol{y}_k$, and then estimating parameters of the model of the system (Fig. 2.5). The model can be generally described by:

$$y_k = \boldsymbol{g}(\boldsymbol{\phi}_k, \boldsymbol{p}_k), \tag{2.1}$$

where $\boldsymbol{\phi}_k$ may contain the previous or current system input $\boldsymbol{u}_k$, the previous system or model output (y or $\hat{y}$), and the previous prediction error. The model (2.1) can also be expressed in the state-space form; however, this does not change the general framework. If a fault now occurs in the system, this causes a change (residual) $\boldsymbol{\Delta p}_k$ in the parameter vector $\boldsymbol{p}_k$. Such a residual can then be used to detect the faults.

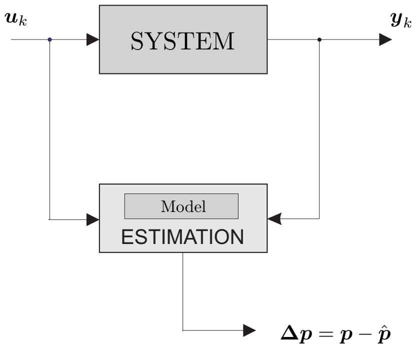
$\boldsymbol{u}_k$
SYSTEM
$\boldsymbol{y}_k$
Model
ESTIMATION
$\boldsymbol{\Delta p} = \boldsymbol{p} - \hat{\boldsymbol{p}}$

Fig. 2.5 . The principle of parameter estimation-based fault detection.

The main drawback to this approach is that the model parameters should have a physical meaning, i.e. they should correspond to the parameters of the system. In such situations, the detection and isolation of faults is very straightforward. If this is not the case, it is usually difficult to distinguish a fault from a change in the parameter vector $\boldsymbol{p}_k$ resulting from time-varying properties of the system. Moreover, the process of fault isolation may become extremely difficult because model parameters do not uniquely correspond to those of the system. Another problem arises when the model structure $\boldsymbol{g}(\cdot)$ is non-linear in its parameters. In this case, non-linear parameter estimation techniques should be applied. For complex models, this may cause serious difficulties with a fast reaction on the faults; consequently, fault detection cannot be performed effectively nor reliably. It should also be pointed out that the detection of faults in sensors and actuators is possible but rather complicated (Patton *et al.* 2000). The robustness with respect to model uncertainty can be tackled relatively easily (especially for linear systems) by employing robust parameter estimation techniques, e.g. the bounded-error approach (Milanese *et al.* 1996).

2.1.2. Parity relation

The basic idea underlying the parity relation approach (Chen and Patton 1999, Gertler 2000) is to check the consistency of the system measurements. In order to illustrate this technique, let us consider a general problem of measuring an n-dimensional vector with m sensors. The measurement equation can be described as:

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{f}_k + \boldsymbol{v}_k, \tag{2.2}$$

where $\boldsymbol{f}_k$ stands for the sensor faults. The application of hardware redundancy means that n signals are measured with m sensors, i.e. $m > n$. Moreover, it is assumed that $\text{rank}(\boldsymbol{C}) = n$. For the purpose of FDI, $\boldsymbol{y}_k$ can be combined into a set of linearly independent parity equations, i.e. $\boldsymbol{r}_k = \boldsymbol{V}\boldsymbol{y}_k$. In order to ensure zero-valued residual, i.e. $\boldsymbol{r}_k = \boldsymbol{0}$, under the normal operating conditions, the matrix $\boldsymbol{V}$ should fulfill the condition $\boldsymbol{V}\boldsymbol{C} = \boldsymbol{0}$. This implies that $\boldsymbol{r}_k = \boldsymbol{V}[\boldsymbol{f}_k + \boldsymbol{v}_k]$. The derivation of the matrix $\boldsymbol{V}$ can be realised by assuming that:

$$\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_m - \boldsymbol{C}\left(\boldsymbol{C}^T\boldsymbol{C}\right)^{-1}\boldsymbol{C}^T, \tag{2.3}$$

and then applying the Gram-Schmidt orthogonalisation scheme (Golub and Van Loan 1989).

The main drawback to such an approach is that it requires additional hardware, i.e. sensors, which may lead to a significant increase in the cost.

In the case when $\text{rank}(C) = m < n$ it is, of course, impossible to apply the approach under consideration. However, an analytical redundancy can be performed by collecting sensor outputs in a data window, i.e. $\{\boldsymbol{y}_{k-i}\}_{i=0}^{s}$. Since the redundancy is related to time, such an approach requires the knowledge of a dynamic model. This concept was introduced by (Chow and Willsky 1984).

In the case of linear systems the following state-space model (it is possible to use different models) can be employed (in a deterministic configuration):

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{L}_1\boldsymbol{f}_k, \tag{2.4}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{D}\boldsymbol{u}_k + \boldsymbol{L}_2\boldsymbol{f}_k. \tag{2.5}$$

The redundancy relation can now analytically be specified as follows. Combining together (2.4)-(2.5) from time instant $k - s$ up to k yields:

$$\underbrace{\begin{bmatrix} \boldsymbol{y}_{k-s} \\ \boldsymbol{y}_{k-s+1} \\ \vdots \\ \boldsymbol{y}_k \end{bmatrix}}_{\boldsymbol{Y}_k} = \boldsymbol{H}\underbrace{\begin{bmatrix} \boldsymbol{u}_{k-s} \\ \boldsymbol{u}_{k-s+1} \\ \vdots \\ \boldsymbol{u}_k \end{bmatrix}}_{\boldsymbol{U}_k} = \boldsymbol{W}\boldsymbol{x}_{k-s} + \boldsymbol{M}\underbrace{\begin{bmatrix} \boldsymbol{f}_{k-s} \\ \boldsymbol{f}_{k-s+1} \\ \vdots \\ \boldsymbol{f}_k \end{bmatrix}}_{\boldsymbol{F}_k}, \tag{2.6}$$

where:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{CB} & \boldsymbol{D} & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{CA}^{s-1}\boldsymbol{B} & \boldsymbol{CA}^{s-1}\boldsymbol{B} & \cdots & \boldsymbol{D} \end{bmatrix}, \quad \boldsymbol{W} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \vdots \\ \boldsymbol{CA}^{s} \end{bmatrix}, \tag{2.7}$$

and:

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{L}_2 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{CL}_1 & \boldsymbol{L}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{CA}^{s-1}\boldsymbol{L}_1 & \boldsymbol{CA}^{s-1}\boldsymbol{L}_1 & \cdots & \boldsymbol{L}_2 \end{bmatrix}. \tag{2.8}$$

The residual signal can now be defined as (Chow and Willsky 1984):

$$\boldsymbol{r}_k = \boldsymbol{V}[\boldsymbol{Y}_k - \boldsymbol{HU}_k] = \boldsymbol{VWx}_{k-s} + \boldsymbol{VMF}_k. \tag{2.9}$$

In order to make (2.9) useful for fault detection, the matrix $\boldsymbol{V}$ should make the residual signal insensitive to system inputs and states, i.e. $\boldsymbol{VW} = \boldsymbol{0}$. On the other hand, to make fault detection possible, the matrix $\boldsymbol{V}$ should also satisfy the condition $\boldsymbol{VM} \neq \boldsymbol{0}$. It can be shown that for an appropriately large s (see (Chen and Patton 1999) for how to obtain the minimum order s) it follows from the Cayley-Hamilton theorem that the solution to $\boldsymbol{VW} = \boldsymbol{0}$ always exists. Finally, fault detection boils down to checking the norm of the residual, i.e. $\|\boldsymbol{r}_k\|$.

The presented approach can further be extended to linear time-varying systems (Chen and Patton 1999), i.e. systems which can be modelled by:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{2.10}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{D}_k\boldsymbol{u}_k + \boldsymbol{L}_{2,k}\boldsymbol{f}_k. \tag{2.11}$$

Its robustness to model uncertainty can be attained by introducing the concept of an unknown input $\boldsymbol{d}_k$ (Chen and Patton 1999, Seliger and Frank 2000), which may represent modelling uncertainty as well as real disturbances acting on the system. In this case, the model can be represented as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k^1\boldsymbol{d}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{2.12}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{D}_k\boldsymbol{u}_k + \boldsymbol{E}_k^2\boldsymbol{d}_k + \boldsymbol{L}_{2,k}\boldsymbol{f}_k. \tag{2.13}$$

The terms $\boldsymbol{E}_k^1\boldsymbol{d}_k$ and $\boldsymbol{E}_k^2\boldsymbol{d}_k$ represent the uncertainty in the state and measurement equations, respectively. One condition for the decoupling of the unknown input $\boldsymbol{d}_k$ from the residual signal $\boldsymbol{r}_k$ is to know the unknown input distribution matrices $\boldsymbol{E}_k^1$

and $\boldsymbol{E}_k^2$, while it is not necessary to have any knowledge concerning $\boldsymbol{d}_k$. There are, of course, many approaches which can be applied to obtain the unknown input distribution matrices (for a comprehensive survey, the reader is referred to (Chen and Patton 1999, Seliger and Frank 2000)).

An extension of parity relation to non-linear polynomial dynamic systems was proposed by (Guernez *et al.* 1997).

In order to describe this approach, let us considered a system modelled by the state-space equations:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{f}_k), \tag{2.14}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{f}_k), \tag{2.15}$$

where $\boldsymbol{g}(\cdot)$ and $\boldsymbol{h}(\cdot)$ are assumed to be polynomials. The equations (2.14)-(2.15) can always be expressed on a time window $[k-s, k]$. As a result, the following structure can be obtained:

$$\boldsymbol{y}_{k-s,k} = \boldsymbol{H}\left(\boldsymbol{x}_{k-s}, \boldsymbol{u}_{k-s,k}, \boldsymbol{f}_{k-s,k}\right), \tag{2.16}$$

where $\boldsymbol{u}_{k-s,k} = \boldsymbol{u}_{k-s}, \ldots, \boldsymbol{u}_k$ and $\boldsymbol{f}_{k-s,k} = \boldsymbol{f}_{k-s}, \ldots, \boldsymbol{f}_k$. In order to check the consistency of the model equations, the state variables have to be eliminated. This results in the following equation:

$$\boldsymbol{\Phi}(\boldsymbol{y}_{k-s,k}, \boldsymbol{u}_{k-s,k}, \boldsymbol{f}_{k-s,k}) = \boldsymbol{0}. \tag{2.17}$$

Since $\boldsymbol{g}(\cdot)$ and $\boldsymbol{h}(\cdot)$ are assumed to be polynomials, elimination theory can be applied to transform (2.16) into (2.17). Knowing that the $\Phi_i(\cdot)$'s are polynomials and therefore they are expressed as sums of monomials, it seems natural to split the expression (2.17) into two parts, i.e.

$$\boldsymbol{r}_k = \boldsymbol{\Phi}_1(\boldsymbol{y}_{k-s,k}, \boldsymbol{u}_{k-s,k}), \tag{2.18}$$

$$\boldsymbol{r}_k = \boldsymbol{\Phi}_2(\boldsymbol{y}_{k-s,k}, \boldsymbol{u}_{k-s,k}, \boldsymbol{f}_{k-s,k}). \tag{2.19}$$

The right-hand side of (2.18) contains all the monomials in $\boldsymbol{y}_{k-s,k}$ and $\boldsymbol{u}_{k-s,k}$ only while (2.19) contains all the monomials involving at least one of the components of $\boldsymbol{f}_{k-s,s}$. The above condition ensures that $\boldsymbol{r}_k = 0$ in the fault-free case. Since the fault signal $\boldsymbol{f}_{k-s,k}$ is not measurable, only equation (2.18) can be applied to generate the residual signal $\boldsymbol{r}_k$ and, consequently, to detect faults.

One drawback to this approach is that it is limited to polynomial models or, more precisely, to models for which the state vector $\boldsymbol{x}_k$ can be eliminated. Another drawback is that it is assumed that a perfect model is available, i.e. there is no model uncertainty. This may cause serious problems while applying the approach to real systems.

Parity relation for a more general class of non-linear systems was proposed by Krishnaswami and Rizzoni (1994). The FDI scheme considered is shown in Fig. 2.6. There are two residual vectors, namely, the forward $\boldsymbol{r}_{f,k}$ residual vector and the backward $\boldsymbol{r}_{b,k}$ residual vector. These residuals are generated using the forward and inverse (backward) models, respectively. Based on these residual vectors, fault detection can (theoretically) easily be performed while fault isolation should be realised according to Tab. 2.1. The authors suggest an extension of the proposed approach to the cases where model uncertainty is considered.

Tab.2.1. The principle of fault isolation with non-linear parity relation.

Fault location	Non-zero element of $\boldsymbol{r}_{f,k}$	Non-zero element of $\boldsymbol{r}_{b,k}$
i-th sensor	r_f^i	all elements dependent on y_i
i-th actuator	all elements dependent on u_i	r_b^i

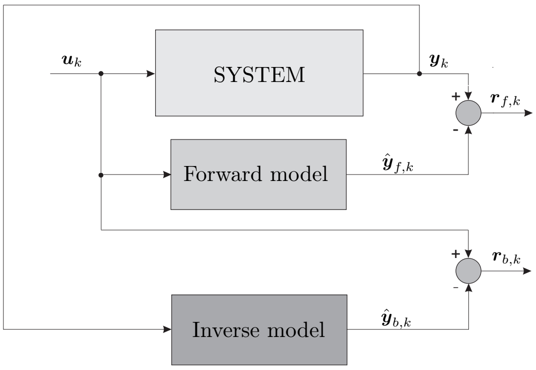

Fig. 2.6 . Non-linear parity relation-based FDI.

Undoubtedly, strict existence conditions for an inverted model as well as possible difficulties with the application of the known identification techniques make the usefulness of this approach for a wide class of non-linear systems questionable.

Another parity relation approach for non-linear systems was proposed by Shumsky (1997). The concepts of parity relation and parameter estimation fault detection techniques are combined. In particular, parity relation is used to detect offsets in the model parameters. The necessary condition is that there exists a transformation $\boldsymbol{x}_k = \boldsymbol{\xi}(\boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k+s}, \boldsymbol{y}_k, \ldots, \boldsymbol{y}_{k+s})$, which may cause serious problems in many practical applications. Another inconvenience is that the approach inherits most drawbacks concerning parameter estimation-based fault detection techniques.

2.1.3. Factorisation techniques

The basic idea underlying the factorisation approach is that the residual generator can be synthesised in the frequency domain (Viswanadham *et al.* 1987). In this case, the linear dynamic system can be described by

$$\boldsymbol{y}(s) = \boldsymbol{G}_u(s)\boldsymbol{u}(s) + \boldsymbol{G}_f(s)\boldsymbol{f}(s), \tag{2.20}$$

where $\boldsymbol{G}_u(s)$ is assumed to be a proper rational transfer function matrix, which can be factorised as

$$\boldsymbol{G}_u(s) = \bar{\boldsymbol{M}}^{-1}(s)\bar{\boldsymbol{N}}(s), \tag{2.21}$$

where $\bar{\boldsymbol{M}}(s)$ and $\bar{\boldsymbol{N}}(s)$ are stable, rational and realizable transfer function matrices. Based on such a factorisation, residual generator can be designed as (cf. Fig. 2.7)

$$\boldsymbol{r}(s) = \boldsymbol{Q}(s)[\bar{\boldsymbol{M}}(s)\boldsymbol{y}(s) - \bar{\boldsymbol{N}}(s)\boldsymbol{u}(s)], \tag{2.22}$$

where $\boldsymbol{Q}(s)$ denotes dynamic residual weighting, which provides an additional level of freedom. Substituting (2.20) into (2.22), the residual generator becomes

$$\boldsymbol{r}(s) = \boldsymbol{Q}(s)\bar{\boldsymbol{M}}(s)\boldsymbol{G}_f(s)\boldsymbol{f}(s). \tag{2.23}$$

Assuming that there is no model uncertainty and no real disturbances acting on the system, it is clear from the above equation that the residual is only affected by faults. Since, in practice no nominal models can perfectly describe a physical system, model uncertainty should be taken into account. This can be realised as follows (Ding and Frank 1991):

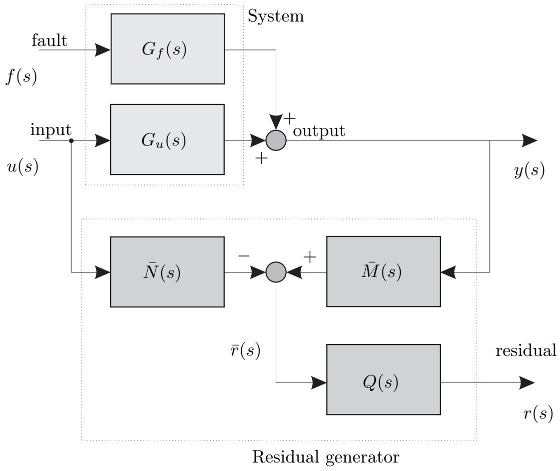

Fig. 2.7 . The principle of factorisation-based residual generation.

$$\boldsymbol{y}(s) = \boldsymbol{G}_u(s)\boldsymbol{u}(s) + \boldsymbol{G}_d(s)\boldsymbol{d}(s) + \boldsymbol{G}_f(s)\boldsymbol{f}(s). \tag{2.24}$$

A comprehensive study regarding applications of the factorisation techniques to systems modelled by (2.24) can be found in (Chen and Patton 1999).

2.1.4. Observers

The basic idea underlying the observer-based (or filter-based, in the stochastic case) approaches to fault detection is to obtain estimates of certain measured and/or unmeasured signals. Then, in the most usual case, the estimates of the measured signals are compared with their originals, i.e. a difference between the original signal and its estimate is used to form a residual signal $\boldsymbol{r}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k$ (Fig. 2.8). To tackle this problem, many different observers (or filters) can be employed, e.g. Luenberger observers, Kalman filters, etc. From the above discussion, it is clear that the main objective is the estimation of the system outputs while the estimation of the entire state vector is unnecessary. Since low-order observers can be employed, state estimation is significantly facilitated. On the other hand, to provide additional freedom to achieve a required diagnostic performance, the observer order is usually larger than the possible minimum one.

Admiration for the observer-based fault detection schemes is caused by the still increasing popularity of state-space models as well as wide applicability of observers in modern control theory and applications. Due to such conditions, the theory of observers (or filters) seems to be well developed (especially for linear systems). This has made a good background for the development of observer-based FDI schemes.

In most robust observer-based fault detection schemes, the problem of robustness to both model uncertainty and real disturbances acting on a system has been tackled by the introduction of the concept of an unknown input (Alcorta *et al.* 1997, Chen *et al.* 1996, Chen and Patton 1999, Kurek 1982, Patton and Chen 1997, Seliger and Frank 2000). In spite of this, there is a large spectrum of candidate solutions. This is the main reason why observer-based approaches deserve special attention.

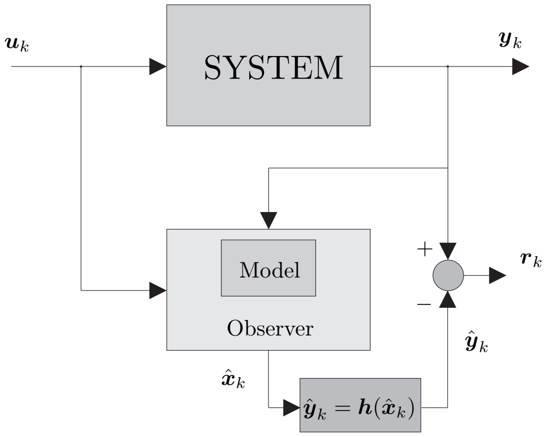

Fig. 2.8. The principle of observer-based residual generation.

2.2. Observers for linear systems

2.2.1. Luenberger observers and Kalman filters

Let us consider a linear system described by the following state-space equations:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{2.25}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}_{k+1}\boldsymbol{x}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}. \tag{2.26}$$

According to the observer-based residual generation scheme (Fig. 2.8), the residual signal can be given as:

$$\begin{aligned}\boldsymbol{r}_{k+1} =& \boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1} = \boldsymbol{C}_{k+1}[\boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1}] + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}\\ =& \boldsymbol{C}_{k+1}[\boldsymbol{A}_k - \boldsymbol{K}_{k+1}\boldsymbol{C}_{k+1}][\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k] + \boldsymbol{C}_{k+1}\boldsymbol{L}_{1,k}\boldsymbol{f}_k\\ &- \boldsymbol{C}_{k+}\boldsymbol{K}_{k+1}\boldsymbol{L}_{2,k}\boldsymbol{f}_k + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}.\end{aligned} \tag{2.27}$$

To tackle the state estimation problem, the Luenberger observer can be used, i.e.

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{K}_{k+1}(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k), \tag{2.28}$$

where $\boldsymbol{K}_k$ stands for the so-called gain matrix and should be obtained in such a way as to ensure an asymptotic convergence of the observer, i.e. $\lim_{k\to\infty}(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k) = 0$ (Paraskevopoulos 1996). If this is the case, i.e. $\hat{\boldsymbol{x}}_k \to \boldsymbol{x}_k$, the state estimation error $\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k$ approaches zero and hence the residual signal (2.27) is only affected by the fault vector $\boldsymbol{f}_k$.

A similar approach can be realised in a stochastic setting, i.e. for systems which can be modelled by:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k + \boldsymbol{w}_k, \tag{2.29}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}_{k+1}\boldsymbol{x}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_k + \boldsymbol{v}_k, \tag{2.30}$$

where $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are zero-mean white noise sequences with covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, respectively. In this case, the observer structure can be similar to that of the Luenberger observer (2.28). To tackle the state estimation problem, the celebrated Kalman filter can be employed (Anderson and Moore 1979). The algorithm of the Kalman filter can be described as follows:

1. Time update:

$$\hat{\boldsymbol{x}}_{k+1/k} = \boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{B}_k\boldsymbol{u}_k, \tag{2.31}$$

$$\boldsymbol{P}_{k+1/k} = \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{Q}_k. \tag{2.32}$$

2. Measurement update:

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1/k} + \boldsymbol{K}_{k+1}[\boldsymbol{y}_{k+1} - \boldsymbol{C}_{k+1}\hat{\boldsymbol{x}}_{k+1/k}], \tag{2.33}$$

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1/k}\boldsymbol{C}_{k+1}^T \left[\boldsymbol{C}_{k+1}\boldsymbol{P}_{k+1/k}\boldsymbol{C}_{k+1}^T + \boldsymbol{R}_{k+1}\right]^{-1}, \tag{2.34}$$

$$\boldsymbol{P}_{k+1} = [\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{C}_{k+1}]\boldsymbol{P}_{k+1/k}. \tag{2.35}$$

Finally, the residual signal can be given as:

$$\begin{aligned}\boldsymbol{r}_{k+1} =&\boldsymbol{C}_{k+1}\boldsymbol{Z}_{k+1}\boldsymbol{A}_k[\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k] + \boldsymbol{C}_{k+1}\boldsymbol{Z}_{k+1}\boldsymbol{L}_{1,k}\boldsymbol{f}_k \\ &+ \boldsymbol{M}_{k+1}\boldsymbol{L}_{2,k}\boldsymbol{f}_{k+1} + \boldsymbol{C}_{k+1}\boldsymbol{Z}_{k+1}\boldsymbol{w}_k + \boldsymbol{M}_{k+1}\boldsymbol{v}_{k+1},\end{aligned} \tag{2.36}$$

where $\boldsymbol{Z}_{k+1} = [\boldsymbol{I} - \boldsymbol{K}_{k+1}\boldsymbol{C}_{k+1}]$ and $\boldsymbol{M}_{k+1} = [\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{K}_{k+1}]$. Since the state estimate $\hat{\boldsymbol{x}}_k$ approaches the real state $\boldsymbol{x}_k$ (in the mean sense) asymptotically, i.e. $\mathcal{E}(\hat{\boldsymbol{x}}_k) \rightarrow \boldsymbol{x}_k$, the residual signal is only affected by the faults and the noise.

In both the deterministic (the Luenberger observer) and stochastic (the Kalman filter) cases fault detection can be performed by checking that the residual norm $\|\boldsymbol{r}_k\|$ exceeds a prespecified threshold, i.e. $\|\boldsymbol{r}_k\| > \varepsilon_H$. In the stochastic case, it is also possible to use more sophisticated, hypothesis-testing approaches such as Generalised Likelihood Ration Testing (GLRT) or Sequential Probability Ratio Testing (SPRT) (Willsky and Jones 1976, Basseville and Nikiforov 1993, Kowalczuk and Suchomski 2001a).

The presented approaches, in spite of their considerable usefulness, suffer from the lack of robustness to model uncertainty. Indeed, in both cases a perfect model of the system is assumed. This problem will be considered in the subsequent sections, where model uncertainty and real disturbances acting on a system are represented by the so-called unknown input.

2.2.2. Unknown input observers

Let us consider a linear system described by the following state-space equations:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{2.37}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}_k\boldsymbol{x}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}, \tag{2.38}$$

where the term $\boldsymbol{E}_k\boldsymbol{d}_k$ stands for model uncertainty as well as real disturbances acting on the system. The general structure of an Unknown Input Observer (UIO) can be given as:

$$\boldsymbol{z}_{k+1} = \boldsymbol{F}_{k+1}\boldsymbol{z}_k + \boldsymbol{T}_{k+1}\boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{K}_{k+1}\boldsymbol{y}_k, \tag{2.39}$$

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{z}_{k+1} + \boldsymbol{H}_{k+1}. \tag{2.40}$$

If the following relations hold true:

$$\boldsymbol{K}_{k+1} = \boldsymbol{K}_{1,k+1} + \boldsymbol{K}_{2,k+2}, \tag{2.41}$$

$$\boldsymbol{T}_{k+1} = \boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1} \tag{2.42}$$

$$\boldsymbol{F}_{k+1} = \boldsymbol{A}_k - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}\boldsymbol{A}_k - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k, \tag{2.43}$$

$$\boldsymbol{K}_{2,k+1} = \boldsymbol{F}_{k+1}\boldsymbol{H}_k, \tag{2.44}$$

then (assuming the fault-free mode, i.e. $\boldsymbol{f}_k = \mathbf{0}$) the state estimation error is:

$$\boldsymbol{e}_{k+1} = \boldsymbol{F}_{k+1}\boldsymbol{e}_k + [\boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}]\boldsymbol{E}_k\boldsymbol{d}_k. \tag{2.45}$$

From the above equation, it is clear that to decouple the effect of an unknown input from the state estimation error (and consequently from the residual), the following relation should be satisfied:

$$[\boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}]\boldsymbol{E}_k = \mathbf{0}. \tag{2.46}$$

The necessary condition for the existence of a solution to (2.46) is $\mathrm{rank}(\boldsymbol{E}_k) = \mathrm{rank}(\boldsymbol{C}_{k+1}\boldsymbol{E}_k)$ (Chen and Patton 1999, p. 72, Lemma 3.1), and a special solution is:

$$\boldsymbol{H}^*_{k+1} = \boldsymbol{E}_k\left[(\boldsymbol{C}_{k+1}\boldsymbol{E}_k)^T\boldsymbol{C}_{k+1}\boldsymbol{E}_k\right]^{-1}(\boldsymbol{C}_{k+1}\boldsymbol{E}_k)^T. \tag{2.47}$$

The remaining task is to design the matrix $\boldsymbol{K}_{1,k+1}$ so as to ensure the convergence of the observer. This can be realised in a similar way as it is done in the case of the Luenberger observer. Finally, the state estimation error and the residual are given by:

$$\begin{aligned}\boldsymbol{e}_{k+1} = {} & \boldsymbol{F}_{k+1}\boldsymbol{e}_k + \boldsymbol{T}_{k+1}\boldsymbol{L}_{1,k}\boldsymbol{f}_k \\ & - \boldsymbol{H}_{k+1}\boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1} - \boldsymbol{K}_{1,k+1}\boldsymbol{L}_{2,k}\boldsymbol{f}_k,\end{aligned} \tag{2.48}$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{C}_{k+1}\boldsymbol{e}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}. \tag{2.49}$$

Since the Kalman filter constitutes a stochastic counterpart of the Luenberger observer, there can also be developed a stochastic counterpart of the UIO, i.e. an observer which can be applied to the following class of systems:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k + \boldsymbol{w}_k, \tag{2.50}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}_k\boldsymbol{x}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1} + \boldsymbol{v}_{k+1}. \tag{2.51}$$

Such observers will be discussed in detail in Chapter 4.

2.2.3. An eigenstructure assignment approach

This section focuses on the problem of designing robust observers using the eigenstructure (eigenvectors and eigenvalues) assignment (Chen and Patton 1999, Chapter 4). The description of the system being considered has the following (continuous-time) state-space form:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{L}_1\boldsymbol{f}(t) + \boldsymbol{E}\boldsymbol{d}(t), \tag{2.52}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{D}\boldsymbol{u}(t) + \boldsymbol{L}_2\boldsymbol{f}(t). \tag{2.53}$$

The observer-based residual generator can be given as:

$$\dot{\hat{\boldsymbol{x}}}(t) = (\boldsymbol{A} - \boldsymbol{K}\boldsymbol{C})\hat{\boldsymbol{x}}(t) + (\boldsymbol{B} - \boldsymbol{K}\boldsymbol{D})\boldsymbol{u}(t) + \boldsymbol{K}\boldsymbol{y}(t), \tag{2.54}$$

$$\hat{\boldsymbol{y}}(t) = \boldsymbol{C}\hat{\boldsymbol{x}}(t) + \boldsymbol{D}\boldsymbol{u}(t), \tag{2.55}$$

$$\boldsymbol{r}(t) = \boldsymbol{Q}(\boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t)), \tag{2.56}$$

where $\boldsymbol{Q} \in \mathbb{R}^{p\times m}$ $(p \leq m)$ stands for the residual weighting matrix, which constitutes additional design freedom. When the above residual generator (2.54)–(2.56) is applied to the system (2.52)–(2.53), the state estimation error ($\boldsymbol{e}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$) and the residual become:

$$\dot{\mathbf{e}}(t) = (\boldsymbol{A} - \boldsymbol{K}\boldsymbol{C})\boldsymbol{e}(t) + \boldsymbol{E}\boldsymbol{d}(t) + \boldsymbol{L}_1\boldsymbol{f}(t) - \boldsymbol{K}\boldsymbol{L}_2\boldsymbol{f}(t), \tag{2.57}$$

$$\boldsymbol{r}(t) = \boldsymbol{Q}\boldsymbol{C}\boldsymbol{e}(t) + \boldsymbol{Q}\boldsymbol{L}_2\boldsymbol{f}(t). \tag{2.58}$$

Unlike in the UIO-based approaches in which the state estimation error (and consequently the residual) is decoupled from the unknown input, here the residual can be decoupled from the unknown input directly. Indeed, using the Laplace transformation, the residual is given by:

$$\begin{aligned}\boldsymbol{r}(s) =&\boldsymbol{Q}\boldsymbol{L}_2\boldsymbol{f}(s) + \boldsymbol{Q}\boldsymbol{C}(s\boldsymbol{I} - \boldsymbol{A}\boldsymbol{K}\boldsymbol{C})^{-1}(\boldsymbol{L}_1 - \boldsymbol{K}\boldsymbol{L}_2)\boldsymbol{f}(s)\\ &+ \boldsymbol{Q}\boldsymbol{C}(s\boldsymbol{I} - \boldsymbol{A}\boldsymbol{K}\boldsymbol{C})^{-1}\boldsymbol{E}\boldsymbol{d}(t).\end{aligned} \tag{2.59}$$

As can be seen from (2.59), in order to decouple the unknown input from the residual, the following relation should be satisfied:

$$\boldsymbol{Q}\boldsymbol{C}(s\boldsymbol{I} - \boldsymbol{A}\boldsymbol{K}\boldsymbol{C})^{-1}\boldsymbol{E}\boldsymbol{d}(t) = \mathbf{0}. \tag{2.60}$$

There are, of course, several different approaches which can be applied to solve (2.60). One of them can be realised as follows (Chen and Patton 1999, Theorem 4.4, p. 127).

The sufficient conditions for satisfying the disturbance decoupling requirement are

1. $\boldsymbol{Q}\boldsymbol{C}\boldsymbol{E} = \mathbf{0}$,
2. All columns of the matrix $\boldsymbol{E}$ are the right eigenvectors of $(\boldsymbol{A} - \boldsymbol{K}\boldsymbol{C})$ corresponding to any eigenvalues.

The remaining design freedom should be used to ensure the convergence of the observer.

2.3. Observers for non-linear systems

Model linearisation is a straightforward way of extending the applicability of linear techniques to non-linear systems. On the other hand, it is well known that such approaches work well when there is no large mismatch between the linearised model and the non-linear system. Two types of linearisation can be distinguished, i.e. linearisation around the constant state and linearisation around the current state estimate. It is obvious that the second type of linearisation usually yields better results. Unfortunately, during such linearisation the influence of terms higher than linear is usually neglected (as in the case of the extended Luenberger observer and the extended Kalman filter). This disqualifies such approaches for most practical applications. Such conditions have led to the development of linearisation-free observers for non-linear systems.

This section briefly reviews the most popular observer-based residual generation techniques for non-linear systems. Their advantages, drawbacks as well as robustness to model uncertainty are discussed.

2.3.1. Extended Luenberger observers and Kalman filters

Let us consider a non-linear discrete-time system modelled by the following state-space equations:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{2.61}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}) + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}. \tag{2.62}$$

In order to apply the Luenberger observer presented in Section 2.2.1, it is necessary to linearise equations (2.61) and (2.62) around either a constant value (e.g. $\boldsymbol{x} = \boldsymbol{0}$) or the current state estimate $\hat{\boldsymbol{x}}_k$. The latter approach seems to be more appropriate as it improves its approximation accuracy as $\hat{\boldsymbol{x}}_k$ tends to $\boldsymbol{x}_k$. In this case the approximation can be realised as follows:

$$\boldsymbol{A}_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k}, \quad \boldsymbol{C}_k = \left.\frac{\partial \boldsymbol{h}(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k}. \tag{2.63}$$

As a result of using the Luneberger observer (2.28), the state estimation error takes the form:

$$\begin{aligned}\boldsymbol{e}_{k+1} = {} & [\boldsymbol{A}_{k+1} - \boldsymbol{K}_{k+1}\boldsymbol{C}_k]\boldsymbol{e}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k - \boldsymbol{K}_{k+1}\boldsymbol{L}_{2,k}\boldsymbol{f}_k + \\ & + \boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k),\end{aligned} \tag{2.64}$$

where $\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)$ stands for the linearisation error caused by the approximation (2.63).

Because of a highly time-varying nature of $\boldsymbol{A}_{k+1}$ and $\boldsymbol{C}_k$ as well as the linearisation error $\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)$, it is usually very difficult to obtain an appropriate form of the gain matrix $\boldsymbol{K}_{k+1}$. This is the main reason why this approach is rarely used in practice.

As the Kalman filter constitutes a stochastic counterpart of the Luenberger observer, the extended Kalman filter can also be designed for the following class of non-linear systems:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{L}_{1,k}\boldsymbol{f}_k + \boldsymbol{w}_k, \tag{2.65}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{h}(\boldsymbol{x}_{k+1}) + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1} + \boldsymbol{v}_{k+1}, \tag{2.66}$$

where, similarly to the linear case, $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are zero-mean white noise sequences. Using the linearisation (2.63) and neglecting the influence of the linearisation error, it is straightforward to use the Kalman filter algorithm described in Section 2.2.1. The main drawback to such an approach is that it works well only when there is no large mismatch between the model linearised around the current state estimate and the non-linear behaviour of the system.

The EKF can also be used for deterministic systems, i.e. as an observer for the system (2.61)–(2.62) (see (Boutayeb and Aubry 1999) and the references therein). In this case, the noise covariance matrices can be set almost arbitrarily. As was proposed in (Boutayeb and Aubry 1999), this possibility can be used to increase the convergence of an observer.

Apart from difficulties regarding linearisation errors, similarly to the case of linear systems, the presented approaches do not take model uncertainty into account. This drawback disqualifies those techniques for most practical applications. Although there are applications for which such techniques work with acceptable efficiency, e.g. (Kowalczuk and Gunawickrama 2000).

2.3.2. The Tau observer

The observer proposed by Tau (1973) can be applied to a special class of non-linear systems which can be modelled by the following state-space equations:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{L}_1\boldsymbol{f}(t) + \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{2.67}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{L}_2\boldsymbol{f}(t). \tag{2.68}$$

This special model class can represent systems with both linear and non-linear parts. The non-linear part is continuously differentiable and locally Lipschitz, i.e.

$$\|\boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t)) - \boldsymbol{g}(\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t))\| \leq \gamma\|\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\|. \tag{2.69}$$

The structure of the Tau observer can be given as:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{g}(\hat{\boldsymbol{x}}(t), \boldsymbol{u}(t)) + \boldsymbol{K}(\boldsymbol{y}(t) - \hat{\boldsymbol{y}}(t)), \tag{2.70}$$

$$\hat{\boldsymbol{y}}(t) = \boldsymbol{C}\hat{\boldsymbol{x}}(t), \tag{2.71}$$

where $\boldsymbol{K} = \boldsymbol{P}_\theta^{-1}\boldsymbol{C}^T$, and $\boldsymbol{P}_\theta$ is the solution to the Lyapunov equation:

$$\boldsymbol{A}^T\boldsymbol{P}_\theta + \boldsymbol{P}_\theta\boldsymbol{A} - \boldsymbol{C}^T\boldsymbol{C} + \theta\boldsymbol{P}_\theta = \boldsymbol{0}, \tag{2.72}$$

where θ is a positive parameter, chosen in such a way as to ensure a positive definite solution of (2.72). In order to satisfy the above condition, the Lipschitz constant γ should satisfy the following condition (Schreier *et al.* 1997):

$$\gamma < \frac{1}{2}\frac{\underline{\sigma}\left(\boldsymbol{C}^T\boldsymbol{C} + \theta\boldsymbol{P}_\theta\right)}{\bar{\sigma}\left(\boldsymbol{P}_\theta\right)}, \tag{2.73}$$

where $\bar{\sigma}(\cdot)$ and $\underline{\sigma}(\cdot)$ stand for the maximum and minimum singular values, respectively.

In spite of the fact that the design procedure does not require any linearisation, the conditions regarding the Lipschitz constant γ are rather restrictive. This may limit any practical application of such an approach. Another difficulty arises from the lack of robustness to model uncertainty.

2.3.3. Observers for bilinear and low-order polynomial systems

A polynomial (and, as a special case, bilinear) system description is a natural extension of linear models. Designs of observers for bilinear and low-order polynomial (up to degree three) systems (Hac 1992, Hou and Pugh 1997, Kinneart 1999, Yu and Shileds 1996, Ashton *et al.* 1999, Shields and Ashton 2000) involve only solutions of non-linear algebraic or Ricatti equations. This allows on-line residual generation.

Let us consider a bilinear continuous-time system modelled by the following state space equations:

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \sum_{i=1}^{r}\boldsymbol{B}_i u_i(t)\boldsymbol{x}t + \boldsymbol{E}_1\boldsymbol{d}(t), \tag{2.74}$$

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t) + \boldsymbol{E}_1\boldsymbol{d}(t). \tag{2.75}$$

With a slight abuse of notation, the influence of faults is neglected. However, faults can very easily be introduced without changing the design procedure.

An observer for the system (2.74)–(2.75) can be given as (Hou and Pugh 1997):

$$\dot{\boldsymbol{\zeta}}(t) = \boldsymbol{F}\boldsymbol{\zeta}(t) + \boldsymbol{G}\boldsymbol{y}(t) + \sum_{i=1}^{r}\boldsymbol{L}_i u_i(t)\boldsymbol{y}(t), \tag{2.76}$$

$$\hat{\boldsymbol{x}}(t) = \boldsymbol{H}\boldsymbol{\zeta}(t) + \boldsymbol{N}\boldsymbol{y}(t). \tag{2.77}$$

Hou and Pugh (1997) established the necessary conditions for the existence of the observer (2.76)–(2.77). Moreover, they proposed a design procedure involving a transformation of the original system (2.74)–(2.75) into an equivalent, quasi-linear one.

An observer for systems which can be described by state-space equations consisting of both linear and polynomial terms was proposed in (Ashton *et al.* 1999, Shields and Ashton 2000). Similarly to the case of the observer (2.74)–(2.75), here robustness to model uncertainty is tackled by means of an unknown input.

2.3.4. Non-linear unknown input observers

This section presents an extension of the unknown input observer for linear systems described in Section 2.2.2. Such an extension can be applied to systems which can be modelled by the following state-space equations:

$$\begin{aligned}\dot{\boldsymbol{x}}(t) =& \boldsymbol{a}(\boldsymbol{x}(t)) + \boldsymbol{B}(\boldsymbol{x}(t))\boldsymbol{u}(t) + \boldsymbol{E}_1(\boldsymbol{x}(t), \boldsymbol{u}(t))\boldsymbol{d}(t) \\ &+ \boldsymbol{K}_1(\boldsymbol{x}(t), \boldsymbol{u}(t))\boldsymbol{f}(t), \qquad (2.78)\\ \boldsymbol{y}(t) =& \boldsymbol{c}(\boldsymbol{x}(t)) + \boldsymbol{E}_2(\boldsymbol{u}(t))\boldsymbol{d}(t) + \boldsymbol{K}_2(\boldsymbol{x}(t))\boldsymbol{f}(t). \qquad (2.79)\end{aligned}$$

For notational convenience, the dependence of time t is neglected (e.g. $\boldsymbol{u} = \boldsymbol{u}(t)$).

The underlying idea is to design an unknown input observer for the system (2.78)-(2.79) without model linearisation. For that purpose the following observer structure is proposed (Alcorta *et al.* 1997, Seliger and Frank 2000):

$$\dot{\hat{\boldsymbol{z}}} = \boldsymbol{l}(\hat{\boldsymbol{z}}, \boldsymbol{y}, \boldsymbol{u}, \dot{\boldsymbol{u}}), \qquad (2.80)$$

$$\boldsymbol{r} = \boldsymbol{m}(\hat{\boldsymbol{z}}, \boldsymbol{y}, \boldsymbol{u}), \qquad (2.81)$$

where:

$$\boldsymbol{z} = \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u}). \qquad (2.82)$$

From (2.78)-(2.79) and (2.80)-(2.81), it can be seen that the estimation error $\boldsymbol{e} = \boldsymbol{z} - \hat{\boldsymbol{z}}$ and the residual $\boldsymbol{r}$ are governed by:

$$\begin{aligned}\dot{\boldsymbol{e}} =& \boldsymbol{l}(\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{e}, \boldsymbol{c}(\boldsymbol{x}) + \boldsymbol{E}_2(\boldsymbol{u})\boldsymbol{d} + \boldsymbol{K}_2(\boldsymbol{x})\boldsymbol{f}, \boldsymbol{u}, \dot{\boldsymbol{u}}) \\ &- \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}(\boldsymbol{c}(\boldsymbol{x}) + \boldsymbol{E}_2(\boldsymbol{u})\boldsymbol{d} + \boldsymbol{K}_2(\boldsymbol{x})\boldsymbol{f}). \qquad (2.83)\\ \boldsymbol{r} =& \boldsymbol{m}(\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u}), \boldsymbol{c}(\boldsymbol{x}) + \boldsymbol{E}_2(\boldsymbol{u})\boldsymbol{d} + \boldsymbol{K}_2(\boldsymbol{x})\boldsymbol{f}, \boldsymbol{u}). \qquad (2.84)\end{aligned}$$

Taking the time derivative of (2.82) yields:

$$\dot{\boldsymbol{z}} = \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\dot{\boldsymbol{u}}. \qquad (2.85)$$

Substituting (2.78) into (2.85) leads to:

$$\begin{aligned}\dot{\boldsymbol{z}} =& \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}(\boldsymbol{a}(\boldsymbol{x}) + \boldsymbol{B}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{E}_1(\boldsymbol{x}, \boldsymbol{u})\boldsymbol{d} + \boldsymbol{K}_1(\boldsymbol{x}, \boldsymbol{u})\boldsymbol{f}) \\ &+ \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\dot{\boldsymbol{u}}. \qquad (2.86)\end{aligned}$$

From the above equation, it is clear that the unknown input decoupling condition can be stated as:

$$\forall \boldsymbol{x}, \boldsymbol{u} \quad \frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}}\boldsymbol{E}_1(\boldsymbol{x}, \boldsymbol{u}) = 0. \qquad (2.87)$$

The unknown input decoupling problem can now be realised by analytically solving a set of linear first-order partial differential equations (2.87).

Moreover, if any fault $\boldsymbol{f}$ is to be reflected by the transformed model, it must be required that:

$$\forall \boldsymbol{x}, \boldsymbol{u} \quad \operatorname{rank}\left(\frac{\partial \boldsymbol{T}(\boldsymbol{x}, \boldsymbol{u})}{\partial \boldsymbol{x}} \boldsymbol{K}_1(\boldsymbol{x}, \boldsymbol{u})\right) = \operatorname{rank}\left(\boldsymbol{K}_1(\boldsymbol{x}, \boldsymbol{u})\right). \tag{2.88}$$

The effect of an unknown input can be decoupled from the output signal (2.79) in a similar way (Seliger and Frank 2000).

The main drawback to the proposed approach is that it requires a relatively complex design procedure, even for simple laboratory systems (Zolghardi *et al.* 1996). This may limit most practical applications of non-linear input observers. Other problems may arise from the application of the presented observer to non-linear discrete-time systems.

2.4. Conclusions

In this chapter, the most popular approaches to residual generation for both linear and non-linear systems were presented. However, perfect coverage and completeness was not the primary concern. The approaches presented here included parameter estimation, parity relation, factorisation, and observers. Their advantages, drawbacks, and robustness to model uncertainty and other factors which may lead to an unreliable fault diagnosis were discussed.

Parameter estimation-based residual techniques are usually used for model structures whose parameters have a physical meaning. Another factor limiting the applicability of this approach is that it can efficiently be applied to model structures which are linear in their parameters. It is so because there are no sufficiently effective methods for on-line parameter estimation for system models which are non-linear in their parameters. The robustness to model uncertainty can be introduced by employing robust parameter estimation techniques.

Parity relation approaches can relatively easily be applied to linear systems. The robustness to model uncertainty can be realised by introducing the concept of the so-called unknown input. Such an unknown input may represent model uncertainty as well as real disturbances acting on a system. The only requirement is the knowledge of the unknown input distribution matrix. It can be attained by one of the efficient methods described in (Chen and Patton 1999, Chapter 5).

As was shown, the concept of parity relation can also be extended to non-linear systems. This, however, involves some requirements regarding model structures (i.e. polynomial model) and the existence of inverse models. This limits the practical application of the approach. Similarly to the case of linear systems, the robustness to model uncertainty can be tackled by the unknown input.

The idea behind factorisation techniques is that the residual signal can be designed in the frequency domain. The approach is limited to linear systems as no really efficient factorisation technique for non-linear systems has been developed.

Observer-based techniques are the most popular way of generating residuals. As in almost all techniques, the robustness to model uncertainty and other factors which may lead to an unreliable fault detection is tackled by means of an unknown input. The popularity of observers also comes from the fact that they are widely used in modern control systems. This means that an observer can be employed for both fault diagnosis and control purposes. Such consideration leads directly to fault-tolerant control.

There are efficient approaches to robust observer-based residual generation for linear systems (e.g. unknown input observers); unfortunately, the existing solutions for non-linear systems are not mature yet. There are, of course, many approaches which can be applied to certain classes of non-linear systems, e.g. bilinear or polynomial systems. However, this requirement limits the applicability of such approaches. On the other hand, the existing non-linear extensions of the UIO (Seliger and Frank 2000) which can be applied to a wider class of systems require a relatively complex design procedure, even for simple laboratory systems (Zolghardi *et al.* 1996).

One way out of this problem is to employ linearisation-based approaches, similar to the extended Kalman filter. In this case, the design procedure is as simple as that for linear systems. On the other hand, it is well known that such a solution works well only when there is no large mismatch between the model linearised around the current state estimate and the non-linear behaviour of the system. Thus, the objective is to improve the convergence of linearisation-based observers. This is to be one of the subjects of Chapter 4.

Another problem arises from the fact that, even for linear systems, research concerning observer-based techniques is strongly oriented towards deterministic systems. Indeed, FDI for systems with both modelling uncertainty and the noise has not attracted enough research attention, although most real systems suffer from both modelling uncertainty and the noise. The existing approaches (see (Chen and Patton 1999, Chen *et al.* 1996, Keller and Darouach 1999) and the references therein), which can be applied to linear stochastic systems, rely on a similar idea to that of the classical Kalman filter. Unfortunately, the only existing approaches to non-linear stochastic systems consist in applying the extended Kalman filter, which, as has already been mention, is usually not a perfect choice. These problems together with the relaxation of the restrictive conditions concerning the process and measurement noises ($\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are assumed to be zero-mean white noise sequences) form another research direction, which is to be the subject of Chapter 4.

Chapter 3

SYSTEM IDENTIFICATION VIA GENETIC PROGRAMMING

The main objective of this chapter is to show how to use the genetic programming approach for system identification purposes. In particular, it is shown how to use trees to represent various model structures. Suitable algorithms which aim at finding a model structure best suited to the real system in terms of the criterion used are presented as well. Both state-space and input-output identification schemes are proposed. Moreover, the importance of GP algorithm control parameters is discussed, and suitable adaptation rules are given.

The chapter is organised as follows. Section 3.1 introduces the idea of evolutionary computation and briefly reviews the most popular algorithms. In Section 3.2, a system identification framework for input-output models is proposed, and the GP algorithm is described in detail. Section 3.3 proposes a system identification framework for state-space models. In Section 3.4, experimental results which confirm the effectiveness and reliability of the proposed approaches are presented. Section 3.5 discusses the importance of an appropriate selection of mutation and crossover probabilities, and proposes their adaptation rules.

3.1. Introduction to evolutionary algorithms

Evolutionary Algorithms (EAs) are a broad class of stochastic optimisation algorithms inspired by some biological processes, which allow populations of organisms to adapt to their surrounding environment. Such algorithms have been influenced by Darwin's theory of natural selection, or the survival of the fittest (published in 1859). The idea behind it is that only certain organisms can survive, i.e. only those which can adapt to the environment and win the competition for food and shelter. Almost at the same time that Darwin's theory was presented (1865), Mendel published a short monograph about experiments with plant hybridisation. He observed how traits of different parents are combined into offspring by sexual reproduction. Darwinian evolutionary theory and Mendel's investigations of heredity in plants became the foundations of evolutionary search methods and led to the creation of the neo-Darwinian paradigm (Fogel 1995).

In order to give a general outline of an evolutionary algorithm, let us introduce a few different concepts and notations (Michalewicz 1996).

An evolutionary algorithm is based on a collective learning process within *a population* of n_p individuals, each of which represents *a genotype* (an underlying genetic code), a search point in the so-called *genotype space*. The environment delivers quantitative information (*fitness value*) regarding an individual based on its *phenotype* (the manner of response contained in the behaviour, physiology and morphology of an organism). Thus, each individual has its own phenotype and genotype.

The general principle of an evolutionary algorithm can be described as follows. At the beginning, a population is randomly initialised and evaluated, i.e. based on a phenotype, fitness of each individual is calculated. Next, the randomised processes of *reproduction*, *recombination*, *mutation* and *succession* are iteratively repeated until a given termination condition is reached. *Reproduction*, called also *preselection*, is a randomised process (deterministic in some algorithms) of parent selection from the entire population, i.e. a temporary population of parent individuals is formed. *Recombination* mechanism (omitted in some algorithms) allows the mixing of parental information while passing it to the descendants. *Mutation* introduces an innovation into current descendants. Finally, *succession*, called also post selection, is applied to choose a new generation of individuals from parents and descendants. All the above operations are repeated until the termination condition is reached.

This is, of course, a general principle, and it can be more or less modified for various types of evolutionary algorithms.

The duality of genotype and phenotype suggests two main approaches to simulated evolution (Michalewicz 1996). In genotypic simulations, the attention focuses on genetic structures. This means that the entire searching process is provided in the genotype space. However, in order to calculate the individual's fitness, its chromosome must be decoded to its phenotype. Nowadays, two kinds of such algorithms can be distinguished, i.e.

- Genetic Algorithms (GAs) (Holland 1975),
- Genetic Programming (GP) (Koza 1992).

In the phenotypic simulations, the attention focuses on the behaviours of the candidate solutions in a population. All operations, i.e. selection, reproduction, and mutation, are performed in the phenotype space. Nowadays, three main kinds of such algorithms can be distinguished, i.e.:

- Evolutionary programming (Fogel *et al.* 1999),
- Evolutionary strategies (Michalewicz 1996),
- Evolutionary search with soft selection (Galar 1989).

In this work, the attention is focused on genotypic algorithms. In particular, Section 3.1.1 outlines genetic algorithms and genetic programming. The possibility of applying genetic programming to system identification is discussed as well. In Section 3.2, the tree-based input-output representation of a model is proposed.

Section 3.2.1 describes the algorithm of genetic programming and some of its modifications. In Section 3.3, an alternative state-space description of a model is proposed. Section 3.4 shows experimental results concerning system identification with genetic programming. A technique of a further improvement of the performance of the GP algorithm is proposed in Section 3.5. Finally, the last section is devoted to conclusions.

3.1.1. Genetic algorithms and genetic programming

Genetic algorithms are computation models that approach the natural evolution perhaps most closely. Many works confirm their effectiveness and recommend their application to various optimisation problems.

A genetic algorithm contains a population of individuals whose DNA is represented by fixed-length binary strings. Inside a computer programme, an individual's fitness is calculated directly from the DNA, and so only the DNA has to be represented. The population of such individuals is evolved through successive generations; individuals in each new generation are bred from the fittest individuals from the previous generation.

The breeding of a new parent is inspired by natural processes, i.e. either asexual or sexual reproduction can be employed. In asexual reproduction, the parent individual is simply copied (possibly with some random changes within a genotype). This process is called *mutation* (Fig. 3.1). In sexual reproduction, couples of parents are randomly chosen and new individuals are created by alternately copying sequences from each parent. This process is known as *crossover* (Fig. 3.2).

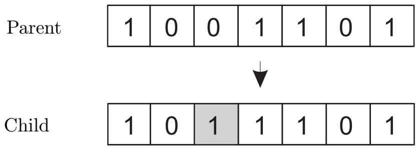

Fig. 3.1 . An exemplary mutation operation.

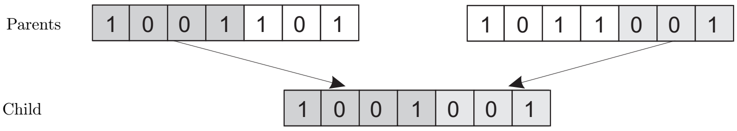

Fig. 3.2 . An exemplary crossover operation.

The main difference between GAs and genetic programming is that in GP the evolving individuals are parse trees rather than fixed-length binary strings (cf. Fig. 3.3). Genetic programming applies the approach of GAs to a pop-

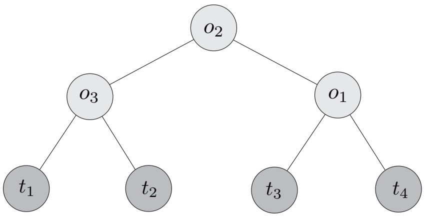

Fig. 3.3. An exemplary GP tree.

ulation of programs which can be described by such trees. Such an approach has demonstrated its potential by evolving simple programs for medical signal filters, or by performing optical character recognition, a target identification, system identification, fault diagnosis, etc. (Esparcia-Alcazar 1998, Koza 1992, Gray *et al.* 1998, Witczak and Korbicz 2000a, Witczak and Korbicz 2000b, Witczak and Korbicz 2001c, Witczak and Korbicz 2002).

In the sequel, it is shown how to employ the genetic programming technique to obtain a mathematical description of a dynamic non-linear system.

3.2. Input-output representation of the system

The characterisation of a set of possible candidate models $\mathbb{M}$ (cf. Section 1.3.2) from which the system model will be obtained constitutes an important preliminary task in any system identification procedure. Knowing that the system exhibits a non-linear characteristic, a choice of a non-linear model set must be made. Let a non-linear input-output MIMO model have the following form:

$$\begin{aligned}\hat{y}_{i,k} =& g_i(\hat{y}_{1,k-1},\dots,\hat{y}_{1,k-n_{1,y}},\dots,\hat{y}_{m,k-1},\dots,\hat{y}_{m,k-n_{m,y}},\\ & u_{1,k-1},\dots,u_{1,k-n_{1,u}},\dots,u_{r,k-1},\dots,u_{r,k-n_{r,u}},\boldsymbol{p}_i),\\ & i=1,\dots,m. \end{aligned} \tag{3.1}$$

Thus the system output is given by:

$$\boldsymbol{y}_k = \hat{\boldsymbol{y}}_k + \boldsymbol{\varepsilon}_k, \tag{3.2}$$

where $\boldsymbol{\varepsilon}_k$ consists of a structural deterministic error caused by the model-reality mismatch, and the stochastic error caused by the measurement noise $\boldsymbol{v}_k$. The problem is to determine the unknown function $\boldsymbol{g}(\cdot) = (g_1(\cdot),\dots g_m(\cdot))$ and to estimate the corresponding parameters vector $\boldsymbol{p} = (\boldsymbol{p}_1,\dots,\boldsymbol{p}_m)$.

One possible solution to this problem is the GP approach. As has already been mentioned, the main ingredient underlying the GP algorithm is a tree. In order to adapt GP to system identification, it is necessary to represent the model (3.1) either as a tree or as a set of trees. Indeed, as is shown in Fig. 3.2, the MISO model can easily be put in the form of a tree, and hence to build the MIMO

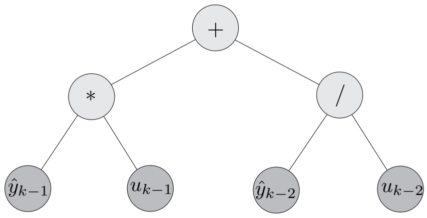

Fig. 3.4. An exemplary GP tree representing the model $\hat{y}_k = \hat{y}_{k-1}u_{k-1} + \hat{y}_{k-2}/u_{k-2}$.

model (3.1), it is necessary to use m trees. In such a tree (see Fig. 3.2), two sets can be distinguished, namely, the terminal set $\mathbb{T}$ and the function set $\mathbb{F}$ (e.g. $\mathbb{T} = \{u_{k-1}, u_{k-2}, \hat{y}_{k-1}, \hat{y}_{k-2}\}$, $\mathbb{F} = \{+, *, /\}$). The language of the trees in GP is formed by a user-defined function $\mathbb{F}$ set and terminal $\mathbb{T}$ set, which form the nodes of the trees. The functions should be chosen so as to be *a priori* useful in solving the problem, i.e. any knowledge concerning the system under consideration should be included in the function set. This function set is very important and should be universal enough to be capable of representing a wide range of non-linear systems. The terminals are usually variables or constants. Thus, the searching space consists of all the possible compositions that can recursively be formed from the elements of $\mathbb{F}$ and $\mathbb{T}$. The selection of variables does not cause any problems, but the handling of numerical parameters (constants) seems very difficult. Even though there are no constant numerical values in the terminal set $\mathbb{T}$, they can be implicitly generated, e.g. the number 0.5 can be expressed as $x/(x + x)$. Unfortunately, such an approach leads to an increase in both the computational burden and evolution time. Another way is to introduce a number of random constants into the terminal set, but this is also an inefficient approach. An alternative way of handling numerical parameters which seems to be more suitable is called *node gains* (Esparcia-Alcazar 1998). A node gain is a numerical parameter associated with the node whose output it multiplies (see Fig. 3.5). Although this technique

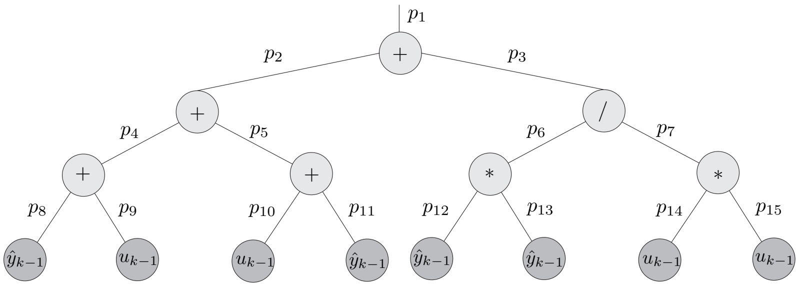

Fig. 3.5. An exemplary parameterised tree.

is straightforward, it leads to an excessive number of parameters, i.e. there are parameters which are not identifiable. Thus, it is necessary to develop a mechanism which prevents such situations from happening. First, let us define the function set $\mathbb{F} = \{+, *, /, \xi_1(\cdot), \ldots, \xi_l(\cdot)\}$, where $\xi_k(\cdot)$ is a non-linear univariate function. To tackle the parameters reduction problem, several simple rules can be established:

$*, /$: A node of type either $*$ or $/$ always has parameters set to unity on the side of its successors. If a node of the above type is a root node of a tree, then the parameter associated with it should be estimated.

$+$: A parameter associated with a node of type $+$ is always equal to unity. If its successor is not of type $+$, then the parameter of the successor should be estimated.

ξ: If a successor of a node of type ξ is a leaf of a tree or is of type $*$ or $/$, then the parameter of the successor should be estimated. If a node of type ξ is a root of a tree, then the associated parameter should be estimated.

As an example, consider the tree shown in Fig. 3.5. Following the above rules, the resulting parameter vector has only five elements $\boldsymbol{p} = (p_3, p_8, p_9, p_{10}, p_{11})$, and the resulting model is $\hat{y}_k = (p_{11}+p_8)\hat{y}_{k-1}+(p_{10}+p_9)u_{k-1}+p_3\hat{y}_{k-1}^2/u_{k-1}^2$. It is obvious that although these rules are not optimal in the sense of parameter identifiability, their application significantly reduces the dimension of the parameter vector, thus making the parameter estimation process much easier. Moreover, the introduction of parameterised trees reduces the terminal set to variables only, i.e. constants are no longer necessary, and hence the terminal set is given by

$$\begin{aligned}\mathbb{T} = \{&\hat{y}_{1,k-1}, \ldots, \hat{y}_{1,k-n_{1,y}}, \ldots, \hat{y}_{m,k-1}, \ldots, \hat{y}_{m,k-n_{m,y}},\\ &u_{1,k-1}, \ldots, u_{1,k-n_{1,u}}, \ldots, u_{r,k-1}, \ldots, u_{r,k-n_{r,u}}\}.\end{aligned} \tag{3.3}$$

The remaining problem is to select appropriate lags in the input and output signals of the model. Assuming that $n_y^{\max}$ i $n_u^{\max}$ are the maximum lags in the output and input signals, the problem boils down to checking $n_y^{\max} \times n_u^{\max}$ possible configurations, which is an extremely time-consuming process. With a slight loss of generality, it is possible to assume that each $n_y = n_u = n$. Thus the problem reduces to finding, throughout experiments, such n for which the model is the best replica of the system.

3.2.1. Model structure determination using GP

If the terminal and function sets are given, populations of GP individuals (trees) can be generated, i.e. the set $\mathbb{M}$ of possible model structures is created. An outline of the GP algorithm implemented in this work is shown in Tab. 3.1. The algorithm works on a set of populations $\mathbb{P} = \{P_i \mid i = 1, \ldots, n_p\}$, and the number of populations n_p depends on the application, e.g. in the case of the model (3.1), the number of populations is equal to the dimension m of the output vector $\boldsymbol{y}_k$, i.e. $n_p = m$. Each of the above populations $P_i = \{b_{ij} \mid j = 1, \ldots, n_m\}$ is composed of a set of n_m trees b_{ij}. Since the number of populations is given, the GP

Tab.3.1. An outline of the GP algorithm

```
I. Initiation
    A. Random generation    $\mathbb{P}(0) = \big\{P_i(0) \mid i = 1,\dots,n_p\big\}$.
    B. Fitness calculation    $\Phi\big(\mathbb{P}(0)\big) = \big\{\Phi\big(P_i(0)\big) \mid i = 1,\dots,n_p\big\}$.
    C. $t = 1$.
II. While    $\big(\iota\big(\mathbb{P}(t)\big) = \mathrm{true}\big)$    do
    A. Selection    $\mathbb{P}'(t) = \big\{P_i'(t) = s_{n_s}\big(P_i(t)\big) \mid i = 1,\dots,n_p\big\}$.
    B. Crossover    $\mathbb{P}''(t) = \big\{P_i''(t) = r_{P_{cross}}\big(P_i'(t)\big) \mid i = 1,\dots,n_p\big\}$.
    C. Mutation    $\mathbb{P}'''(t) = \big\{P_i'''(t) = m_{P_{mut}}\big(P_i''(t)\big) \mid i = 1,\dots,n_p\big\}$.
    D. Fitness calculation    $\Phi\big(\mathbb{P}'''(t)\big) = \big\{\Phi\big(P_i'''(t)\big) \mid i = 1,\dots,n_p\big\}$.
    E. New generation    $\mathbb{P}(t+1) = \big\{P_i(t+1) = P_i'''(t) \mid i = 1,\dots,n_p\big\}$.
    F. $t = t + 1$.
```

algorithm can be started (*initiation*) by randomly generating individuals, i.e. n_m individuals are created in each population whose trees are of a desired depth n_d. The tree generating process can be performed in several different ways, resulting in trees of different shapes. The basic approaches are the "full" and "grow" methods (Koza 1992). The "full" method generates trees for which the length of every nonbacktracking path from the root to an endpoint is equal to the prespecified depth n_d. The "grow" method generates trees of various shapes. The length of a path between the root and an endpoint is not greater than the prespecified depth n_d. Because of the fact that, in general, the shape of the true solution is unknown, it seems desirable to combine both of the above methods. Such a combination is called "ramped half-and-half". Moreover, it is assumed that the parameters $\boldsymbol{p} = (\boldsymbol{p}_1, \dots, \boldsymbol{p}_m)$ of each tree are initially set to unity (although it is possible to set the parameters randomly). In the first step (*fitness calculation*), estimation of the parameter vector $\boldsymbol{p}$ of each individual is performed, according to (1.47). In the case of parameter estimation, many algorithms can be employed; more precisely, as GP models are usually non-linear in their parameters, the choice reduces to one of non-linear optimisation techniques. Unfortunately, because the models are randomly generated, they can contain linearly dependent parameters (even after the application of parameter reduction rules) and parameters which have very little influence on the model output. In many cases, this may lead to a very pure performance of gradient-based algorithms. Owing to the above-mentioned problems, the spectrum of possible non-linear optimisation techniques reduces to gradient-free techniques, which usually require a large number of cost evaluations. On the other hand, the application of stochastic gradient-free algorithms, apart from the simplicity of the approach, decreases the chance to get stuck in a local optimum, and hence it may give more suitable parameter estimates. Based on numerous computer experiments, it has been found that the extremely simple Adaptive Random Search (ARS) algorithm (Walter and Pronzato 1997) is especially well suited for that purpose. The routine chooses the initial parameter vector $\boldsymbol{p}^0$, e.g. $\boldsymbol{p}^0 = \mathbf{1}$. After q iterations, given the current best estimate $\boldsymbol{p}^q$, a random displacement vector $\boldsymbol{\Delta p}$ is generated and the trial point:

$$\boldsymbol{p}^* = \boldsymbol{p}^q + \boldsymbol{\Delta p}, \tag{3.4}$$

is checked, with $\boldsymbol{\Delta p}$ following normal distribution with zero-mean and covariance:

$$\boldsymbol{\Sigma} = \mathrm{diag}[\sigma_1, \dots, \sigma_{\dim \boldsymbol{p}}]. \tag{3.5}$$

If $j(M(\boldsymbol{p}^*)) > j(M(\boldsymbol{p}^q))$, then $\boldsymbol{p}^*$ is rejected and, consequently, $\boldsymbol{p}^{q+1} = \boldsymbol{p}^q$ is set; otherwise, $\boldsymbol{p}^{q+1} = \boldsymbol{p}^*$ The adaptive strategy consists in repeatedly alternating two phases. During the first one (*variance selection*), $\boldsymbol{\Sigma}$ is selected from the sequence ${}^1\sigma, \dots, {}^5\sigma$, where ${}^1\sigma$ is set by the user in such a way as to allow an easy exploration of the parameter space, and:

$${}^i\sigma = {}^{i-1}\sigma/10, \quad i = 2, \dots, 5. \tag{3.6}$$

In order to allow a comparison to be drawn, all the possible ${}^i\sigma$'s are used $100/i$ times, starting from the same initial value of $\boldsymbol{p}$. The largest ${}^i\sigma$'s, designed to escape the local minimum, are therefore used more often than the smaller ones. During the second (exploration) phase, the most successful ${}^{i}\sigma$ is used to perform 100 random trials starting from the best $\boldsymbol{p}$ obtained so far.

In the next step, using (1.45), fitness of each model is obtained and the best-suited model is selected by the use of (1.44). If the selected model satisfies the prespecified requirements, then the algorithm is stopped. In the second step, the *selection* process is applied to create a new intermediate population of parent individuals. For that purpose, various approaches can be employed, e.g. proportional selection, rank selection, tournament selection (Koza 1992, Michalewicz 1996). The selection method used in the present work is the tournament selection, and it works as follows: select randomly n_s models, i.e. trees which represent the models, and copy the best of them into the intermediate set of models (intermediate populations). The above procedure is repeated n_m times.
The individuals for the new population (the next generation) are produced through the application of *crossover* and *mutation.* To apply *crossover* $r_{P_{\text{cross}}}$, random couples of individuals which have the same position in each population are formed. Then, with a probability P_{cross}, each couple undergoes crossover, i.e. a random crossover point (node) is selected and then the corresponding sub-trees are exchanged (Fig. 3.6). *Mutation* $m_{P_{mut}}$ (Fig. 3.7) is implemented such that for each entry of each individual a sub-tree at a selected point is removed with probability P_{mut} and replaced with a randomly generated tree. The parameter vectors of individuals which have been modified by means of either crossover or mutation are set to unity (although other choice is possible), and the other node parameters vectors remain unchanged. The GP algorithm is repeated until the best-suited model satisfies the prespecified requirements $\iota(\mathbb{P}(t))$, or until the number of maximum admissible iterations has been exceeded. It should also be pointed out that the simulation programme must ensure robustness to unstable models. This can

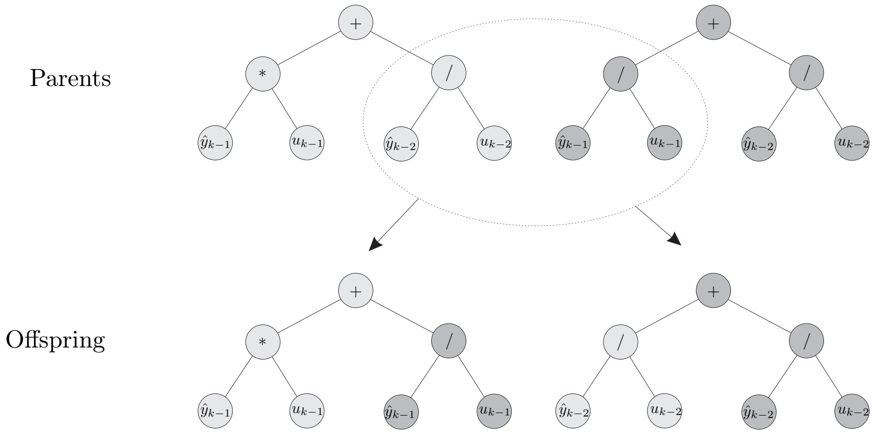

Fig. 3.6 . An exemplary crossover operation.

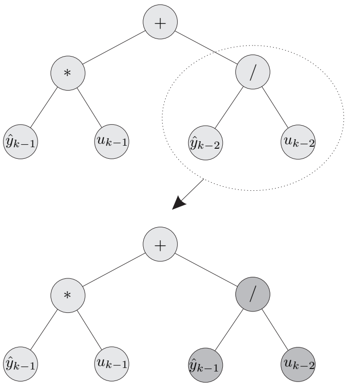

Fig. 3.7. An exemplary mutation operation.

easily be attained when (1.47) is bounded by a certain maximum admissible value. This means that each individual which exceeds the above bound is penalised by stopping the calculation of its fitness, and then $J_m(M_i)$ is set to a sufficiently large positive number. This problem is especially important in the case of the input-output representation of the system. Unfortunately, the stability of models resulting from this approach is very difficult to prove. However, this is a common problem with non-linear input-output dynamic models. To overcome it, an alternative model structure is presented in the subsequent section.

3.3. State-space representation of the system

Let us consider the following class of non-linear discrete-time systems:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \tag{3.7}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}\boldsymbol{x}_{k+1} + \boldsymbol{v}_k. \tag{3.8}$$

Assume that the function $\boldsymbol{g}(\cdot)$ has the form:

$$\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}(\boldsymbol{x}_k)\boldsymbol{x}_k + \boldsymbol{h}(\boldsymbol{u}_k). \tag{3.9}$$

The choice of the structure (3.9) is caused by the fact that the resulting model is to be used in FDI systems. The algorithm presented below though can also, with minor modifications, be applied to the following structures of $\boldsymbol{g}(\cdot)$:

$$\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}(\boldsymbol{x}_k, \boldsymbol{u}_k)\boldsymbol{x}_k, \tag{3.10}$$

$$\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}(\boldsymbol{x}_k, \boldsymbol{u}_k)\boldsymbol{x}_k + \boldsymbol{h}(\boldsymbol{u}_k), \tag{3.11}$$

$$\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}(\boldsymbol{x}_k, \boldsymbol{u}_k)\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{x}_k)\boldsymbol{u}_k, \tag{3.12}$$

$$\boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}(\boldsymbol{x}_k)\boldsymbol{x}_k + \boldsymbol{B}(\boldsymbol{x}_k)\boldsymbol{u}_k. \tag{3.13}$$

The state-space model of the system (3.7)-(3.8) can be expressed as:

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}(\hat{\boldsymbol{x}}_k)\hat{\boldsymbol{x}}_k + \boldsymbol{h}(\boldsymbol{u}_k), \tag{3.14}$$

$$\hat{\boldsymbol{y}}_{k+1} = \boldsymbol{C}\hat{\boldsymbol{x}}_{k+1}. \tag{3.15}$$

The problem is to determine the matrices $\boldsymbol{A}(\cdot)$, $\boldsymbol{C}$ and the vector $\boldsymbol{h}(\cdot)$, given the sets of input-output measurements $\{(\boldsymbol{u}_k, \boldsymbol{y}_k)\}_{k=0}^{n_t-1}$ and $\{(\boldsymbol{u}_k, \boldsymbol{y}_k)\}_{k=0}^{n_v-1}$. Moreover, it is assumed that the true state vector $\boldsymbol{x}_k$ is, in particular, unknown. Without loss of generality, it is possible to assume that:

$$\boldsymbol{A}(\hat{\boldsymbol{x}}_k) = \mathrm{diag}[a_{1,1}(\hat{\boldsymbol{x}}_k), a_{2,2}(\hat{\boldsymbol{x}}_k), \ldots, a_{n,n}(\hat{\boldsymbol{x}}_k)]. \tag{3.16}$$

Thus, the problem reduces to identifying the non-linear functions $a_{i,i}(\hat{\boldsymbol{x}}_k),\ h_i(\boldsymbol{u}_k)\ i = 1, \ldots, n$, and the matrix $\boldsymbol{C}$. Now it is possible to establish the conditions under which the model (3.14)-(3.15) is globally asymptotically stable. The following theorem is based on the theorems presented in (Bubnicki 2000).

Theorem 3.3.1. *If, for $\boldsymbol{h}(\boldsymbol{u}_k) = \boldsymbol{0}$,*

$$\forall k \geq 0, \quad \forall \hat{\boldsymbol{x}}_k \in \mathbb{R}^n, \quad \max_{i=1,\ldots,n} |a_{i,i}(\hat{\boldsymbol{x}}_k)| < 1, \tag{3.17}$$

then the model (3.14)-(3.15) is globally asymptotically stable, i.e. $\hat{\boldsymbol{x}}_k$ converges to the equilibrium point $\hat{\boldsymbol{x}}^*$ for any $\hat{\boldsymbol{x}}_0$.

Proof. Since the matrix $\boldsymbol{A}(\hat{\boldsymbol{x}}_k)$ is a diagonal one,

$$\|\boldsymbol{A}(\hat{\boldsymbol{x}}_k)\| = \max_{i=1,\ldots,n} |\lambda_i(\boldsymbol{A}(\hat{\boldsymbol{x}}_k))| = \max_{i=1,\ldots,n} |a_{i,i}(\hat{\boldsymbol{x}}_k)|, \tag{3.18}$$

where the norm $\|\boldsymbol{A}(\cdot)\|$ may have one of the following forms:

$$\|\boldsymbol{A}(\cdot)\|_2 = \sqrt{\lambda_{\max}(\boldsymbol{A}(\cdot)^T\boldsymbol{A}(\cdot))}, \tag{3.19}$$

$$\|\boldsymbol{A}(\cdot)\|_1 = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{i,j}(\cdot)|, \tag{3.20}$$

$$\|\boldsymbol{A}(\cdot)\|_\infty = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{i,j}(\cdot)|. \tag{3.21}$$

Finally, using (Bubnicki 2000, Proof of Theorem 1) yields the condition (3.17). □

Since the stability conditions are established, it is possible to give a general framework for the identification of (3.14)-(3.15). Since $a_{i,i}(\hat{\boldsymbol{x}}_k),\ h_i(\boldsymbol{u}_k),\ i = 1, \ldots, n$ are assumed to be non-linear (in general) functions, it is necessary to use n populations to represent $a_{i,i}(\hat{\boldsymbol{x}}_k),\ i = 1, \ldots, n$, and another n populations to represent $h_i(\boldsymbol{u}_k),\ i = 1, \ldots, n$. Thus the number of populations is $n_p = 2n$. The terminal sets for these two kinds of populations are different, i.e. the first terminal set is defined as $\mathbb{T}_A = \{\hat{\boldsymbol{x}}_k\}$, and the second one as $\mathbb{T}_h = \{\boldsymbol{u}_k\}$. The parameter vector $\boldsymbol{p}$ consists of the parameters of both $a_{i,i}(\hat{\boldsymbol{x}}_k)$ and $h_i(\boldsymbol{u}_k)$. Unfortunately, estimation of $\boldsymbol{p}$ is not as simple as in the input-output representation case. This means that checking the trial point in the ARS algorithm (see Section 3.2.1) involves computation of $\boldsymbol{C}$, which is necessary to obtain the output error $\boldsymbol{\varepsilon}_k$ and, consequently, the value of the fitness function (1.44). To tackle this problem, for each trial point $\boldsymbol{p}$ it is necessary to first set an initial state estimate $\hat{\boldsymbol{x}}_0$, and then to obtain the state estimate $\hat{\boldsymbol{x}}_k$, $k = 1, \ldots, n_t - 1$. Knowing the state estimate and using the least-square method, it is possible to obtain $\boldsymbol{C}$ by solving the following equation:

$$\boldsymbol{C} \sum_{k=0}^{n_t-1} \hat{\boldsymbol{x}}_k \hat{\boldsymbol{x}}_k^T = \sum_{k=0}^{n_t-1} \boldsymbol{y}_k \hat{\boldsymbol{x}}_k^T, \tag{3.22}$$

or, equivalently, by using:

$$\boldsymbol{C} = \sum_{k=0}^{n_t-1} \boldsymbol{y}_k \hat{\boldsymbol{x}}_k^T \left[\sum_{k=0}^{n_t-1} \hat{\boldsymbol{x}}_k \hat{\boldsymbol{x}}_k^T \right]^{-1}. \tag{3.23}$$

Since the identification procedure of (3.14)-(3.15) is given, it is possible to establish the structure of $\boldsymbol{A}(\cdot)$, which guarantees that the condition of *Theorem 1* is always satisfied, i.e. $\max_{i=1,\ldots,n} |a_{i,i}(\hat{\boldsymbol{x}}_k)| < 1$. This can easily be achieved with the following structure of $a_{i,i}(\hat{\boldsymbol{x}}_k)$:

$$a_{i,i}(\hat{\boldsymbol{x}}_k) = \tanh(s_{i,i}(\hat{\boldsymbol{x}}_k)), \quad i = 1, \ldots, n, \tag{3.24}$$

where $\tanh(\cdot)$ is a hyperbolic tangent function, and $s_{i,i}(\hat{\boldsymbol{x}}_k)$ is a function represented by the GP-tree. It should be also pointed out that the order n of the model is in general unknown and hence should be determined throughout experiments.

3.4. Experimental results

The main objective of further investigations is to show the reliability and effectiveness of the system identification technique proposed in the present chapter. In particular, real data from an industrial plant were employed to identify both the input-output and state-space models of chosen parts of the plant. The plant to be considered is an evaporation station at the Lublin Sugar Factory (Poland) (Kościelny *et al.* 1999, Edelmayer 2000). Fig. 3.8 shows a scheme of the plant with all the available process variables. These process variables are described in Tab. 3.2. The models to be obtained are as follows (cf. Fig. 3.8):

- The vapour model
  The input and output vectors: $\boldsymbol{u}_k = (T51\_07)$, $\boldsymbol{y}_k = (P51\_03)$,

- The apparatus model
  The input and output vectors:
  $\boldsymbol{u}_k = (T51\_06, TC51\_05, F51\_01, F51\_02)$, $\boldsymbol{y}_k = (T51\_08)$.

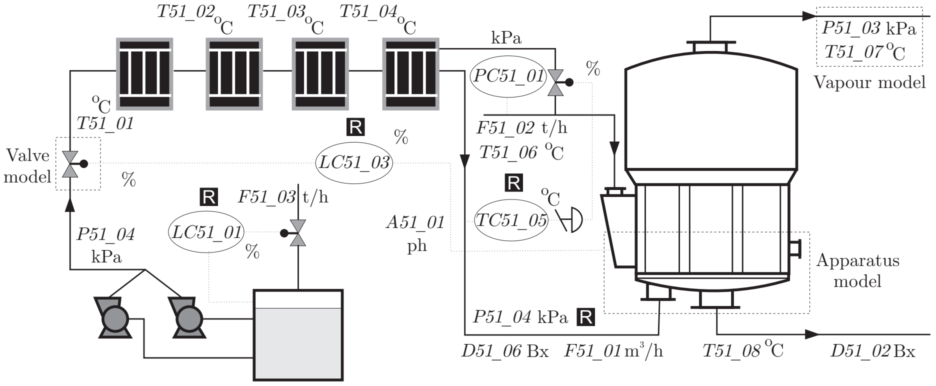

Fig. 3.8 . A scheme of the evaporation station.

The data used for the identification and validation data sets were collected from two different shifts in November 1998. The data from the first shift were used for identification, and the data from the second one formed the validation data set. Unfortunately, the data turned out to be sampled too fast (the sampling rate was 10s). Thus, every 10-th value was picked, after proper prefiltering, resulting in the $n_v = n_t = 700$-th element identification and validation data sets. After this, the offset levels were removed with the use of *MATLAB System Identification Toolbox*.

3.4.1. The vapour model

The objective of this section is to design an input-output vapour model according to the approach described in Section 3.2. The parameters used during the identification process were $P_{\text{cross}} = 0.8$, $P_{\text{mut}} = 0.01$, $n_m = 200$, $n_d = 10$, $n_s = 10$, $\mathbb{F} = \{+, *, /\}$. Moreover, for the sake of comparison, the ARX model was obtained. In both the ARX and non-linear cases the order of the model was tested between $n_y = n_u = 1, \ldots, 4$.

Experimental results have shown that the best-suited ARX model is of order $n_y = n_u = 4$. On the contrary, after the 50 runs of the GP algorithm performed for each model order, it was found that the order of the model which provides the best approximation quality is $n_y = n_u = 2$. The best obtained model structure is given by:

$$\begin{aligned}\hat{y}_k =& ((\hat{p}_2 u_{k-2} + \hat{p}_1 \hat{y}_{k-2}) u_{k-1}^2 + (\hat{p}_5 u_{k-2} \hat{y}_{k-1} \\ &+ \hat{p}_6 u_{k-2}^2 + \hat{p}_3 \hat{y}_{k-1}^2 + \hat{p}_4 \hat{y}_{k-1} u_{k-2} + \hat{p}_9) u_{k-1} \\ &+ \hat{p}_7 u_{k-2} \hat{y}_{k-1}^2 + \hat{p}_8 \hat{y}_{k-1} u_{k-2}^2)/(\hat{p}_{10} \hat{y}_{k-1} + \hat{p}_{11} \hat{y}_{k-1}^2 \\ &+ \hat{p}_{12} \hat{y}_{k-1} u_{k-2} + \hat{p}_{13}),\end{aligned} \tag{3.25}$$

Tab.3.2. A specification of the process variables

$F51\_01$	Thin juice flow at the inlet of the evaporation station
$F51\_02$	Steam flow at the inlet of the evaporation station
$LC51\_03$	Juice level in the first section of the evaporation station
$P51\_03$	Vapour pressure in the first section of the evaporation station
$P51\_04$	Juice pressure at the inlet of the evaporation station
$T51\_06$	Input steam temperature
$T51\_07$	Vapour temperature in the first section of the evaporation station
$T51\_08$	Juice temperature at the outlet of the first section of the evaporation station
$TC51\_05$	Thin juice temperature at the outlet of the heater

where:

$$\begin{aligned}\hat{\boldsymbol{p}} = (&-0.021, 0.495, 1.682, -0.832, 0.601,\\ &0.877, -1.396, 1.206, 1.931, -0.091, 0.067, -0.038, 0.495). \quad (3.26)\end{aligned}$$

A comparative study performed for the ARX and GP-based models shows that the GP model is superior to the ARX model. Indeed, the mean-squared output error was 1.5 and 3.77 for the GP and ARX models, respectively. However, this superiority can be seen especially clearly in the case of the validation data set, for which the mean-squared error was 6.7 and 21.5 for the GP and ARX models, respectively. From these results it can be seen that the introduction of the non-linear model has significantly improved modelling performance. While a linear model may be acceptable in the case of the identification data set, it is clear that its generalisation abilities are rather unsatisfactory, which has been shown throughout the test on the validation data set. The response of the model obtained for both the identification and validation data sets is given in Fig. 3.9.

The main drawback to the GP-based identification algorithm concerns its convergence abilities. Indeed, it seems very difficult to establish the convergence conditions which can guarantee the convergence of the proposed algorithm. On the other hand, many examples treated in the literature, cf. (Esparcia-Alcazar 1998, Gray *et al.* 1998, Koza 1992) and the references therein, as well as the author's experience with GP (Witczak and Korbicz 2000a, Witczak and Korbicz 2000b, Witczak and Korbicz 2001c, Witczak and Korbicz 2002) confirm its particular usefulness, in spite of the lack of the convergence proof. In the case of the presented example, the average fitness (mean-squared output error for the identification data set), Fig. 3.10, for the 50 runs of the algorithm confirms the modelling abilities of the approach. Moreover, based on the fitness attained by each of the 50 models (resulting from the 50 runs), it is possible to obtain a histogram representing the

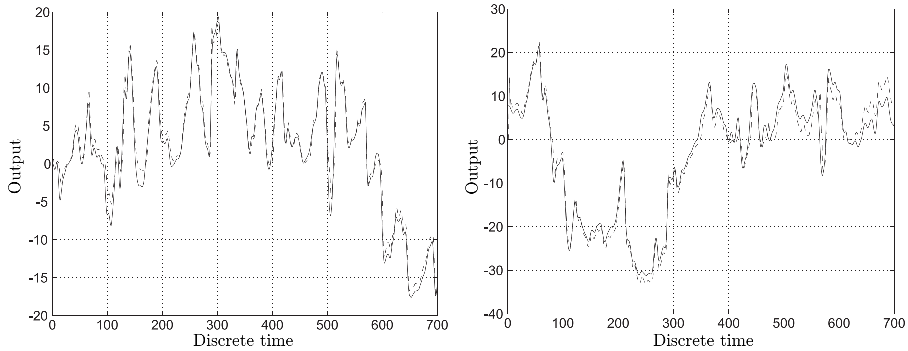

Fig. 3.9 . The system (solid line) and model (dashed line) output for the identification (left) and validation (right) data sets.

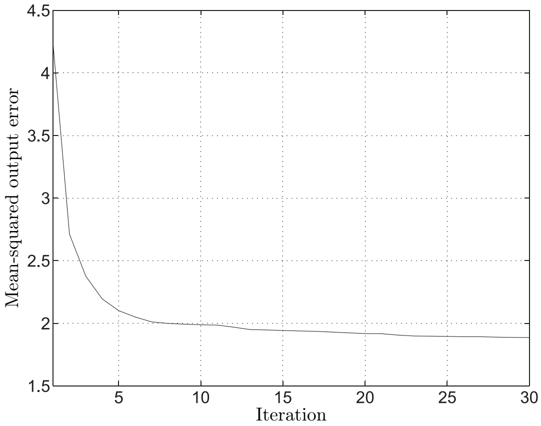

Fig. 3.10 . The average fitness for the 50 runs of the algorithm.

achieved fitness values (Fig. 3.11) as well as the fitness confidence region. Let $\alpha = 0.99$ denote the confidence level. Then the corresponding confidence region can be defined as:

$$\bar{J}_m \in [\bar{j}_m - t_\alpha \frac{s}{\sqrt{50}}, \bar{j}_m + t_\alpha \frac{s}{\sqrt{50}}], \tag{3.27}$$

where $\bar{j}_m = 1.89$ and $s = 0.64$ denote the arithmetic mean and standard deviation of the fitness of the 50 models, while $t_\alpha = 2.58$ is the normal distribution quantile. According to (3.27), the fitness confidence region is $\bar{J}_m \in [1.65, 2.12]$, which means that the probability that the true mean fitness $\bar{J}_m$ belongs to this region is 99%. On the other hand, owing to the multimodal properties of the identification index, it can be observed (Fig. 3.11) that there are two optima resulting in models of different quality. However, it should be pointed out that, on average (Fig. 3.10),

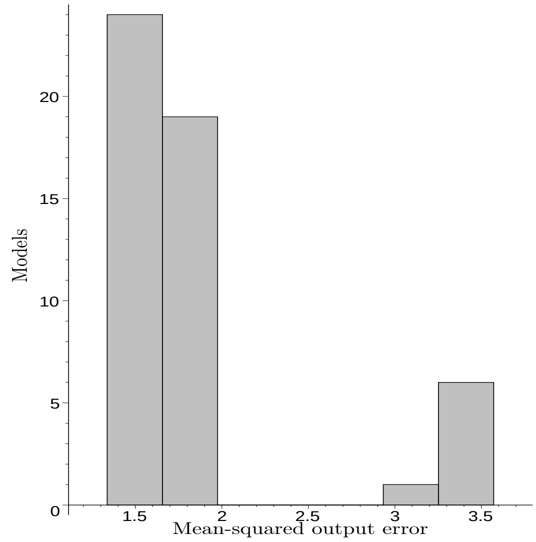

Fig. 3.11 . A histogram representing the fitness of the 50 models.

the algorithm converges to the optimum resulting in models of better quality. The convergence abilities of the algorithm can be further increased by the application of various parameters, e.g. P_{cross}, P_{mut}, control strategies (see, e.g., (Eiben *et al.* 1999)), but this is beyond the scope of the present section.

The above results confirm that, even if there is no convergence proof, the proposed approach can be successfully used to tackle the non-linear system identification problem.

3.4.2. The apparatus model

The objective of this section is to design the state-space apparatus model (cf. Fig. 3.8) according to the approach described in Section 3.3. The parameters used in the GP algorithm are the same as in Section 3.4.1. Similarly, for the sake of comparison, the linear state-space model was obtained. In both the linear and non-linear cases the order of the model was tested between $n = 2, \ldots, 4$.

Experimental results showed that the best-suited linear model is of order $n = 4$. After the 50 runs of the GP algorithm performed for each model order, it was found that the order of the model which provides the best approximation quality is $n = 2$. The best obtained model structure is given by:

$$\hat{x}_{1,k+1} = \mathrm{tgh}(s_{1,1})\hat{x}_{1,k} + h_1(\boldsymbol{u}_k), \tag{3.28}$$

$$\hat{x}_{2,k+1} = \mathrm{tgh}(s_{2,2})\hat{x}_{2,k} + h_2(\boldsymbol{u}_k), \tag{3.29}$$

where:

$$s_{1,1} = -0.13\hat{x}_{2,k}, \quad s_{2,2} = \frac{\hat{x}_{2,k}}{\hat{x}_{1,k}(\hat{x}_{2,k}\hat{x}_{1,k}+2\hat{x}^2_{2,k}\hat{x}^2_{1,k}+1)}, \tag{3.30}$$

$$\begin{aligned} h_1(\boldsymbol{u}_k) =& (u_{1,k} + (u_{1,k} + 2u_{4,k} + u_{4,k}u_{1,k})(u_{1,k} \\ & + u_{4,k} + u_{3,k} + u_{4,k}u_{1,k}))u_{3,k}+ \\ & + u_{3,k}(u_{1,k} + (u_{1,k} + u_{4,k} + u_{3,k} + u_{4,k}u_{1,k}) \\ & (u_{1,k} + \frac{u_{4,k}u_{3,k}u_{1,k}}{u_{4,k}+u_{2,k}} + 2u_{4,k})), \end{aligned} \tag{3.31}$$

$$h_2(\boldsymbol{u}_k) = u_{1,k} + u_{2,k}, \tag{3.32}$$

and:

$$\boldsymbol{C} = [0.21 * 10^{-5}, 0.51]. \tag{3.33}$$

A comparative study performed for the linear model and the GP model shows that the GP model is superior to the linear state-space model. Indeed, the mean-squared output error was 0.05 and 0.2 for the GP and linear state-space models, respectively. This superiority was also confirmed in the case of the validation data set, for which the mean-squared error was 0.5 and 2.1 for the GP and linear state-space models, respectively. From these results it can be seen that the proposed non-linear state-space model identification approach can effectively be applied to various system identification tasks. The response of the model obtained for both the identification and validation data sets is given in Fig. 3.12. The average fitness

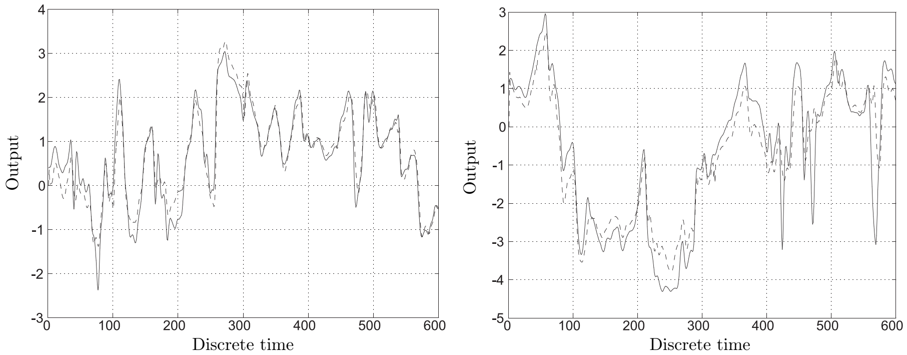

Fig. 3.12. The system (solid line) and model (dashed line) output for the identification (left) and validation (right) data sets.

(mean-squared output error for the identification data set), Fig. 3.13, for the 50 runs of the algorithm confirms the modelling abilities of the approach. As previ-

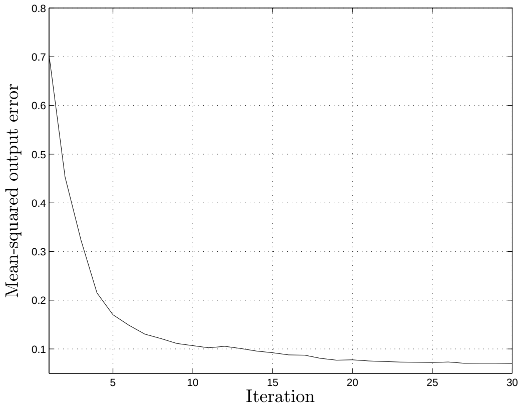

Fig. 3.13. The average fitness for the 50 runs of the algorithm.

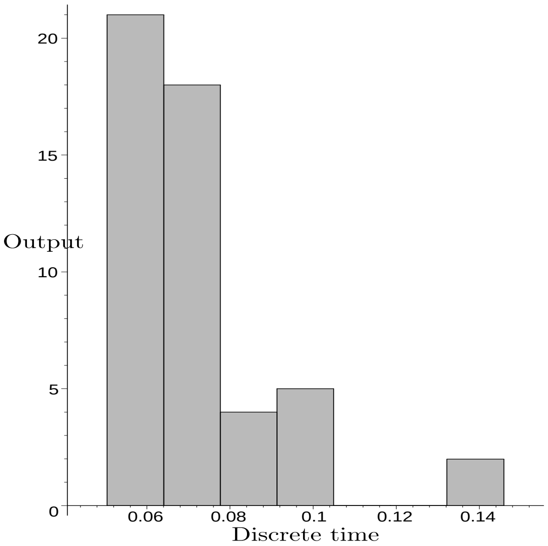

Fig. 3.14. A histogram representing the fitness of the 50 models.

ously, based on the fitness attained by each of the 50 models (resulting from the 50 runs), it is possible to obtain a histogram representing the achieved fitness values (Fig. 3.14) as well as the fitness confidence region. According to (3.27), the fitness confidence region is $\bar{J}_m \in [0.06, 0.078]$ (for $s = 0.02$, $\bar{j}_m = 0.07$), which means that the probability that the true mean fitness $\bar{J}_m$ belongs to this region is 99%. As in the previous section, it can be observed (Fig. 3.14) that there are two optima in the space of the models. However, on average, the algorithm is convergent to the optima resulting in models of better quality.

3.5. Further improvements: Selection strategies of control parameters

The preliminary operation while applying genetic programming algorithm is the specification of its components such as the mutation and crossover probabilities P_{mut} and P_{cross}, population size n_p, fitness function, etc. These parameters greatly determine whether or not the algorithm will find a near optimum solution and whether it will find such a solution efficiently. The rules of setting most of them have already been described.

Undoubtedly, it seems profitable to have as large a population as possible. The size is limited by the abilities of computers and the maximum admissible evolution time. Thus, a population usually consists of $100 - 1000$ individuals.

On the other hand, the mutation and crossover probabilities P_{mut} and P_{cross} can be set arbitrarily within the range $[0, 1]$. The most common approach to employed set these parameters consists in setting them before the run of the algorithm and then running the algorithm using these values, which remain fixed during the run. In such cases, typically exploited values, e.g. $P_{\mathrm{mut}} = 0.01$, $P_{\mathrm{cross}} = 0.8$, are used without much justification of the choice made. A general drawback to such techniques can be deduced from the observation of a run of the genetic programming algorithm, which is an intrinsically dynamic, adaptive process. Moreover, it seems intuitively obvious that different values of parameters may be advantageous in different stages of an evolution process.

Such parameter adaptation rules can be described as:

$$P_{\mathrm{cross},k+1} = f_1(P_{\mathrm{cross},k}), \quad P_{\mathrm{mut},k+1} = f_2(P_{\mathrm{mut},k}), \tag{3.34}$$

where k stands for the generation number. It seems, of course, impossible to analytically derive the rules of controlling the parameter values $f_1(\cdot)$, $f_2(\cdot)$.

One way out of this problem is to incorporate the parameters $P_{\mathrm{mut},k}$ and $P_{\mathrm{cross},k}$ into the chromosomes (see (Eiben *et al.* 1999) for a survey). But in the case of GP this seems inconvenient because the chromosome is represented by a tree, which is rather inappropriate for representing simple numbers. Another possibility is to use some heuristic rule which employs the feedback from the current search state to modify the parameters. There are, of course, many more or less sophisticated approaches which can be applied to settle this problem (Eiben *et al.* 1999). For example, the celebrated Rechenberg "1/5 success rule", which states that the ratio ξ of successful mutations to all mutations should be $1/5$. If this is not the case, the probability of mutation should be modified as follows:

$$\text{if} \quad (k \bmod n = 0) \quad \text{then}$$

$$P_{\mathrm{mut},k} = \begin{cases} P_{\mathrm{mut},k-n}/c, & \xi > 1/5, \\ P_{\mathrm{mut},k-n} \cdot c, & \xi < 1/5, \\ P_{\mathrm{mut},k}, & \xi = 1/5, \end{cases}$$

$$\text{else} \quad P_{\mathrm{mut},k} = P_{\mathrm{mut},k-1},$$

where $0.817 \leq c \leq 1$. Using this mechanism means that changes in the probability of mutation happen every n-th generation.

Unfortunately, the known approaches (Eiben *et al.* 1999) neglect the statistical properties of a population. The purpose of further consideration is to establish a connection between such properties and probabilities of crossover and mutation resulting in adaptation rules $f_1(\cdot)$ and $f_2(\cdot)$.

It is clear that the arithmetic mean of the individuals' fitness should decrease from generation to generation in order to locate a sub-optimal solution to the problem under consideration. This means that the probability of crossover should be appropriately high while the probability of mutation should be low. On the other hand, the variance of the individuals' fitness should be as large as possible in order to guarantee a large spectrum of possible structures (genotypes) from which a sub-optimal solution is to be obtained. Moreover, it is clear that the probability of crossover should be decreased when the algorithms get stuck in unsatisfactory local minima, i.e. when the arithmetic mean remains constant from generation to generation, while the probability of mutation should be increased. Such an increase is performed in order to generate new individuals which help to escape local optima. Finally, $P_{\text{mut},k}$ and $P_{\text{cross},k}$ should belong to prespecified ranges.

All these requirements can be fulfilled by the following adaptation rules:

$$P_{\text{cross},k+2} = \begin{cases} \text{for } m_{k+1} < m_k \\ P_{\text{cross}}^{\text{min}} + (P_{\text{cross},k+1} - P_{\text{cross}}^{\text{min}}) \left(1 - e^{\frac{-p_1 m_k}{m_{k+1}}}\right), \\ \text{otherwise} \\ P_{\text{cross},k+1} + (1 - P_{\text{cross},k+1}) e^{\frac{-p_1 m_k}{m_{k+1}}}, \end{cases} \tag{3.35}$$

and:

$$P_{\text{mut},k+2} = \begin{cases} \text{for } v_{k+1} > v_k \\ P_{\text{mut}}^{\text{min}} + (P_{\text{mut},k+1} - P_{\text{mut}}^{\text{min}}) \left(1 - e^{\frac{-p_2 v_{k+1}}{v_k}}\right) \\ \text{otherwise} \\ P_{\text{mut},k+1} + (P_{\text{mut}}^{\text{max}} - P_{\text{mut},k+1}) e^{\frac{-p_2 v_{k+1}}{v_k}}, \end{cases} \tag{3.36}$$

where m_k and v_k are respectively the arithmetic mean and the coefficient of variation of the individuals' fitness, p_1 and p_2 stands for additional design parameters which can be applied to modify the probabilities' changes, $P_{\text{mut}}^{\text{min}}$ and $P_{\text{cross}}^{\text{min}}$ are the minimum allowable probabilities of mutation and crossover, respectively, and $P_{\text{mut}}^{\text{max}}$ is the maximum probability of mutation. It should also be mentioned that $P_{\text{mut},0} = P_{\text{mut},1} = P_{\text{mut}}^{\text{init}}$ and $P_{\text{cross},0} = P_{\text{cross},1} = P_{\text{cross}}^{\text{init}}$, where $P_{\text{mut}}^{\text{init}}$ and $P_{\text{cross}}^{\text{init}}$ stand for initial values of the probabilities of mutation and crossover, respectively.

Although the rules (3.35) and (3.36) are not optimal, it is expected that their application will improve the process of model building. In order to verify this practically, let us consider the following example, for which genetic programming

with constant values of P_{mut} and P_{cross} almost always finds the optimal solution. This means that any further improvements are really difficult to attain.
Example 3.1. The problem is to model the relation:

$$y = -\frac{1}{0.1 + u^2}, \tag{3.37}$$

with genetic programming, given a set of input-output measurements $\{u_i, y_i\}_{i=1}^{n_t}$, where u_i is generated according to the uniform distribution, i.e. $u_i \in \mathcal{U}(-2, 2)$, and $n_t = 10$. In particular, two cases were considered:

Case 1: Constant values: $P_{\text{cross}} = 0.8$, $P_{\text{cross}} = 0.01$,

Case 2: Selection according to (3.35) and (3.36).

The average fitness (mean-squared output error for data set), Fig. 3.15, for the 50 runs of the algorithm confirms the modelling abilities of the approach. Indeed, the results of the proposed approach provide faster convergence rate. Moreover, as can be seen from Fig. 3.16, for the proposed approach there are no local optima representing models of unacceptable quality, which is the case in the traditional approach.

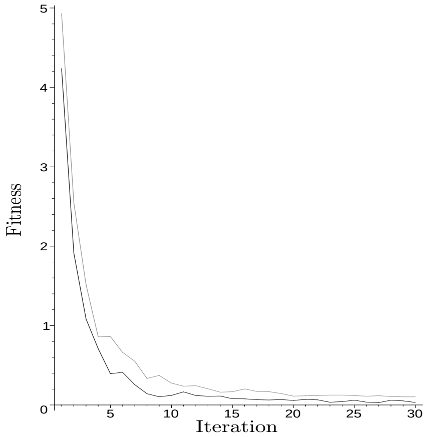

Fig. 3.15 . The average fitness for the 50 runs of the algorithm. Case 1 (dashed), Case 2 (solid).

3.6. Conclusions

The purpose of this chapter was to propose a unified framework for the identification of non-linear dynamic systems. To tackle this problem, a relatively new genetic programming technique was employed. In particular, it was shown how to represent various model structures as parameterised trees, and how to identify

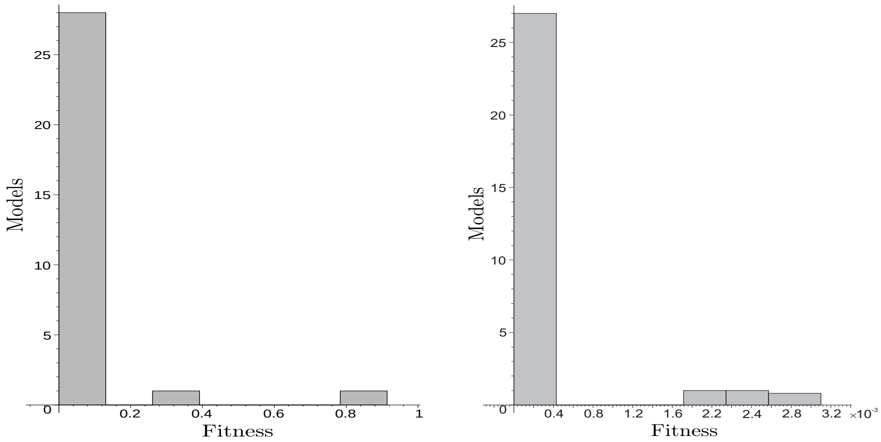

Fig. 3.16. Histograms representing the fitness of the 50 models. Case 1 (left), Case 2 (right).

their structure as well as estimate their parameters. Both the input-output and state-space model structures were presented. Moreover, it was proven that the proposed state-space model identification scheme provides asymptotically stable models.

The experimental results, covering model construction of chosen parts of an evaporation station at the Lublin Sugar Factory, confirm the reliability and effectiveness of the proposed identification framework.

The main drawback to this approach is its computational cost resulting in a relatively long identification time. However, as usual, the model construction procedure is realised *off-line*, and hence the identification time is not very important. Another drawback is that the model order has to be determined by a time-consuming trial-and-error process. This is, however, a problem with all nonlinear schemes. There are, of course, some approaches (Nelles 2001) which can be applied to estimate such a model order implicitly.

Moreover, an adaptation technique of the probabilities of crossover and mutation was proposed as well. The experimental results suggest that its application may result in models of better quality and that it reduces the required computation time.

Chapter 4

A ROBUST OBSERVER-BASED RESIDUAL GENERATION

Irrespective of the identification method used, there always exists the problem of model uncertainty, i.e. the model-reality mismatch. To overcome it, many approaches have been proposed (Chen and Patton 1999, Patton *et al.* 2000). Undoubtedly, the most common one is to use robust observers, such as the Unknown Input Observer (UIO) (Alcorta *et al.* 1997, Chen and Patton 1999, Chen *et al.* 1996, Patton and Chen 1997, Patton *et al.* 2000), which can tolerate a degree of model uncertainty and hence increase the reliability of fault diagnosis. Unfortunately, the design procedure for Non-linear Unknown Input Observers (NUIOs) (Alcorta *et al.* 1997, Seliger and Frank 2000) is usually very complex, even for simple laboratory systems (Zolghardi *et al.* 1996). One way out of this problem is to employ linearisation-based approaches, similar to the extended Kalman filter (Anderson and Moore 1979). In this case, the design procedure is almost as simple as that for linear systems. On the other hand, it is well known that such a solution works well only when there is no large mismatch between the model linearised around the current state estimate and the non-linear behaviour of the system. Thus, the problem is to improve the convergence of linearisation-based observers.

The application of the EKF to the state estimation of non-linear deterministic systems has received considerable attention during the last two decades (see (Boutayeb and Aubry 1999) and the references therein). This is mainly because the EKF can directly be applied to a large class of non-linear systems. Moreover, it is possible to show that the convergence of such a deterministic observer is ensured under certain conditions.

The main objective of further investigations is to show how to employ a modified version of the well-known UIO, which can be applied to linear stochastic systems, to form a non-linear deterministic observer. Moreover, it is shown that the convergence of the proposed observer is ensured under certain conditions (Witczak 2001a, Witczak and Korbicz 2001c, Witczak and Korbicz 2002), and that the convergence rate can dramatically be increased, compared to the classical approach, by the application of the genetic programming technique(Witczak and Korbicz 2001b, Witczak and Korbicz 2002). Moreover, it is shown how to use the proposed observer to tackle the problem of both sensor and actuator fault diagnosis.

Another problem arises for systems with both modelling uncertainty and the noise. Indeed, this problem has not attracted enough research attention, although real systems suffer from both modelling uncertainty and the noise. The existing approaches (see (Chen and Patton 1999, Chen *et al.* 1996, Keller and Darouach 1999) and the references therein) which can be applied to linear stochastic systems rely on a similar idea to that of the classical Kalman filter (Anderson and Moore 1979). The main drawback to such techniques lies in their restrictive assumptions concerning the noise distribution, i.e. it is assumed that the process and measurement noises are zero-mean white noise sequences. However, in many practical situations it is more natural to assume that only the bounds on the noise signals are available (for a detailed description of such approaches, we refer the reader to (Maksarow and Norton 1996a, Maksarow and Norton 1996b, Milanese *et al.* 1996, Walter and Pronzato 1997) and the references therein). This bounded-error approach describes the set of all the states that are consistent with the model, the measured data and the error (or the noise) bounds. All members of this feasible set are then possible solutions to the state estimation problem. Unfortunately, the set obtained in thsi way may become extremely complex. For the sake of computational complexity, this feasible set is usually characterised by the smallest (in some sense) ellipsoid that encloses it. Although, in the case of the observers of this type, the so-called unknown input can be treated in a similar way as the process noise, i.e. the only requirements are the bounds of the unknown input, it seems especially attractive to employ the bounded-error approach to design an UIO for linear stochastic systems (Witczak and Korbicz 2002a). This is especially true from the point of view of fault isolation. Indeed, in order to design a fault diagnosis system which is based on a bank of observers, each of the observers should be insensitive to one fault while sensitive to the others. This can be achieved by combining the classical UIO with bounded-error techniques, resulting in an observer for a wide class of linear stochastic systems.

Another problem arises from the application of fault diagnosis to non-linear stochastic systems. Unfortunately, the only existing approaches to this class of systems consist in the application of the Extended Kalman Filter (EKF) (Anderson and Moore 1979). Thus, it seems especially attractive to extend the proposed bounded-error UIO in such a way that it can be applied to non-linear stochastic systems (Witczak and Korbicz 2002a).

The chapter is organised as follows: Sections 5.1 and 5.2 propose unknown input observers for non-linear deterministic and stochastic systems, respectively.

4.1. An unknown input observer for non-linear deterministic systems

4.1.1. Preliminaries

This section presents a special version of the UIO which can be employed to tackle the fault detection problem of linear stochastic systems. The information presented below is necessary to explain further results of this work, and the reader is referred to (Chen and Patton 1999) and the references therein for further explanations. Following a common nomenclature, such an UIO will be called an Unknown Input Filter (UIF).

Let us consider the following linear discrete-time system:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k + \boldsymbol{w}_k, \tag{4.1}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1} + \boldsymbol{v}_k. \tag{4.2}$$

In this case, $\boldsymbol{v}_k$ and $\boldsymbol{w}_k$ are independent zero-mean white noise sequences. The matrices $\boldsymbol{A}_k$, $\boldsymbol{B}_k$, $\boldsymbol{C}_k$, $\boldsymbol{E}_k$ are assumed to be known and have appropriate dimensions. As has already been mentioned, the robustness to model uncertainty and to other factors which may lead to an unreliable fault detection is of great importance. In the case of the UIF, the robustness problem is tackled by introducing the concept of the unknown input $\boldsymbol{d}_k$; hence the term $\boldsymbol{E}_k\boldsymbol{d}_k$ may represent various kinds of modelling uncertainty, as well as real disturbances affecting the real system. To overcome the state estimation problem of (4.1)-(4.2), an UIF with the following structure can be employed:

$$\boldsymbol{z}_{k+1} = \boldsymbol{F}_{k+1}\boldsymbol{z}_k + \boldsymbol{T}_{k+1}\boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{K}_{k+1}\boldsymbol{y}_k, \tag{4.3}$$

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{z}_{k+1} + \boldsymbol{H}_{k+1}\boldsymbol{y}_{k+1}, \tag{4.4}$$

where:

$$\boldsymbol{K}_{k+1} = \boldsymbol{K}_{1,k+1} + \boldsymbol{K}_{2,k+1}, \tag{4.5}$$

$$\boldsymbol{E}_k = \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}\boldsymbol{E}_k, \tag{4.6}$$

$$\boldsymbol{T}_{k+1} = \boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}, \tag{4.7}$$

$$\boldsymbol{F}_{k+1} = \boldsymbol{T}_{k+1}\boldsymbol{A}_k - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k. \tag{4.8}$$

The above matrices are designed in such a way as to ensure the unknown input decoupling as well as the minimisation of the state estimation error:

$$\boldsymbol{e}_{k+1} = \boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1}. \tag{4.9}$$

It should also be pointed out that the necessary condition for the existence of a solution to (4.6) is $\mathrm{rank}(\boldsymbol{C}_{k+1}\boldsymbol{E}_k) = \mathrm{rank}(\boldsymbol{E}_k)$ (Chen and Patton 1999, p. 72, Lemma 3.1), and a special solution is:

$$\boldsymbol{H}^*_{k+1} = \boldsymbol{E}_k\left[(\boldsymbol{C}_{k+1}\boldsymbol{E}_k)^T\boldsymbol{C}_{k+1}\boldsymbol{E}_k\right]^{-1}(\boldsymbol{C}_{k+1}\boldsymbol{E}_k)^T. \tag{4.10}$$

If the conditions (4.5)-(4.8) are fulfilled, then the fault-free, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$, state estimation error is given by:

$$\boldsymbol{e}_{k+1} = \boldsymbol{F}_{k+1}\boldsymbol{e}_k - \boldsymbol{K}_{1,k+1}\boldsymbol{v}_k - \boldsymbol{H}_{k+1}\boldsymbol{v}_{k+1} + \boldsymbol{T}_{k+1}\boldsymbol{w}_k. \tag{4.11}$$

In order to obtain the gain matrix $\boldsymbol{K}_{1,k+1}$, let us first define the state estimation covariance matrix:

$$\boldsymbol{P}_k = \mathcal{E}\left\{[\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k][\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k]^T\right\}. \tag{4.12}$$

Using (4.11), the update of (4.12) can be defined as:

$$\begin{aligned}\boldsymbol{P}_{k+1} =& \boldsymbol{A}_{1,k+1}\boldsymbol{P}_k\boldsymbol{A}_{1,k+1}^T + \boldsymbol{T}_{k+1}\boldsymbol{Q}_k\boldsymbol{T}_{k+1}^T + \boldsymbol{H}_{k+1}\boldsymbol{R}_{k+1}\boldsymbol{H}_{k+1}^T \\ & - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{A}_{1,k+1}^T - \boldsymbol{A}_{1,k+1}\boldsymbol{P}_k\boldsymbol{C}_k^T\boldsymbol{K}_{1,k+1}^T \\ & + \boldsymbol{K}_{1,k+1}\left[\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T + \boldsymbol{R}_k\right]\boldsymbol{K}_{1,k+1}^T,\end{aligned} \tag{4.13}$$

where:

$$\boldsymbol{A}_{1,k+1} = \boldsymbol{T}_{k+1}\boldsymbol{A}_k. \tag{4.14}$$

To give the state estimation error $\boldsymbol{e}_{k+1}$ the minimum variance, it can be shown that the gain matrix $\boldsymbol{K}_{1,k+1}$ should be determined by:

$$\boldsymbol{K}_{1,k+1} = \boldsymbol{A}_{1,k+1}\boldsymbol{P}_k\boldsymbol{C}_k^T\left[\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1}. \tag{4.15}$$

In this case, the corresponding covariance matrix is given by:

$$\begin{aligned}\boldsymbol{P}_{k+1} =& \boldsymbol{A}_{1,k+1}\boldsymbol{P}'_{k+1}\boldsymbol{A}_{1,k+1}^T + \boldsymbol{T}_{k+1}\boldsymbol{Q}_k\boldsymbol{T}_{k+1}^T \\ & + \boldsymbol{H}_{k+1}\boldsymbol{R}_{k+1}\boldsymbol{H}_{k+1}^T,\end{aligned} \tag{4.16}$$

$$\boldsymbol{P}'_{k+1} = \boldsymbol{P}_k - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{A}_{1,k+1}^T. \tag{4.17}$$

The above derivation is very similar to that which has to be performed for the classical Kalman filter (Anderson and Moore 1979). Indeed, the UIF can be transformed to the KF-like form as follows:

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1} =& \boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{B}_k\boldsymbol{u}_k - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}[\boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{B}_k\boldsymbol{u}_k] \\ & - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\hat{\boldsymbol{x}}_k - \boldsymbol{F}_{k+1}\boldsymbol{H}_k\boldsymbol{y}_k \\ & + [\boldsymbol{K}_{1,k+1} + \boldsymbol{F}_{k+1}\boldsymbol{H}_k]\boldsymbol{y}_k + \boldsymbol{H}_{k+1}\boldsymbol{y}_{k+1},\end{aligned} \tag{4.18}$$

or, equivalently:

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1/k} + \boldsymbol{H}_{k+1}\boldsymbol{\varepsilon}_{k+1/k} + \boldsymbol{K}_{1,k+1}\boldsymbol{\varepsilon}_k, \tag{4.19}$$

where:

$$\hat{\boldsymbol{x}}_{k+1/k} = \boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{B}_k\boldsymbol{u}_k, \tag{4.20}$$

$$\boldsymbol{\varepsilon}_{k+1/k} = \boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1/k} = \boldsymbol{y}_{k+1} - \boldsymbol{C}_{k+1}\hat{\boldsymbol{x}}_{k+1/k}, \tag{4.21}$$

$$\boldsymbol{\varepsilon}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k. \tag{4.22}$$

The above transformation can be performed by substituting (4.4) into (4.3) and then using (4.7) and (4.8). As can be seen, the structure of the observer (4.19) is very similar to that of the Kalman filter. The only difference is the term $\boldsymbol{H}_{k+1}\boldsymbol{\varepsilon}_{k+1/k}$, which vanishes when no unknown input is considered.

4.1.2. An extended unknown input observer

As has already been mentioned, the application of the EKF to the state estimation of non-linear deterministic systems has received considerable attention during the last two decades (see (Boutayeb and Aubry 1999) and the references therein). This is mainly because the EKF can directly be applied to a large class of non-linear systems, and its implementation procedure is almost as simple as that for linear systems. Moreover, in the case of deterministic systems, the instrumental matrices $\boldsymbol{R}_k$ and $\boldsymbol{Q}_k$ can be set almost arbitrarily. This opportunity makes it possible to use them to improve the convergence of the observer, which is the main drawback to linearisation-based approaches. This section presents an Extended Unknown Input Observer (EUIO) for a class of non-linear systems which can be modelled by the following equations (Witczak 2001a, Witczak and Korbicz 2001b, Witczak and Korbicz 2001c, Witczak and Korbicz 2002):

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}(\boldsymbol{u}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k) + \boldsymbol{E}_k\boldsymbol{d}_k, \tag{4.23}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}_{k+1}\boldsymbol{x}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}, \tag{4.24}$$

where $\boldsymbol{g}(\boldsymbol{x}_k)$ is assumed to be continuously differentiable with respect to $\boldsymbol{x}_k$. Similarly to the EKF, the observer (4.19) can be extended to the class of non-linear systems (4.23)-(4.24). (The algorithm presented below though can also, with minor modifications, be applied to a more general structure. Such a restriction is caused by the need to employ it for FDI purposes.) This leads to the following structure of the EUIO:

$$\hat{\boldsymbol{x}}_{k+1/k} = \boldsymbol{g}(\hat{\boldsymbol{x}}_k) + \boldsymbol{h}(\boldsymbol{u}_k), \tag{4.25}$$

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1/k} + \boldsymbol{H}_{k+1}\boldsymbol{\varepsilon}_{k+1/k} + \boldsymbol{K}_{1,k+1}\boldsymbol{\varepsilon}_k, \tag{4.26}$$

It should also be pointed out that the matrix $\boldsymbol{A}_k$ used in (4.14) is now defined by:

$$\boldsymbol{A}_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k}. \tag{4.27}$$

4.1.3. Convergence of the EUIO

In this section the Lyapunov approach is employed for the convergence analysis of the EUIO. The approach presented here is similar to that described in (Boutayeb and Aubry 1999), which was used in the case of the EKF-based deterministic observer. The main objective of this section is to show that the convergence of the EUIO strongly depends on the appropriate choice of the instrumental matrices $\boldsymbol{R}_k$ and $\boldsymbol{Q}_k$. Subsequently, the fault-free mode is assumed, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$.

For notational convenience, let us define the *a priori* state estimation error:

$$\boldsymbol{e}_{k+1/k} = \boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1/k}. \tag{4.28}$$

Substituting (4.23)-(4.24) and (4.25)-(4.26) into (4.9), one can obtain the following form of the state estimation error:

$$\boldsymbol{e}_{k+1} = \boldsymbol{e}_{k+1/k} - \boldsymbol{H}_{k+1}\boldsymbol{\varepsilon}_{k+1/k} - \boldsymbol{K}_{1,k+1}\boldsymbol{\varepsilon}_{k}. \tag{4.29}$$

As usual, to perform further derivation, it is necessary to linearize the model around the current state estimate $\hat{\boldsymbol{x}}_k$. This leads to the classical approximation:

$$\boldsymbol{e}_{k+1/k} \approx \boldsymbol{A}_k\boldsymbol{e}_k + \boldsymbol{E}_k\boldsymbol{d}_k. \tag{4.30}$$

In order to avoid the above approximation, the diagonal matrix $\boldsymbol{\alpha}_k = \mathrm{diag}(\alpha_{1,k},\ldots,\alpha_{n,k})$ is introduced, which makes it possible to establish the following exact equality:

$$\boldsymbol{e}_{k+1/k} = \boldsymbol{\alpha}_k\boldsymbol{A}_k\boldsymbol{e}_k + \boldsymbol{E}_k\boldsymbol{d}_k, \tag{4.31}$$

and hence (4.29) can be expressed as:

$$\begin{aligned}
\boldsymbol{e}_{k+1} &= \boldsymbol{e}_{k+1/k} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}\boldsymbol{e}_{k+1/k} - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{e}_k \\
&= [\boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}][\boldsymbol{\alpha}_k\boldsymbol{A}_k\boldsymbol{e}_k + \boldsymbol{E}_k\boldsymbol{d}_k] - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{e}_k \\
&= [\boldsymbol{T}_{k+1}\boldsymbol{\alpha}_k\boldsymbol{A}_k - \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k]\boldsymbol{e}_k.
\end{aligned} \tag{4.32}$$

The main objective of further consideration is to determine conditions under which the sequence $\{V_k\}_{k=1}^{\infty}$, defined by the Lyapunov candidate function:

$$V_{k+1} = \boldsymbol{e}_{k+1}^T\boldsymbol{A}_{1,k+1}^{-T}[\boldsymbol{P}'_{k+1}]^{-1}\boldsymbol{A}_{1,k+1}^{-1}\boldsymbol{e}_{k+1}, \tag{4.33}$$

is a decreasing one. It should be pointed out that the Lyapunov function (4.33) involves a very restrictive assumption regarding an inverse of the matrix $\boldsymbol{A}_{1,k+1}^{-1}$. Indeed, from (4.14) and (4.7), (4.6) it is clear that the matrix $\boldsymbol{A}_{1,k+1}$ is singular when $\boldsymbol{E}_k \neq \boldsymbol{0}$. Thus, the convergence conditions can formally be obtained only when $\boldsymbol{E}_k = \boldsymbol{0}$. This means that the practical solution regarding the choice of the instrumental matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ is to be obtained for the case when $\boldsymbol{E}_k = \boldsymbol{0}$ and generalized to other cases, i.e. $\boldsymbol{E}_k \neq \boldsymbol{0}$.

First, let us define an alternative form of $\boldsymbol{K}_{1,k}$, and the inverse of $\boldsymbol{P}'_{k+1}$. Substituting (4.17) into (4.16) and then comparing it with (4.13), one can obtain that:

$$\boldsymbol{A}_{1,k+1}\boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{A}_{1,k+1}^T = \boldsymbol{K}_{1,k+1}\boldsymbol{C}_k\boldsymbol{P}_k. \tag{4.34}$$

Next, from (4.34), (4.17) and (4.15), we have that the gain matrix is:

$$\boldsymbol{K}_{1,k+1} = \boldsymbol{A}_{1,k+1}\boldsymbol{P}'_{k+1}\boldsymbol{C}_k^T\boldsymbol{R}_k^{-1}. \tag{4.35}$$

Similarly, from (4.34) and (4.17), we have that the inverse of $\boldsymbol{P}'_{k+1}$ is:

$$[\boldsymbol{P}'_{k+1}]^{-1} = \boldsymbol{P}_k^{-1} + \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k. \tag{4.36}$$

Substituting (4.32), and then (4.35) and (4.36) into (4.33), the Lyapunov candidate function is:

$$\begin{aligned} V_{k+1} =& \boldsymbol{e}_k^T [\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k \\ &+ \boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k \\ &- \boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k - \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k \\ &+ \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \boldsymbol{P}'_{k+1} \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k] \boldsymbol{e}_k. \end{aligned} \tag{4.37}$$

Let:

$$\boldsymbol{G} = \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k, \quad \boldsymbol{L} = \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k, \tag{4.38}$$

then:

$$\boldsymbol{G}^T \boldsymbol{L} \boldsymbol{G} - \boldsymbol{G}^T \boldsymbol{L} - \boldsymbol{L} \boldsymbol{G} = \left[\boldsymbol{G}^T - \boldsymbol{I}\right] \boldsymbol{L} \left[\boldsymbol{G} - \boldsymbol{I}\right] - \boldsymbol{L}. \tag{4.39}$$

Using (4.39) and (4.15), the expression (4.37) becomes:

$$\begin{aligned} V_{k+1} =& \boldsymbol{e}_k^T \Big[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k \\ &+ \left[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} - \boldsymbol{I}\right] \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \left[\boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k - \boldsymbol{I}\right] \\ &- \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \left[\boldsymbol{I} - \boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1}\right]\Big] \boldsymbol{e}_k. \end{aligned} \tag{4.40}$$

Using the identity in (4.40):

$$\boldsymbol{I} = \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right] \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1}, \tag{4.41}$$

the Lyapunov candidate function can be written as:

$$\begin{aligned} V_{k+1} =& \boldsymbol{e}_k^T \Big[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k \\ &+ \left[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} - \boldsymbol{I}\right] \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \left[\boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k - \boldsymbol{I}\right] \\ &- \boldsymbol{C}_k^T \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1} \boldsymbol{C}_k\Big] \boldsymbol{e}_k. \end{aligned} \tag{4.42}$$

The sequence $\{V_k\}_{k=1}^{\infty}$ is a decreasing one when there exists a scalar ζ, $0 < \zeta < 1$, such that:

$$V_{k+1} - (1 - \zeta) V_k \leq 0. \tag{4.43}$$

Using (4.42), the inequality (4.43) becomes:

$$V_{k+1} - (1-\zeta)V_k = \boldsymbol{e}_k^T \boldsymbol{X}_k \boldsymbol{e}_k + \boldsymbol{e}_k^T \boldsymbol{Y}_k \boldsymbol{e}_k \leq 0, \tag{4.44}$$

where:

$$\boldsymbol{X}_k = \boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k - (1-\zeta) \boldsymbol{A}_{1,k}^{-T} [\boldsymbol{P}_k']^{-1} \boldsymbol{A}_{1,k}^{-1}, \tag{4.45}$$

$$\begin{aligned}\boldsymbol{Y}_k = &\left[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} - \boldsymbol{I}\right] \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \left[\boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k - \boldsymbol{I}\right] \\ &- \boldsymbol{C}_k^T \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1} \boldsymbol{C}_k.\end{aligned} \tag{4.46}$$

In order to satisfy (4.44), the matrices $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$ should be semi-negative defined. This is equivalent to:

$$\bar{\sigma}\left(\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k\right) \leq \underline{\sigma}\left((1-\zeta) \boldsymbol{A}_{1,k}^{-T} [\boldsymbol{P}_k']^{-1} \boldsymbol{A}_{1,k}^{-1}\right), \tag{4.47}$$

and:

$$\begin{aligned}&\bar{\sigma}\left(\left[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} - \boldsymbol{I}\right] \boldsymbol{C}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{C}_k \left[\boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k - \boldsymbol{I}\right]\right) \\ &\leq \underline{\sigma}\left(\boldsymbol{C}_k^T \left[\boldsymbol{C}_k \boldsymbol{P}_k \boldsymbol{C}_k^T + \boldsymbol{R}_k\right]^{-1} \boldsymbol{C}_k\right),\end{aligned} \tag{4.48}$$

where $\bar{\sigma}(\cdot)$ and $\underline{\sigma}(\cdot)$ denote the maximum and minimum singular values, respectively. The inequalities (4.47) and (4.48) determine the bounds of the diagonal matrix $\boldsymbol{\alpha}_k$, for which the condition (4.44) is satisfied. The objective of further consideration is to obtain a more convenient form of the above bounds.

Using the fact that:

$$\begin{aligned}\bar{\sigma}\left(\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} \boldsymbol{P}_k^{-1} \boldsymbol{A}_k^{-1} \boldsymbol{\alpha}_k \boldsymbol{A}_k\right) &\leq \bar{\sigma}^2(\boldsymbol{A}_k)\, \bar{\sigma}^2\left(\boldsymbol{A}_k^{-1}\right) \bar{\sigma}^2(\boldsymbol{\alpha}_k)\, \bar{\sigma}\left(\boldsymbol{P}_k^{-1}\right) \\ &= \frac{\bar{\sigma}^2(\boldsymbol{A}_k)}{\underline{\sigma}^2(\boldsymbol{A}_k)} \frac{\bar{\sigma}^2(\boldsymbol{\alpha}_k)}{\underline{\sigma}(\boldsymbol{P}_k)},\end{aligned} \tag{4.49}$$

expression (4.47) becomes:

$$\bar{\sigma}(\boldsymbol{\alpha}_k) \leq \gamma_1 = \frac{\underline{\sigma}(\boldsymbol{A}_k)}{\bar{\sigma}(\boldsymbol{A}_k)} \left(\frac{(1-\zeta)\underline{\sigma}(\boldsymbol{P}_k)}{\bar{\sigma}\left(\boldsymbol{A}_{1,k} \boldsymbol{P}_k' \boldsymbol{A}_{1,k}^T\right)}\right)^{\frac{1}{2}}. \tag{4.50}$$

Similarly, using:

$$\begin{aligned}\bar{\sigma}\left(\left[\boldsymbol{A}_k^T \boldsymbol{\alpha}_k \boldsymbol{A}_k^{-T} - \boldsymbol{I}\right]\right) &= \bar{\sigma}\left(\boldsymbol{A}_k^T \left[\boldsymbol{\alpha}_k - \boldsymbol{I}\right] \boldsymbol{A}_k^{-T}\right) \\ &\leq \frac{\bar{\sigma}(\boldsymbol{A}_k)}{\underline{\sigma}(\boldsymbol{A}_k)} \bar{\sigma}(\boldsymbol{\alpha}_k - \boldsymbol{I}),\end{aligned} \tag{4.51}$$

and then:

$$\begin{aligned}
&\bar{\sigma}\left(\left[\boldsymbol{A}_k^T\boldsymbol{\alpha}_k\boldsymbol{A}_k^{-T}-\boldsymbol{I}\right]\boldsymbol{C}_k^T\boldsymbol{R}_k^{-1}\boldsymbol{C}_k\left[\boldsymbol{A}_k^{-1}\boldsymbol{\alpha}_k\boldsymbol{A}_k-\boldsymbol{I}\right]\right)\\
&\le\frac{\bar{\sigma}^2\left(\boldsymbol{A}_k\right)}{\underline{\sigma}^2\left(\boldsymbol{A}_k\right)}\frac{\bar{\sigma}\left(\boldsymbol{C}_k^T\right)\bar{\sigma}\left(\boldsymbol{C}_k\right)}{\underline{\sigma}\left(\boldsymbol{R}_k\right)}\bar{\sigma}^2\left(\boldsymbol{\alpha}_k-\boldsymbol{I}\right)
\end{aligned}\tag{4.52}$$

and:

$$\underline{\sigma}\left(\boldsymbol{C}_k^T\left[\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T+\boldsymbol{R}_k\right]^{-1}\boldsymbol{C}_k\right)\ge\frac{\underline{\sigma}\left(\boldsymbol{C}_k^T\right)\underline{\sigma}\left(\boldsymbol{C}_k\right)}{\bar{\sigma}\left(\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T+\boldsymbol{R}_k\right)},\tag{4.53}$$

the expression (4.48) becomes:

$$\begin{aligned}
&\bar{\sigma}\left(\boldsymbol{\alpha}_k-\boldsymbol{I}\right)\le\gamma_2\\
&=\frac{\underline{\sigma}\left(\boldsymbol{A}_k\right)}{\bar{\sigma}\left(\boldsymbol{A}_k\right)}\left(\frac{\underline{\sigma}\left(\boldsymbol{C}_k^T\right)\underline{\sigma}\left(\boldsymbol{C}_k\right)}{\bar{\sigma}\left(\boldsymbol{C}_k^T\right)\bar{\sigma}\left(\boldsymbol{C}_k\right)}\frac{\underline{\sigma}\left(\boldsymbol{R}_k\right)}{\bar{\sigma}\left(\boldsymbol{C}_k\boldsymbol{P}_k\boldsymbol{C}_k^T+\boldsymbol{R}_k\right)}\right)^{\frac{1}{2}}.
\end{aligned}\tag{4.54}$$

Bearing in mind that $\boldsymbol{\alpha}_k$ is a diagonal matrix, the above inequalities can be expressed as:

$$\max_{i=1,\ldots,n}|\alpha_{i,k}|\le\gamma_1\quad\text{and}\quad\max_{i=1,\ldots,n}|\alpha_{i,k}-1|\le\gamma_2.\tag{4.55}$$

Since:

$$\boldsymbol{P}_k=\boldsymbol{A}_{1,k}\boldsymbol{P}_k'\boldsymbol{A}_{1,k}^T+\boldsymbol{T}_k\boldsymbol{Q}_{k-1}\boldsymbol{T}_k^T+\boldsymbol{H}_k\boldsymbol{R}_k\boldsymbol{H}_k^T,\tag{4.56}$$

it is clear that an appropriate selection of the instrumental matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ may enlarge the bounds γ_1 and γ_2 and, consequently, the domain of attraction. Indeed, if the conditions (4.55) are satisfied, then $\hat{\boldsymbol{x}}_k$ converges to $\boldsymbol{x}_k$.

4.1.4. Increasing the convergence rate via genetic programming

Unfortunately, an analytical derivation of the matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ seems to be an extremely difficult problem. However, it is possible to set the above matrices as follows: $\boldsymbol{Q}_{k-1}=\beta_1\boldsymbol{I}$, $\boldsymbol{R}_k=\beta_1\boldsymbol{I}$, with β_1 and β_1 large enough. On the other hand, it is well known that the convergence rate of such an EKF-like approach can be increased by an appropriate selection of the covariance matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$, i.e. the more accurate (near "true" values) the covariance matrices, the better the convergence rate. This means that in the deterministic case ($\boldsymbol{w}_k=\boldsymbol{0}$ and $\boldsymbol{v}_k=\boldsymbol{0}$), both matrices should be zero ones. Unfortunately, such an approach usually leads to the divergence of the observer as well as other computational problems. To tackle this, a compromise between the convergence and the convergence rate should be established. This can easily be done by setting the instrumental matrices as:

$$\boldsymbol{Q}_{k-1}=\beta_1\boldsymbol{\varepsilon}_{k-1}^T\boldsymbol{\varepsilon}_{k-1}\boldsymbol{I}+\delta_1\boldsymbol{I},\quad\boldsymbol{R}_k=\beta_2\boldsymbol{\varepsilon}_k^T\boldsymbol{\varepsilon}_k\boldsymbol{I}+\delta_2\boldsymbol{I},\tag{4.57}$$

with β_1, β_2 large enough, and δ_1, δ_2 small enough. Although this approach is very simple, it is possible to increase the convergence rate further. Indeed, the instrumental matrices can be set as follows:

$$\boldsymbol{Q}_{k-1} = q^2(\boldsymbol{\varepsilon}_{k-1})\boldsymbol{I} + \delta_1\boldsymbol{I}, \quad \boldsymbol{R}_k = r^2(\boldsymbol{\varepsilon}_k)\boldsymbol{I} + \delta_2\boldsymbol{I}, \tag{4.58}$$

where $q(\boldsymbol{\varepsilon}_{k-1})$ and $r(\boldsymbol{\varepsilon}_k)$ are non-linear functions of the output error $\boldsymbol{\varepsilon}_k$ (the squares are used to ensure the positive definiteness of $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$). Thus, the problem reduces to identifying the above functions. To tackle it, genetic programming can be employed. The unknown functions $q(\boldsymbol{\varepsilon}_{k-1})$ and $r(\boldsymbol{\varepsilon}_k)$ can be expressed as trees, as shown in Fig. 4.1. Thus, in the case of $q(\cdot)$ and $r(\cdot)$, the terminal sets are $\mathbb{T} = \{\boldsymbol{\varepsilon}_{k-1}\}$ and $\mathbb{T} = \{\boldsymbol{\varepsilon}_k\}$, respectively. In both cases, the function set can be defined as $\mathbb{F} = \{+, *, /, \xi_1(\cdot), \ldots, \xi_l(\cdot)\}$, where $\xi_k(\cdot)$ is a non-linear univariate function and, consequently, the number of populations is $n_p = 2$. Since the terminal and function sets are given, the approach described in Chapter 3 can easily be adopted for the identification purpose of $q(\cdot)$ and $r(\cdot)$. First, let us define the identification criterion constituting a necessary ingredient of the $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ selection process.

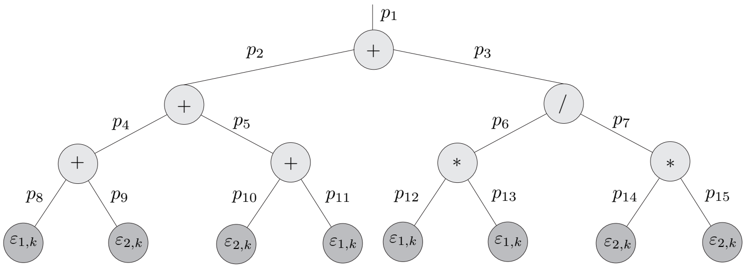

Fig. 4.1 . An exemplary tree representing $r(\boldsymbol{\varepsilon}_k)$.

Since the instrumental matrices should be chosen so as to satisfy (**??**), the selection of $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ can be performed according to:

$$\left(\boldsymbol{Q}_{k-1}, \boldsymbol{R}_k\right) = \arg \max_{q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)} j_{\text{obs},1}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)), \tag{4.59}$$

where:

$$j_{\text{obs},1}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)) = \sum_{k=0}^{n_t-1} \text{trace}\boldsymbol{P}_k. \tag{4.60}$$

On the other hand, owing to the FDI requirements, it is clear that the output error should be near zero in the fault-free mode. In this case, one can define another identification criterion:

$$\left(\boldsymbol{Q}_{k-1}, \boldsymbol{R}_k\right) = \arg \min_{q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)} j_{\text{obs},2}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)), \tag{4.61}$$

where:

$$j_{\mathrm{obs},2}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)) = \sum_{k=0}^{n_t-1} \boldsymbol{\varepsilon}_k^T \boldsymbol{\varepsilon}_k. \tag{4.62}$$

Therefore, in order to join (4.59) and (4.61), the following identification criterion is employed:

$$(\boldsymbol{Q}_{k-1}, \boldsymbol{R}_k) = \arg \min_{q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)} j_{\mathrm{obs},3}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)), \tag{4.63}$$

where:

$$j_{\mathrm{obs},3}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k)) = \frac{j_{\mathrm{obs},2}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k))}{j_{\mathrm{obs},1}(q(\boldsymbol{\varepsilon}_{k-1}), r(\boldsymbol{\varepsilon}_k))}. \tag{4.64}$$

Since the identification criterion is established, it is straightforward to use the GP algorithm detailed in Chapter 3.

4.1.5. EUIO-based sensor FDI

In order to design a fault detection and isolation scheme for a real industrial system, it is insufficient to only design an observer and check that the norm of the residual (or the output error) has exceeded a prespecified maximum admissible value T (threshold). Indeed, this condition is necessary only for fault detection. For the purpose of fault isolation, it will be desirable to design a bank of observers where each of the observers is sensitive to one fault while insensitive to others. Unfortunately, such a requirement is rather difficult to attain. A more realistic approach is to design a bank of observers where each of the observers is sensitive to all faults but one. In this section, the faults are divided into two categories, i.e. the sensor and actuator faults. First, the sensor fault detection scheme is described. In this case, the actuators are assumed to be fault-free, and hence, for each of the observers, the system can be characterized as follows:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}(\boldsymbol{u}_k) + \boldsymbol{E}_k \boldsymbol{d}_k, \tag{4.65}$$

$$\boldsymbol{y}_{k+1}^j = \boldsymbol{C}_k^j \boldsymbol{x}_{k+1} + \boldsymbol{f}_{k+1}^j, \qquad j = 1, \ldots, m, \tag{4.66}$$

$$y_{j,k+1} = c_{j,k} \boldsymbol{x}_{k+1} + f_{j,k+1}, \tag{4.67}$$

where, similarly to the way it was done in (Chen and Patton 1999), $c_{j,k} \in \mathbb{R}^n$ is the j-th row of the matrix $\boldsymbol{C}_k$, $\boldsymbol{C}_k^j \in \mathbb{R}^{m-1\times n}$ is obtained from the matrix $\boldsymbol{C}_k$ by deleting the j-th row, $c_{j,k}$, $y_{j,k+1}$ is the j-th element of $\boldsymbol{y}_{k+1}$, and $\boldsymbol{y}_{k+1}^j \in \mathbb{R}^{m-1}$ is obtained from the vector $\boldsymbol{y}_{k+1}$ by deleting the j-th component $y_{j,k+1}$. Thus, the problem reduces to designing m EUIOs (Fig. 4.2), where each of the observers is constructed using all input data sets and all output data sets but one $\{\boldsymbol{y}_k, \boldsymbol{u}_k\}_{k=0}^{n_t-1}$. Similarly, after the design procedure, when the observers are employed on-line, each residual generator (observer) is driven by all inputs and all outputs but one.

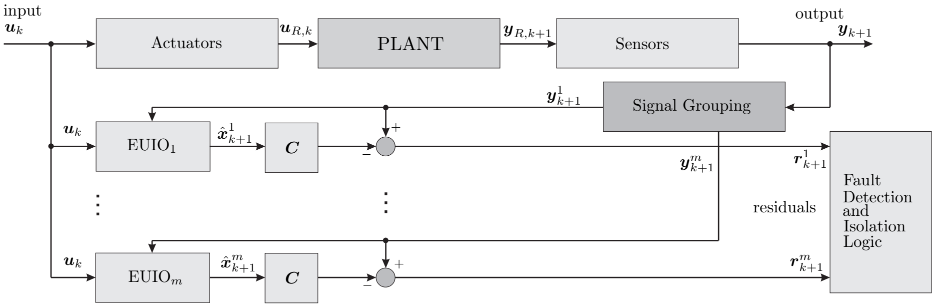

Fig. 4.2 . Sensor fault detection and isolation scheme.

When all sensors but the j-th one are fault-free, and all actuators are fault-free, then the residual $\boldsymbol{r}_k = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k$ will satisfy the following isolation logic:

$$\begin{cases} \|r_{j,k}\| < \varepsilon_j^H \\ \|r_{l,k}\| \geq \varepsilon_l^H, & l = 1, \dots, j-1, j+1, \dots, m. \end{cases} \tag{4.68}$$

where ε_i^H denotes a prespecified threshold.

4.1.6. EUIO-based actuator FDI

Similarly to the case of the sensor fault isolation scheme, in order to design the actuator fault isolation scheme, it is necessary to assume that all sensors are fault-free. Moreover, the term $\boldsymbol{h}(\boldsymbol{u}_k)$ in (4.23)-(4.24) should have the following structure:

$$\boldsymbol{h}(\boldsymbol{u}_k) = \boldsymbol{B}_k \boldsymbol{u}_k. \tag{4.69}$$

In this case, for each of the observers, the system can be characterized as follows:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}^i(\boldsymbol{u}_k^i + \boldsymbol{f}_k^i) + \boldsymbol{h}_i(u_{i,k} + f_{i,k}) + \boldsymbol{E}_k \boldsymbol{d}_k, \\ &= \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}^i(\boldsymbol{u}_k + \boldsymbol{f}_k^i) + \boldsymbol{E}_k^i \boldsymbol{d}_k^i, & (4.70) \\ \boldsymbol{y}_{k+1} &= \boldsymbol{C}_k \boldsymbol{x}_{k+1}, \qquad i = 1, \dots, r, & (4.71) \\ & & (4.72) \end{aligned}$$

where:

$$\boldsymbol{h}^i(\boldsymbol{u}_k^i + \boldsymbol{f}_k^i) = [\boldsymbol{b}_k^1, \dots \boldsymbol{b}_k^{i-1}, \boldsymbol{b}_k^{i+1}, \dots, \boldsymbol{b}_k^r](\boldsymbol{u}_k^i + \boldsymbol{f}_k^i) \tag{4.73}$$

$$\boldsymbol{h}_i(u_{i,k} + f_{i,k}) = \boldsymbol{b}_k^i(u_{i,k} + f_{i,k}) \tag{4.74}$$

$$\boldsymbol{E}_k^i = [\boldsymbol{E}_k \ \boldsymbol{b}_k^i], \quad \boldsymbol{d}_k^i = \begin{bmatrix} \boldsymbol{d}_k \\ u_{i,k} + f_{i,k} \end{bmatrix}. \tag{4.75}$$

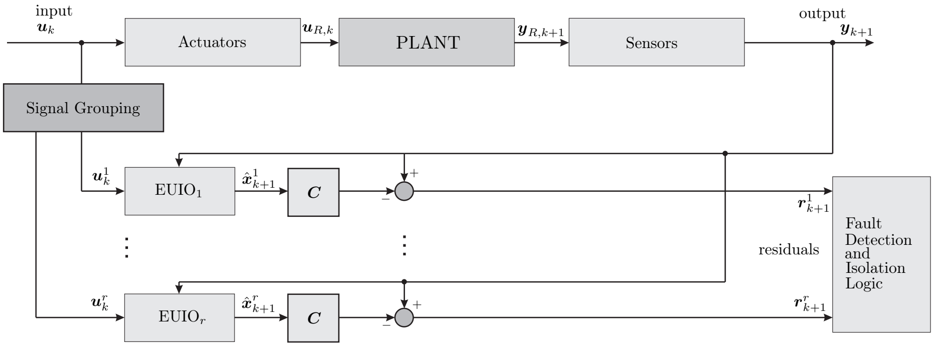

Fig. 4.3 . Actuator fault detection and isolation scheme.

Thus, the problem reduces to designing r EUIOs (Fig. 4.3). When all actuators but the i-th one are fault-free, and all sensors are fault-free, then the residual $\boldsymbol{r} = \boldsymbol{y}_k - \hat{\boldsymbol{y}}_k$ will satisfy the following isolation logic:

$$\begin{cases} \|r_{i,k}\| < \varepsilon_i^H \\ \|r_{l,k}\| \geq \varepsilon_l^H, \qquad\qquad l = 1, \ldots, i-1, i+1, \ldots, r. \end{cases} \tag{4.76}$$

4.1.7. Experimental results

The purpose of this section is to show the reliability and effectiveness of the observer-based fault diagnosis scheme described in the present chapter. The numerical example considered here is a fifth-order two-phase non-linear model of an induction motor, which has already been the subject of a large number of various control design applications (see (Boutayeb and Aubry 1999) and the references therein).

The complete discrete time model in a stator-fixed (a,b) reference frame is:

$$x_{1,k+1} = x_{1,k} + h(-\gamma x_{1k} + \frac{K}{T_r} x_{3k} + K p x_{5k} x_{4k} + \frac{1}{\sigma L_s} u_{1k}), \tag{4.77}$$

$$x_{2,k+1} = x_{2,k} + h(-\gamma x_{2k} - K p x_{5k} x_{3k} + \frac{K}{T_r} x_{4k} + \frac{1}{\sigma L_s} u_{2k}), \tag{4.78}$$

$$x_{3,k+1} = x_{3,k} + h(\frac{M}{T_r} x_{1k} - \frac{1}{T_r} x_{3k} - p x_{5k} x_{4k}), \tag{4.79}$$

$$x_{4,k+1} = x_{4,k} + h(\frac{M}{T_r} x_{2k} + p x_{5k} x_{3k} - \frac{1}{T_r} x_{4k}), \tag{4.80}$$

$$x_{5,k+1} = x_{5,k} + h(\frac{pM}{JL_r}(x_{3k} x_{2k} - x_{4k} x_{1k}) - \frac{T_L}{J}), \tag{4.81}$$

$$y_{1,k+1} = x_{1,k+1}, \; y_{2,k+1} = x_{2,k+1}. \tag{4.82}$$

where $\boldsymbol{x}_k = (x_{1,k}, \ldots, x_{n,k}) = (i_{\mathrm{sak}}, i_{\mathrm{sbk}}, \psi_{\mathrm{rak}}, \psi_{\mathrm{rbk}}, \omega_k)$ represents the currents, the rotor fluxes, and the angular speed, respectively, while $\boldsymbol{u}_k = (u_{\mathrm{sak}}, u_{\mathrm{sbk}})$ is the stator voltage control vector, p is the number of the pairs of poles, and T_L is the load torque. The rotor time constant T_r and the remaining parameters are defined as:

$$T_r = \frac{L_r}{R_r},\ \sigma = 1 - \frac{M^2}{L_s L_r},\ K = \frac{M}{\sigma L_s L_r^2},\ \gamma = \frac{R_s}{\sigma L_s} + \frac{R_r M^2}{\sigma L_s L_r^2}, \tag{4.83}$$

where R_s, R_r and L_s, L_r are stator and rotor per-phase resistances and inductances, respectively, and J is the rotor moment inertia.
The numerical values of the above parameters are as follows: $R_s = 0.18\ \Omega$, $R_r = 0.15\ \Omega$, $M = 0.068$ H, $L_s = 0.0699$ H, $L_r = 0.0699$ H, $J = 0.0586$ kgm^2, $T_L = 10$ Nm, $p = 1$, and $h = 0.1$ ms. The initial conditions for the observer and the system are $\hat{\boldsymbol{x}}_k = (200, 200, 50, 50, 300)$ and $\boldsymbol{x}_k = \mathbf{0}$. The unknown input distribution matrix is:

$$\boldsymbol{E}_k = \begin{bmatrix} 0.1 & 0 & 1 & 0 & 0 \\ 0 & 0.1 & 0 & 1 & 0 \end{bmatrix}^T, \tag{4.84}$$

and hence, according to (2.47), the matrix $\boldsymbol{H}_k$ is:

$$\boldsymbol{H}_k = \begin{bmatrix} 1 & 0 & 10 & 0 & 0 \\ 0 & 1 & 0 & 10 & 0 \end{bmatrix}^T. \tag{4.85}$$

The input signals are:

$$u_{1,k} = 300\cos(0.03k), \quad u_{2,k} = 300\sin(0.03k). \tag{4.86}$$

The unknown input is defined as:

$$d_{1,k} = 0.09\sin(0.5\pi k)\cos(0.3\pi k), \quad d_{2,k} = 0.09\sin(0.01k), \tag{4.87}$$

and $\boldsymbol{P}_0 = 10^3\boldsymbol{I}$.

The following three cases concerning $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ were considered:

Case 1: Classical approach (constant values), i.e. $\boldsymbol{Q}_{k-1} = 0.1$, $\boldsymbol{R}_k = 0.1$,

Case 2: Selection according to (4.57), i.e.

$$\boldsymbol{Q}_{k-1} = 10^3\boldsymbol{\varepsilon}_{k-1}^T\boldsymbol{\varepsilon}_{k-1}\boldsymbol{I} + 0.01\boldsymbol{I},\ \boldsymbol{R}_k = 10\boldsymbol{\varepsilon}_k^T\boldsymbol{\varepsilon}_k\boldsymbol{I} + 0.01\boldsymbol{I}, \tag{4.88}$$

Case 3: GP-based approach presented in Section 4.1.4.

It should be pointed out that in all the cases the unknown input-free mode (i.e., $\boldsymbol{d}_k = \mathbf{0}$) is considered. This is because the main purpose of this example is to show the importance of an appropriate selection of the instrumental matrices but not the abilities of disturbance de-coupling.

In order to obtain the matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ using the GP-based approach (Case 3), a set of $n_t = 300$ input-output measurements was generated according to (4.77)-(4.82), and then the approach from Section 4.1.4 was applied. As a result, the following form of the instrumental matrices was obtained:

$$\boldsymbol{Q}_k = \left(10^2\varepsilon_{1,k}^2\varepsilon_{2,k}^2 + 1012\varepsilon_{1,k} + 103.45\varepsilon_{1,k} + 0.01\right)^2 \boldsymbol{I}, \tag{4.89}$$

$$\boldsymbol{R}_k = \left(112\varepsilon_{1,k}^2 + 0.1\varepsilon_{1,k}\varepsilon_{2,k} + 0.12\right)^2 \boldsymbol{I}. \tag{4.90}$$

The parameters used in the GP algorithm presented in Section 4.1.4 were $n_m = 200$, $n_d = 10$, $n_s = 10$, $\mathbb{F} = \{+, *, /\}$. It should also be pointed out that the above matrices (4.89)-(4.90) are formed by simple polynomials. This, however, may not be the case for other applications.

Simulation results (for all the cases) are shown in Fig. 4.4. The numerical values of the optimisation index (4.64) are as follows: Case 1 $j_{\text{obs},3} = 1.49 * 10^5$, Case 2 $j_{\text{obs},3} = 1.55$, Case 3 $j_{\text{obs},3} = 1.2 * 10^{-16}$. Both the above results and the plots shown in Fig. 4.4 confirm the relevance of the appropriate selection of the instrumental matrices. Indeed, as can be seen, the proposed approach is superior to the classical technique of selecting the instrumental matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$.

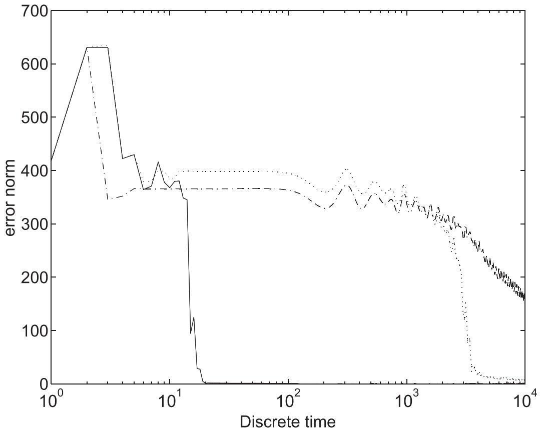

Fig. 4.4. The state estimation error norm $\|\boldsymbol{e}_k\|_2$ for Case 1 (dash-doted line), Case 2 (dotted line) and Case 3 (solid line).

The objective of presenting the next example is to show the abilities of disturbance decoupling. In this case, the unknown input $\boldsymbol{d}_k$ acts on the system according to (4.87). All the simulations were performed with the instrumental matrices set according to (4.89)-(4.90). Figure 4.5 shows the residual signal for an observer without unknown input decoupling, i.e. $\boldsymbol{H}_k = \boldsymbol{0}$. From this figure, it is clear that the unknown input influences the residual signal and hence it may cause an unreliable fault detection (and consequently fault isolation). On the contrary, Fig. 4.6 shows the residual signal for an observer with unknown input decoupling, i.e. $\boldsymbol{H}_k$

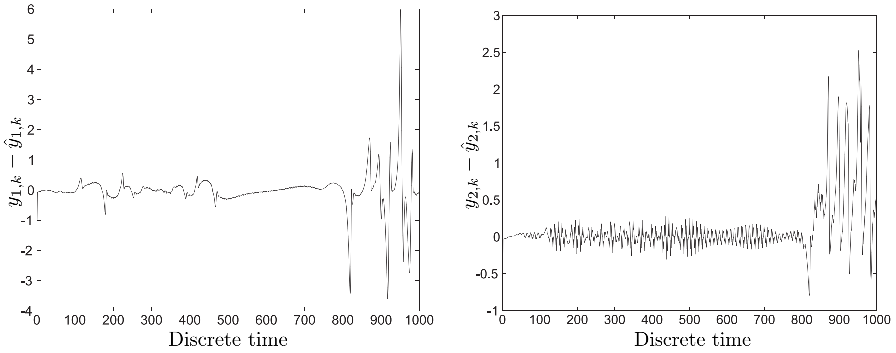

Fig. 4.5 . Residuals for an observer without unknown input decoupling.

was set according to (4.85). In this case, the residual is almost zero. This confirms the importance of disturbance decoupling.

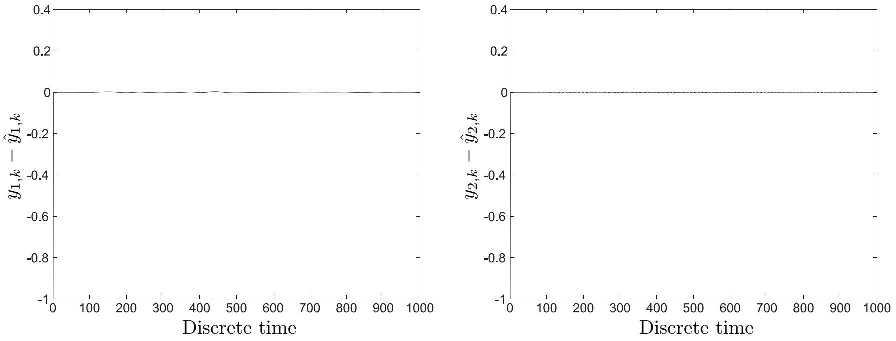

Fig. 4.6 . Residuals for an observer with unknown input decoupling.

The objective of presenting the next example is to show the effectiveness of the proposed observer as a residual generator in the presence of an unknown input. For that purpose, the following fault scenarios were considered:

Case 1: An abrupt fault of $y_{1,k}$ sensor:

$$f_{s,k} = \begin{cases} 0, & k < 140, \\ -0.1y_{1,k}, & \text{otherwise,} \end{cases} \tag{4.91}$$

Case 2: An abrupt fault of $u_{1,k}$ actuator:

$$f_{a,k} = \begin{cases} 0, & k < 140, \\ -0.2u_{1,k}, & \text{otherwise.} \end{cases} \tag{4.92}$$

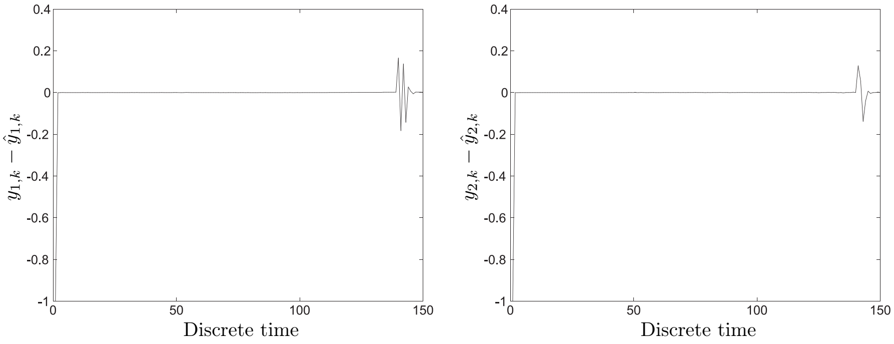

Fig. 4.7. Residuals for a sensor fault.

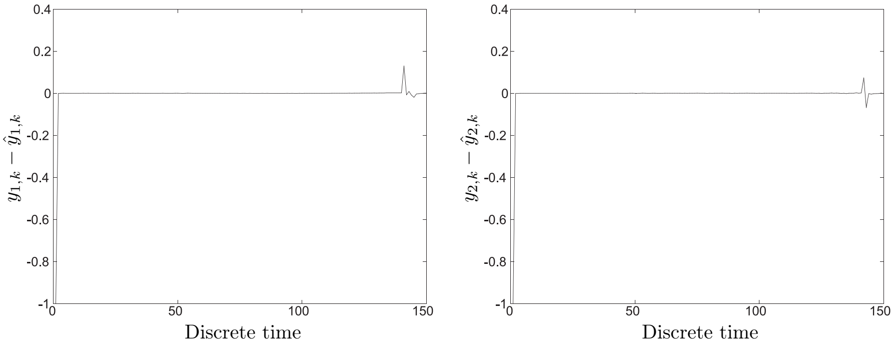

Fig. 4.8. Residuals for an actuator fault.

From Figs. 4.7 and 4.8 it can be observed that the residual signal is sensitive to the faults under consideration. This, together with unknown input decoupling, implies that the process of fault detection becomes a relatively easy task.

4.1.7.1. Sensor FDI with EUIO

In this section the sensor fault diagnosis scheme proposed in Section 4.1.5 is implemented and tested against simulated faults. Based on the above approach, the matrices $\boldsymbol{C}_k^1$ and $\boldsymbol{C}_k^2$ for $m = 2$ observers were defined as:

$$\boldsymbol{C}_k^1 = [\ 1 \quad 0 \quad 0 \quad 0 \quad 0\ ], \quad \boldsymbol{C}_k^2 = [\ 0 \quad 1 \quad 0 \quad 0 \quad 0\ ], \tag{4.93}$$

and this time the initial condition for both observers was selected as $\hat{\boldsymbol{x}}_0 = \mathbf{1}$. The matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ for the observers were obtained by simply modifying the equations (4.89)-(4.90), i.e.

$$\boldsymbol{Q}_{k-1} = \left(10^2\varepsilon_{k-1}^2 + 1112\varepsilon_{k-1} + 0.01\right)^2 \boldsymbol{I}, \tag{4.94}$$

$$\boldsymbol{R}_k = \left(112\varepsilon_k^2 + 0.1\varepsilon_{1,k} + 0.12\right)^2 \boldsymbol{I}. \tag{4.95}$$

However, it should be pointed out that, although such a simple reduction works quite well in the proposed example, there may be cases for which it is necessary to design the instrumental matrices for each of the observers separately. The fault signals were simulated according to the formulaes

$$f_{1,k} = \begin{cases} -100, & k = 100, \dots, 150, \\ 0, & \text{otherwise}, \end{cases} \tag{4.96}$$

$$f_{2,k} = \begin{cases} 10, & k = 200, \dots, 250, \\ 0, & \text{otherwise}. \end{cases} \tag{4.97}$$

Since $\boldsymbol{d}_k \in \mathbb{R}^q$, $q \leq m = 1$, the unknown input distribution matrix takes the following form: $\boldsymbol{E}_k = [e_{1,k}, e_{2,k}]^T$; then the matrix $\boldsymbol{H}_k$ obtained with (4.10) contributes to the fact that $[\boldsymbol{I} - \boldsymbol{C}_k\boldsymbol{H}_k] = \boldsymbol{0}$ and $[\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{H}_{k+1}] = \boldsymbol{0}$. This leads to the following form of the state estimation error:

$$\boldsymbol{e}_{k+1} = -\boldsymbol{K}_{k+1}\boldsymbol{C}_k\boldsymbol{e}_k - \boldsymbol{K}_{k+1}\boldsymbol{L}_{2,k}\boldsymbol{f}_k - \boldsymbol{H}_{k+1}\boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}, \tag{4.98}$$

and, consequently, the residual is:

$$\boldsymbol{r}_{k+1} = -\boldsymbol{C}_{k+1}\boldsymbol{K}_{k+1}\boldsymbol{r}_k. \tag{4.99}$$

This is the reason why UIOs cannot be applied to MISO and other systems for which $[\boldsymbol{I} - \boldsymbol{C}_k\boldsymbol{H}_k] = \boldsymbol{0}$ and $[\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{H}_{k+1}] = \boldsymbol{0}$. If the effect of an unknown input is not considered though, i.e. $\boldsymbol{d}_k = \boldsymbol{0}$, then UIOs can successfully be employed. This is the case for the present example.

The simulation results are shown in Fig. 4.9, from which it can be seen that the residual signal is almost zero in the fault-free case and increases significantly when a fault occurs, thus making the process of fault detection a relatively easy task. Moreover, it should be pointed out that each of the observers is sensitive to the faults of one sensor only, while remaining insensitive to the other sensors' faults. This possibility facilities the process of fault isolation. Indeed, as can be seen in Fig. 4.9, the sensor fault isolation problem is relatively easy to solve.

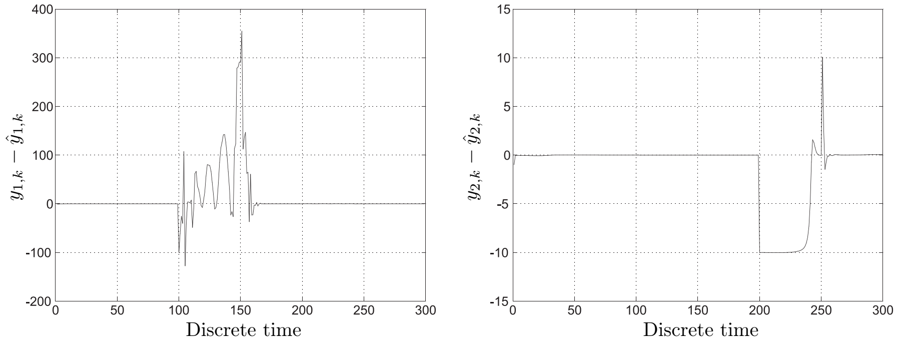

Fig. 4.9 . The residual signals for the Observer 1 (left) and 2 (right).

4.2. An unknown input observer for non-linear stochastic systems

4.2.1. Problem statement

Let us consider the following discrete-time linear system:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{w}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k, \tag{4.100}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{v}_k + \boldsymbol{L}_{2,k}\boldsymbol{f}_k. \tag{4.101}$$

As has already been mentioned, the robustness to model uncertainty and other factors which may lead to an unreliable fault detection is of paramount importance. In the case of the UIO, the robustness problem is tackled by introducing the concept of an unknown input $\boldsymbol{d}_k$, and hence the term $\boldsymbol{E}_k\boldsymbol{d}_k$ may represent various kinds of modelling uncertainty. The remaining factors can be modelled by $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$. Indeed, it is only necessary to know the bounds of $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$, which can be defined by the following sets:

$$\mathbb{W}_k = \bigcap_i \left\{\boldsymbol{w}_k : -b_i \leq w_k^i \leq b_i\right\} \tag{4.102}$$

and:

$$\mathbb{V}_k = \bigcap_i \left\{\boldsymbol{v}_k : -r_i \leq v_k^i \leq r_i\right\}. \tag{4.103}$$

In order to use the bounded-error algorithm described in (Maksarow and Norton 1996a) for the state estimation problem of the system (4.100)-(4.101), it is necessary to introduce some modifications concerning the unknown input. In the existing approaches the unknown input is treated in two different ways. The first one (Chen and Patton 1999) consists in introducing an additional matrix into the state estimation equation, which is then used for decoupling the effect of the unknown input on the state estimation error (and consequently the residual signal). In the second one (Keller and Darouach 1999), the system (4.100)-(4.101) is suitably transformed into a system without an unknown input. In the case of the algorithm (Maksarow and Norton 1996a) both the approaches can be applied.

Owing to the fact that the approach of (Chen and Patton 1999) was presented in the previous section, by simply mimicking this technique the unknown input effect can be decoupled by changing the state estimation equation as:

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_{k+1/k} + \boldsymbol{H}_{k+1}(\boldsymbol{y}_{k+1} - \boldsymbol{C}_{k+1}\hat{\boldsymbol{x}}_{k+1/k}), \tag{4.104}$$

where $\boldsymbol{H}_{k+1}$ is set according to (4.10).

The second approach can be realised as follows (the fault-free mode is assumed, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$). Let us assume that $\operatorname{rank}(\boldsymbol{C}_k\boldsymbol{E}_k) = q$ and:

$$\boldsymbol{T}_k = \boldsymbol{\beta}_k(\boldsymbol{I} - \boldsymbol{H}_k^{+}\boldsymbol{H}_k), \tag{4.105}$$

where $\boldsymbol{H}_k = (\boldsymbol{C}_k\boldsymbol{E}_k)^{+}$ denotes the generalised inverse or pseudo-inverse of $\boldsymbol{C}_k\boldsymbol{E}_k$, and $\boldsymbol{\beta}_k \in \mathbb{R}^{m-q\times m}$ is a chosen arbitrary matrix such that $\boldsymbol{T}_k$ is a full-row rank matrix. Since $\operatorname{rank}\left([\boldsymbol{H}_k \; \boldsymbol{T}_k]^T\right) = m$, the system (4.100)-(4.101) can be transformed into an equivalent form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{w}_k, \tag{4.106}$$

$$\boldsymbol{H}_k\boldsymbol{y}_{k+1} = \boldsymbol{H}_k\boldsymbol{C}_k\boldsymbol{x}_{k+1} + \boldsymbol{H}_k\boldsymbol{v}_{k+1}, \tag{4.107}$$

$$\boldsymbol{T}_k\boldsymbol{y}_{k+1} = \boldsymbol{T}_k\boldsymbol{C}_k\boldsymbol{x}_{k+1} + \boldsymbol{T}_k\boldsymbol{v}_{k+1}. \tag{4.108}$$

Substituting the relation (4.106) into (4.107) leads to:

$$\boldsymbol{H}_k\boldsymbol{y}_{k+1} = \boldsymbol{H}_k\boldsymbol{C}_k\left[\boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{w}_k\right] + \boldsymbol{d}_k + \boldsymbol{H}_k\boldsymbol{v}_{k+1}, \tag{4.109}$$

or, equivalently:

$$\boldsymbol{d}_k = \boldsymbol{H}_k\left[\boldsymbol{y}_{k+1} - \boldsymbol{C}_k\left[\boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{w}_k\right] - \boldsymbol{v}_{k+1}\right]. \tag{4.110}$$

Inserting (4.110) into (4.106) leads to an alternative form of the system (4.100)-(4.101):

$$\boldsymbol{x}_{k+1} = \bar{\boldsymbol{A}}_k\boldsymbol{x}_k + \bar{\boldsymbol{B}}_k\boldsymbol{u}_k + \bar{\boldsymbol{E}}_k\boldsymbol{y}_{k+1} + \bar{\boldsymbol{w}}_k, \tag{4.111}$$

$$\bar{\boldsymbol{y}}_{k+1} = \bar{\boldsymbol{C}}_k\boldsymbol{x}_{k+1} + \bar{\boldsymbol{v}}_{k+1}, \tag{4.112}$$

where:

$$\bar{\boldsymbol{A}}_k = [\boldsymbol{I} - \boldsymbol{E}_k\boldsymbol{H}_k\boldsymbol{C}_k]\boldsymbol{A}_k, \tag{4.113}$$

$$\bar{\boldsymbol{B}}_k = [\boldsymbol{I} - \boldsymbol{E}_k\boldsymbol{H}_k\boldsymbol{C}_k]\boldsymbol{B}_k, \quad \bar{\boldsymbol{E}}_k = \boldsymbol{E}_k\boldsymbol{H}_k, \tag{4.114}$$

$$\bar{\boldsymbol{w}}_k = [\boldsymbol{I} - \boldsymbol{E}_k\boldsymbol{H}_k\boldsymbol{C}_k]\boldsymbol{w}_k - \bar{\boldsymbol{E}}_k\boldsymbol{v}_{k+1}, \tag{4.115}$$

$$\bar{\boldsymbol{y}}_{k+1} = \boldsymbol{T}_k\boldsymbol{y}_{k+1}, \quad \bar{\boldsymbol{C}}_k = \boldsymbol{T}_k\boldsymbol{C}_k, \tag{4.116}$$

$$\bar{\boldsymbol{v}}_{k+1} = \boldsymbol{T}_k\boldsymbol{v}_{k+1}. \tag{4.117}$$

The bounds of $\bar{\boldsymbol{w}}_k$ and $\bar{\boldsymbol{v}}_{k+1}$, i.e.

$$\bar{\mathbb{W}}_k = \bigcap_i \left\{ \bar{\boldsymbol{w}}_k : -\bar{b}_i \le \bar{w}_k^i \le \bar{b}_i \right\}, \tag{4.118}$$

and

$$\bar{\mathbb{V}}_k = \bigcap_i \left\{ \bar{\boldsymbol{v}}_k : -\bar{r}_i \le \bar{v}_k^i \le \bar{r}_i \right\}, \tag{4.119}$$

can easily be obtained using the equations (4.102) and (4.103).

Since the system (4.100)-(4.101) was transformed into the equivalent form (4.111)-(4.112), it is straightforward to use the state estimation algorithm described in (Maksarow and Norton 1996a) and, as a result, to design a Bounded-error Unknown Input Observer (BUIO). The purpose of the subsequent section is to give a detailed description of the above algorithm.

4.2.2. A bounded-error approach to state estimation

In a manner similar to the classical Kalman filtering procedure, the bounded-error approach consists in alternating two phases, i.e. the time and measurement updates. Unlike in the classical approach, where the initial state estimate $\hat{\boldsymbol{x}}_0$ is assumed to be a random variable, here it is assumed that $\hat{\boldsymbol{x}}_0$ belongs to an ellipsoidal set defined as:

$$\mathbb{E}_0 = \left\{ \boldsymbol{x}_0 : (\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0)^T \boldsymbol{P}_0^{-1} (\boldsymbol{x}_0 - \hat{\boldsymbol{x}}_0) \le 1 \right\}, \tag{4.120}$$

where $\hat{\boldsymbol{x}}_k$ denotes the centre of the ellipsoid (the state estimate), and $\boldsymbol{P}_0$ is a positive definite matrix describing its size and orientation. Thus, the ellipsoid containing all the admissible states at time $(k-1)$ is:

$$\mathbb{E}_{k-1} = \left\{ \boldsymbol{x}_{k-1} : (\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1})^T \boldsymbol{P}_{k-1}^{-1} (\boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1}) \le 1 \right\}. \tag{4.121}$$

As a result of the *time update*, which is a consequence of transforming the ellipsoid $\mathbb{E}_{k-1}$ according to the state transition equation (Fig. 4.10a), the ellipsoid $\mathbb{E}_{k/k-1}$ is obtained. The centre of the new ellipsoid is:

$$\hat{\boldsymbol{x}}_{k/k-1} = \bar{\boldsymbol{A}}_{k-1}\hat{\boldsymbol{x}}_{k-1} + \bar{\boldsymbol{B}}_{k-1}\boldsymbol{u}_{k-1} + \bar{\boldsymbol{E}}_k\boldsymbol{y}_k. \tag{4.122}$$

The matrix defining its size and orientation is successively computed by:

$$\boldsymbol{P}_{k/k-1}^0 = \bar{\boldsymbol{A}}_{k-1}\boldsymbol{P}_{k-1}\bar{\boldsymbol{A}}_{k-1}^T, \tag{4.123}$$

$$\boldsymbol{P}_{k/k-1}^{i+1} = (1+p_i)\boldsymbol{P}_{k/k-1}^i + (1+p_i^{-1})\bar{b}_i^2\boldsymbol{l}_i\boldsymbol{l}_i^T, \tag{4.124}$$

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

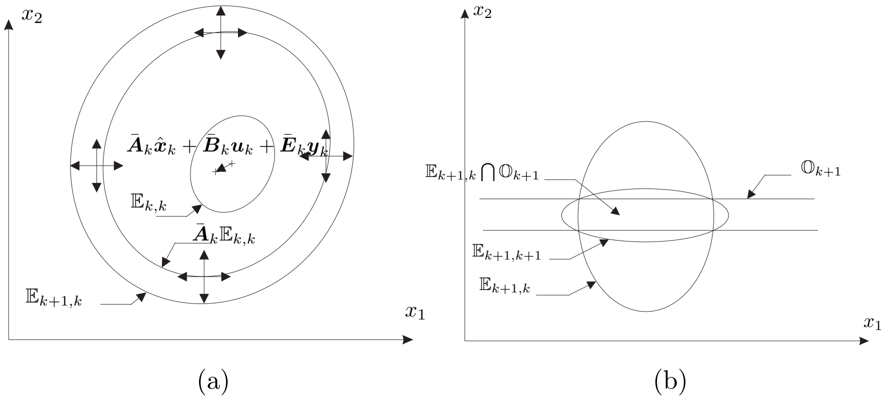
x_2
$\bar{A}_k\hat{x}_k + \bar{B}_k u_k + \bar{E}_k y_k$
$\mathbb{E}_{k,k}$
$\bar{A}_k\mathbb{E}_{k,k}$
$\mathbb{E}_{k+1,k}$
x_1
(a)
x_2
$\mathbb{E}_{k+1,k} \bigcap \mathbb{O}_{k+1}$
$\mathbb{O}_{k+1}$
$\mathbb{E}_{k+1,k+1}$
$\mathbb{E}_{k+1,k}$
x_1
(b)

Fig. 4.10 . The principle of bounded-error state estimation. Time (a) and measurement (b) updates.

where $l_i = [0,\ldots,\overbrace{1}^{i},\ldots,0]^T$. The value of the parameter $p_i > 0$ in (4.124) is the positive root of:

$$np_i^2 + (n-1)\mathrm{trace}(\boldsymbol{Q}_i)p_i - \mathrm{trace}(\boldsymbol{Q}_i) = 0, \tag{4.125}$$

where:

$$\boldsymbol{Q}_i = \bar{b}_i \left(\boldsymbol{P}^i_{k/k-1}\right)^{-1} \boldsymbol{l}_i \boldsymbol{l}_i^T. \tag{4.126}$$

Finally, $\boldsymbol{P}_{k/k-1} = \boldsymbol{P}^n_{k/k-1}$.

The *measurement update* intersects the predicted ellipsoid $\mathbb{E}_{k/k-1}$ with the pairs of parallel hyperplanes defined using (4.119), i.e.

$$\mathbb{O}_k = \bigcap_i \left\{\boldsymbol{x}_k : \bar{y}^i_k - \bar{r}_i \leq (\bar{\boldsymbol{c}}^i)^T \boldsymbol{x}_k \leq \bar{y}^i_k + \bar{r}_i\right\}, \tag{4.127}$$

where $\bar{\boldsymbol{C}}^T_k = [\bar{\boldsymbol{c}}_1,\ldots,\bar{\boldsymbol{c}}_m]$. As a result, the ellipsoid $\mathbb{E}_k \subset \mathbb{E}_{k/k-1} \cap \mathbb{O}_k$ is obtained (Fig.4.10b). The centre, size and orientation defining matrix is successively computed as:

$$\hat{\boldsymbol{x}}^0_k = \hat{\boldsymbol{x}}_{k/k-1},\ \boldsymbol{P}^0_k = \boldsymbol{P}_{k/k-1}, \tag{4.128}$$

$$\hat{\boldsymbol{x}}^{i+1}_k = \hat{\boldsymbol{x}}^i_k + q_i \frac{\boldsymbol{S}^i_k \bar{\boldsymbol{c}}_i e_i}{d_i^2}, \tag{4.129}$$

$$\boldsymbol{P}^{i+1}_k = \left(1 + q_i - \frac{q_i e_i^2}{d_i^2 + q_i g_i}\right) \boldsymbol{S}^i_k, \tag{4.130}$$

where:

$$e_i = \sqrt{g_i}\frac{\alpha_i^+ + \alpha_i^-}{2}, \quad d_i = \sqrt{g_i}\frac{\alpha_i^+ - \alpha_i^-}{2}, \tag{4.131}$$

$$g_i = \bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i, \ i = 0, \dots, m-1. \tag{4.132}$$

In the standard procedure any hyperplane bound which does not intersect $\mathbb{E}_k^i$ is replaced by the closest parallel hyperplane touching $\mathbb{E}_k^i$. The parameters α_i^+ and α_i^- denote the normalised distances from the centre of the ellipsoid $\mathbb{E}_k^i$ to each of the i-th hyperplane after such a replacement. The above parameters can be obtained in the following way.

Let the i-th hyperplane bound be:

$$\mathbb{V}_k^i = \left\{ \bar{\boldsymbol{v}}_k : \beta_i^+ \le \bar{\boldsymbol{c}}_i^T \boldsymbol{x}_k \le \beta_i^- \right\}. \tag{4.133}$$

For each ellipsoid $\mathbb{E}_k^j$, $j = 0, \dots, i$, the normalised distances are:

$$\alpha_j^+ = \frac{\bar{y}_k^i - \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^j + \bar{r}_k^i}{\sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}}, \tag{4.134}$$

$$\alpha_j^- = \frac{\bar{y}_k^i - \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^j - \bar{r}_k^i}{\sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}}. \tag{4.135}$$

In the next step

$$\text{If } \alpha_j^+ > 1 \text{ then } \beta_j^+ = \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^j + \sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}, \tag{4.136}$$

$$\text{If } \alpha_j^- < -1 \text{ then } \beta_j^- = \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^j - \sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}, \tag{4.137}$$

$$\text{If } -1 \le \alpha_j^+ \le 1 \text{ then } \beta_j^+ = \bar{y}_k^i + \bar{r}_k^i, \tag{4.138}$$

$$\text{If } -1 \le \alpha_j^+ \le 1 \text{ then } \beta_j^+ = \bar{y}_k^i - \bar{r}_k^i. \tag{4.139}$$

If $\alpha_j^+ < -1$ or $\alpha_j^- > 1$, then $\mathbb{V}_k^i$ does not intersect the j-th intermediate ellipsoid at time k. This may correspond to an inaccurate selection of the noise bounds. Such a property makes it possible to check the consistency of the whole model with the measured data (Fig.4.11). Faults can be detected in a similar way. Indeed, a fault occurrence may (in some sense) be equivalent to the inconsistency of the model with the measured data. The parameters β_i^+ and β_i^+ in (4.133) are defined as:

$$\beta_i^+ = \min_j \beta_j^+, \ \beta_i^- = \max_j \beta_j^-, \ j = 0, \dots, i, \tag{4.140}$$

and, finally, the parameters α_i^+ and α_i^- in (4.131) are defined as:

$$\alpha_i^+ = \frac{\beta_i^+ - \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^i}{\sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}}, \quad \alpha_i^- = \frac{\beta_i^- - \bar{\boldsymbol{c}}_i^T \hat{\boldsymbol{x}}_k^i}{\sqrt{\bar{\boldsymbol{c}}_i^T \boldsymbol{P}_k^i \bar{\boldsymbol{c}}_i}}. \tag{4.141}$$

If $\alpha_i^+\alpha_i^- \leq -1/n$, then $\mathbb{E}_k^{i+1} = \mathbb{E}_k^i$; otherwise, the parameter q_i minimizing the volume of $\mathbb{E}_k^{i+1}$ should be obtained as positive root of:

$$a_1 q_i^2 + a_2 q_i + a_3 = 0, \tag{4.142}$$

where:

$$a_1 = (n-1)g_i^2, \tag{4.143}$$

$$a_2 = \left((2n-1)d_i^2 - g_i + e_i^2\right) g_i, \tag{4.144}$$

$$a_3 = \left(n(d_i^2 - e_i^2) - g_i\right) d_i^2. \tag{4.145}$$

Finally, $\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^m$, and $\boldsymbol{P}_k = \boldsymbol{P}_k^m$.

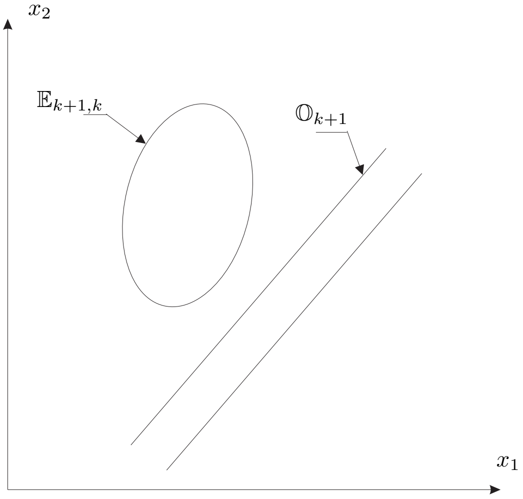

Fig. 4.11 . Fault detection with the bounded-error approach.

4.2.3. An extended BUIO

As has already been mentioned, the application of the EKF to the state estimation of non-linear systems has received considerable attention during the last two decades. This is mainly because the EKF can directly be applied to a large class of non-linear systems, and its implementation procedure is almost as simple as that for linear systems. The main drawback to such an approach is that its performance strongly depends on the difference between the non-linear system and the model linearised around the current state estimate. This is mainly because in the EKF case the linearisation errors are neglected. In the proposed approach, as in the EKF, the state equation is linearised around the current state estimate. The main difference between these two approaches is that in the proposed technique the linearisation errors are taken into account as additional disturbances and, as a result, the Bounded-error Extended Unknown Input Observer (BEUIO) is obtained.

Let us consider a class of non-linear systems which can be modeled by the following equations:

$$\boldsymbol{x}_{k+1} = \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{h}(\boldsymbol{u}_k + \boldsymbol{L}_{1,k}\boldsymbol{f}_k) + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{w}_k, \tag{4.146}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}\boldsymbol{x}_{k+1} + \boldsymbol{v}_{k+1} + \boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}, \tag{4.147}$$

where $\boldsymbol{g}(\boldsymbol{x}_k)$ is assumed to be continuously differentiable with respect to $\boldsymbol{x}_k$. For the sake of notational simplicity, the fault-free mode is assumed, i.e. $\boldsymbol{f}_k = \boldsymbol{0}$. In order to linearise the system (4.146)-(4.147) around the current state estimate, let us define the following matrix:

$$\boldsymbol{A}_k = \left.\frac{\partial \boldsymbol{g}(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_k}\right|_{\boldsymbol{x}_k=\hat{\boldsymbol{x}}_k}; \tag{4.148}$$

then the state equation of the system (4.146)-(4.147) can be transformed into an equivalent form:

$$\begin{aligned}\boldsymbol{x}_{k+1} =& \boldsymbol{g}(\hat{\boldsymbol{x}}_k) + \boldsymbol{A}_k(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k) + \boldsymbol{h}(\boldsymbol{u}_k) + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{w}_k \\ &+ \boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k),\end{aligned} \tag{4.149}$$

with the linearisation error $\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)$ satisfying:

$$\|\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)\|_\infty \leq \gamma\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|_\infty, \quad \boldsymbol{x}_k, \hat{\boldsymbol{x}}_k \in \mathbb{E}_k, \tag{4.150}$$

where $\|\boldsymbol{x}\|_\infty = \max_{i=1,\ldots,n}|x_i|$. The equation (4.149) can be expressed in a more convenient form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{u}_{k,f} + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{w}_k + \boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k), \tag{4.151}$$

where:

$$\boldsymbol{u}_{k,f} = \boldsymbol{g}(\hat{\boldsymbol{x}}_k) - \boldsymbol{A}_k\hat{\boldsymbol{x}}_k + \boldsymbol{h}(\boldsymbol{u}_k). \tag{4.152}$$

For the purpose of further consideration, it will be more convenient to express the ellipsoid $\mathbb{E}_k$ as a deviation from its centre:

$$\bar{\mathbb{E}}_k = \{\boldsymbol{z} : \hat{\boldsymbol{x}}_k + \boldsymbol{z} \in \mathbb{E}_k\}. \tag{4.153}$$

Let:

$$\phi_k = \sup_{\boldsymbol{z}\in\bar{\mathbb{E}}_k} \|\boldsymbol{z}\|_\infty, \tag{4.154}$$

$$\phi_k = \|\sqrt{\boldsymbol{P}_k(1,1)}, \ldots, \sqrt{\boldsymbol{P}_k(n,n)}\|_\infty, \tag{4.155}$$

then, using (4.150), the following relation is satisfied:

$$\|\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)\|_\infty \leq \gamma\phi_k, \tag{4.156}$$

which implies that:

$$-\gamma\phi_k \leq o(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)^i \leq \gamma\phi_k, \quad i = 1, \ldots, n. \tag{4.157}$$

This means that the linearisation error $\boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)$ can be treated as an additional disturbance vector with known bounds (4.157). Finally, the system (4.146)-(4.147) can be put in the following form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{u}_{k,f} + \boldsymbol{E}\boldsymbol{d}_k + \breve{\boldsymbol{w}}_k, \tag{4.158}$$

$$\boldsymbol{y}_{k+1} = \boldsymbol{C}\boldsymbol{x}_{k+1} + \boldsymbol{v}_{k+1}, \tag{4.159}$$

where:

$$\breve{\boldsymbol{w}}_k = \boldsymbol{w}_k + \boldsymbol{o}(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k). \tag{4.160}$$

The bounds of $\breve{\boldsymbol{w}}$, i.e.

$$\breve{\mathbb{W}}_k = \bigcap_i \left\{\breve{\boldsymbol{w}}_k : -\breve{b}_i \leq \breve{w}_k^i \leq \breve{b}_i\right\}, \tag{4.161}$$

can easily be computed using (4.102) and (4.157). Since the system (4.146)-(4.147) is expressed in the form (4.158)-(4.159), it is straightforward to perform the system transformation detailed in Section 4.2.1 and then use the state estimation algorithm described in Section 4.2.2. Without this transformation, though, the proposed observer can be used for the state estimation of non-linear unknown input-free systems.

4.2.4. Experimental results

4.2.4.1. State estimation and fault detection of linear systems

The purpose of this section is to show the reliability and effectiveness of the proposed approach. The numerical example considered here is a discrete-time model of a simplified longitudinal flight control (for a detailed description of the above model, we refer the reader to (Chen and Patton 1999, pp. 105-107)), which can be expressed in the following form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k + \boldsymbol{E}_k\boldsymbol{d}_k + \boldsymbol{w}_k, \tag{4.162}$$

$$\boldsymbol{y}_k = \boldsymbol{C}_k\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{4.163}$$

where:

$$\boldsymbol{A}_k = \begin{bmatrix} 0.9944 & -0.1203 & -0.4302 \\ 0.0017 & 0.9902 & -0.0747 \\ 0 & 0.8187 & 0 \end{bmatrix}, \tag{4.164}$$

$$\boldsymbol{B}_k = \begin{bmatrix} 0.4252 \\ -0.0082 \\ 0.1813 \end{bmatrix}, \boldsymbol{E}_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \tag{4.165}$$

$$\boldsymbol{C}_k = \boldsymbol{I}_{3\times 3},\ \boldsymbol{u}_k = 10, \tag{4.166}$$

$$\tag{4.167}$$

and $\boldsymbol{w}_k - \mathcal{U}(-0.1, 0.1)$, $\boldsymbol{v}_k - \mathcal{U}(-0.1, 0.1)$, where $\mathcal{U}(\cdot)$ denotes the uniform distribution. The initial conditions for the observer and the system to be observed were $\boldsymbol{x}_0 = \boldsymbol{0}$ and $\hat{\boldsymbol{x}}_0 = \boldsymbol{1}$. The bounds in (4.102) and (4.103) were selected according to

$$b = [0.3, 0.05, 0.05], \tag{4.168}$$

$$r_i = 0.5,\ i = 1, \ldots, m. \tag{4.169}$$

The simulation results are shown in Figs. 4.12 and 4.13. It can be observed that the state estimation method proposed in this paper can give satisfactory results. Indeed, unlike in the case of the classical Kalman filter (cf. Fig. 4.13), the effect of the unknown input was decoupled.

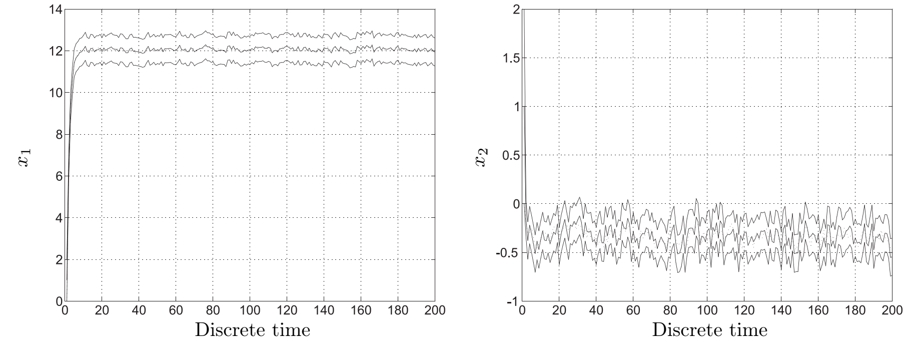

Fig. 4.12. The states (dashed) x_1 (left) and x_2 (right), their estimates and the corresponding sets of admissible states (solid).

In order to illustrate the principle of fault detection with the bounded-error unknown input observer, the following fault scenarios were considered:

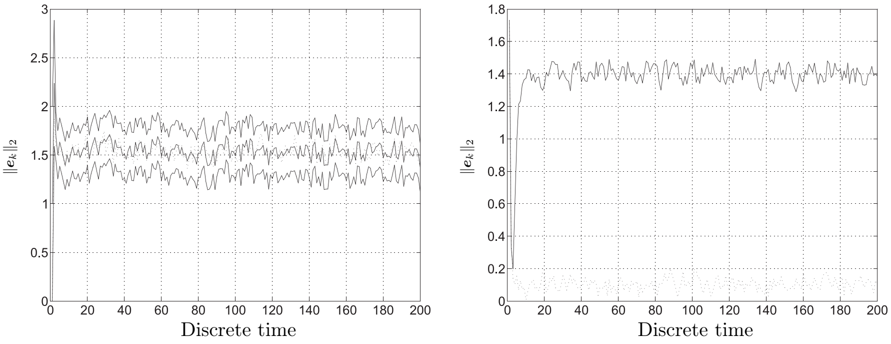

Fig. 4.13 . The state (dashed) x_3 (left), its estimate and the corresponding set of admissible states (solid). The state estimation error norm (right) for the proposed observer (dotted) and the Kalman filter (solid).

Case 1: An abrupt fault of the actuator:

$$f_{a,k} = \begin{cases} 0, & k < 100, \\ 3, & \text{otherwise}, \end{cases} \tag{4.170}$$

Case 2: An incipient fault of the actuator:

$$f_{a,k} = \begin{cases} 0, & k < 100, \\ 0.02e^{0.03k}, & \text{otherwise}, \end{cases} \tag{4.171}$$

Case 3: An abrupt fault of the $y_{2,k}$ sensor:

$$f_{s,k} = \begin{cases} 0, & k < 100, \\ 1, & \text{otherwise}, \end{cases} \tag{4.172}$$

Case 4: An incipient fault of the $y_{2,k}$ sensor:

$$f_{s,k} = \begin{cases} 0, & k < 100, \\ -0.005e^{0.03k}, & \text{otherwise}. \end{cases} \tag{4.173}$$

All the above faults were simulated using the model:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k(u_k + f_{a,k}) + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{w}_k, \tag{4.174}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k + [0\ 1\ 0]^T f_{s,k}. \tag{4.175}$$

First of all, as can be seen from Figs. 4.14-4.17, all the residuals are sensitive to all the faults. This is a condition necessary for fault detection, but to design

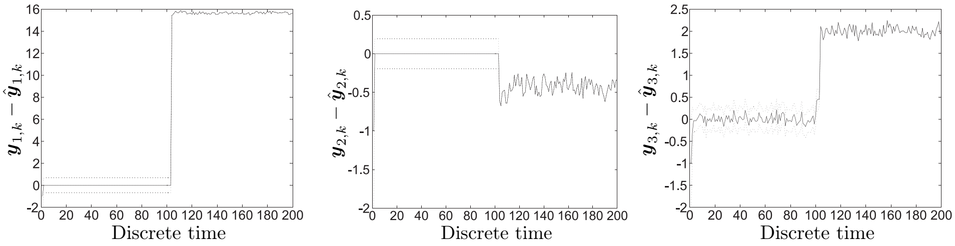

Fig. 4.14 . Residuals and their bounds for an abrupt fault of an actuator (Case 1).

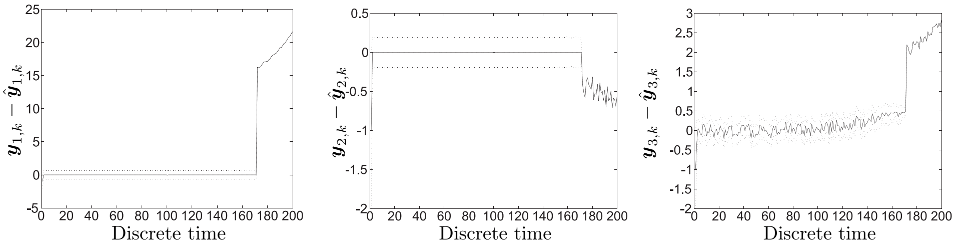

Fig. 4.15 . Residuals and their bounds for an incipient fault of an actuator (Case 2).

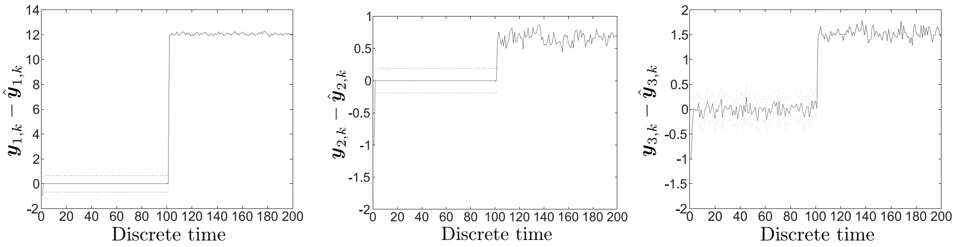

Fig. 4.16 . Residuals and their bounds for an abrupt fault of a sensor (Case 3).

a reliable fault isolation scheme (for sensors or actuators) it is necessary to employ the approaches proposed in Secs. 4.1.5 and 4.1.6.

Moreover, as can be seen from Figs. 4.14-4.17, the detection of the faults under consideration was performed relatively fast, especially for abrupt faults. In order to increase the sensitivity to incipient faults, it is necessary to use tighter noise bounds. On the other hand, such a modification may cause various false alarms.

4.2.4.2. State estimation of non-linear systems

In this section, in order to allow a comparison with the extended Kalman filter to be drawn, an unknown input-free system is considered, i.e. $\boldsymbol{E}_k\boldsymbol{d}_k = \boldsymbol{0}$ (a

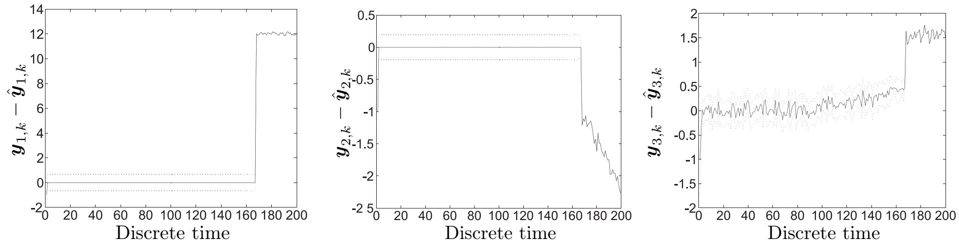

Fig. 4.17 . Residuals and their bounds for an incipient fault of a sensor (Case 4).

perfect model is assumed). The non-linear system was modelled by the following equations:

$$x_{1,k+1} = x_{1,k} + 0.02x_{2,k} + u_k + w_{1,k}, \tag{4.176}$$

$$x_{2,k+1} = x_{2,k} + 0.02(-9x_{1,k} + 2(1 - x_{1,k}^2)x_{2,k}) + w_{2,k}, \tag{4.177}$$

$$y_{k+1} = x_{1,k+1} + v_k, \tag{4.178}$$

where $w_{i,k} - \mathcal{U}(-0.1, 0.1)$, $v_k = 3$, the input signal was $u_k = 0.04 * \sin(0.02k)$, and the initial condition for the system and the observers was $\boldsymbol{x}_0 = (5, 0)$ and $\hat{\boldsymbol{x}}_0 = (0, 0)$, respectively. The parameters of the proposed observer were $\gamma = 0.2$, $\boldsymbol{b} = (0.1, 0.1)$, $r = 0.3$, while the parameters of the extended Kalman filter were $\boldsymbol{R}_k = 0.3$, $\boldsymbol{Q}_k = 0.1$.

The simulation results performed for both the proposed observer and the extended Kalman filter are shown in Figs. 4.18-4.20. From Figs. 4.19-4.20, it can be seen that the proposed observer is superior to the EKF. Indeed, the mean-squared state estimation error was 0.64 and 0.81 for the BEUIO and the EKF, respectively. As was expected, it turned out to be profitable to take the linearisation error into account.

The main drawback to the proposed approach is that there is no really practical method of selecting an appropriate value for the parameter γ. Thus, many experiments have to be carried out to obtain the appropriate value.

4.2.4.3. Fault detection of a chosen part of an evaporation station

The purpose of this section is to design a fault detection system for an apparatus that is a part of the evaporation station described in Section 3.4. To design an observer-based residual generator, the model (3.28)-(3.33), obtained via genetic programing, was utilised. Unfortunately, for such an MISO model it is impossible to design any UIO. Indeed, if $\boldsymbol{d}_k \in \mathbb{R}^q$, $q \leq m = 1$, then the matrix (4.105) equals zero, i.e. $\boldsymbol{T}_k = \boldsymbol{0}$, which makes it impossible to perform the system transformation described in Section 4.2.1. The same problem occurs with the decoupling approach realised according to (4.104). Since the unknown input distribution matrix has the

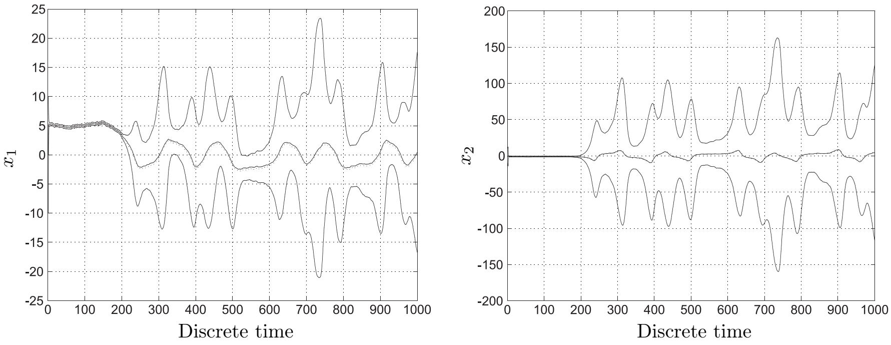

Fig. 4.18 . States estimates and the corresponding sets of admissible states obtained with the BEUIO.

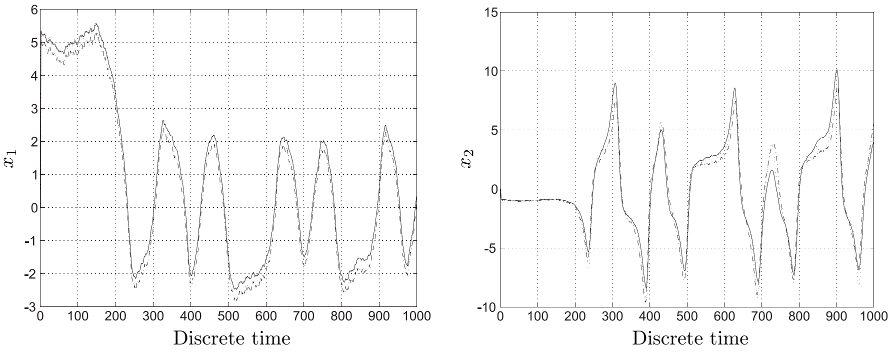

Fig. 4.19 . The true states (dash-dotted) and their estimates obtained by the proposed observer (dotted) and the extended Kalman filter (solid).

form $\boldsymbol{E}_k = [e_{1,k}, e_{2,k}]^T$, then using (4.10), the matrix $\boldsymbol{H}_k$ in (4.104) becomes:

$$\boldsymbol{H}_k = \left[\frac{e_{1,k}}{c_{1,k}e_{1,k} + c_{2,k}e_{2,k}}, \frac{e_{2,k}}{c_{1,k}e_{1,k} + c_{2,k}e_{2,k}}\right]^T. \tag{4.179}$$

In such a case the matrix $(\boldsymbol{I} - \boldsymbol{C}_k\boldsymbol{H}_k)$ equals zero, and hence the state estimate (4.104) is:

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}\boldsymbol{x}_{k+1} + \boldsymbol{H}_{k+1}\boldsymbol{v}_{k+1} + \boldsymbol{H}_{k+1}\boldsymbol{L}_{2,k+1}\boldsymbol{f}_{k+1}, \tag{4.180}$$

for which the residual signal becomes:

$$\boldsymbol{r}_{k+1} = \boldsymbol{y}_{k+1} - \hat{\boldsymbol{y}}_{k+1} = \boldsymbol{C}_{k+1}[\boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}]\boldsymbol{x}_{k+1} \tag{4.181}$$

$$+ [\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{H}_{k+1}][\boldsymbol{v}_{k+1} + \boldsymbol{L}_{1,k+1}\boldsymbol{f}_{k+1}]. \tag{4.182}$$

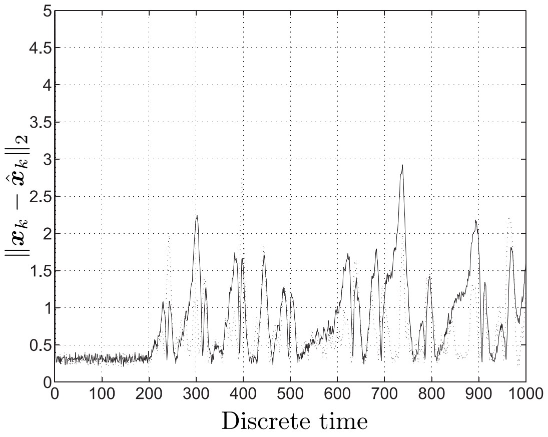

Fig. 4.20 . The state estimation error norm $\|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k\|_2$ for the proposed observer (dotted) and the extended Kalman filter (solid).

Unfortunately, $[\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{H}_{k+1}] = \boldsymbol{0}$, and hence everything is perfectly decoupled from the residual, including the faults. Indeed, irrespective of what happens with the system, the residual (4.182) is always zero, i.e. $\boldsymbol{r}_{k+1} = \boldsymbol{0}$. Nevertheless, this drawback pertains to all unknown input observers, not only those presented in this chapter. In other words, to use unknown input observers as residual generators it is necessary to ensure that:

$$[\boldsymbol{I} - \boldsymbol{C}_{k+1}\boldsymbol{H}_{k+1}] \neq \boldsymbol{0} \quad \text{and} \quad [\boldsymbol{I} - \boldsymbol{H}_{k+1}\boldsymbol{C}_{k+1}] \neq \boldsymbol{0}. \tag{4.183}$$

Irrespective of the above consideration, it is possible to use the model (3.28)-(3.33). Indeed, assuming that the matrix $\boldsymbol{C}$ in (3.33) has the form:

$$\boldsymbol{C} = \begin{bmatrix} 0.21 * 10^{-5} & 0.51 \\ 0 & 1 \end{bmatrix}, \tag{4.184}$$

the second output of the system $y_{2,k}$ can be simulated by the model (3.28)-(3.33) while the first output $y_{2,k}$ remains original.

The unknown input distribution matrix was obtained using the approach described in (Chen and Patton 1999) (assuming that $q = 1$) and, as a result, the matrix $\boldsymbol{E}_k$ was $\boldsymbol{E}_k = [11, 95.8]^T$. The constant γ was assumed to be $\gamma = 0.2$. The unknown input decoupling was realised according to (4.104).

To demonstrate the effectiveness of the obtained fault detection scheme, the following fault scenarios were considered:

Case 1: An abrupt fault of an actuator:

$$f_{a,k} = \begin{cases} 0, & k < 250, \\ -0.25u_{1,k}, & \text{otherwise}, \end{cases} \qquad \boldsymbol{L}_1 = [1, 0, 0, 0]^T, \tag{4.185}$$

Case 2: An abrupt fault of an actuator:

$$f_{a,k} = \begin{cases} 0, & k < 100, \\ 0.3u_{2,k}, & \text{otherwise}, \end{cases} \qquad \boldsymbol{L}_1 = [0, 1, 0, 0]^T, \tag{4.186}$$

Case 3: An abrupt fault of a sensor:

$$f_{s,k} = \begin{cases} 0, & k < 100, \\ -0.1y_{1,k}, & \text{otherwise}. \end{cases} \tag{4.187}$$

All the above faults were simulated using the model:

$$\boldsymbol{x}_{k+1} = \boldsymbol{h}(\boldsymbol{u}_k + \boldsymbol{L}_1 f_{a,k}) + \boldsymbol{E}\boldsymbol{d}_k + \boldsymbol{w}_k, \tag{4.188}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{v}_k + [1\ 0]^T f_{s,k}. \tag{4.189}$$

As can be seen from Figs. 4.21-4.22, the residual is sensitive to all the faults. As previously, this is a condition necessary for fault detection, but to design a reliable fault isolation scheme (for sensors or actuators) it is necessary to employ the approaches proposed in Setions 4.1.5 and 4.1.6.

Moreover, as can be seen from Figs. 4.21-4.22, the detection of the faults under consideration was performed relatively fast.

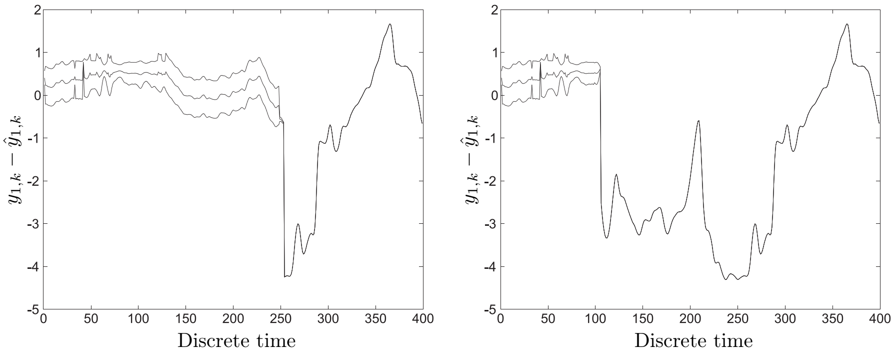

Fig. 4.21. A residual and its bounds for actuator faults: Case 1 (left), Case 2 (right).

4.3. Conclusions

The objective of this chapter was to propose robust observers for both deterministic and stochastic non-linear systems which can be applied as residual generators for FDI purposes.

To tackle the observer designing problem for deterministic systems, the concept of the extended unknown input observer was introduced. It was shown, with

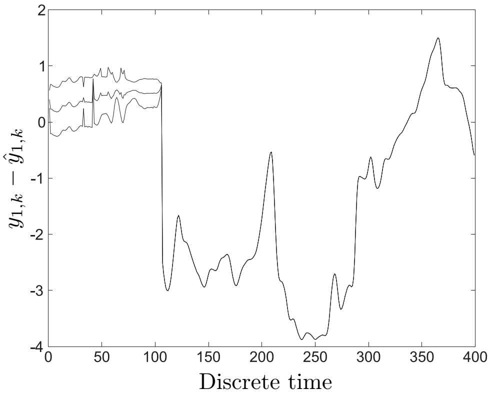

Fig. 4.22 . A residual and its bounds for a sensor fault (Case 3).

the use of the Lyapunov approach, that the proposed fault detection observer is convergent under certain conditions. Moreover, it was shown that an appropriate selection of the instrumental matrices $\boldsymbol{Q}_{k-1}$ and $\boldsymbol{R}_k$ strongly influences the convergence properties. To tackle the instrumental matrix selection problem, a genetic programming-based approach was proposed. It was shown by means of an example with an induction motor that the proposed observer can be a useful tool for both state estimation and fault diagnosis of non-linear deterministic systems. This is mainly because of the convergence properties of the observer, which confirm its superiority over classical approaches. Moreover, experimental results confirm that the proposed fault diagnosis scheme provides fast and reliable fault detection and isolation.

To tackle the observer design problem for stochastic systems, the bounded-error state estimation technique and a suitable transformation of the system equations were applied, resulting in the bounded-error unknown input observer. An extension of the proposed observer which can be applied to the state estimation of non-linear stochastic systems was proposed as well. This was performed by applying a linearisation technique similar to that of the classical EKF. Unlike the EKF, the proposed approach does not neglect the linearisation errors. Indeed, these errors were taken into account as additional disturbances. The drawbacks and advantages of the proposed residual generation technique were discussed in Section 4.2.4 during its application to state estimation and fault diagnosis of simulated and real industrial systems.

CONCLUSIONS

From the point of view of engineering, it is clear that providing fast and reliable fault detection and isolation is an integral part of control design, particularly as far as the control of complex industrial systems is considered.

Unfortunately, most of such systems exhibit a non-linear behaviour, which makes it impossible to use the well-developed techniques for linear systems. If it is assumed that the system is linear, which is not true in general, and even if the robust techniques for linear systems are used (e.g. unknown input observers), it is clear that such an approximation may lead to an unreliable fault detection and, because of this, an early indication of which faults are developing is rather impossible. Such a situation increases the probability of the occurence of faults, which can be extremely serious in terms of economic loss, environmental impact, or even human mortality. Indeed, robust techniques are able to tolerate a certain degree of model uncertainty. In other words, they are not robust to everything, i.e. robust to an arbitrary degree of model uncertainty. This real world development pressure creates the need for new techniques which are able to tackle fault diagnosis of non-linear systems. In spite of the fact that the problem has been attacked from various angles by many authors and a number of relevant results have already been reported in the literature, there is no general framework which can be simply and conveniently applied to maintain fault diagnosis for non-linear systems.

As was indicated in Chapter 2, observers are immensely popular as residual generators for fault detection (and consequently for fault isolation) of both linear and non-linear dynamic systems. Their popularity lies in the fact that they can also be employed for control purposes. There are, of course, many different observers which can be applied to non-linear, and especially non-linear deterministic systems, and the best-known of them were briefly reviewed in Section 2.3. Logically, the number of "real world" applications (not only simulated examples) should proliferate, yet this is not the case. It seems that there are two main reasons why strong formal methods are not accepted in engineering practice. First, the design complexity of most observers for non-linear systems does not encourage engineers to apply them in practice. Second, the application of observers is limited by the need for non-linear state-space models of the system being considered, which is usually a serious problem in complex industrial systems. This explains why most of the examples considered in the literature are devoted to simulated or laboratory systems, e.g. the celebrated three- (two- or even four-) tank system, an inverted pendulum, a traveling crane, etc.

As was indicated in Chapter 1, there is no practical approach which can be applied to obtain a state-space model of a system. In fact, the designer is left with a time-consuming trial-and-error approach.

Therefore, one original objective of this work was simply to develop effective and reliable methods to solve the practical problem of non-linear system identification. Another original objective was to develop robust observers for both deterministic and stochastic non-linear systems. In the process of executing these tasks, two new system identification algorithms for input-output and state-space models as well as the extended unknown input observer and the bounded-error extended unknown input observer for deterministic and stochastic non-linear systems were proposed. The following is a concise summary of the contributions provided by this book to the state-of-the-art in both non-linear system identification and fault detection (and especially to residual generation):

- Adaptation of the genetic programming technique to a discrete-time model construction and, especially, introduction of parameterised trees together with the rules reducing an excessive number of parameters. In particular, effective and genetic programming-based algorithms for designing both input-output and state-space models were developed. It was proven that the state-space models resulting from the above algorithms are asymptotically stable. In order to increase the convergence rate of the proposed techniques, the problem of adapting the probabilities of crossover and mutation was formulated and solved.
- Application of the unknown input observer to linear stochastic systems to form an Extended Unknown Input Observer (EUIO) for non-linear deterministic systems. A comprehensive convergence analysis with the Lyapunov approach was performed, resulting in the convergence conditions for the EUIO. Based on these conditions, a GP-based technique for increasing the convergence rate of the EUIO was proposed.
- Extension of the bounded-error state estimation approach to linear stochastic systems with unknown inputs, resulting in the Bounded-error Unknown Input Observer (BUIO). Development of an extension of the BUIO which can be applied to non-linear stochastic systems, resulting in the Bounded-error Extended Unknown Input Observer BEUIO.

The main advantage of the proposed system identification framework is that in spite of the fact that the models resulting from this approach are of a behavioural type, they are more transparent than the most popular rival structures, which is what neural networks undoubtedly are. This makes it possible, after a suitable analysis, to employ the models to FDI of system faults. This means that their transparency may allow to find some internal connections between the system and the model. Another advantage is that the state-space models resulting from this approach are asymptotically stable, which is *a priori* guaranteed by a suitable model structure.

The main drawback to the proposed system identification technique is that it is relatively time consuming. This is caused mainly by the fact that for each of the models, in each generation it is necessary to perform parameter estimation, which is relatively time consuming for models non-linear in their parameters. In addition to that, the computational requirements grow together with the model order n (for state-space models) or the dimension of the output vector m (for input-output). To tackle this problem, the adaptation rules of crossover and mutation probabilities were proposed, which made the identification process faster and more efficient. However, as usual, the model construction procedure is realised *off-line*, and hence the identification time is not extremely important.

The main advantage of the proposed extended unknown input observer for non-linear deterministic systems is that its design procedure is as simple as that for the extended Kalman filter. Unlike in the case of the extended Kalman filter, during the convergence analysis of the EUIO the linearisation errors were taken into account, which made it possible to establish the convergence conditions and, consequently, the GP-based approach to increasing the convergence rate of the EUIO. The practical advantages of the proposed approach were confirmed by examples regarding state estimation and fault diagnosis of an induction motor.

It is, of course, with some minor modifications, possible to apply the EUIO to non-linear stochastic systems. However, such an observer will inherit all the drawbacks of the Kalman filter regarding the restrictive assumptions of the noise distribution. To tackle this problem, an extension of the bounded-error state estimation technique to systems with unknown inputs was proposed. In this case, the only assumption is that the noise (or error) lies between given prior bounds. This is, undoubtedly, a less restrictive condition than the assumption that the process and measurement noises are zero-mean white noise sequences. The proposed observer was extended to non-linear deterministic systems. In the proposed approach, the linearisation errors were taken into account as additional disturbances with known bounds. The practical usefulness of the proposed techniques (for both linear and non-linear systems) was confirmed by experimental results concerning state estimation as well as fault detection for both simulated and real industrial systems.

The main drawback to the proposed observers is that they inherit the drawback of linear UIOs, which pertains the system dimensionality. Indeed, it is impossible to apply the proposed observers as residual generators for MISO systems and all systems for which the conditions (4.183) are not satisfied. From these conditions it is clear that it is possible to apply the proposed observers as residuals generators without, however, considering unknown inputs (no robustness to model uncertainty). This means that in such cases the robustness should be realised in the fault isolation stage, which for simple MISO systems seems to be a relatively easy task.

BIBLIOGRAPHY

Alcorta Garcia E. and Frank P. M. (1997): *Deterministic nonlinear observer-based approaches to fault diagnosis.* – Control. Eng. Practice. Vol. 5, No. 5, pp. 663-670.

Anderson B. D. O. and Moore J. B. (1979): *Optimal Filtering.* – New Jersey: Prentice-Hall.

Ashton S.A., Shields D.N. and Daley S. (1999): *Design of a robust fault detection observer for polynomial non-linearities.* – Proc. 14th IFAC World Congress, Beijing, P.R. China, CD-ROM.

Ayoubi M. (1994): *Fault diagnosis with dynamic neural structure and application to a turbo-charger.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'94*, Espoo, Finland, Vol. 2, pp. 618-623.

Back A. D. and Tsoi A. C. (1991): *FIR and IIR synapses. A new neural network architecture for time series modelling.* – Neural Computation, Vol. 3, pp. 375-385.

Basseville M. and Nikiforov I.V. (1993): *Detection of Abrubt Changes: Theory and Applications.* – New York: Prentice Hall.

Billings, S. A., M. J. Korenberg and S. Chen (1989): *Identification of MIMO nonlinear systems using a forward regression orthogonal estimator.* – Int. J. Contr., Vol. 49, pp. 2157-2189.

Blanke M., Bogh S., Jorgensen R.B., Patton R.J. (1994): *Fault detection for diesel engine actuator - a benchmark for FDI.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'94*, Espoo, Finland, Vol. 2, pp. 498-506.

Boutayeb M. and Aubry D. (1999): *A strong tracking extended Kalman observer for nonlinear discrete-time systems.* – IEEE Trans. Automat. Contr., Vol. 44, No. 8, pp. 1550-1556.

Bubnicki, Z. (2000): *General approach to stability and stabilization for a class of uncertain discrete non-linear systems.* – Int. J. Contr., Vol. 73, No. 14, pp. 1298-1306.

Chen J. and Patton R. J. (1999): *Robust Model-based Fault Diagnosis for Dynamic Systems.* – London: Kluwer Academic Publishers.

Chen J., Patton R. J. and Zhang H. (1996): *Design of unknown input observers and fault detection filters.* – Int. J. Contr., Vol. 63, No. 1, pp. 85-105.

Chow E.Y. and Willsky A.S. (1984): *Analytical redundancy and the design of robust detection systems.* – IEEE Trans. Automat. Contr., Vol. 29, No. 7, pp. 603-614.

Ding X. and Frank P.M. (1991): *Frequency domain approach and threshold selector for robust model-based fault detection and isolation.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'91*, Baden-Baden, Germany, Vol. 1, pp. 307–312.

Duch W., Korbicz J., Rutkowski L. and Tadeusiewicz R. (2000): *Biocybernetyka i Inżynieria Biomedyczna 2000: T. 6: Sieci Neuronowe.* – Warsaw: Akademicka Oficyna Wydawnicza EXIT. (in Polish)

Edelmayer A. (Editor) (2000). *Research of Quantitative and Qualitative FDI Methods Based on Data from Lublin Sugar Factory. – Special Session of the IFAC Symp. SAFEPROCESS'2000.* Prep. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'2000*, June 14-16, Budapest, Hungary, Vol. 1, pp. 331-358.

Eiben A. E., Hinterding R. and Michalewicz Z. (1999): *Parameter control in evolutionary algorithms.* – IEEE Trans. Evolutionary Computation, Vol. 3, No. 2, pp. 124-141.

Esparcia-Alcazar A. I. (1998): *Genetic Programming for Adaptive Digital Signal Processing.* – Glasgow University: Ph.D. thesis.

Farlow S. J. (1984): *Self-organizing Methods in Modelling - GMDH Type Algorithms.* – New York: Marcel Dekker.

Fasconi P., Gori M. and Soda G. (1992): *Local feedback multilayered networks.* – Neural Computation, Vol. 4, pp. 120-130.

Fogel D.B. (1995): *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* – IEEE Press.

Fogel L.J., Owens A.J. and Walsh M.J. (1999): *An overview of evolutionary programming.* – In: Evolutionary Algorithms (De Jong K., Vose M.D. and Whitley L.D., Eds.) – Heidelberg: Springer-Verlag.

Galar R. (1989): *Evolutionary search with soft selection.* – Bio. Cyber., Vol.51, pp.1-9.

Gertler J.J. (2000): *Structured parity equations in fault detection and isolation.* – In: Issues of Fault Diagnosis for Dynamic Systems (Patton, R. J., P. Frank and R. N. Clark, Eds.) – Berlin: Springer-Verlag.

Golub D.E. and Van Loan G.F. (1989): *Matrix computations.* – Baltimore and London: The John Hopkins University Press.

Gori M., Bengio Y. and Mori R. D. (1989): *BPS: A learning algorithm for capturing the dynamic nature of speech.* – Proc. Int. Joint Conf. Neural Networks, Vol. II, pp. 417-423.

Gray G. J, Murray-Smith D. J., Li Y., Sharman K. C., and Weinbrenner T. (1998): *Nonlinear model structure identification using genetic programming.* – Control Eng. Practice, Vol. 6, pp. 1341-1352.

Greblicki W. (1994): *Nonparametric identification of Wiener systems by orthogonal series.* – IEEE Trans. Automat. Contr., Vol. 39, No. 10, pp. 2077-2086.

Greblicki W. (1996): *Nonlinearity estimation in Hammerstein systems based on ordered observations.* – IEEE Trans. Automat. Contr., Vol. 44, No. 5, pp. 1224-1233.

Greblicki W. (2001): *Recursive identification of Wiener systems.* – Int. J. Appl. Math. Comput. Sci., Vol. 11, No. 4, pp. 101-115.

Guernez C., Cassar J. Ph. and Staroswiecki M. (1997): *Extension of parity space to non-linear polynomial dynamic systems.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'97*, Hull, UK, Vol. 2, pp. 861-866.

Hac A. (1992): *Design of disturbance decoupled observers for bilinear systems.* – ASME J. Dynamic Sys. Measure. Contr., Vol. 114, pp. 556-562.

Hertz J., Krogh R. and Palmer G. (1991): *Introduction to the Neural Computation.* – New York: Addison-Wesley.

Holland J.H. (1975): *Adaptation in natural and artificial systems.* – Ann Arbor, MI: The University of Michigan Press.

Hornik K., Stinchombe M. and White H. (1989): *Multi-layer feed-forward networks are universal approximators.* – Neural Computation, Vol. 2, pp. 359-366.

Hou M. and Pugh A.C. (1997): *Observing state in bilinear systems: an UIO approach.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'97*, Hull, UK, Vol. 2, pp. 783-788.

Ivakhnenko A. G. T (1968): *The group method of data handling - A rival of stochastic approximation.* – Sov. Autom. Contr., Vol. 3, p. 43.

Kalman R.E. and Bucy R.S. (1961): *New results in linear filtering and prediction theory.* – Trans. ASME J. Basic Eng. (ser. D), Vol. 83, pp. 85-108.

Keller J. Y. and Darouach M. (1999): *Two-stage Kalman estimator with unknown exogenous inputs.* – Automatica, Vol. 35, pp. 339-342.

Kinneart M. (1999): *Robust fault detection based on observers for bilinear systems.* – Automatica, Vol. 35, pp. 1829-1824.

Kościelny J. M., Jankowska A., Syfert M., Ostasz A. and Nowak J. (1999): *Fault detection methods based on fuzzy logic and neural networks - application and research in chosen industrial plants.* – Proc. 4th Nat. Conf.: *Diagnostics of Industrial Processes: DPP 1999*, September 13-16, Kazimierz Dolny, Poland, pp. 49-64. (in Polish)

Korbicz J. and Bidyuk P. (1993): *State and Parameter Estimation. Digital and Optimal Filtering. Applications.* – Zielona Góra: Technical University Press.

Kowalczuk Z. and Gunawickrama K. (2000): *Leak detection and isolation for tranmission pipelines via non-linear state estimation.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS 2000*, Budapest, Hungary, Vol. 2, pp. 943-948.

Kowalczuk Z. and Suchomski P. (2001): *Analytical redundancy: Kalman filter as a basis for robust detection.* – Proc. 5th Nat. Conf.: *Diagnostics of Industrial Processes: DPP 2001*, September 17-19, Łagów Lubuski, Poland. (in Polish).

Koza J. R. (1992): *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* – Cambridge: The MIT Press.

Krishnaswami V. and Rizzoni G. (1994): *Non-linear parity equation residual generation for fault detection and isolation.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'94*, Espoo, Finland, Vol. 1, pp. 317-322.

Kurek J. (1982): *Observations of the state vector of linear multivariable systems with unknown inputs.* – Int. J. Contr., Vol. 36, No. 3, pp. 511-515.

Ljung L. (1987): *System Identification. Theory for the Users.* – New Jersey: Prentice-Hall.

Ljung, L. (1988): *System Identification Toolbox. For Use with MATLAB.* – Natick, MA: The MathWorks Inc.

Lunze J. and Schiller F. (2000): *Fault diagnosis based on a predicate logic description of dynamical systems.* – In: Issues of Fault Diagnosis for Dynamic Systems (Patton, R. J., P. Frank and R. N. Clark, Eds.) – Berlin: Springer-Verlag.

Maksarow D. and Norton J. (1996): *Tuning noise bounds in state bounding.* – Proc. *CESA'96* IMACS Multiconference, July 9-12, Lille, France, pp. 837-842.

Maksarow D. and Norton J. (1996): *State bounding with ellipsoidal set description of the uncertainty.* Int. J. Contr., Vol. 65, No. 5, pp. 847-866.

Michalewicz Z. (1996): *Genetic Algorithms + Data Structures = Evolution Programs.* Springer. Berlin.

Milanese M., Norton J., Piet-Lahanier H. and Walter E. (Eds.) (1996): *Bounding Approaches to System Identification.* – New York: Plenum Press.

Mrugalski M. and Witczak M. (2002): *Parameter estimation of dynamic GMDH neural networks with the bounded-error technique.* J. Appl. Comput. Sci. (in print).

Nelles O. (2001): *Non-linear Systems Identification. From Classical Approaches to Neural Networks and Fuzzy Models.* – Berlin: Springer.

Nuck D., Klawonn F. and Krause R. (1997): *Foundations of Neuro-Fuzzy Systems.* – Chichester: John Wiley & Sons.

Van Overschee P. and De Moor B. (1994): *N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems.* Automatica, Vol. 30, No. 1, pp. 75-93.

Paraskevopoulos P.N. (1996): *Digital Control Systems.* – London: Prentice Hall.

Patan K. (2000): *Artificial Dynamic Neural Networks and Their Application in Modelling of Industrial Processes.* – Warsaw University of Technology: Ph.D. thesis. (via Internet `http://www.issi.uz.zgora.pl/~patan/`)

Patton R. J. and Chen J. (1997): *Observer-based fault detection and isolation: robustness and applications.* Control Eng. Practice, Vol. 5, No. 5, pp. 671-682.

Patton R. J., Frank P. and Clark R. N. (Eds.) (2000): *Issues of Fault Diagnosis for Dynamic Systems.* – Berlin: Springer-Verlag.

Patton R. J. and Korbicz J. (Eds.) (1999): *Advances in Computational Intelligence for Fault Diagnosis Systems.* – Special issue of Int. J. Appl. Math. Comput. Sci., Vol. 9, No. 3.

Rafajłowicz E. (1989): *Time-domain optimization of input signals for distributed-parameter system identification.* J. Optimization Theory and Applications, Vol. 60, No. 1, pp. 67-79.

Rafajłowicz E. (1996): *Algorithms of Experimental Design with Implementations in MATHEMATICA.* – Warsaw: Akademicka Oficyna PLJ. (in Polish)

Schreier G., Ragot J., Patton R.J. and Frank P.M. (1997): *Observer design for a class of non-linear systems.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'97*, Hull, UK, Vol. 1, pp. 483-488.

Seliger R. and P. Frank (2000): *Robust observer-based fault diagnosis in non-linear uncertain systems.* – In: Issues of Fault Diagnosis for Dynamic Systems (Patton, R. J., P. Frank and R. N. Clark, Eds.) – Berlin: Springer-Verlag.

Shields D.N. and Ashton S. (2000): *A fault detection observer method for non-linear systems.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS 2000*, Budapest, Hungary, Vol. 1, pp. 226-231.

Shumsky A. Ye. (1997): *Robust residual generation for diagnosis of non-linear systems: parity relation approach.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS'97*, Hull, UK, Vol. 2, pp. 867-872.

Tau F.E. (1973): *Observing the state of non-linear dynamic systems.* – Int. J. Contr., Vol. 17, No. 3.

Uciński D. (1999): *Measurement Optimization for Parameter Estimation in Distributed Systems.* – Zielona Góra: Technical University Press. (via Internet: `http://www.issi.uz.zgora.pl/~ucinski/`).

Uciński D. (2000): *Optimal sensor location for parameter estimation in distributed processes.* – Int. J. Contr. Vol. 73, No. 13, pp. 1235-1248.

Viswanadham N., Taylor J.H and Luce E.C. (1987): *A frequency-domain approach to failure detection and isolation with application to GE-21 turbine engine control systems.* – Control - Theory and Advanced Technology, Vol. 3, No. 1, pp. 45–72.

Walter E. and Pronzato L. (1997): *Identification of Parametric Models from Experimental Data.* – Berlin: Springer.

Williams R.J. and Zipser D. (1989): *A learning algorithm for continually running fully recurrent neural networks.* – Neural Computation, Vol. 1, pp. 270-289.

Willsky A.S. and Jones H.L. (1976): *A generalized likelihood ratio approach to the detection of jumps in linear systems.* – IEEE Trans. Automat. Contr., Vol. 21, pp. 108-121.

Witczak M. (2001): *Design of an extended unknown input observer for non-linear systems.* – Proc. 5th Nat. Conf.: *Diagnostics of Industrial Processes: DPP 2001*, September 17-19, Łagów Lubuski, Poland, pp. 65-71. (in Polish).

Witczak M. and Korbicz J. (2000a): *Genetic programming based observers for nonlinear systems.* – Proc. IFAC Symp.: *Fault Detection, Supervision and Safety of Technical Processes: SAFEPROCESS 2000*, June 14-16, Budapest, Hungary, Vol. 2, pp. 967-972.

Witczak M. and Korbicz J. (2000b): *Identification of nonlinear systems using genetic programming.* – Proc. 6th Int. Conf. *Methods and Models in Automation and Robotics: MMAR 2000*, August 28-31, Międzyzdroje, Poland, Vol. 2, pp. 795-800.

Witczak M. and Korbicz J. (2001a): *An extended unknown input observer for nonlinear discrete-time systems.* Proc. *European Control Conf. ECC 2001*, September 7-9, Porto, Portugal, CD-ROM.

Witczak M. and Korbicz J. (2001b): *Robustifying an extended unknown input observer with genetic programming.* – Proc. 7th IEEE Int. Conf. *Methods and Models in Automation and Robotics: MMAR'2001*, August 28-31, Międzyzdroje, Poland, Vol. 2, pp. 1061-1066.

Witczak M. and Korbicz J. (2001c): *An evolutionary approach to identification of non-linear dynamic systems.* – Springer-Verlag: Proc. Int. Conf. on *Artificial Neural Nets and Genetic Algorithms ICANNGA'01*, May, Prague, Czech Republic, pp. 240-243.

Witczak M. and Korbicz J. (2002a): *A bounded-error approach to designing unknown input observers.* – *IFAC World Congress*, Barcelona, Spain. (accepted for publication)

Witczak M. and Korbicz J. (2002b): *Genetic programming in fault diagnosis and identification of non-linear dynamic systems.* – In: Diagnostics of Processes. Models, Methods of Artificial Intelligence. Applications. (Korbicz J., Kościelny J. M. and Kowalczuk Z., Cholewa W. Eds.) – Warsaw: WNT. (in print) (in Polish)

Yu D. and Shileds D.N. (1996): *Bilinear fault detection observer and its application to a hydraulic system.* – Int. J. Contr., Vol. 64, pp. 1023-1047.

Zhang J. and Roberts P.D. (2000): *A hierarhical structure for on-line process fault diagnosis based on deep qualitative modelling.* – In: Issues of Fault Diagnosis for Dynamic Systems (Patton, R. J., P. Frank and R. N. Clark, Eds.) – Berlin: Springer-Verlag.

Zolghardi A., Henry D. and Monision M. (1996): *Design of nonlinear observers for fault diagnosis. A case study.* – Control. Eng. Practice. Vol. 4, No. 11, pp. 1535-1544.

STRESZCZENIE

W pracy rozpatruje się zagadnienia związane z detekcją uszkodzeń i identyfikacją nieliniowych systemów dynamicznych.

Cel pracy można podzielić na dwie części. Pierwsza z nich dotyczy opracowania metodologii konstruowania modeli nieliniowych systemów dynamicznych z zastosowaniem programowania genetycznego. Natomiast druga projektowania odpornych obserwatorów stanu do zadań detekcji uszkodzeń.

W szczególności zostały przeanalizowane i zbadane następujące problemy:

1. Badania teoretyczne

W ramach badań teoretycznych zrealizowano:

• **Zaadaptowano technikę programowania genetycznego do konstruowania modeli nieliniowych systemów dynamicznych.** W szczególności, pokazano jak różne struktury modeli można przedstawić w postaci drzew oraz jak wykorzystać programowanie genetyczne do wyznaczania ich odpowiedniej postaci. Przeprowadzone badania wykazały, że programowanie genetyczne w „klasycznej" postaci powoduje trudności z wyznaczaniem parametrów modeli. Do rozwiązania tego zadania zastosowano sparametryzowane drzewa oraz zaproponowano reguły umożliwiające redukcję nadmiernej liczby parametrów. Opracowano efektywne algorytmy służące wyznaczaniu modeli zarówno w konfiguracji wejściowo-wyjściowej jak i w przestrzeni stanów. Wykazano również, że otrzymane modele opisane w przestrzeni stanów są asymptotycznie stabilne. Podstawową wadą proponowanych algorytmów jest ich duża złożoność obliczeniowa, odzwierciedlająca się w relatywnie długim czasie procesu identyfikacji. W celu poprawienia szybkości zbieżności proponowanych algorytmów sformułowano i rozwiązano problem doboru prawdopodobieństw krzyżowania i mutacji.

• **Z wykorzystaniem obserwatora o nieznanym wejściu dla liniowych systemów stochastycznych opracowano rozszerzony obserwator o nieznanym wejściu dla deterministycznych systemów nieliniowych.** Podobnie jak w przypadku klasycznego rozszerzonego filtru Kalmana, w celu zaadaptowania liniowego obserwatora do estymacji stanu, a w konsekwencji do generacji sygnałów residuum dla potrzeb detekcji uszkodzeń systemów nieliniowych, zaproponowano linearyzację modelu wokół aktualnej estymaty stanu. W odróżnieniu od klasycznego rozszerzonego filtru Kalmana, w proponowanym rozwiązaniu uwzględnia się błędy linearyzacji. Ponieważ rozważany obserwator dla systemów stochastycznych został zaadaptowany dla systemów deterministycznych, pozostawiło to pewien stopień swobody, który został wykorzystany do wykazania zbieżności proponowanego obserwatora. W szczególności, przeprowadzono szczegółową analizę zbieżności obserwatora z zastosowaniem metody Lapunowa, w rezultacie której otrzymano odpowiednie warunki zbieżności. Opierając się na powyższych warunkach zaproponowano technikę zwiększania szybkości zbieżności obserwatora z zastosowaniem programowania genetycznego.

• **Z wykorzystaniem techniki estymacji stanu przy ograniczonych**

wartościach błędów opracowano obserwatory o nieznanym wejściu zarówno dla liniowych jak i nieliniowych systemów stochastycznych. W odróżnieniu od klasycznego filtru Kalmana (w przypadku którego zakłada się, że szumy procesu i pomiarów posiadają rozkład gaussowski) zastosowanie techniki estymacji stanu przy ograniczonych wartościach błędów umożliwia wprowadzenie mniej restrykcyjnych warunków (tzn. zakłada się, że szumy procesu i pomiarów ograniczone są przez pewne znane z góry wartości).

W pracy pokazano jak poprzez odpowiednią transformację modelu można zastosować powyższą technikę do estymacji stanu, a w konsekwencji do generowania sygnałów residuum dla potrzeb detekcji uszkodzeń przy założeniu wpływu niepewności modelu i innych czynników (wpływu nieznanego wejścia) mogących prowadzić do błędnej diagnozy. Opracowany obserwator o nieznanym wejściu dla stochastycznych systemów nieliniowych stanowi rozszerzenie wcześniej zaproponowanego obserwatora liniowego. Podobnie jak w przypadku klasycznego rozszerzonego filtru Kalmana, przeprowadzono linearyzację modelu wokół aktualnej estymaty stanu. W odróżnieniu od rozszerzonego filtru Kalmana w procedurze projektowania obserwatora uwzględnia się błędy linearyzacji. W szczególności, pokazuje się jak uwzględnić błędy linearyzacji jako zakłócenia o znanych wartościach granicznych.

2. Badanie aplikacyjne

W ramach badań aplikacyjnych zrealizowano:

- **Zastosowanie algorytmów opierających się na programowaniu genetycznym do identyfikacji nieliniowych systemów dynamicznych.** W szczególności, wyznaczono modele wybranych urządzeń stacji wyparnej cukrowni Lublin S.A., zarówno w konfiguracji wejściowo-wyjściowej jak i w przestrzeni stanów. Przeprowadzono porównania z klasycznymi modelami liniowymi, które wykazały zasadność stosowania proponowanej techniki identyfikacji. Na podstawie przeprowadzonych badań statystycznych pokazano, że pomimo braku dowodu zbieżności proponowane algorytmy można z powodzeniem wykorzystywać do różnych zadań identyfikacji. Również na podstawie przykładu wykazano zasadność stosowania reguł adaptacji prawdopodobieństw krzyżowania i mutacji.
- **Zastosowanie rozszerzonego obserwatora o nieznanym wejściu do estymacji stanu i detekcji uszkodzeń silnika elektrycznego.** Na przykładzie z modelem silnika elektrycznego pokazano wyższość proponowanego obserwatora nad klasycznymi rozwiązaniami. W szczególności pokazano, iż proponowane rozwiązanie umożliwia uzyskanie znaczącego zwiększenia szybkości zbieżności, zapewniając jednocześnie samą zbieżność. Na tym samym przykładzie wykazano zasadność stosowania odpornych obserwatorów w diagnostyce uszkodzeń.
- **Zastosowanie obserwatorów o nieznanym wejściu wykorzystujących technikę estymacji stanu przy ograniczonych wartościach błędów do estymacji stanu i detekcji uszkodzeń systemów stochastycznych.** W szczególności, pokazano zastosowanie proponowanego obserwatora do detekcji uszkodzeń urządzeń wykonawczych, jak i czujników pomiarowych liniowego systemu stochastycznego. Na przykładzie z nieliniowym modelem systemu wykazano zasadność uwzględniania błędów linaryzacji jako dodatkowych zakłóceń

o znanych wartościach granicznych. Pokazano również przykład zastosowania proponowanego obserwatora w detekcji uszkodzeń wybranego elementu stacji wyparnej cukrowni Lublin S.A. W rozważanym przykładzie, wykorzystano model opisany w przestrzeni stanów, do którego wyznaczenia posłużono się proponowaną techniką identyfikacji systemów opierającą się na programowaniu genetycznym. Książka została podzielona na 4 rozdziły. W rozdziale 1 przedstawiono dobrze znane i często wykorzystywane struktury modeli, począwszy od klasycznych rozwiązań, a skończywszy na modelach opierających się na sieciach neuronowych. Rozważano zarówno statyczne jak i dynamiczne modele. Omówiono również zalety i wady każdej ze struktur. W końcowej części rozdziału omówiono problemy przygotowywania danych pomiarowych i doboru kryterium identyfikacji. Rozdział 2 zawiera przegląd najbardziej popularnych metod generacji sygnału residuum dla potrzeb diagnostyki uszkodzeń. W rozdziale skupiono się na detekcji uszkodzeń, która stanowi najważniejszą część procesu diagnostyki uszkodzeń (bez detekcji uszkodzeń nie jest możliwe przeprowadzenie lokalizacji uszkodzeń). W szczególności, przedstawiono różne techniki generacji sygnału residuum bazujące na modelu matematycznym systemu, począwszy od bezpośredniego zastosowania modelu, a skończywszy na rozwiązaniach bazujących na odpornych obserwatorach stanu. W rozdziale opisano rozwiązania zarówno dla systemów liniowych jak i nieliniowych. Rozważano również wady, zalety jak i możliwe dziedziny zastosowań każdego z rozwiązań.

W rozdziale 3 zaproponowano techniki identyfikacji systemów bazujące na programowaniu genetycznym. Przedstawiono krótki przegląd dobrze znanych i często stosowanych algorytmów ewolucyjnych. Szczególną uwagę skupiono na algorytmach genetycznych i programowaniu genetycznym. Zaproponowano i zastosowano zmodyfikowaną wersję programowania genetycznego do rozwiązania problemu wyznaczania modeli nieliniowych systemów dynamicznych. W szczególności, opracowano algorytmy służące wyznaczaniu modeli zarówno w konfiguracji wejściowo-wyjściowej jak i w przestrzeni stanów. W rozdziale rozważa się również stabilność modeli opisanych w przestrzeni stanów otrzymywanych za pomocą proponowanego algorytmu. Końcowa część rozdziału zawiera rezultaty eksperymentów potwierdzające efektywność proponowanego podejścia.

W rozdziale 4 zaproponowano koncepcję rozszerzonego obserwatora o nieznanym wejściu dla nieliniowych systemów deterministycznych oraz szczegółowo opisano procedurę jego projektowania. Z wykorzystaniem metody Lapunowa, przeprowadzono szczegółową analizę zbieżności obserwatora. Otrzymane rezultaty posłużyły zwiększeniu szybkości zbieżności obserwatora. W tym celu zaproponowano podejście bazujące na programowaniu genetycznym. Rozdział zawiera wyniki numerycznych symulacji dotyczących estymacji stanu i diagnostyki uszkodzeń silnika elektrycznego. W rozdziale rozważano również zastosowanie techniki estymacji stanu przy ograniczonych wartościach błędów do projektowania obserwatorów o nieznanym wejściu zarówno dla liniowych jak i nieliniowych systemów stochastycznych. Przedstawiono problem estymacji stanu systemów liniowych w przypadku znanych wartości granicznych szumów procesu i pomiarów oraz zamieszczono odpowiedni algorytm. Pokazano jak zastosować powyższy algorytm przy projektowaniu obserwatora o nieznanym wejściu dla szerokiej klasy liniowych systemów stochastycznych. Zaproponowano również rozszerzoną wersję rozważanego obserwatora dla nieliniowych systemów stochastycznych. Końcowa część rozdziału zawiera rezultaty eksperymentów dotyczące estymacji stanu i diagnostyki uszkodzeń liniowych i nieliniowych systemów stochastycznych.

Książka została częściowo opracowana w ramach realizacji projektu badawczego KBN pt. *Detekcja uszkodzeń i identyfikacja nieliniowych systemów dynamicznych* (2001-2002) oraz międzynarodowego projektu pt. EU FP 5 Research Training Network project DAMADICS: *Development and Application of Methods for Actuator Diagnosis in Industrial Control Systems* (2000-2003).