A STRUCTURED APPROACH TO IDENTIFICATION TECHNIQUES FOR THE ANALYSIS OF INDUSTRIAL PROCESSES

Steffen Körner

A thesis submitted in partial fulfilment of the requirements of the
University of Glamorgan / Prifysgol Morgannwg
for the degree of

Doctor of Philosophy
in Process Automation

The research project was carried out
in collaboration with the Fachhochschule Hannover.

School of Design and Advanced Technology

August 1999

Acknowledgement

At first I wish to express my sincere gratitude to my supervisors Mr. Keith J. Baker, the Director of Studies of this research project, and Prof. Dennis L. Hawkes for the many fruitful discussions and their help within my research work.

I like to thank especially Prof. Dr.-Ing. Reimar Schumann for his enthusiastic supervision, help and encouragement during the course of my PhD program. He has provided great stimulation for this work and we shared many hours of productive work.

Many thanks are due to all my colleagues in the applied research program REPAM and to Dr. Matthias Strickrodt, a member of the collaborative research project, for good discussions not only in the field of automation. I would also like to thank my research colleagues Birga Syska and Frank Dziembowski for their support in the control laboratory and the extensive tests of the software developed within this research project.

I also like to thank the University of Glamorgan, the School of Design and Advanced Technology and the members of staff for supporting my work. The financial support from the Volkswagen foundation is gratefully acknowledged. It also allowed me to present and discuss different parts of this work at international conferences.

Furthermore I like to thank my parents and my brother, who greatly supported me also in this phase of my life.

I owe particular gratefulness to my beautiful wife Corinna, who did not hesitate to marry me in this thrilling period of time, ready to share my ups and downs and to encourage me whenever necessary.

Certificate of Research

This is to certify that, except where specific reference is made, the work described in this thesis is the result of the candidate. Neither this thesis, nor any part of it, is currently being submitted for the award of any degree at any other University.

date, Steffen Körner (Author)

date, Keith J Baker (Director of Studies)

Abstract

Currently process industry faces a paradoxical situation. On the one hand there is the urgent need to optimise the performance of processes by increasing throughput, decreasing operating costs while increasing the product quality. On the other hand there are only few specialists in industry who are able to develop and apply appropriate control strategies for the increasingly complex processes in the process industry. Generally, these specialists work in research and development departments necessitating a considerable amount of time to develop sophisticated solutions for specific processes. However, in the process industry control design and fine-tuning are mostly done by practitioners more than by specialists, directly at the process and in a minimum of time. Within this commissioning phase the process is assembled and set into operation, often with sub-optimally tuned controllers.

Efforts have been undertaken to support these commissioners doing their tasks, and for single-variable processes practically applicable methods have been developed. Nevertheless, for more complex processes the generation of mathematical process models as an appropriate base for control system design still is a major problem in practice.

The subject of this work is the development of a *structured approach to identification techniques for the analysis of industrial processes* that enables industrial users with limited control engineering knowledge to design process models suitable for the design of industrial controllers. This latter aspect has been addressed within the collaborative research project between the University of Glamorgan and the Fachhochschule Hannover, of which the work presented in this thesis is a substantial part.

Therefore, an industrially suitable scheme for computer aided control system design (CACSD) has firstly been developed in agreement with industrial users in order to set the frame for the research project. This scheme has been based on simple block-oriented model structures composed from nonlinear static and linear dynamic characteristics. The scheme is simple in use and intuitive to understand and follow. Therefore, it can be directly applied also by inexperienced engineers, who look for quick and efficient solutions as a basis even for nonlinear controller design.

Beyond this a standardised identification procedure for nonlinear processes has been elaborated in order to provide process models fitting to the CACSD scheme. This standardised identification procedure has been equipped with two improved algorithms. For the approximation of even multi-dimensional static characteristics a capable method has been developed necessitating neither a-priori information nor user interaction. For the identification of discrete-time linear dynamic models a two-step identification method has been improved by a numerically efficient least squares estimator that allows the parallel estimation of a set of model structures, which is evaluated automatically. For the validation of the proposed approach and the developed methods a prototype identification tool has been programmed, which also lays the ground for the integration of the whole CACSD scheme into a block-oriented simulation environment.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Abbreviations

AI	artificial intelligence
AR	autoregressive
ARMA	autoregressive moving average
ARMAX	autoregressive moving average with external input
ARX	autoregressive with external input
CACE	computer aided control engineering
CACSD	computer aided control system design
ELS	extended least squares
FIR	finite impulse response
GLS	generalised least squares
GMA	Gesellschaft für Meß- und Automatisierungstechnik (German measurement and automation society)
GUI	graphical user interface
GWM	generalised weighted mean
ICAC	industrial computer aided control
ICACSD	industrial computer aided control system design
ICAI	industrial computer aided identification
ID block	identification block
I/O	input/output
IV	instrumental variable
LD	linear dynamic
LS	least squares
MA	moving average
MES	model evolution scheme
MIMO	multiple input, multiple output
MIMO-SC	multi-dimensional static characteristic
MISO	multiple input, single output
MPC	model-based predictive control
MRAS	model reference adaptive identification systems
MGWM	modified generalised weighted mean
OE	output error
OEM	output error method
OLE	object linking and embedding
OOP	object oriented programming
PC	personal computer
PCS	process control system
PID	proportional plus integral plus derivative
PRBS	pseudo random binary signal
RBF	radial-basis function
RBS	random binary signal
SC	static characteristic
SISO	single input, single output
SISO-SC	single-dimensional static characteristic

SITB	system identification toolbox
SNIP	standardised nonlinear identification procedure
VDI	Verein Deutscher Ingenieure
XPS	expert system

Latin Symbols

Δx	$\Delta x = x_i - x_{i-1}$
a,b,c,d	parameter vectors
$\mathbf{A}^T$	transpose of the matrix **A**
c	form parameter (constant)
$\mathbf{C}$	augmented covariance matrix
d	discrete dead time: $T_d = d\ T_0$ (integer valued)
e	output error (residual)
$e_w(t)$	control deviation: $e_w(t) = w(t)-y(t)$
$F(q^{-1})$	data prefilter
$f(x)$	function of x
g	weight (coefficient)
$G(s),G(z),G(q^{-1})$	transfer function operator
$\hat{G}(q^{-1})$	estimated transfer function operator
i	index for number of inputs n_x
J	loss function
k	index for number of outputs n_y
k	time variable for discrete time: $t = k\ T_0$ (integer valued)
$\mathbf{L}$	loss function matrix
m	order of numerator polynomial
mv	manipulated variable (actuator variable)
N	number of measurements
n	model order
$n(t),\ n(k)$	coloured noise
n_u	number of inputs (for dynamic models)
n_x	number of inputs (for static characteristics)
n_y	number of outputs
p	polynomial coefficient
p	number of estimated parameters
q^{-1}	backward shift operator: $q^{-1}u(k) = u(k-1)$
$A(q^{-1})$	transfer operator model in q^{-1}
$\mathbf{P}$	parameter matrix
R	ratio
$\mathbf{S}$	augmented information matrix
t	time variable for continuous time
T	time constant
T_0	sampling time
T_d	dead time
$u(t),u(s),u(z),u(k)$	input signal (vector of dimension n_u)
U_0	input steady state (operating point: (U_0, Y_0))

$w(t)$	set point
x	independent variable
$y(t),y(s),y(z),y(k)$	output signal (vector of dimension n_y)
Y_0	output steady state (operating point: (U_0, Y_0))
y_c	controlled variable
y_u	undisturbed output signal
z	Disturbance

Greek Symbols and Conventions

δ	error
θ	parameter vector
$\nu(t)$	white noise
μ	truth value of membership function
ψ	data vector
Ψ	data matrix
ε	value close to zero
τ	time shift in discrete time
^	estimated

1 Introduction

Worldwide competition in the process industry requires processes which are made more profitable by improving quality, increasing throughput, decreasing maintenance effort and other operating costs while maximising profits. Moreover environmental problems demand the best possible use of resources and the minimisation of waste. Conventional standard controllers being widely used in the process industry cannot meet these demands in all cases because of increasingly complex process designs that mostly exhibit nonlinear behaviour and couplings (Hahn and Nöth 1997). Nevertheless, it has been shown in some practical applications that modern process control strategies are capable of tackling even very complex and difficult control tasks quite well (see for example, Gawthrop and Ponton 1996).

Therefore process industry is urged to invest in automation strategies which utilise the wide potential of control. The often applied process control systems, for example, provide the necessary means but their possibilities for the improvement of control strategies are rarely used to full extent, simply because appropriate process models usually do not exist.

Nowadays, several approaches to process modelling are accessible. Since the development of conventional theoretical models proved to be time-consuming and expensive the advantages of experimental identification methods for already built processes are obvious. Although the identified models are only valid in the analysed range and do not necessarily comprise physically relevant parameters they are mostly appropriate for control design. Unfortunately no constructive and systematic method for the experimental determination of suitable process models for industrial engineers has been established so that the problem of '*moving from parameter estimation to system identification*' (Ljung 1991) still exists. According to Zhu (1998) it is an '*astonishing fact that most identification results developed in the last 30 years are not used by industrial control engineers*'. Zhu remarks that too many researchers concentrate on parameter estimation and convergence analysis and that the emerging toolbox approaches of identification software are difficult to use by practical control engineers who do not have academic training in system identification. The reason for this situation may lay in the complexity of identification theory and the resulting difficulties in application, especially for untrained personnel.

1.1 The Need for a New Approach to Industrial Process Identification

The subject of this work is the development of a new structured approach to identification techniques for the analysis of industrial processes, which enables process engineers and process personnel to easily build up even multi-variable nonlinear process models utilising their broad capabilities and the knowledge about the process under investigation.

Especially where control experts are rare - this being the case in many small and medium size companies - this approach can provide a solid base for automated controller design within easily usable simulation and optimisation programs, which already exist. To satisfy these needs it is essential to support this approach by computer aids being aimed at simple application.

Numerous identification programs have been already developed as part of CACSD-(Computer Aided Control System Design) tools. Nevertheless these programs do not address practical design aspects sufficiently, because these programs are mostly of academic origin and meant to serve as test beds providing a comprehensive collection of sophisticated aids for the main CACSD-tasks like process modelling, controller design and in some cases also for the validation at the process (Figure 1-1).

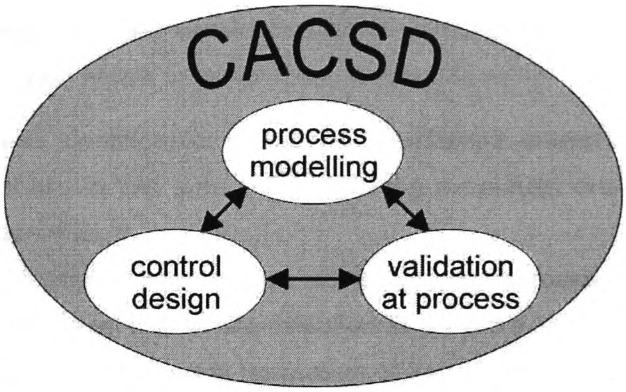


Figure 1-1. CACSD tasks

Graphical user interfaces (GUI) have also been developed for such CACSD and identification tools but these are mostly aimed at the support of expert users and therefore are only of marginal help for uninitiated users, who lack the understanding of the offered methods. This should not imply that all identification programs should be simplified for inexperienced users. Moreover it means that there is still scope for solutions bridging the gap between advanced identification methods and their application in industry.

Consequently a new approach to identification is needed that is tailored to the 'way of thinking' and the capabilities of process engineers as well as the working conditions in industry.

1.2 Contributions of this Thesis

The main aim of this research has been the development of a streamlined approach to identification that concentrates on the determination of sensible models for good controller design. Especially in the case of nonlinear multi-variable systems the variety of possible model structures is virtually endless. Nevertheless non-expert users should also have the possibility to cope with nonlinear multi-variable processes. Therefore, in this work, a specific class of process models and structures has been selected and supported by a standardised identification procedure that is capable of representing a variety of nonlinear multi-variable processes. The new approach centres on solutions for the 'area'-engineer in the process industry and not on the control expert or identification specialist. To validate the new approach a software prototype has been developed for *industrial computer aided identification (consequently named ICAI)*.

1.2.1 Relevance to Industrial Applications

For the development of practically applicable yet progressive approaches it has been especially important to understand the specific constraints of the application of computer aided control system design in industry. Interviews revealed that most process engineers and process personnel in the process industry are often extremely constrained in time and relatively inexperienced in modelling, simulation and control. At first glance this contradicts the observation that process engineers or process personnel in industry have a fairly good knowledge of their processes. Mostly, however, this knowledge cannot be used systematically for traditional control design because it is generally an intuitive and abstract model in the process engineer's mind.

Therefore the new structured approach to process identification considers the different aspects necessary to allow successful industrial application. In this sense the following properties are specifically relevant for the ICAI prototype development:

- The software tool has been designed to be straightforward in use without requiring special training or repeated familiarisation phases. A guided tour of identification is provided that is geared to the user's knowledge. This implies that the software realisation of this identification approach has been created in such a way that the available functionality is task-oriented and intuitive to use in order to clarify each next step during the identification procedure.
- As part of an integrated industrial concept different user levels are provided that differ in complexity and functionality. For example for process personnel a standardised identification procedure is provided that displays the results in the time domain according to the user's understanding. However, control specialists need more degrees of freedom and functions while performing the identification task in order to utilise their whole creative potential. This includes, for example, direct access to frequency and *z*-domain functions and representations.
- The software tool utilises selected identification techniques, which have been tailored to the needs of industrial 'area'-engineers with broad but shallow knowledge in an effective and economic way. This means that the complexity of advanced methods is hidden behind an easily accessible user interface by utilising sensible defaults.

1.2.2 Relevance to Identification Theory

In order to achieve the research objectives as outlined before, it was necessary to access a wide range of different but complementary research fields. Consequently this research has a broad focus and utilises available schemes and methods where possible.

However, with the focus set on industrial application a *new industrial CACSD scheme* has been developed that formalises a widely used approach to multi-variable control system design as it is carried out in the process industry. Additionally it introduces a *standardised CACSD procedure* that is aimed at a specific class of process models. This procedure has set the frame for the elaboration of a *standardised nonlinear identification procedure (SNIP)* that is aimed at easy application, separating the modelling of the static characteristic and the linear dynamic characteristic.

This standardised identification procedure has been equipped with two intensively modified methods. The first identification method combines multiple model least squares with correlation techniques to allow a robust identification of linear dynamic models (Körner and Schumann 1998b) whereas the second one supports the approximation of static characteristics being based on the generalised weighted mean and standardised for simple application. Both methods have been extended to multi-dimensional problems.

1.2.3 Relevance to Future Developments

This research project is part of the collaborative development of an industrial CACSD tool carried out at the University of Glamorgan and the Fachhochschule Hannover. Hence the software concept has been created to be open, extensible and modular to allow the collaborating partners the incorporation of new ideas in the future.

The resulting process models have been designed to be a fruitful base for other CACSD modules. For example the control design tool ICAC (Industrial Computer Aided Control) currently under development at the Fachhochschule Hanover will be capable of incorporating the results coming from the *ICAI* software tool directly. The combined modules *ICAI* and ICAC will have the potential for improved commissioning in the process industry. In the near future these modules will be connected to a commercial process control system[1].

1.3 Organisation of this Thesis

The organisation of this thesis follows a top down approach starting with a broad description of the background that initiated the work and going into detail, where necessary in order to accomplish a prototype realisation of an industrially suitable identification software (Figure 1-2).

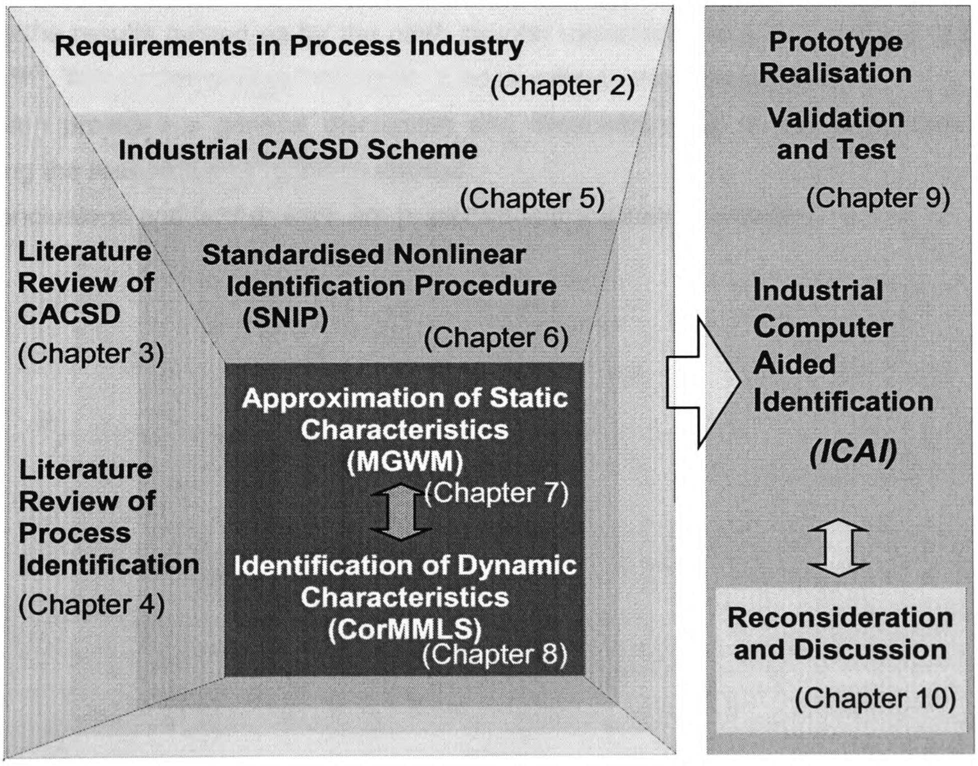


Figure 1-2. Organisation of the thesis

[1] Because of a close cooperation with ABB - Hartmann&Braun the integration will be directed to the proprietary PCS Freelance 2000™

The thesis is organised as follows:

- The *second chapter* is based on interviews with industrial users in the process industry and references to authors working in industry. The possibilities and limitations of CACSD in industrial environments are analysed and the main requirements for industrial CACSD are formulated considering expertise and needs of industrial users.
- The *third chapter* reviews current control design in the process industry and technological developments in the field of CACSD. It also demonstrates the urgent need for system identification tools to be part of CACSD tools, both aimed at industrial needs.
- In the *fourth chapter* current literature about methods and methodologies in identification are critically reviewed with respect to industrial applicability and a substantial amount of the utilised terminology and methods is introduced.
- Based on the foregoing work a concept for an industrial CACSD scheme is proposed in the *fifth chapter.*
- In the *sixth chapter* a scheme for a nonlinear identification procedure is specified, which is used to further improve the ICACSD scheme. The following two chapters introduce modified identification methods that are especially useful for the *SNIP*.
- Hence the *seventh chapter* outlines an effective method for the identification of multi-dimensional static characteristics.
- In the *eighth chapter* an efficient method for the identification of linear dynamic models is provided.
- Based on the results gained so far the *ninth chapter* describes the *ICAI* prototype realised in MATLAB™[2]. This prototype has been also used to validate the new approach.
- *Chapter ten* provides a general discussion and reconsideration of the work done so far illuminating the research methodology applied.
- Finally conclusions and further work are reasoned in the *eleventh chapter.*

[2] MATLAB is a registered trademark of The MathWorks, Inc.

2 Computer Aided Control System Design (CACSD) for Process Industry

"*An integrated environment must support the user's work.*" This is one of the main ideas of Andersson *et al.* (1991), who recommended that the design of CACSD systems needs a substantial amount of "*understanding of the application*". As the new identification approach must be seen in the industrial context, numerous interviews in industry were arranged for this project to clarify the requirements for an industrial approach to identification in the context of CACSD. This chapter presents the results of interviews and discussions at the following companies, where process control systems are frequently utilised:

- o Siemens (Karlsruhe)
- o Hartmann&Braun (Minden)
- o Bayer (Leverkusen)
- o Sensycon (Hanover)
- o Preussen Elektra (Hanover)

It also considers the work of academic and industrial members of the GMA (German Measurement and Automation Society). Naturally, these interviews discussed the engineers' view, the tools at hand, the facilities, the problems and possible extensions.

2.1 Industrial CACSD (ICACSD) for Commissioning and Operation

Nowadays, process control systems (PCSs) are widely used in the process industry. PCSs are very flexible but also complex tools that are used, for example, for control, supervision, visualisation and documentation of various processes in industry (see for example, Bieling 1997, Uecker and Epple 1997, Brownlie *et al.* 1997 and Brucker *et al.* 1997).

Figure 2-1 visualises where CACSD systems can support the planning and use of PCSs. Before a PCS is set into operation three main phases have to be accomplished, namely the concept, design and commissioning phase. Körner and Schumann (1996a) discussed the support of the control tasks within these phases by CACSD systems.

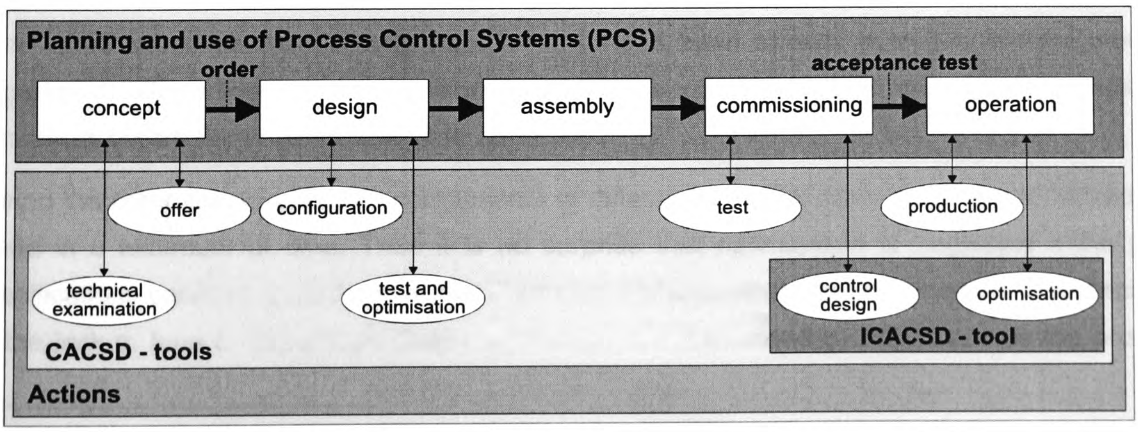


Figure 2-1. CACSD support for process control systems (Körner and Schumann 1996b)

CACSD systems can already provide considerable support for the concept and design phase by clarifying the capabilities or limitations of the control aspects of the automation project before the process is set up. Naturally, it would be desirable to utilise process models that can be used for all phases of the PCS set up, because the additional modelling effort for every phase is hardly accepted in industry. Therefore CACSD systems should be integrated into PCS-concepts, so that information and data can be useable in all phases and redundancies in the data input and storage are avoided. The use of paper based methods to transfer and store information being still being practised in industry should be replaced by efficient computer based concepts. This is long overdue. For this work it will be concentrated on the industrially oriented CACSD support during the commissioning phase, which will be called industrial CACSD (ICACSD) in the following (Figure 2-1).

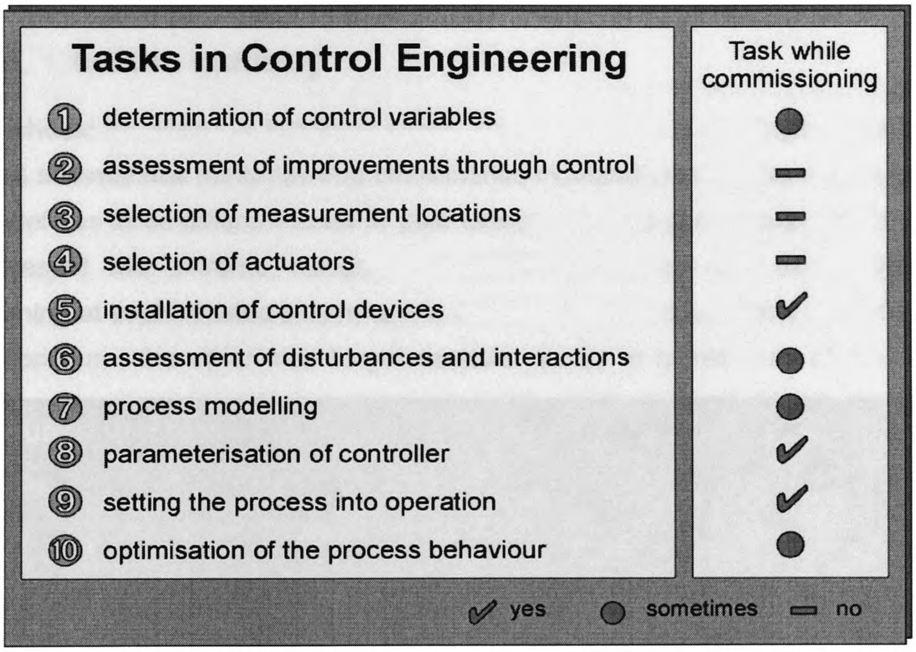


Figure 2-2. Tasks while commissioning

After the assembly of the process the commissioning of the PCS commences. Figure 2-2 shows the main control engineering tasks starting with the determination of control variables and ending up with the optimisation of the control system. The right column shows, which of these tasks are done while commissioning. Mostly steps 1 to 4 have been already solved in the engineering department. Generally, the commissioner installs the control devices and parameterises the controllers before the process is set into operation.

During the commissioning phase components of different suppliers have to be set up, tested and tuned in a minimum of time. Thus it is no surprise that optimisation is neglected if the plant specifications can be reached somehow. Further optimisation potential is not considered because of the lack of time or expertise[3]. Only occasionally are the following undertaken on the process

[3] This has been reported by Alig and Spies from their work at LURGI at the 3rd GMA - FA 5.6 meeting in 1994.

itself for optimisation: determination of control variables, assessment of disturbances and interactions or modelling of the process.

An ICACSD tool could help to parameterise and optimise control concepts in the commissioning phase providing functions for process identification, controller design and validation. Also hardware in-the-loop experiments with a simulated controller working at the process would be beneficial for first controller tests on the process (Section 5.1.3). If process models are available as outlined above, the commissioning could be much safer and quicker and control strategies could be partly evaluated on simulated process parts.

ICACSD support must be strongly oriented towards industrial environment. Therefore the user's needs are reflected in the next section before the ICACSD requirements in the process industry are derived.

2.2 ICACSD Users in Industry

ICACSD tools should be aimed at industrial users. However, there is no single class of industrial user. Figure 2-3 summarises attributes that characterise industrial users, who put the process into operation and defines three different kinds of user categories. The *control expert* and the *process personnel* represent two extreme categories of users. While the first one prefers to work theoretically aiming at sophisticated solutions and utilising advanced functions for each step within the CACSD procedure, the other tries to get useable results in a minimum of time looking for

attribute / user	way of thinking	envisaged solution	utilised function	amount of functionality
control expert	theoretical	sophisticated	advanced, complex	large
area engineer				
process personnel	practical	pragmatic	standardised, preparametrised	small

Figure 2-3. How CACSD-tasks are tackled by different groups of users

pragmatic solutions and utilising standard functions that are automated as far as possible. The *area engineer* is someone, who fits in between these two groups having a broad and also some detailed knowledge. These '*area engineers*' mostly have a chemical or process engineering background. Their main task is the supervision and optimisation of production processes but they are normally also confronted with control engineering issues, although on a less frequent and somewhat 'peripheral' basis.

Most medium and small size companies cannot afford a control department and even bigger companies with a control department have employed only few experts and some more area engineers. Considering that process personnel or area engineers are mostly responsible for the set up and maintenance of a process, it is clear that ICACSD tools should be especially tailored towards this group.

This has also been confirmed by a questionnaire which has been developed within a project in cooperation with this project. This has been sent out to users of control systems from different branches of process industry (Strickrodt 1997).

The main results of this questionnaire and the interviews are:

- Basic SISO control systems are generally manually and experimentally adjusted during process operation.
- More advanced control systems are normally designed after the new process has been in operation because of the lack of process models.
- Multi-variable control design is still rarely employed and MIMO processes are generally treated as separate SISO processes neglecting the increasing complexity of MIMO processes.
- In companies without a special control department, design and tuning of control systems is normally not done systematically and simulation environments are rarely used.
- The main shortcoming of CACSD programs is seen in the lack of modelling support.
- The experience of process personnel (the model in the process engineer's mind) is rarely used for modelling.
- Usually linear approximations are used for modelling although virtually every process in industry exhibits nonlinearities of some kind.

From these results it can be seen that ICACSD systems should be so simple and efficient for the industrial user that manual tuning is not the only alternative.

2.3 Requirements for ICACSD in the process industry

The following requirements for ICACSD tools were elaborated as result of the interviews:

- The ICACSD tool must systematically support the whole CACSD procedure, considering the experience of process personnel.
- The process modelling or identification must become so simple such that it can be intuitively applied even by process personnel inexperienced in identification.
- The resulting process model must be as simple and transparent as possible to allow practical control design.
- Reliable simple routines must be provided even for nonlinear multi-variable processes.
- The results must be presented in the time domain graphically including intuitively understandable characteristic values. Frequency or z-domain graphics should be offered only for experienced users.

If these requirements are met industrial CACSD will provide a valuable alternative to manual tuning by rules of thumb.

2.4 Conclusion for this Chapter

Several practical aspects have to be considered to develop a structured approach to identification suitable for process industry. It has been important for the direction of this research to define a clear frame looking at the industrial application. The results of these efforts have been summarised in the preceding sections, where technological side conditions of CACSD and user requirements have been discussed.

Considering that many processes in the process industry are already supplied with process control systems offering the possibility of implementing various control strategies it is obvious that the unused potential in control is enormous, if suboptimal control strategies are replaced. Therefore the control performance could be considerably improved by ICACSD tools, tailored to industrial needs. Such a tool must also provide a quick and transparent approach to process modelling and identification because this is seen as the main application obstacle of most existing CACSD systems.

3 Review of Computer Aided Control System Design

'*Potentially fruitful interactions between the control and computer science communities in industry and academia are not properly handled today*'. According to Benveniste and Åström (1993) this was one of the main motivations to start the IEEE project "Facing the Challenge of Computer Science in the Industrial Applications of Control" and it impressively shows the need for applied research. Additionally Benveniste and Åström identified the '*significant need for tools that support all aspects of modelling*' being a main precondition to utilise the potential of CACSD in industry. Absolutely, this literature review cannot address all aspects of CACSD systems but it will show the need for practically applicable system identification as part of an industrially oriented CACSD procedure.

3.1 Control Design in the process industry

Nowadays approximately 95% of all industrial controllers in the process industry are PID-based (Nöth and Keuchel 1996). This popularity of PID control is due to its simplicity and transparency. Experienced process personnel tune PID controllers at a single-variable processes based on experience and by '*rules of thumb*' (Schuler 1992). Then the process is indirectly regarded as a second order system which is not explicitly modelled. Another advantage is that rules and tuning aids are provided resulting in a well appreciated performance to cost ratio (Richalet 1993). Well established simple parameterisation techniques like Chien-Hrones-Reswick or Ziegler-Nichols (Piwinger 1975) are mostly based on simple step response experiments and are easily comprehensible. Also robust tuning of PID controllers based on step responses is possible as described by Maffezzoni and Rocco (1995) but this technique is rather advanced and rarely applied. Increased ease of use of PID controllers can be achieved by the implementation of automatic tuning and adaptation techniques being introduced in the eighties. Åström *et al.* (1993) offer a comprehensive insight into this particular subject. They stress that automatic tuning (being mostly based on step response identification as well) is quite helpful for industrial users to build up gain schedules and to initialise adaptive controllers, thus providing the base for the control of nonlinear SISO processes.

However, according to the experiences of Nöth (1998) in the process industry only 20% of the applied PID-controllers work satisfactorily. The remaining 80% do not work properly because in 30% sensors and actuators are wrongly dimensioned, in another 30% the controllers are tuned poorly and the last 20% have other reasons. Mostly major malfunctions could have been avoided, if process and control engineers would closely work together in order to detect process design faults or equipment unsuitable for control, but this is generally not practised.

In the case of nonlinear multi-variable processes the control performance of multiple single-variable controllers is often inadequate, even though this is the standard control design in the process industry. With the tendency to improve the effectiveness of industrial processes, for example through the feedback of resources (as heat and (by-)products) couplings are implemented, such that manual readjustments of poorly designed controllers have often to be made during operation. This reduces product quality and can lead to unsafe process operation. It also requires extra process staff to undertake the regular adjustments (Hahn and Nöth 1997). Especially in the case of process disturbances the poor control performance can lead to critical situations, such that the process personnel have to react quickly to switch off the controller and to manually adjust the process. Only for a few multi-variable processes some more complicated control schemes have been designed, which are widely based on PID control but they often comprise extra linear and also nonlinear elements like multipliers and min/max selection, which make the control system difficult to analyse. Once the control strategy has been sufficiently successful these schemes become standard and are supplemented only if the final performance test on the process fails. However, if these standard control schemes are not available or applicable most companies run their processes with simple sub-optimal control strategies because they fear the enormous cost of modelling (Froese 1995). Of course, such controlled processes have to be frequently supervised in case of disturbances or drifts. The frequent use of sub-optimal control strategies – neglecting the wide range of control methods that modern control theory provides - is generally justified by the effort needed to improve the control performance (Funk 1994).

3.2 Control Design Effort versus Performance

Perne (1995) compared design effort and control performance of different control system design techniques. Figure 3-1 shows that installation and commissioning effort for *'standard PID'* controllers are relatively small. However, if better performance is required other techniques have to be applied.

'Advanced PID' control, for example, is based on the same PID-algorithm but includes simple parameter adaptation, gain scheduling, disturbance cancellation or the possibility of extending the control structure by different linear or nonlinear elements, features that would be extremely costly

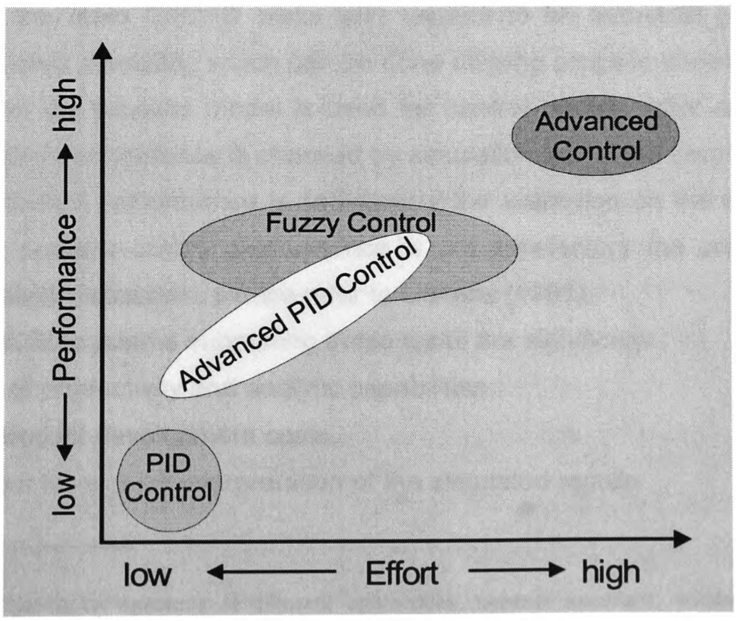


Figure 3-1. Performance and effort of different control techniques

if realised in analogue techniques. Most digital industrial controllers offer these possibilities to increase the control performance, but these extras are rarely used because appropriate process models are missing. Some digital controllers already provide simple SISO identification methods mostly based on aperiodic step responses (Kuhn 1993). With respect to multi-variable systems Rake and Enning (1996) criticise the fact that no single industrial controller or PCS supports the identification of coupled systems and the corresponding PID control system design.

'Advanced control' stands for the high end in control, like state-space control or model based predictive control (MPC), most of which require a good model, high tuning effort and control expertise and are therefore not suitable for everyday industrial control. According to Funk (1994) approximately five percent of all control loops in the process industry are based on advanced control strategies with increasing tendency.

'Fuzzy control' design differs notably from the aforementioned methods incorporating rule-based process knowledge. It can show good performance with little modelling effort, if process knowledge is already available but otherwise it needs much optimisation effort for a performance which possibly could have been gained through 'advanced PID' control more easily. Furthermore advanced and fuzzy control can only be implemented if supported by the control device.

3.3 CACSD Systems

The previous sections clearly demonstrated the need for improved control especially in case of multi-variable processes. Modern CACSD systems provide the means to considerably optimise the control performance. Nevertheless there is still a gap between possible and applied solutions in control. Therefore currently available CACSD systems have been critically examined including an analysis of deficits with respect to industrial application[4].

3.3.1 CACSD System Functions and Potential

Figure 3-2 visualises the main CACSD tasks with respect to an industrial process. The first CACSD task is the *process modelling* which can be done utilising process knowledge and/or data from the process. Then the process model is used for *control design*. After an analysis of the control system the control performance is checked by simulation within the control system model. Only if the simulated control performance is satisfactory the *validation* on the industrial process will take place. If the process control performance is not satisfactory the procedure must be iterated. For more detailed discussion, please refer to Linkens (1993).

The advantages of CACSD systems supporting these tasks are significant:

- Enhancement of productivity and analytic capabilities.
- Reduction of product development costs.
- Graphical output for a quick interpretation of the simulated results.

[4] This was supported by 'hands on'-sessions at different universities, product seminars, conferences and the VDI-Workshop 'Regelungstechnische Programmpakete' ('Control Engineering Programs'); March 1993, Düsseldorf; Chair: Prof. Dr.-Ing. R. Schumann

- Huge storage capability for better storage and context related retrieval of technical information.
- Optimisation of productivity and environmental process behaviour, i.e. better use of the resources.
- Reproducible results with automatic documentation.
- Increase of reliability of the gained results.

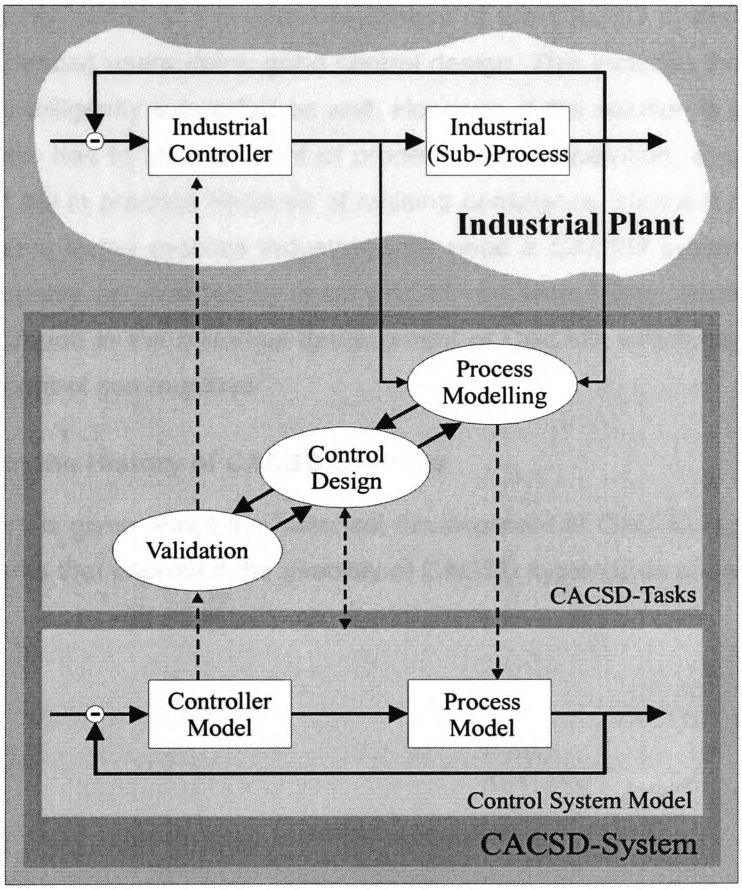


Figure 3-2. CACSD for an industrial process

MacFarlane *et al.* (1989) described the ideal CACSD system so comprehensively that the following passage became a main guideline for the development of CACSD systems. These should enable the designer to:

1) "*build; analyse; browse; search; compare and evaluate; reason and hypothesize; synthesize; design; manipulate and modify; experiment; catalogue, store and retrieve*"

while being:

2) "*easily comprehensible to a single individual; wide in scope; modular with a manageable number of distinct parts; predictable in its behaviour; integrated and coherent in the ways in which its different parts relate; helpful, with quick and efficient access to relevant information; tolerant of errors and supportive in enabling the effect of errors to be easily undone; extensible and adaptable; self documenting.*"

Naturally, not all of these objectives have been achieved yet. While most properties specified in 1) could be realised in many CACSD systems to different degrees, those specified in 2) have not been achieved fully, although some efforts have been undertaken (reviewed in Section 3.3.2).

For efficient use in industry two main prerequisites must be satisfied by CACSD systems:

- o The effort needed to reach a better control performance must be reasonably small.
- o The resulting control technique must be transparent and trustworthy.

The first point is directly aimed at the user-friendliness of the CACSD system and its ability to support even inexperienced users doing good control design. This includes the crucial modelling phase that must be intelligently supported as well. However, if the solution is not transparent for the industrial user who has to take account of proper process operation, even 'good academic control design' might fail in practice because of missing confidence. Hence it must be examined whether industrial users in the process industry really need a CACSD system that supplies an overwhelming functionality as provided by most CACSD systems today. Some explanations for this situation can be found in the historical development of CACSD, which has been carried out mostly by academic control communities.

3.3.2 Milestones in the History of CACSD Systems

Here a short summary is given about the historical development of CACSD highlighting the most important developments that improved the usability of CACSD systems as shown in Figure 3-3.

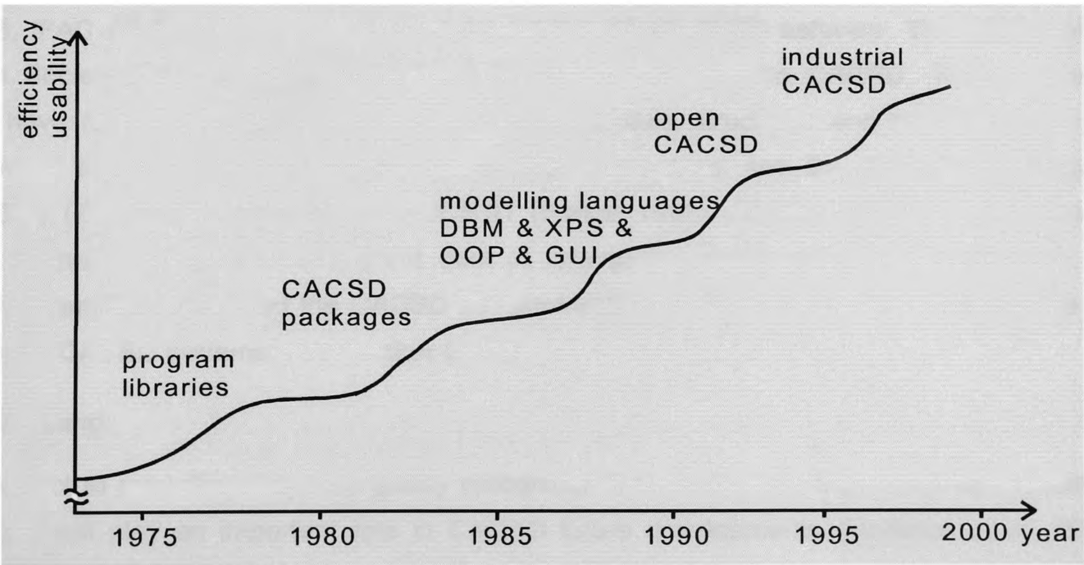


Figure 3-3. Impacts on the CACSD development

Program Libraries

In all probability the automatic synthesis program (ASP) was the first CACSD program designed by Kalman and Englar (1966) to calculate optimal linear state space feed back systems. In the 1970s many control program libraries were developed starting in 1971 with VASP (Variable

Dimension Automatic Synthesis Program). EISPACK (Garbow *et al.* 1977), Linpack (Dongarra *et al.* 1979), LAPACK (Anderson *et al.* 1995) followed. Nearly all libraries were written in FORTRAN, being quite suitable for the mathematically oriented needs of the researchers in the 70's and 80's. Control program libraries worked successfully for testing discussing and exchanging the collected tools with interested scientists.

CACSD Packages

Based on these reliable libraries, CACSD packages like MATLAB, $MATRIX_x$ and Ctrl-C have been developed as interpreter interfaces to the numerical linear algebra libraries. With regard to nonlinear systems $MATRIX_x$ and Ctrl-C were first with the block-oriented modelling extensions SystemBuild and Model-C respectively. Later on MATLAB followed with SIMULAB (now: SIMULINK). For more details on these developments please refer to Taylor *et al.* (1991). However, the inherent suboptimal numerical performance due to interpreted execution of the above mentioned commercial programs and restricted programming semantics triggered the development of heterogeneous packages starting in the mid eighties. These packages are also called CACE (Computer Aided Control Engineering) systems. Packages like CADACS (Schmid 1993) or ANDECS (Grübel 1995) are good examples of this trend comprising functions for identification, system analysis, model transformation, controller synthesis, optimisation and sometimes even online facilities.

Efforts to Handle the CACSD Complexity

In 1986, IFAC set up the working group on Guidelines for CACSD software. The IEEE Control System Society (IEEE-CSS) established a technical committee on CACSD. Their combined efforts have focused on three areas: user interfaces, data structures and algorithms. For an overview of the state of CACSD developments at these days, see the IFAC Proceedings of CADCS'88 (Zhen-Yu 1988) and of CADCS'91 (Barker 1991). As the iterative control system design methodology generates a great deal of information database management tools and expert systems have directed the CACSD development in the late eighties in order to ease the handling of CACSD systems (see Section 4.8.1).

Modelling Languages

With respect to modelling it is now widely recognised that particularly object-oriented modelling languages will play an important role in CACSD future developments (Mattsson *et al.* 1993). Equation based modelling languages like Dymola (Elmqvist 1994), Omola (Andersson 1989) or the currently developed unified modelling language Modelica (Elmqvist *et al.* 1998) provide concepts for model structuring to support multidisciplinary modelling and to facilitate reuse. However, these developments are only interesting for industrial users if appropriate user interfaces are available that allow the use apart from the equation based level. Naturally it would be advantageous for extensive industrial use, if one of these languages would at least become de facto standard.

Object-Oriented Programming

Nowadays one of the most popular ways to realise software, also for CACSD, is the application of object-oriented methodologies. It provides the potential to increase the software productivity and to decrease the maintenance effort while reflecting natural concepts as comprehensively described by Jobling *et al.* (1994). Object-oriented programming (OOP) is useful to realise graphical user interfaces (GUI) as outlined in Section 4.8.2 and also to realise concepts like Open CACSD.

Open CACSD Developments

A topic of special importance is the standardisation and development of 'open software' for CACSD that is easy to interface and maintain. In a panel discussion on Open CACSD several routes to Open CACSD have been discussed (Taylor *et al.* 1994). Herewith DSblock (Otter 1998) was recommended as an independent model bus in order to interface separate modelling and simulation environments (Grübel and Jobling 1994). To realise the Open CACSD paradigms Barker *et al.* (1993) proposed a scheme that heavily borrows ideas from computer aided software engineering standardisation efforts. Schmid and Schumann (1994) presented a data interface standard for CACSD resulting from the work in the GMA (German measurement and automation society). Lately, Barker *et al.* (1996) proposed an object-oriented approach to project management in CACSD incorporating many valuable ideas considering multiple users, hierarchical systems, restrictions on model access, documentation and a supervisory mechanism. They based the model on a hierarchical block schematic diagram, which is described in the same issue by Versamides *et al.* (1996), who introduced a very promising object-oriented information model and suggested a possible architecture for its implementation. Nevertheless they end their paper very realistically: *"However, to actually make use of this (or other similar) models, a wider consensus about fundamental aspects, like that of the CACE information of various levels, must be first achieved"*. This clearly shows that the results of these efforts have no major effect on commercial packages as long as there is no consensus. Thus there has been a quest for de facto standards in CACSD that profit from their generally favourable acceptance at universities and in industry.

3.3.3 MATLAB – De Facto Standard in CACSD

In recent years especially the MATLAB package has been very successful because of its simple and flexible extendibility via C-like m-files (MATLAB-files). Collections of these files are available as MATLAB toolboxes supporting special aspects of scientific problems. Rimvall (1988) detailed the toolbox concept and showed possible extensions. MATLAB has also been so successful compared to other packages like MATRIX$_x$ and Ctrl-C because of the synergetic relationship between the signal processing, system identification and control systems toolbox covering many CACSD problems (Sørli 1994). Hence it is possible to develop packages in this proprietary environment utilising a large and well established software base.

Besides, MATLAB's user interface is especially geared to researchers in control, who prefer to communicate with computers using numerical data models such as state space or polynomials, which are traditionally represented by arrays. Many scientific publications incorporate MATLAB simulation examples in the field of control and researchers often exchange their ideas in form of m-files. MATLAB's dominance in the field of CACSD is also demonstrated within the extended list of control software ELCS (Frederick *et al.* 1992), where all MATLAB compatible programs are summarised in an extra section *'MATLAB-family software packages'* because of their large number and relative importance.

MATLAB also provides the block-oriented simulation environment SIMULINK, which is increasingly accepted in industry and which can handle linear and nonlinear, continuous and discrete models as well as a combination of them (Hortsch and Schlüter 1996). Nevertheless MATLAB is not geared to the Open CACSD efforts as outlined above, but it provides an OLE interface allowing execution of MATLAB functions from external programs and the ability to generate C-code from SIMULINK block diagrams (Schmoll 1997). Because of these means a slight openness is achieved. With respect to Open CACSD it was possible for Barker *et al.* (1996) to link their project manager to SIMULINK. In the context of modelling languages it is possible to link the widely used physical modelling tool Dymola to SIMULINK, for example for use with controllers implemented in SIMULINK.

3.4 Modelling as Crucial Part of the CACSD Procedure

In a study about industrial applications of model based predictive control modelling Richalet (1993) quantified that modelling, experiments and identification require more than half the effort needed for advanced control system design in the process industry. Foss *et al.* (1998) also detected modelling as *'the main bottleneck for the application of advanced control'* and investigated the requirements for computer-based tools supporting the modelling process. They performed a field study of the industrial modelling process, interviewing experienced modellers. However useful as this approach is, it still focuses on the demands of specialists. Takutsu *et al.* (1998) reported on the future needs for the Japanese control industry. They detected that the lack of process analysis is the key factor for failure of automation strategies and that modelling tools are still often seen as the most lacking engineering environments. Therefore it is necessary to investigate strategies that support process modelling, especially for industrial users, as done in the following.

3.5 Process Modelling and Identification

A model is *'a representation of the essential aspects of an existing system (or a system to be constructed) which presents the knowledge of that system in a useable form'* (Eykhoff 1974). For the solution of many scientific and technological problems mathematical process models are required, which imitate the most interesting properties of the process.

The two basic approaches to modelling are *theoretical* and *experimental modelling.*

- *Theoretical modelling* is based on physical laws and offers an excellent insight into the process behaviour. It utilises abstraction, decomposition and aggregation of sub-models describing the process partly to compose a structured process model (Cellier 1991), which also provides information about the internal state of the process. Therefore the evolving process models are called *white or transparent boxes.* Theoretical modelling is supported by the modelling languages like Dymola (Elmqvist 1994), Omola (Mattsson et al. 1993) or Modelica (Elmqvist et al. 1998) and it is already possible during the planning phase such that the influence of process design parameters can already be evaluated in this phase.
- By contrast *experimental modelling* - widely called identification – is concerned with the generation of mathematical process models from data. Naturally, the process must already exist and it must be regarded that the identified process model is only valid within the analysed range and does not necessarily comprise physically relevant parameters. Therefore these inductively derived models are called *black boxes.* Due to the rapid and context related results identification is an efficient method for modelling aimed at control design, especially during commissioning. However, as the process is already built, changes in process design are expensive or even impossible.

While the knowledge of the internal process structure has to be profound for the theoretical modelling in order to yield satisfactory model performance, less knowledge is needed for the application of identification methods to obtain models enabling satisfactory control performance to be achieved.

In this context three types of processes can be distinguished with respect to the knowledge available of the internal process structure (Isermann 1991):

1. *good* knowledge (for example, mechanical and electrical processes)
2. *less* knowledge (for example, processes in power engineering)
3. *poor* knowledge (for example, processes in chemical or process engineering)

Especially for processes of type 2 and 3 it is sensible to apply identification techniques. Furthermore in practical application identification methods can be applied without requiring too much effort and expertise and thus cost.

3.6 Conclusion for this Chapter

The review of CACSD systems has provided a taste of the immense amount of research carried out in this field. However, the transfer of the scientific results into industrial application is rather slow, a situation which can be also found in other scientific fields and which is visualised in Figure 3-4.

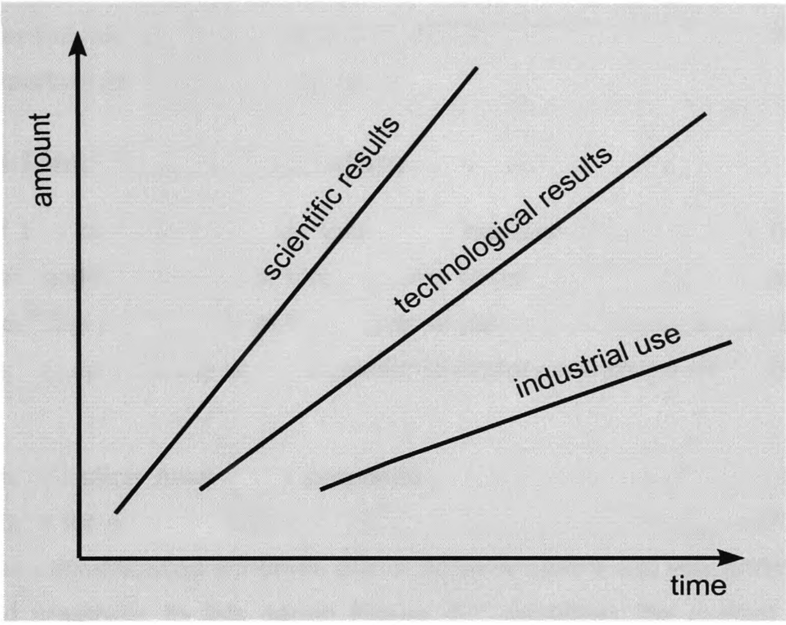


Figure 3-4. The difference between scientific and technological results and their application in industry

Currently industry faces a paradoxical situation:

The more advanced automatic control becomes, the more advanced must be the human operator.

This should not be necessarily the case. Occasionally the CAD support of control issues already is quite good and straightforward in application even for inexperienced users. However, the necessary precondition is an appropriate process model.

As this research copes with processes in the process industry (mostly type 3; Section 3.5) identification is especially useful to derive mathematical models to be used by CACSD tools. Furthermore identification can be even applied by practitioners with broad but shallow process knowledge. Therefore the field of identification from linear SISO up to nonlinear MIMO is reviewed in the following Chapter in order to determine those methods necessary for industrial CACSD.

4 Review of Identification

For various reasons identification is not a foolproof methodology that can be used without user interaction. For example the experiment design must be planned carefully to acquire suitable data, an appropriate model structure must be found and a suitable parameter estimation method must be applied. Besides, the identified process model must be validated against the real process behaviour. This chapter provides an overview of process identification highlighting those aspects that are particularly important for this research project.

4.1 The Process Identification Procedure

In the second half of this century a steadily increasing number of publications have become available covering many aspects of process identification. Unfortunately most publications are not concerned with the industrial applicability of the proposed methods. Instead, simulation has been used to prove the principal applicability under scientific laboratory conditions neglecting real world problems.

Many schemes for identification have been proposed describing the main steps necessary in system identification (see for example, Eykhoff, 1990, Wernstedt 1989, Ljung 1987, Söderström and Stoica 1989). These identification schemes are of iterative nature and rely particularly on the user's experience and creativity. In this sense Figure 4-1 visualises the system identification procedure as it is treated within this research project.

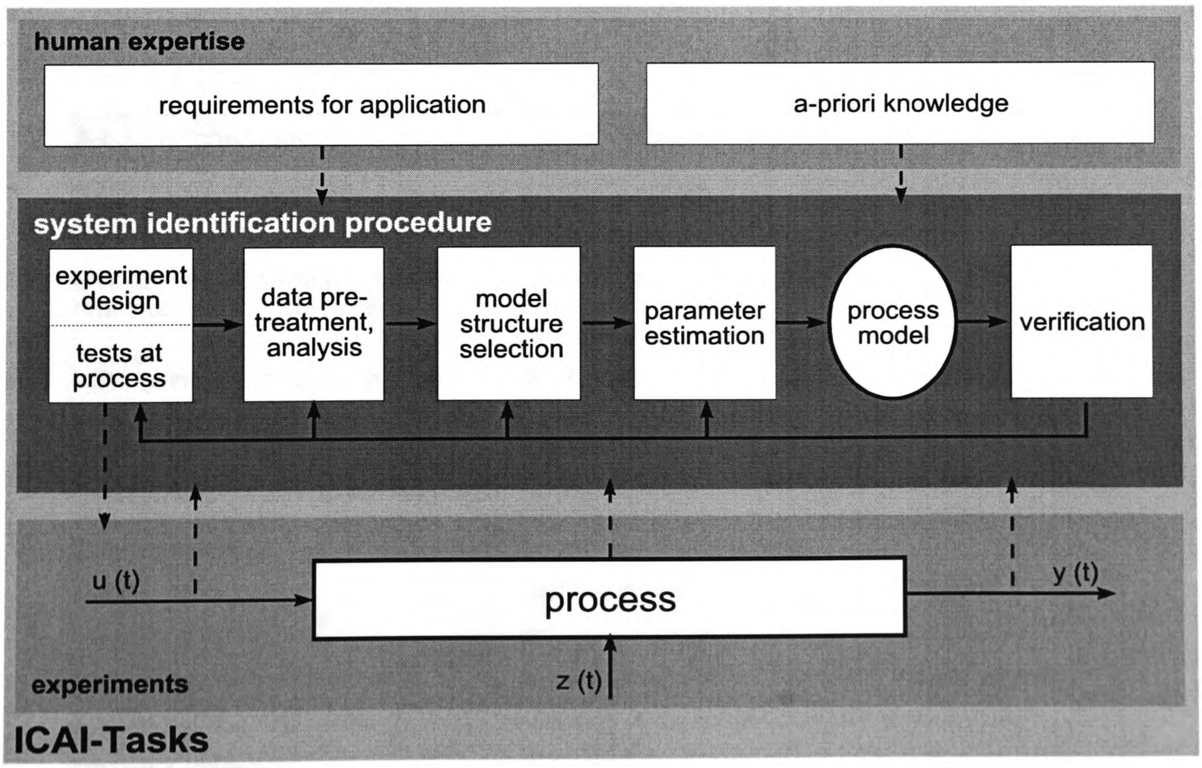


Figure 4-1. System identification procedure

The system identification procedure starts with the *experiment design* which depends on human expertise, the specified aim and available a-priori knowledge. Experiments at the process generate data suitable to describe the process behaviour in the analysed range. After *data pre-treatment* the *structure* of the mathematical process description is selected. Then an appropriate *parameter estimation* method is applied to parameterise the mathematical model. The resulting process model must finally be *verified* with respect to the envisaged aim. If an action within this scheme fails, earlier steps have to be revised until the verification is satisfactory. All these steps rely heavily on the varying side-conditions, as for example required model quality, experimental conditions, process disturbances, operation modes, purpose and modelling.

This section describes the system identification procedure as outlined in Figure 4-1. It details the steps of the system identification procedure and pinpoints where improvement in process identification is necessary to tailor it to industrial needs and lays ground for main ideas of this thesis.

4.1.1 Experiment Design

As the quality of the process model is largely determined by the quality of the measured data, the experiments have to be planned very carefully. Available knowledge of the process – called a-priori knowledge – should be utilised for the design of identification experiments whenever available (Table 4-1). An essential consideration for experiments on industrial processes is the safe operation of the plant. The design of such experiments must include the development of safe working procedures and mechanisms. Nöth and Keuchel (1996) proposed solutions for process control systems and programmable logic controllers.

Table 4-1. Effects of a-priori knowledge

A-Priori Knowledge	Effects on Experiment Design and Identification
sensitivity of the process	critical processes require carefully designed tests (formal tests), whereas some plants accept iterative trial and error tests (informal tests)
dominant time constants and the range of significant dynamics	determination of the sampling interval for each variable to be recorded, input excitation signal and the parameterisation of the data pretreatment
settling time (time to steady state)	determination of the experiment duration
time invariance	influencing experiment design, model structure and selection of on-line or off-line experiments
constraints on input/output variables, including operating, equipment and safety constraints	influencing experiment design with respect to process output and constrain the operating conditions for which the process model is valid
stability	decision if open or closed loop experiment
integrating dynamics	selection of model structure and input signal
maximum possible time delays	provides the range for searching the correct time delay
approximate process order	selection of the model structure and input signal
nonlinear behaviour	decision if linear approximation is sufficient
oscillations, non-minimum phase characteristics	influencing selection of model structure

A-Priori Knowledge

A-priori knowledge may include exact or approximate values for static or dynamic process characteristics, constraints or external disturbances of the process. Table 4-1 summarises the effects of different process characteristics on the experiment design.

Further it is sensible to determine:

- possible *sources of disturbances* in order to avoid these while experimenting or to consider these effects formally.
- the *relative importance of each input*: For example, one unit of change in input number 1 and one unit of change in input number 2 affects the output like *A* to *B*. This measure allows to determine that input with the biggest influence on the output and the relative influence of the other inputs.
- the *relative importance of each output*: For example, *A* units of error in output 1 are as serious as *B* units of error in output 2 with respect to the influence on process performance, product quality, safety etc..

Strickrodt *et al.* (1996) proposed the *MODEL^{ing}* approach, which systematically exploits the practical experience among industrial users and which can help to reduce the number of (pre-)experiments. However, as this approach is not readily applicable simple practically oriented questions or standardised questionnaires can yield valuable a-priori information. It is also possible to determine much of the above information from the process directly by conducting pre-experiments or by evaluating past process data. Naturally a-priori knowledge can be also used for rough validation of the process model (Section 9.4.3).

Pre-Experiments

If sufficient a-priori knowledge cannot be collected simple pre-experiments can provide useful information for better experiment design. Sometimes even 'normal' process changes during operation can be evaluated for this purpose.

Step response identification is probably the most widely used method for process identification in industry due to its simple application and the intuitively understandable picture that is provided about the process behaviour as a reaction to an input step change[5]. Even in the near future step response identification will be the preferred method in industry for the practical evaluation of linear dynamic process characteristics according to Bretthauer (in Bretthauer *et al.* 1998). The reason can be found in the traditional relation between step response identification and the widely applied PID tuning rules (Section 3.1). Therefore it is worthwhile considering advantages and disadvantages of this method. Körner and Schumann (1998a) assessed nine methods for step response identification with respect to the models produced, evaluation problems due to noise, and computer implementation aspects. One especially appealing numerical method has been

[5] The nature of step responses is also of prime concern for the development of several predictive control strategies (see e.g. Moellenhoff and Erickson (1992) and for multivariable processes Zhu (1998)).

proposed by Hankel and Reiner (1992). This method can be perfectly used to determine dominant time constant, settling time, approximate deadtime and signal-to-noise ratio.

Even for integrating processes step response identification methods can be used either applied to the differentiated step response or to the process response to a narrow impulse input change with an impulse width W_i much smaller than the time constant T of the process ($W_i < T/10$ according to Wernstedt 1989).

Input Signal Design

If sufficient a-priori knowledge could be collected, a more advanced input signal superior to step or impulse input change can be designed directed at the desired model and its purpose. As identification results can be only as good as the provided data, identification becomes more a problem of the synthesis of optimal test signals rather than data analysis (Richalet 1993). Of special importance are:

- Selection of the appropriate form of the input signal that provides excitation of all relevant frequencies of the process utilising a-priori knowledge, for example, about sampling time and time constants.
  - the steeper the flanks of the signal the higher the frequencies that are excited.
  - the smaller the breadth of impulses in the signal the higher the frequencies that are excited.
- Determination of the input signal magnitude to produce a reasonable signal-to-noise ratio at the process output utilising the knowledge of the steady-state gains and noise level of the process. The input-signal magnitude should be as big as possible.
- Sufficient length of the experiment is required. If small amplitude test signals are needed that do not affect the product quality long test duration are required to extract signal from noise (Söderström and Stoica 1989). According to Zhu and Backx (1993) the experiment duration should be longer than 10 times the largest time constant in order to get a reliable estimate at low frequencies.
- Identification is secured by persistently exciting input (Ljung 1987).
- In case of closed-loop identification a test signal should be added to the normal set point to ensure identifiability (Söderström and Stoica 1989).
- A suitable sampling frequency lies in the range of ten times of the estimated bandwidth of the process (Ljung 1987). Another empirical method for the determination of the sampling frequency evaluates the rise time of a step response, during which two to four samples should occur.

Three type of test signals are commonly used for identification of linear dynamic models:

- Filtered white noise.
- Pseudo random binary signals (PRBS) (Eykhoff 1974 and Söderström and Stoica 1989).
- Sum of sinusoids.

The filtered white noise signal is persistently exciting over the bandwidth and has a continuous spectrum. The PRBS is persistently exciting when its order (i.e. the period) is higher than the order of the process model. The sum of sinusoids signal should contain a much higher number of sinusoids than the process model order. For more information about input signal design please refer to Godfrey (1993).

Implementation of the Experiment

Careful design and implementation of the experiment are the basis for experiment data with good information suitable for successful identification.

- In order to capture the application-relevant frequency band it is important that the experimental condition resembles the situation for which the model is going to be used. The sampling time T_0 must satisfy Shannon's theorem, which relates to the maximum frequency f_{max} of interest (Lechner and Lohl 1990):

$$T_0 << 0.5/f_{max} \qquad (4\text{-}1)$$

- Additionally an analog anti-aliasing filter should be used during the measurements to avoid aliasing effects. Sometimes the existing instrumentation provides already a low-pass filter to eliminate high frequency noise and disturbances being above f_{max}. The cut-off frequency f_c must be smaller than half the sampling frequency f_0.

$$f_c < 0.5\, f_0 \qquad (4\text{-}2)$$

  For low and medium sized frequencies the anti-aliasing filter should have a constant gain and a phase close to zero in order not to distort the signal unnecessarily. For high frequencies the gain should drop quickly. This will decrease the aliasing effect and may also increase the signal-to-noise ratio. In case the input signal is not in a sampled form held constant over the sampling intervals it is useful to prefilter it as well (Ljung 1987).

- The quantisation accuracy of the utilised A/D (analog to digital) and D/A (digital to analog) converters must be sufficient. The A/D-converter accuracy (Δy) influences directly the measured accuracy of the control variable (Δe). Practically Δy should be much smaller than Δe. Mostly the test signal is converted from digital to analog (D/A conversion). Therefore the D/A-converter affects also the input signal accuracy Δu but here the process gain K_s has to be considered as follows:

$$\Delta u << \Delta e/K_s. \qquad (4\text{-}3)$$

  Similarly actuators and sensors have to satisfy accuracy limits (Isermann 1988).

- Plant operations and interactions by process personnel must be considered because these can badly affect the data, for example manual interaction like adding ingredients, etc..

Furthermore active and passive experiment can be distinguished (Nöth and Keuchel 1996). The active experiment needs perturbation signals as described above whereas the passive experiment uses normal process changes that are suitable for identification. However, also these signals have to be stationary, uncorrelated with disturbances and must sufficiently excite the relevant frequencies for satisfactory identification results.

4.1.2 Data Pre-treatment

Mostly experiment data are not directly suited for parameter estimation, because of:

1. high-frequency disturbances in the data record, above the frequencies of interest
2. drift and offset, low-frequency disturbances
3. outliers, bursts or missing data

It is always advantageous to examine a time history of data. This can often help in devising data pre-treatment schemes to improve the identification result. Specific action can be determined to address each of the above points as follows:

to 1): If it turns out that the sampling interval is unnecessarily short or the anti-aliasing filter is not properly chosen then low pass filtering of input and output is necessary.

to 2): These variations mostly come from external sources that:

- can be modelled if necessary. However, estimating the offset or drift explicitly is unnecessarily complicated and increases the order of the estimation problem (Ljung 1987).
- can be removed by proper data pre-treatment. There are several ways to do so (Isermann 1992). For example differencing the data will push the fit into a high-frequency region unsuitable for many applications. It is better to determine the offset directly from the data and to prefilter the data utilising a high-pass filter, in case of trends, drift or low frequency disturbances.

to 3): These can be detected by manual inspection but also failure-detection algorithms can be used as outlined by Wernstedt (1989).

Another important aspect is the scaling of the signals because in industry inputs and outputs do not necessarily have the same order of magnitude. These are related to physical quantities which, in general, do not have the same dimensions such that the signals with the largest numerical values will automatically get highest weight in the loss function. Normalisation of the signals helps to avoid this effect.

4.1.3 Model Structure Selection

The selection of an appropriate model structure in practice is greatly influenced by the intended use of the model. Therefore the structure of a process model should fit the aim of modelling to a large extent (Söderström and Stoica 1989). The model structure should not require an excessive number of parameters. If the increase of the number of model parameters does not result in significant decrease of the corresponding loss function then the parameter set should be selected that covers the process behaviour with the smallest parameter set.

Generally a compromise between *complexity* and *flexibility* must be found:

- *complexity*. An over-parameterised model structure can lead to unnecessary complicated computations for finding the parameter estimates and for using the estimated model. An underparameterised model may be very inaccurate. As the number of model parameters directly determines the complexity of the calculation simple models are often preferred that compromise between model accuracy and simplicity. This is especially important for real-time applications.
- *flexibility*. The model structure must be flexible enough to represent the system dynamics over the required range of operating conditions. The flexibility of a model set is often related to the number of model parameters and how the parameters enter the model. For example, a finite impulse response (FIR) model (Section 4.3.2) - although it allows unbiased estimates - is not suitable for systems with slow dynamics if the sampling rate is relatively small because of the large number of parameters required. Therefore it is not a flexible model structure for slow processes under specific conditions.

The model structure is strongly related to specific parameter estimation method as explained in Section 4.4.

4.1.4 Parameter Estimation

After the determination of the model structure the model parameters can be estimated utilising a parameter estimation method that fits to the corresponding difference equation. Numerous parameter estimation methods have been developed allowing proper estimation even for poor signal-to-noise ratio. For a detailed discussion see Section 4.4.

4.1.5 Model Verification

In practice, the identified model always only is an approximation to the real process. Therefore the identified model must be verified to see if it is an adequate representation of the actual process. If the model is not acceptable at least parts of the identification procedure must be repeated. For the model verification it is sensible to use a-priori knowledge, engineering insights or physical laws for consistency checks. For very complex processes sub-systems of the process often have to be validated step by step. The verification is best done on a fresh data set.

Normally industrial users have difficulties in interpreting the different loss function values which identification methods provide because these can differ among the methods considerably. Therefore a unified model validation criterion helps the user considerably to assess the model quality. In this sense the output error (OE) is an intuitively understandable model verification criterion, which is also suitable to assess the quality of nonlinear process identification.

The identified model is run in parallel with the process and model output versus actual process output are compared (open-loop test).

$$\delta_{OE} = \frac{\|\hat{\mathbf{y}} - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} = \frac{\sqrt{\sum_{k=1}^{N} (\hat{y}(k) - y(k))^2}}{\sqrt{\sum_{k=1}^{N} y^2(k)}} \qquad (4\text{-}4)$$

The OE rates the normalised quadratic error between process and model output excited by the same input signal. A good model should be a compromise of simple model structure (small order n) and sufficient agreement with the measurements (small loss). Therefore it must be assessed if the improvement of fit is significant when the model order is increased. This is described in Section 8.3.2.

The final validation of the model is purpose-oriented. If the process model is the basis for control design then the model quality is sufficient if the control performance meets or exceeds the control requirements. Otherwise it is necessary to determine whether the control design can be improved or if the identification has to be repeated.

Some causes for model deficiency are:

- The data contain insufficient information due to poor experiments.
- A model structure unsuitable for describing the system behaviour has been used.
- Numerical problems occurred in the estimation procedure.

Many methods for the identification of linear dynamic processes have been developed with the aim to lead to successful identification results, even if the side conditions are not perfect. The practically most relevant identification methods are surveyed in the following and conclusions are drawn.

4.2 Methods for the Identification of Linear Processes

Identification methods can be separated into *nonparametric* and *parametric* methods, which are characterised by the property of the resulting model. A parametric model can be described by a finite-dimensional parameter vector, whereas this is not necessarily the case for nonparametric methods.

4.2.1 Nonparametric Identification Methods

An advantage of using nonparametric identification methods is that it is not necessary to determine a model structure prior to use. Regularly used nonparametric methods for the identification of linear processes are:

- *Transient analysis*. The recorded process response on a step (step response) or impulse (weighting function) input change constitutes the model. Transient analysis is very sensitive to noise but provides some fairly good idea of the process behaviour.

- *Frequency analysis*. The frequency response is determined by recording the output sinusoids' changes in amplitude and phase for different input sinusoids. Frequency analysis requires rather long identification experiments, especially for processes with large time constants.
- *Correlation analysis*. A normalised cross-covariance function between input and output provides an estimate of the weighting function if the input is white noise. Correlation analysis is rather insensitive to additive noise on the output signal (Ljung 1987).
- *Spectral analysis*. The frequency response is estimated for arbitrary inputs by dividing the cross-spectrum between output and input to the input spectrum. For practical application a lag window must be applied leading to limited frequency resolution.

For more details on these methods see for example Isermann (1992). Mostly nonparametric methods can be easily applied but only give moderately accurate models. For higher accuracy parametric models should be utilised (Söderström and Stoica 1989).

4.2.2 Parametric Identification Methods

For the application of parametric identification methods a specific model structure must be assumed. It is beneficial if the model structure can be based on physical insight into the process because appropriate model structures lead to higher accuracy. Two important classes of parametric identification methods are:

- *Table based methods*. From the process response on a non-periodic input signal some characteristic values are determined, for example deadtime, rise time, settling time and so on. Utilising the characteristic values simple parametric models can be determined from tables. However the model quality decreases considerably in case of disturbances (Körner and Schumann 1998a).
- *Parameter estimation methods*. Parameter estimation methods are based on differential or difference equations of arbitrary order and deadtime. Specially designed methods minimise error signals for arbitrary inputs. If properly applied the model quality can be quite good even for low signal-to-noise ratios.

Because of the characteristics of parameter estimation methods advantageous for industrial application these will be considered particularly in the following.

4.3 Parametric Mathematical Models of Dynamic Processes

Parametric mathematical models of dynamic systems can be classified in various ways. Some classes of dynamic systems are especially important:

- *deterministic – stochastic*. For a deterministic model the output can be exactly calculated from the input signal, whereas stochastic models contain random terms, which describe the disturbances.
- *continuous-time – discrete-time*. Continuous-time models describe the relation between input and output for each time point whereas discrete-time models only describe equidistant discrete-time points. The latter is the dominating class discussed here.

- *time-invariant – time-variant*. Time-invariant models are certainly more common than time-variant models, for which special identification methods are needed, especially if not only the parameters but also the model structure varies with time. Only time-invariant models will be discussed here.
- *linear – nonlinear*. A model is linear if the output depends linearly on the input and possible disturbances. In practice nonlinear processes dominate. Only a special class of nonlinear models will be considered in this research (see Section 4.5).
- *single-variable – multi-variable*. In the following all model structures will be introduced for single-variable models that contain a single-input and a single-output (SISO). However, most theory will also hold for multi-variable models. Multi-input, single-output (MISO) and multi-input multi-output (MIMO) models are easily derived from SISO methods (see Section 8.3.3).

In the following mathematical models are described that are relevant to the identification approach presented in this work.

4.3.1 Deterministic Dynamic Models

In the following sections deterministic continuous and discrete-time linear dynamic SISO models are discussed (Figure 4-2).

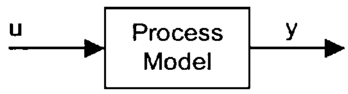


Figure 4-2. Deterministic process model

Deterministic Continuous-Time Dynamic Model

The basic mathematical representation of a dynamic model is a linear differential equation, which describes the linear(ised) dynamic behaviour about the operating point (U_0, Y_0) for the input and output variable $u(t)$ and $y(t)$ respectively:

$$u(t) = U(t) - U_0 \qquad y(t) = Y(t) - Y_0 \tag{4-5}$$

in the form:

$$\alpha_n \cdot \frac{d^n y(t)}{dt^n} + \cdots + \alpha_1 \cdot \frac{dy(t)}{dt} + \alpha_0 \cdot y(t) = \beta_0 \cdot u(t) + \beta_1 \cdot \frac{du(t)}{dt} + \cdots + \beta_m \cdot \frac{d^m y(t)}{dt^m} \tag{4-6}$$

The Laplace-transform of the differential equation becomes in *s*-domain:

$$G(s) = \frac{y(s)}{u(s)} = \frac{\beta_0 + \beta_1 s + \beta_2 s^2 + \cdots + \beta_m s^m}{\alpha_0 + \alpha_1 s + \alpha_2 s^2 + \cdots + \alpha_n s^n} = \frac{B(s)}{A(s)} \tag{4-7}$$

It is beneficial that theoretically derived differential equations are often described by physically relevant parameters (α and β). For digital processing continuous-time models are generally solved by numeric integration. For example the Euler or Runge-Kutta method can be utilised for the solution of differential equations (James 1992). Many methods have been developed also for specific problems like stiff systems. However, it should be appreciated that all numerical solutions are only approximations to the exact solution.

Deterministic Discrete-Time Dynamic Model

If the input is sampled at equidistant discrete-time points $t_k = k\ T_0$ and held constant between the sampling points it can be shown that the sampled input-output behaviour of linear processes can be exactly described by discrete dynamic models. These models can be mathematically derived from continuous-time models utilising the z-transform (Reuter 1991). Discrete models have the same order as the differential equation (Isermann 1988) but their parameters *a* and *b* relate to a specific sampling time T_0. With $z=e^{T_o s}$ a discrete-time dynamic model is yielded, which describes the linear(ised) dynamic behaviour about the operating point (Reuter 1991):

$$G(z) = \frac{y(z)}{u(z)} = \frac{b_1 z^{-1} + b_2 z^{-2} + \cdots + b_n z^{-n}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n}} = \frac{B(z^{-1})}{A(z^{-1})} \qquad (4\text{-}8)$$

If the deadtime $T_d = d\ T_0$ is part of the discrete-time model this includes the integer valued discrete deadtime *d*:

$$G(z) = \frac{y(z)}{u(z)} = \frac{B(z^{-1})}{A(z^{-1})} z^{-d} \qquad (4\text{-}9)$$

In the following the model structures are discussed without this extra term.
Introducing the backward shift operator q^{-1} ($q^{-1}u(k) = u(k-1)$, etc.) for a simplified representation in time domain:

$$G(q^{-1}) = \frac{y(k)}{u(k)} = \frac{B(q^{-1})}{A(q^{-1})} \qquad (4\text{-}10)$$

where:

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_n q^{-n} \qquad (4\text{-}11)$$

$$B(q^{-1}) = 1 + b_1 q^{-1} + \ldots + b_n q^{-n}$$

Utilising the backward shift operation following difference equation can be used for a recursive solution:

$$a_n \cdot y(k-n) + \cdots + a_1 \cdot y(k-1) + y(k) = b_1 \cdot u(k-1) + \cdots + b_n \cdot u(k-n) \qquad (4\text{-}12)$$

Continuous- versus Discrete-Time Models

With respect to the selection of continuous or discrete models it was decided for this project to concentrate on *discrete dynamic identification* techniques. The resulting discrete-time models have no relation to the real process structure, which is possible for continuous-time process models. However, the identification techniques for discrete models and the models themselves are very close to digital computer scenery and digital controllers and therefore efficient with respect to computing effort and quality of results (Kortmann 1988).

If the application requires a specific model, several methods are available to transform continuous to discrete-time models and vice versa[6].

4.3.2 Stochastic Discrete-Time Dynamic Models

A quite general structure of a stochastic discrete time model for the description of a linear finite-order system is depicted in Figure 4-3 (Isermann 1992). The difference to the deterministic discrete-time model is the inclusion of noise via a disturbance filter $G_v(z)$, which transforms the stochastic white noise signal v into the coloured noise n according to this difference equation:

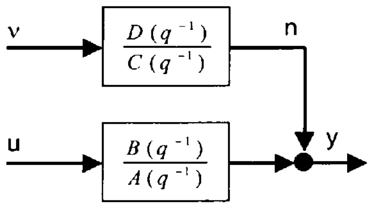


Figure 4-3. Stochastic discrete-time process model

$$G_v(q^{-1}) = \frac{n(k)}{v(k)} = \frac{D(q^{-1})}{C(q^{-1})} = \frac{d_0 + d_1 q^{-1} + d_2 q^{-2} + \cdots + d_n q^{-n}}{1 + c_1 q^{-1} + c_2 q^{-2} + \cdots + c_n q^{-n}} \tag{4-13}$$

The whole model becomes:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})} u(k) + \frac{D(q^{-1})}{C(q^{-1})} v(k) \tag{4-14}$$

There are several special cases relevant for the analysis of stochastic signals.

In case of $C(q^{-1}) = A(q^{-1})$ an ARMAX model is yielded, which is the abbreviation of ARMA (autoregressive moving average) with an exogenous (X) signal being the control variable u:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})} u(k) + \frac{D(q^{-1})}{A(q^{-1})} v(k) \tag{4-15}$$

This ARMAX model is capable of describing any linear finite-order system with stationary disturbances having rational spectral density (Söderström and Stoica 1989). From this model three important models can be derived that solely describe the noise:

- the autoregressive (AR) model is obtained for $D(q^{-1}) = 1$ *and* $B(q^{-1}) = 0$. Then a pure time series without input signal is modelled:

$$y(k) = \frac{1}{A(q^{-1})} v(k) \tag{4-16}$$

- A moving average (MA) model is obtained for $A(q\text{-}1) = 1$ and $B(q\text{-}1) = 0$

$$y(k) = D(q^{-1}) v(k) \tag{4-17}$$

- An autoregressive moving average (ARMA) model is obtained for $B(q\text{-}1) = 0$:

$$y(k) = \frac{D(q^{-1})}{A(q^{-1})} v(k) \tag{4-18}$$

[6] The *ICAI* toolbox described in Chapter 9.2 contains a utility for model conversion (see ICAI help system).

Furthermore special models can be derived from the ARMAX model that utilise the input u.

- A finite impulse response (FIR) model is obtained for $A(q\text{-}1)$ = 1 and $D(q\text{-}1)$ = 1:

$$y(k) = B(q^{-1})\mathrm{u(k)} + v(k) \tag{4-19}$$

  This is also called truncated weighting function.

- The controlled autoregressive (ARX) model is obtained for $D(q\text{-}1)$ = 1:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})}\mathrm{u(k)} + \frac{1}{A(q^{-1})}v(k) \tag{4-20}$$

The ARX model structure is especially important for the least squares estimation because of the special noise filter structure.

A practically important model is output error model (OEM) which is similar to the ARX model but which does not assume a special noise filter:

$$y(k) = \frac{B(q^{-1})}{A(q^{-1})}\mathrm{u(k)} + v(k) \tag{4-21}$$

Here the noise $v(k)$ can be interpreted as the difference between measured output $y(k)$ and the model output $B(q^{-1})/A(q^{-1})u(k)$, which in turn is equivalent to the deterministic model.

4.4 Parametric Identification of Discrete-Time Models

The model parameters can be estimated for a model structure utilising parameter estimation methods. In practice most investigated processes are stochastic because the measurable process output is mostly corrupted by noise. Numerous parameter estimation methods have been developed allowing proper estimation even for poor signal-to-noise ratio. Identification methods can be separated into following categories:

- *recursive (online) – batch (offline).* In recursive identification methods the parameter estimates are computed recursively in time. The counterparts are batch methods, in which all the recorded data are used simultaneously to compute the parameter estimates. Recursive identification methods are especially suited for adaptive systems or in fault-detection to detect changes in the process characteristic. Furthermore these require only limited memory and are appropriate for slightly time-variant processes. However, the starting conditions have to be carefully selected then (Isermann 1992). Furthermore many recursive identification methods are derived as approximations of batch methods and therefore might be less accurate (Söderström and Stoica 1989). In this thesis only batch methods have been investigated because it was not aimed at online parameter estimation or time-invariant processes.
- *deterministic - stochastic.* Identification methods for deterministic models, called deterministic identification methods, try to eliminate the influence of stochastic signals in order to generate a model according to Equation (4-8). By contrast, stochastic identification methods explicitly estimate a specific model of the disturbance signal in addition (Equation (4-13)).

Important aspects of the most widely used deterministic and stochastic identification methods are described in the following.

4.4.1 Deterministic Identification Methods

The most widely used method has been the least-squares (LS) method, which has been developed by Gauss (1809) to calculate the orbits of planets. Nevertheless the direct implementation of the LS method for the determination of deterministic process models suffers from various numerical problems. For example, if the input signal excitation is small, the computer accuracy is low or the model is over-parameterised. In case of process noise the LS method only leads to unbiased estimates for the ARX model structure (Section 4.3.2) being rarely met in practice (Isermann 1992). Therefore extensions of the LS methods are described, namely the instrumental variable and the correlation least squares method, which aim at a reduction of the influence of stochastic disturbances without modelling the disturbance filter G_v.

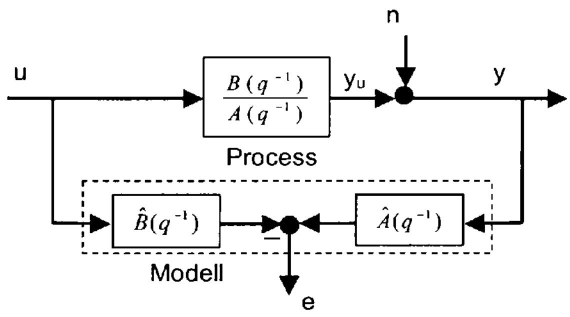


Figure 4-4. LS identification based upon equation error

Least-Squares Estimator (LS)

This method, also called equation error method, has many advantages like model parsimony, robustness, simplicity, fast and global convergence (Niu 1994). The LS estimation is unbiased if the ARX model structure applies. The LS-method can be applied, if the experimental data are in steady state for $k<0$, if model order n, deadtime d and steady states U_0 and Y_0 are known, while n being stationary with $E\{n(k)\}=0$.

Assume that the process behaviour under investigation can be described by the following linear discrete time difference equation $A(q^{-1})=B(q^{-1})u(k)+n(k)$, which can be rewritten as:

$$y(k) + a_1 y(k-1) + ... + a_n y(k-n) = b_1 u(k-1) + ... + b_n u(k-n) + n(k) \tag{4-22}$$

Considering the structure of Figure 4-4 and inserting the estimated parameter vectors $\hat{a}$ and $\hat{b}$ up to $(k-1)$:

$$e(k) = y(k) - \{-\hat{a}_1 y(k-1) - ... - \hat{a}_n y(k-n) + \hat{b}_1 u(k-1) + ... + \hat{b}_n u(k-n)\} \tag{4-23}$$

The equation error *e(k)*, also called residual, can be interpreted as the new observation minus the one-step-ahead prediction. Introducing the data vector Ψ and the parameter vector Θ:

$$\psi(k) = [-y(k-1), -y(k-2), \ldots, -y(k-n), u(k-1), u(k-2), \ldots, u(k-n)]^T \tag{4-24}$$

$$\hat{\theta} = [\hat{a}_1, \ldots, \hat{a}_n, \hat{b}_1, \ldots, \hat{b}_n]^T \tag{4-25}$$

For $k = m, \ldots, m+N$, $N+1$ equations evolve as:

$$\mathbf{e}(k) = \mathbf{y}(k) - \psi^{\mathrm{T}}(k)\hat{\theta} \tag{4-26}$$

The purpose of the parameter estimation is to find the best estimate of the parameter vector $\hat{\theta}$ that minimises a specific loss function. Different loss functions lead to different methods. The least-squares estimate is defined as the vector $\hat{\theta}$ that minimises following loss function *J*:

$$J(k, \hat{\theta}) = \sum_{k=m}^{m+N} \mathbf{e}^2(k) = \sum_{k=m}^{m+N} \left[\mathbf{y}(k) - \psi^{\mathrm{T}}(k)\hat{\theta}\right]^2 \tag{4-27}$$

For this loss function the unique minimum point is given by:

$$\hat{\theta} = \left[\Psi^{\mathrm{T}}\Psi\right]^{-1} \Psi^{\mathrm{T}}\mathbf{y} \tag{4-28}$$

$$\text{with } \Psi = \begin{bmatrix} \psi^T(n) \\ \psi^T(n+1) \\ \vdots \\ \psi^T(n+N) \end{bmatrix} \text{ and } \mathbf{y} = \begin{bmatrix} y(n) \\ y(n+1) \\ \vdots \\ y(n+N) \end{bmatrix}$$

A solution can be only guaranteed if the process is stable and the information matrix $\Psi^{\mathrm{T}}\Psi$ is positive definite ($\det(\Psi^{\mathrm{T}}\Psi)>0$). A good condition of the information matrix can be reached if :

- the sampling interval is not too small.
- the signals are normalised.
- the input signal is persistently exciting.

A comprehensive discussion of the LS estimator and accompanying proofs are given by Isermann (1992) or Ljung (1987). The numerical efficiency can be drastically increased by different mathematical operations. For example the orthogonal triangularisation (the QR method) is a sound way to compute least squares estimates (Söderström and Stoica 1989).

Instrumental Variable Method (IV)

The instrumental variable (IV) method is a modification of the LS estimator producing consistent estimates. As the disturbance filter is not modelled it is not necessary to assume a specific disturbance filter as well. Equation (4-27) is multiplied by a transposed instrumental variable W^T.

$$\mathbf{W}^T\mathbf{e} = \mathbf{W}^T\mathbf{y} - \mathbf{W}^T\Psi^{\mathrm{T}}\hat{\theta} \tag{4-29}$$

The problem is to find an instrumental variable with following probabilities p:

$$p\lim_{N\to\infty}\mathbf{W}^T\mathbf{e}=0 \quad \text{and} \quad p\lim_{N\to\infty}\mathbf{W}^T\mathbf{\Psi}^\mathbf{T} \text{ positive definite,}$$

such that the parameters can be determined by:

$$\hat{\boldsymbol{\theta}}=\left[\mathbf{W}^T\mathbf{\Psi}^\mathbf{T}\right]^{-1}\mathbf{W}^T\mathbf{y} \tag{4-30}$$

The instrumental variable must be uncorrelated with the disturbance *n(k)* and it should be well correlated with *u(k)* and $y_u(k)$. The convergence heavily relies on the instrumental variable. It has been shown that the application of this method requires an iterative search for the instrumental variable until the estimated parameters do not change any more (Isermann 1992).

Two-step LS Method with Correlation Analysis (CorLS)

A further method for the determination of deterministic models in the presence of noise is the correlation least squares (CorLS) method, which is especially appealing because the correlation analysis reduces in a first step the influence of stochastic disturbances and can provide process insight. It can also considerably reduce the data vector for the second step, which is the LS estimation (Körner and Schumann 1998b). For a finite number of samples the correlation functions can be estimated by (Isermann *et al.* 1974):

$$\begin{aligned}\hat{\phi}_{uy}(\tau)&=\frac{1}{N+1}\sum_{k=0}^{N}u(k-\tau)y(k)\\ \hat{\phi}_{uu}(\tau)&=\frac{1}{N+1}\sum_{k=0}^{N}u(k-\tau)u(k)\end{aligned} \tag{4-31}$$

If the difference equation (4-22) of the process model is multiplied by $u(k-\tau)$ and summed up over time the following equation for the correlation function estimates arises:

$$\hat{\phi}_{uy}(\tau)+a_1\hat{\phi}_{uy}(\tau-1)+\ldots+a_n\hat{\phi}_{uy}(\tau-n)=b_1\hat{\phi}_{uu}(\tau-1)+\ldots+b_n\hat{\phi}_{uu}(\tau-n)+\hat{\phi}_{un}(\tau)\ , \tag{4-32}$$

where the last term represents the estimated correlation of input signal and noise. According to the assumption that the input signal u is independent from the noise signal n the following holds:

$$\lim_{N\to\infty}\hat{\phi}_{un}(\tau)=0 \tag{4-33}$$

This means that the influence of the corresponding estimated cross-correlation function in Equation (4-33) disappears with increasing N. It is now straightforward to determine the model parameters utilising the LS method:

$$\varphi(\tau)=[-\hat{\phi}_{uy}(\tau-1),\ldots,-\hat{\phi}_{uy}(\tau-n),\hat{\phi}_{uu}(\tau-1),\ldots,\hat{\phi}_{uu}(\tau-n)]^T \tag{4-34}$$

In this context correlation can be interpreted as a special prefiltering technique that removes the influence of zero mean disturbances. More details are discussed in Chapter 8.

Other Deterministic Identification Methods

There are also other methods like the output error method (OEM), which is nonlinear-in-the parameters and therefore has no direct solution. Therefore optimisation methods are required to calculate the parameter vector. The OEM is widely used because the identification procedure is not affected by the noise as long as it is uncorrelated with the system input.

Another method that has gained relative importance in the context of adaptive control is model reference adaptive identification systems (MRAS). These are normally based on the output error that is minimised by tuning the parameter vector. A literature review about MRAS methods is provided by Eykhoff (1974). Many optimisation techniques can be utilised but especially suitable for large parameter vectors are generic algorithms (Kahlert 1995b).

4.4.2 Stochastic Identification Methods

Generally, stochastic identification methods require more measurements than the LS method because the parameters of the disturbance filter G_υ are also estimated. Examples for stochastic identification methods are the generalised least squares (GLS) and the extended least squares (ELS) method, which are based on the LS method.

Generalised Least Squares (GLS)

The idea of the GLS is to replace the possibly correlated error signal *e(k)* of Figure 4-4 by a correlated coloured noise signal $\xi(k)$ that is gained from the uncorrelated error signal $e'(k)$ by:

$$\xi(k) = \frac{1}{F(q^{-1})} e'(k) \tag{4-35}$$

such that:

$$\xi(k) = A(q^{-1})y(k) - B(q^{-1})\mathrm{u(k)} \tag{4-36}$$

However, as the operator $F(q^{-1})$ is not known an iterative procedure must be applied (Isermann 1992). It can be shown that the estimate is only unbiased if:

$$G_v(q^{-1}) = \frac{\mathrm{n}(k)}{v(k)} = \frac{D(q^{-1})}{C(q^{-1})} = \frac{1}{A(q^{-1})F(q^{-1})} \tag{4-37}$$

The computational effort is much higher than for the standard LS and the estimates are only consistent if the disturbance filter has the form of (4-37).

Extended Least Squares (ELS)

In contrast to the GLS the ELS assumes a general disturbance filter to allow a correlated *e(k)*:

$$\xi(k) = \frac{D(q^{-1})}{C(q^{-1})} e'(k) \tag{4-38}$$

The most general realisation is the *extended matrix method* (Eykhoff 1974), where data and parameter vector are extended with p being the order of the disturbance filter:

$$\psi(k) = [-y(k-1),\ldots,-y(k-n),u(k-1),\ldots,u(k-n), \\ -\xi(k-1),\ldots,-\xi(k-n),e'(k-1),\ldots,e'(k-n)]^T \tag{4-39}$$

$$\hat{\theta} = [\hat{a}_1,\ldots,\hat{a}_n,\hat{b}_1,\ldots,\hat{b}_n,\hat{c}_1,\ldots,\hat{c}_p,\hat{d}_1,\ldots,\hat{d}_p]^T \tag{4-40}$$

Here the signals $\xi(k)$ and $e'(k)$ are not known and have to be iteratively determined from Equations (4-38) and (4-36). Naturally, this requires considerable computational effort and for a large dimension of the parameter vector the method converges quite slowly, if at all.

Other Stochastic Identification Methods

Alternatively, the maximum likelihood (ML) method is an efficient method (Isermann 1992). It assumes a normal distribution of the error signal such that a loss function evolves, that can be minimised by optimisation methods. However, it can be also utilised for nonlinear parametric model structures (Jategaonkar 1985). Because of the large number of parameters required to estimate the starting parameters have to be carefully selected.

Prediction error methods (PEM) are another generalisation of the LS method (Ljung 1987). The parameter estimate is determined as the minimising vector of a suitable scalar-valued function of the sample covariance matrix of the prediction errors. The user has to choose the model structure, a suitable predictor, being a type of filter, and a criterion to assess the performance of the predictor. Söderström and Stoica (1989) showed that the PEM for an ARMAX model can be interpreted as a ML method with quadratic loss function and that the OEM can be seen as a special realisation of the (PEM). Naturally numerical optimisation is applied in all cases.

4.4.3 Comparison of Different Identification Methods

In Table 4-2 the properties of the most important methods for batch system identification have been summarised. Nevertheless most properties are also valid for the recursive methods. This table is mainly based on simulative experiments, a discussion by Wernstedt (1989) and results gained by Isermann (1992).

It can be seen that stochastic identification methods in particular require much computer power. However, the application of stochastic identification methods is only useful if it is aimed at stochastic control requiring the disturbance filter (for example minimum-variance-controller, Isermann 1988). Considering that the disturbance filter is not necessary for the design of industrial PID controllers the CorLS seems to be a superior method. Also the standard LS method is inviting because it is comparable or even better than all other methods for small data sets. According to Rosenthal (1985) the LS estimation is successful for noise-to-signal ratios up to 20% and therefore a good alternative for practical identification. Especially if it is aimed at regulator design some bias is generally acceptable because the regulator should make the closed-loop system insensitive to parameter variations in the open loop part (Söderström and Stoica 1989).

Table 4-2. Comparison of different identification methods

property / method	consistent estimate with disturbance filter	estimation of disturbance filter	linear-in-the parameters	computational effort	necessary a-priori knowledge	negative characteristic	positive characteristic
LS	$\frac{1}{A(q^{-1})}$	no	yes	+	no	o generally biased estimates o sensitive if Y_0 unknown o poor results for noisy systems	o good results for few samples[1] o reliable convergence
IV	$\frac{D(q^{-1})}{C(q^{-1})}$	no	yes	o	no	o not sensible for closed loop o convergence depends on IV o iterative procedure	o good for various disturbances o insensitive for Y_0 if *mean(u)*=0
CorLS	$\frac{D(q^{-1})}{C(q^{-1})}$	no	yes	+	number of cor-relation values	o two-step method	o good for various disturbances o insensitive for Y_0 if *mean(u)*=0 o simple order and delay deter-mination from intermediate model o simple verification o quick convergence
OEM	$\frac{D(q^{-1})}{A(q^{-1})}$	no	no	-	no	o local minima possible	o good results if disturbance filter applies
GLS	$\frac{1}{A(q^{-1})\cdot F(q^{-1})}$	yes	yes	-	filter order	o biased estimates quite possible	o bias smaller then LS-bias
ELS	$\frac{D(q^{-1})}{C(q^{-1})}$	yes	yes	-	filter order	o problematic convergence possible	o good results because of general disturbance filter
ML PEM	$\frac{D(q^{-1})}{A(q^{-1})}$	yes	no	-	depends on opti-misation method	o optimal properties only if n is normally distributed o local minima possible	o good results if disturbance filter applies o flexible process structure
	legend (computational effort):	+: few	o: medium	-: much			

[1]in case of small data samples the results of consistent methods are not superior

4.5 Methods for the Identification of Nonlinear Processes

Most real world processes do not allow the globalisation of local results as it is for linear systems, in other words, the linearity principle cannot be applied. These nonlinear systems may be stable, unstable, oscillatory or even chaotic. Mostly their stability is difficult to analyse (James 1992). For example, many physical devices have nonlinear characteristics outside a limited linear range. Chemical processes contain valves that saturate, connecting lines whose time delays vary with flow rate, reacting mixtures which obey a power law, and separation units being very sensitive to input changes and disturbances. For most chemical processes understanding the nonlinear characteristic is important for the control system design (Eskinat *et al.* 1991).

Various mathematical descriptions and concepts exist to handle nonlinear effects due to the manifold occurrence. Haber and Keviczky (1978) summarised conventional nonlinear model structures in a comprehensive table with a rather practical view. They pointed out that exact mathematical solutions for nonlinear systems are rarely possible, because it is generally difficult to select the right structure regarding the system properties under investigation and the envisaged purpose.

4.5.1 Model Structures for Nonlinear Processes

In a survey about the identification of nonlinear systems Billings (1980) points out that Volterra-series (functional series expansions) provide an adequate representation for a wide class of nonlinear systems, while it can be solved with standard LS methods (Section 4.4). It's discrete time realisation is (Isermann 1992):

$$y(k) = g_0 + \sum_{\tau_1=0}^{k} g_1(\tau_1)u(k-\tau_1) + \sum_{\tau_1=0}^{k}\sum_{\tau_2=0}^{k} g_2(\tau_1,\tau_2)u(k-\tau_1)u(k-\tau_2)$$
$$+\sum\sum\sum g_3(\tau_1,\tau_2,\tau_2)\cdots \tag{4-41}$$

Obviously, the first two terms characterise the linear behaviour in the form of a constant plus the weighting sequence. The Volterra-series is a nonparametric model and requires no determination of the model structure but often several hundred parameters to characterise even simple nonlinear systems (Leontaritis and Billings 1985). The excessive computational effort required for parameter estimation, the difficulty of interpreting the results and the necessity to use special input signals are disadvantages that hinder practical application. To increase its applicability the Volterra-series should be restricted to few kernels as outlined by (Lachmann 1983) but then it's generality is lost. Alternatively, automatic structure selection can be used to reduce the number of estimated parameters. This can be based on different approaches like the heuristically working group method of data handling (GMDH, Ivakhnenko 1968) or on statistical evaluation (Kortmann 1988). However, this is based on a reduction of the complete model and the resulting mathematical descriptions are still quite complex.

A unified survey about nonlinear dynamic black-box modelling is offered by Sjöberg *et al.* (1995), who also include model structures based on neural networks, wavelet networks and fuzzy methods. They conclude that physical insight should be used whenever possible for the identification of complex nonlinear models. Only if no physical insight is available black-box approaches should be used. A systematic approach to the incorporation of a-priori knowledge is developed by Johansen and Foss (1997), who decompose the whole operating range of complex systems into a number of operating regimes, which can be represented by simple subsystems. Nevertheless the approaches mentioned require much user interaction and lead to parameter-rich models, which can cause difficulties in the context of industrial PID control.

The number of parameters in nonlinear process models can be considerably reduced, if only block-oriented nonlinear system models are considered. In this field Chen (1995) provides an overview about structural classification and parameter estimation techniques that are based on the functional series expansions. He points out that the biggest problem is still the selection of the right model structure. To increase the applicability of the block-oriented approach Marenbach and Bettenhausen (1995) proposed a heuristic way to produce parametric block-oriented models utilising generic algorithms, which compose the model structures based on a model block library. The model-blocks are parameterised according to an output error criterion within this optimisation procedure. However, this approach needs much a-priori knowledge in the form of a model block library and it is still rather time-consuming and unreliable even for small problems because the fitting of the measurement data to varying model structures is not trivial with respect to the optimisation issue.

4.5.2 Simple Block-Oriented Nonlinear Models

Often it is sufficient to describe a block-oriented nonlinear dynamic process by a static characteristic in series with a linear dynamic characteristic. This is also supported by Heiss (1995), who investigated the applications of static characteristics for industrial control. He concluded that in practice nonlinear systems are mainly described by their static characteristic, which can be modelled differently according to the purpose (see also Chapter 7).

If the linear dynamic block follows the nonlinear static block this is called a simplified Hammerstein-model. If the nonlinear follows the linear block then a simplified Wiener-model is yielded. These models are often applicable in industrial environments and can directly be used within block-oriented simulation environments. Naturally, these standard structures are not able to cover all problems but they should be applicable to the majority of them.

Simplified Hammerstein-Model

Generally, simplified Hammerstein-models have a memoryless static characteristic in front of the linear dynamics (Figure 4-5). Hammerstein-models are usually suitable, for example for plants with a nonlinear actuator with neglectable small dynamics but also for processes as complex as distillation columns and heat exchangers as demonstrated by Eskinat *et al.* (1991).

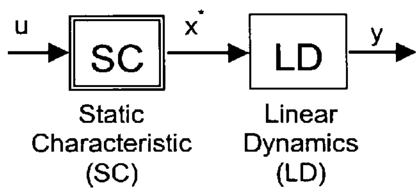


Figure 4-5. Hammerstein-model structure

When it comes to Hammerstein-models literature provides numerous identification methods for the combined identification of a polynomial static characteristic and a linear dynamic. The model structure represents a special case of the Volterra-series (Isermann 1992). The polynomial is described by:

$$x^*(k) = r_0 + r_1 u(k) + r_2 u^2(k) + \ldots + r_p u^p(k) = \sum_{i=0}^{p} r_i u^i(k) \tag{4-42}$$

with the linear part:

$$A(q^{-1})y(k) = B^*(q^{-1})x^*(k) \tag{4-43}$$

Therefore the data vector for the direct estimation becomes:

$$\Psi^T(k) = [-y(k-1)\ldots - y(k-n), u^1(k-1)\ldots u^1(k-n),\ldots, u^p(k-1)\ldots u^p(k-n)] \tag{4-44}$$

Narendra and Gallmann (1966) proposed a first iterative identification method for Hammerstein-models, which firstly approximates a linear model. This is utilised to calculate the static characteristic, from which a new output is calculated being used to refine the linear model. This iteration is repeated until the parameters converge. An early batch method was developed by Chang and Luus (1971) and Stoica and Söderström (1982) developed a consistent IV method for the identification of Hammerstein-models. However, a necessary condition is that the model is not over-parameterised and that the input signal is persistently exciting. The proper order selection of the polynomial is especially important because of oscillating effects that may occur (Zi-Qiang 1993). Correlation analysis was used by Billings and Fakhouri (1978) to decouple the identification of the linear subsystems from the identification of the nonlinear elements. They estimate a nonparametric model, gaining FIR coefficients for the linear part and a set of data points for the nonlinear part. However, they mention that this method can only be applied for white noise input.

Simplified Wiener-Model

Simplified Wiener-models also find their application in chemical process industry, for example to model pH-neutralisation (Norquay 1996). However, there does not exist as much literature as for the Hammerstein-models, probably because the estimation is nonlinear-in-the-parameters and iterative methods are required. The simple Wiener-model is described by a polynomial:

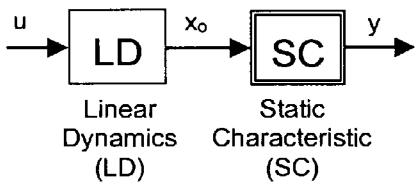


Figure 4-6. Wiener-model structure

$$y(k) = r = \sum_{i=0}^{p} r_i x_o^{\ i}(k) \tag{4-45}$$

with the linear part:

$$A(q^{-1})x_0(k) = B(q^{-1})\mathrm{u(k)} \tag{4-46}$$

For parameter estimation different methods can be used. For example Goldberg and Durling (1971) proposed an iterative search algorithm but it's convergence is not guaranteed. Correlation analysis was used by Billings and Fakhouri (1977) resulting in a nonparametric impulse response model. The model reference technique developed by Parjunen (1982) is applicable to online identification but the parameters converge very slowly. Kortmann and Unbehauen (1987) presented a recursive PEM for the simplified Wiener-models, however it can only be applied if a-priori knowledge is available to determine the right model order of the linear dynamics.

Wiener-Hammerstein-models

A Wiener-Hammerstein-model, sometimes called G-model, consists of a static characteristic lying in between two linear dynamics (Figure 4-7). The identification algorithm of Billings and Fakhouri (1978) also covers this kind of model. However, the

Figure 4-7. Wiener-Hammerstein-model structure

restrictions on the input sequence needed to satisfy the separability condition hinder practical application. Boutayeb and Darouach (1995) proposed a more efficient recursive algorithm based on a weighted ELS algorithm but this needs proper initialisation.

For Hammerstein-Wiener-models that consist of a linear dynamic surrounded by two static characteristics Bai (1998) proposed an *'optimal two-stage identification algorithm'*, which combines a RLS algorithm with a singular value decomposition. However, this algorithm requires special weighting matrices for arbitrary noise at the process output and Bai mentions that it is still not clear how these matrices can be found.

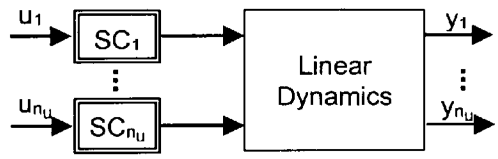


Figure 4-8. Hammerstein-model with separate single-dimensional static characteristics (SISO-SCs)

Multi-variable Wiener- and Hammerstein-Models

Several realisations for MISO Hammerstein-models exist. Mostly a *p*-canonic structure is assumed for linear dynamics as shown in Figure 4-9.

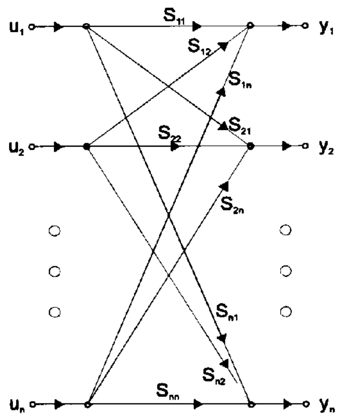


Figure 4-9. Signal flow of p-canonic structure

Boutayeb *et al.* (1993) propose one-dimensional polynomials for each input (Figure 4-8), whereas Ayoubi (1998) proposes a more general approach approximating an n_u-dimensional polynomial, where n_u is the number of inputs (Figure 4-10). He yielded good results for the application to a loading process of a diesel engine. Al-Duwaish and Nazmul Karim (1997) model the multi-dimensional static characteristic as a multilayer feedforward neural network such that a variety of nonlinearities can be modelled. Naturally, more parameters have to be estimated for multi-dimensional static characteristics with the possibly accompanying estimation problems and it is not guaranteed that this model can be used for industrial control system design.

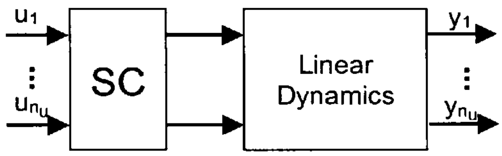


Figure 4-10. Hammerstein-model with multi-dimensional static characteristics (MIMO-SC)

Input Signals

Another important aspect is the design of a suitable persistently exciting input signal covering the whole working range under investigation. Pseudo random binary signals are not adequate for the excitation of nonlinear processes. These would lead to one simple gain instead of a static characteristic if a Hammerstein-model structure applies, because only two points of the static characteristic would be excited.

According to Kortmann (1988) the best input signal for the determination of nonlinearities mostly is a sampled white noise with equally distributed amplitude spectrum. This signal excites the whole frequency range of the process utilising different persistently exciting amplitudes. A dilemma of this signal is that there is no qualitative difference to the disturbance signal. Another appropriate signal is the multilevel PRS (Kurth 1995). For more details about the design of input signals please refer to Wernstedt (1989) or Rosenthal (1985).

4.6 Alternative Identification Procedures for Hammerstein- and Wiener-Models

Most of the identification methods described in the preceding sections focused on polynomial static characteristics. However, already Atkinson and Williams (1978) questioned the *'practical value in the use of polynomial representations of no memory nonlinear elements'*. Although they *'feel'* that Wiener and Hammerstein structures are appropriate for many chemical and other industrial processes, they stress that polynomials are not appropriate especially for discontinuous nonlinearities, like saturations or dead zones. The polynomial order becomes too high and the risk of oscillations considerably increases. Alternatively two-step procedures can be applied that determine static characteristic and linear dynamic separately in two steps (Table 4-3).

Table 4-3. Two-step procedures for Wiener- and Hammerstein-modelling

name of two-step procedure	first step	second step
SC-LD procedure	determination of static characteristic (SC)	determination of linear dynamic (LD)
LD-SC procedure	determination of linear dynamic (LD)	determination of static characteristic (SC)

Although single step procedures are more direct in application there are several advantages of two-step procedures as shown in the following. Two strategies are possible.

4.6.1 SC-LD Procedure

The SC-LD modelling is especially suited for Hammerstein-models and requires an invertible static characteristic for Wiener-models. At first the static characteristic is modelled in the relevant working range. In the case of a Hammerstein-model the parameters of the dynamic part are estimated with the intermediate input signal x^* (of Figure 4-5) and the measured output signal y. In case of a Wiener-model the output signal has to be utilised to calculate the intermediate signal x_o (of Figure 4-6), which necessitates an inversion of the static characteristic. If the static characteristic is not invertible MRAS techniques can well be utilised (Section 4.4). As the gain is comprised in the static characteristic the gain of the linear dynamic block is normalised to one.

4.6.2 LD-SC Procedure

This strategy requires a linear region of the SC and is well suited for Hammerstein and Wiener-modelling (Wernstedt 1989) but it is especially aimed at polynomial SC. At first the linear dynamic block is identified from an experiment, which only excites the process in the linear operating range. Then an experiment follows that excites the complete relevant operating range. The intermediate signals can be calculated utilising the linear dynamic model identified from the first experiment. Herewith the parameters of the polynomial SC can be determined by regression analysis. This procedure is especially useful if steady-state experiments are not possible (Rosenthal 1985).

4.6.3 Comparison of Direct and Two-Step Identification Procedures

A comprehensive investigation of simplified Hammerstein- and Wiener-models has been carried out by Rosenthal (1985). Here some important aspects of this work are presented and critical remarks are added.

Simplified Hammerstein-Models

The *direct (one-step) identification* of Hammerstein-models utilising stochastic estimation methods causes no major problems. However, well-known effects, like increasing errors with increasing disturbances, biased estimates with correlated error signal are more significant because of the lengthy parameter vector. Rosenthal (1985) experienced for the direct identification of the Hammerstein-model that modified LS-estimators like the IV- or GLS-estimators are not superior to the standard LS-estimator in the presence of disturbances for nonlinear models with model order of two and bigger.

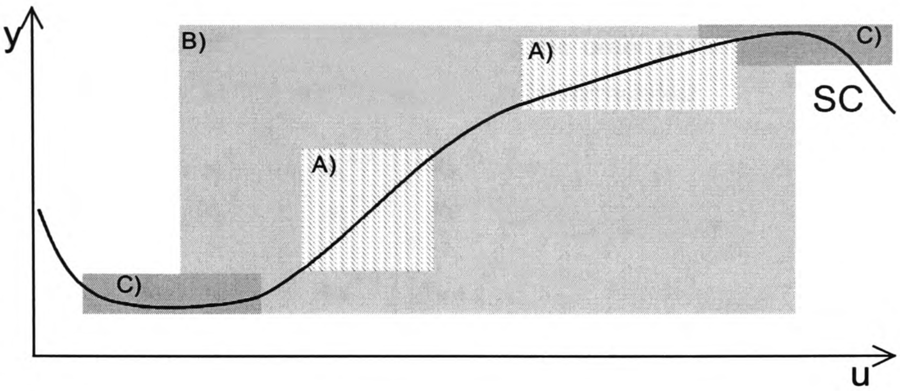
The *two-step SC-LD procedure* has some major advantages:

- o Knowledge about static process behaviour can directly be implemented.
- o The SC is not restricted to polynomials, its representation can appropriately be chosen.
- o Arbitrary nonlinearities without memory can be modelled.

Rosenthal (1985) remarks that errors in the representation of the SC affect the identification of the linear dynamic model as input disturbances and produce biased estimates. However, this is only a theoretically relevant statement, because the approximation of the SC within two-step methods is much more flexible than the approximation of polynomials within one-step procedures.

Simplified Wiener-Models

As already remarked the two-step SC-LD method is best applied for Wiener-models, if the range of the SC being touched by the experiment is linear or at least invertible. The segments *A* and *B* of Figure 4-11 shows these regions of the SC and the identification methods necessary to identify the linear dynamic part of the model. If the SC is invertible the intermediate signal can be calculated, which then includes the output disturbances. Rosenthal (1985) experienced that the quality of the estimation results is generally inferior to the Hammerstein results and that this method is not recommended for heavily disturbed processes. However, Norquay *et al.* (1996) remark that the SC-LD procedure still leads to better model quality than the direct identification, especially for highly nonlinear systems.



part of static characteristic (SC)	*identification of linear dynamic part*
A) linear	standard identification
B) nonlinear but invertible	inversion of SC & standard identification
C) nonlinear and not invertible part	application of an optimisation method

Figure 4-11. Influence of the SC-shape on the identification of Wiener-models

A direct identification of Wiener-models is possible if these are regarded as a subclass of a Volterra-series (Section 4.5). The Volterra-series is not restricted to polynomial SC and the information about static and dynamic parts of the model are combined. The estimation results are very sensitive to disturbances because of the large parameter vector. Mostly these models are over-parameterised leading to a good fit of the signal data but a model which generalises poorly (Wernstedt 1989).

4.7 Modelling Static Characteristics

In the preceding sections it was proposed to describe nonlinear processes by model structures which are composed of a linear dynamic and nonlinear static characteristic. In industry complex processes are often solely described by their static (and not their dynamic) behaviour, which is sometimes implemented into a simulation tool (Wozny and Jeromin 1991). If static characteristics are modelled from experiments as black boxes (Section 3.5), mathematical functions are fitted to available data points (for example gained by measuring different steady states of the process). This can be done by interpolation or approximation depending on the number of data and the purpose. Interpolation is concerned with the exact fit of a curve to generally few data points (Figure 7-2). Linear interpolation is the most simple method for interpolation (Bronstein *et al.* 1995). By contrast, approximation is concerned with an approximate fit of a smooth and simple curve to numerous data points. An approximated curve does not necessarily meet all data points, which is practically more sensible in the case of faulty data points. Some widely used methods are outlined in the following.

4.7.1 Polynomial Fits

Interpolation and approximation are often based on algebraic polynomial functions *f(x)*:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1} = \sum_{i=0}^{n-1} a_i x^i \qquad (4\text{-}47)$$

where n is the number of coefficients a_i and $n-1$ being the order of the polynomial. With N being the number of data points a set of linear equations can be formulated:

$$\begin{bmatrix} 1 & x_1 & x_1^{\,2} & \cdots & x_1^{\,n-1} \\ 1 & x_2 & x_2^{\,2} & \cdots & x_2^{\,n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & x_N^{\,2} & \cdots & x_N^{\,n-1} \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \quad \text{or} \quad \mathbf{Xa} = \mathbf{y} \qquad (4\text{-}48)$$

Looking for the minimal quadratic error and assuming that the matrix $\mathbf{X}$ is regular, a compact solution is (Wernstedt 1989):

$$\mathbf{a} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \qquad (4\text{-}49)$$

Often the matrix $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ is ill conditioned. Therefore numerically more reliable methods are utilised for the solution that avoid matrix inversion. If $N<n$ no solution can be found, for $N=n$ an exact solution can be found that interpolates the data points, whereas for $N>n$ an approximation takes place. Further it can be recognised that the approximation result becomes more and more oscillatory with increasing polynomial order. Therefore it is practically sensible only for $n \le 4$. Due to the parametric nature of polynomial approximation the curve can be described by a short parameter vector. It is characteristic for polynomial approximation that the fit of the function *f(x)* is oriented at its overall behaviour considering all data points at a time, whereas spline methods are more locally active and have more degrees of freedom as described in the following.

4.7.2 Cubic Splines

Cubic spline interpolation methods are able to avoid the oscillatory effects that may occur with polynomial interpolation. Furthermore they provide an efficient arithmetic if applied partially, which means that for every interval $x \in [x_i, x_{i+1}]$ one interpolating polynomial $S_i(x)$ is calculated that must have the same first and second derivative in x_i and x_{i+1} as the neighbouring polynomials. This guarantees a smooth curve as it would be drawn with a flexible ruler. If it is, for example, aimed at third order (cubic) polynomials, then *(4N-4)* equations are used to determine the same number of coefficients (a_i, b_i, c_i, d_i) for the $N-1$ spline polynomials $S_i(x)$ with $x \in [x_i, x_{i+1}]$ as follows:

$$S(x) = S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad i = 1,2,\ldots,N-1 \qquad (4\text{-}50)$$

In Bronstein *et al.* (1995) some realisations for cubic spline interpolation can be found as well. For example, an extra condition can be introduced controlling the smoothness of the resulting curve, like:

$$\sum_{i=1}^{N}\left[\frac{y_i - S_i(x_i)}{\sigma_i}\right]^2 + \lambda \int_{x_1}^{x_N} [S''(x)]^2 dx = Min \tag{4-51}$$

with σ_i representing the standard deviation of the ordinate y_i. For more details please refer to Reinsch (1967). Here λ is the parameter responsible for the smoothness of the curve. For λ=0 cubic spline interpolation is performed. The bigger λ the smoother the function so that for $\lambda=\infty$ a straight line results. Up to now λ must be chosen interactively with computer aid (Bronstein *et al.* 1995). The spline approximation exhibits a local behaviour and therefore a good local fit around the data points, because of the partial generation of the curve.

4.7.3 Limitations of Traditional Methods

In practice spline interpolation or approximation is preferred to polynomial methods because of its ability to fit local areas. This is especially apparent in computer graphics. For multi-dimensional fittings polynomial and spline methods as discussed in the previous sections have only weak performance (Preuß 1994). Solutions for interpolations can only be guaranteed if the data points fit to a rectangular grid, which is not the case for most measurement data – hence an extra transformation of the data is necessary beforehand. Therefore the application of polynomials and splines becomes quite arduous for multi-dimensional fittings.

4.7.4 Alternative Approaches

It is well known that *neural networks and fuzzy methods* are capable of representing multi-dimensional static characteristics in various ways. A comprehensive overview of neural net and fuzzy approaches would be out of the scope of the thesis but some interesting properties are named, because these methods can be transformed into the modified generalised weighted mean (MGWM) method introduced in Chapter 7.

Neural net methods utilise the terminology and some structures of human brain research. However, a closer look reveals that neural nets are not as intelligent as the term implies. The capability to learn is solely based on optimisation strategies that allow a stepwise adaptation of the input/output behaviour with the accompanying problems like local minima or excessive computation (Flaton *et al.* 1995). Nevertheless neural networks have the facility to perform interpolation and approximation quite well and are therefore an alternative for describing multi-dimensional static characteristics. The most important structure of neural networks is the feed-forward network that can be realised for example as a Multilayer Perceptron or as a Radial Basis Function (RBF) network, with a more global or local approximation behaviour respectively (Kahlert 1995a). The RBF network will be described in more detail in context with the MGWM in Section 7.4.2.

It took a long time for the initial work on *fuzzy sets* elaborated by Zadeh (1965) to be applied in practice. Nowadays fuzzy methods are not only fashionable – they even offer useful methods for practitioners. Fuzzy methods describe the generalisation of binary logic. The application of fuzzy methods can be separated into *knowledge based* and *algorithmic*. The first allows the processing of qualitative knowledge offering a way to translate linguistic descriptions into mathematics. The latter deals with data and therefore is an alternative to model multi-dimensional static characteristics. In Section 7.4 this will be detailed in context with the MGWM method. For a comprehensive overview of fuzzy methods please refer to Zimmermann (1993).

4.7.5 Quality Index

The quality index δ_{SC} can be used to determine and assess the quality of fit for the static characteristic. The error in the static characteristic is similarly defined as the OE for dynamic models. The deviation of the estimated and measured static characteristic is described by:

$$\delta_{SC} = \frac{\left\|\hat{f}(\mathbf{x}) - f(\mathbf{x})\right\|_2}{\left\|f(\mathbf{x})\right\|_2} = \frac{\sqrt{\sum_{i=1}^{N}(\hat{f}(x_i) - f(x_i))^2}}{\sqrt{\sum_{i=1}^{N} f^2(x_i)}} \tag{4-52}$$

4.8 Software Support for Identification

Different methods have been tried to assist the user in the complex task of system identification. The most important concepts of identification support are briefly analysed in the following sections and discussed with respect to applicability in industry.

4.8.1 Intelligent Help and Expert Systems

In the mid eighties it was widely recognised that user guidance and passive help systems were not sufficient to support the increasingly complex CACSD systems (including the identification tools). The standard *passive help systems* were suitable for getting quick index information but they were not suitable for inexperienced users, who did not know which sequence of commands lead to the solution of their tasks. Two approaches emerged form artificial intelligence (AI) to overcome this problem (Rimvall and Kündig 1990):

1. *Intelligent help.* This is the installation of an intelligent help in form of an information retrieval system. Intelligent help systems provide context sensitive help upon request or on the occurrence of mistakes and act as a consultant, proposing decisions to the user who has to decide ultimately. Three different types of systems have been developed, subdivided by the type of "user modelling", the event triggering, the activation of the help function and the behaviour after activation (Hvelplund 1986):

   - The mode of dialogue can be differentiated between *explicit* querying of the user and *implicit* modelling of the skill level, the latter is determined based upon the monitored user actions.

- The help function can be triggered *actively* by the user model, if the need is automatically recognised or *passively* upon user request as it is done in passive help systems.
- Hoffmann and Rimvall (1988) differentiated the behaviour after activation between a *passive help* mode (providing specific information on help topics), a *query mode* (checking actions interactively), and an *extended guiding* mode (defined as a mixed informational-executional mode).

To define the right mixture of the latter modes for a specific user category with respect to the supported software is a major difficulty because the needs of the users increasingly vary with the complexity of the supported software. Although intelligent help support for the identification procedure has been implemented by Nagy and Ljung (1989) or Betta and Linkens (1990) these do not address the general problems of industrial users, namely the time to become familiar with the program package and the confusing number of functions.

2. *Expert system* (XPS). This is the introduction of a guiding system as a new front end. Contrarily to the intelligent help system approach the XPS takes over the key role in the man-machine relationship. It leads the user through the whole identification procedure or solves problems automatically. XPSs work on underlying rule bases, which consist of weighted paths between problem and solution but are not able to cover the complete variety of industrial identification tasks due to the enormous number of rules which are already necessary to implement even for specific identification tasks. Therefore the amount as well as the type of AI employed in system identification have to be carefully selected (Taylor *et al.* 1991 and Rimvall and Kündig 1990). Furthermore the method of reasoning is not transparent for the user for complex problems, such that a close process understanding may be hindered. This problem has been addressed by Wernstedt *et al.* (1991), who provide a tracking facility that enables the user to view the complete decision process finally. Many XPSs have been applied to identification in order to handle the vast number of model structures and the methods for data pre-processing and identification. Meier zu Farwig and Unbehauen (1991) provide a good introduction into that subject and different realisations are reported in detail by for example Haest *et al.* (1990), Gentil *et al.* (1990), Niederlinski *et al.* (1991) and Bergeon *et al.* (1991). For more information about XPS in identification and a comparison of different approaches please refer to Maier zu Farwig (1992).

In a paper about AI in identification Niederlinski *et al.* (1994) confirm that the identification procedure can be intelligently supported by an expert system approach as well as by an approach utilising decision trees. However, they could not find logical advantage for the use of XPSs over the use of decision trees. They conclude that knowledge can be implemented more economically and transparently in conventional software structures like decision trees without problems. This is the more convincing as these authors had already implemented the XPS approach some years ago (Niederlinski *et al.* 1991). Also the change or refinement of existing decision trees is easier through the clear structure within the trees. It might be due to this fact that the number of publications about AI in process identification decreased considerably the last years. Benveniste and Åström (1993) point out that the AI community itself questions expert systems: "*Rules or knowledge bases are not the universal method of reasoning!*". Nevertheless the research on XPS for system identification has been extremely valuable as it has produced considerable insight into the details of the identification procedure.

4.8.2 Graphical Support

Barker and Linn (1979) very early postulated that the greatest potential for reducing the gap between technological and theoretical development in CACSD is provided by *graphical user interfaces (GUIs)*. In general GUIs are one of the most successful approaches to support users of computer programs. In the domain of control engineering this approach has been pursued by different research groups, most notably at the University College of Swansea (Barker *et al.* 1988, Barker *et al.* 1989, Barker *et al.* 1990, Grant *et al.* 1991, Chen 1991) and the University of Salford (King and Gray 1986, Li and Gray 1993). With respect to system identification, the basic idea is to offer facilities that are well suited to support and simplify the identification task, reflecting approaches that are familiar to the industrial user. A historical background of graphical user interfaces in system identification can be found in De Moor and Van Overschee (1994).

As the identification task is part of CACSD and thereby may be a part of modelling as well, it is necessary to review the main graphical aids for CACSD. Graphically oriented CACSD environments like SIMULINK, BLOCKSIM (Winter *et al.* 1994), DORA (Kiendl *et al.* 1998) and many more support block-oriented modelling because this is geared to the understanding of control engineers and reflects the commonly used pencil-and-paper approach. Most of these modelling and simulation environments support solely the bottom-up modelling, which is the structural composition of systems from basic modelling blocks combining subsystems to yield the complete model. However, research projects like ECSTASY (Munro 1993) or HIBLITZ (Elmqvist and Mattsson 1989) have already demonstrated the merits of providing top-down modelling facilities as well, which is the stepwise refinement or decomposition of a system structure according to an envisaged granularity of the process. For complex systems it is helpful that the direct verification of graphical systems representations is mostly featured by current CACSD systems such that the consistency of block-oriented modelling environments is ensured.

Apart from the modelling environment there are some basic features of GUI, which increasingly converge towards the Windows styleguide standards (Polzer 1998). The inherent *'look and feel'* concept being concerned with the appearance and behaviour of the application could be utilised to provide visual guidance for uninitiated or infrequent users, who have to feel their way forward in complex CACSD tasks. Helpful interface control features have been developed on these standards, including facilities to browse, select, zoom, copy and so on. However, the potential of GUI design is not fully employed for industrial users yet.

Especially in the field of system identification most GUI-realisations are aimed at experts.

For example the user-interfaces shown in Figure 4-12 and Figure 4-13 are quite complex providing much information and many functions in one window. The user cannot feel through these GUIs at once and needs some time to get acquainted to them because of the many facilities that are provided simultaneously regardless the current situation within the identification task. For inexperienced users the next step within the identification procedure is mostly unclear.

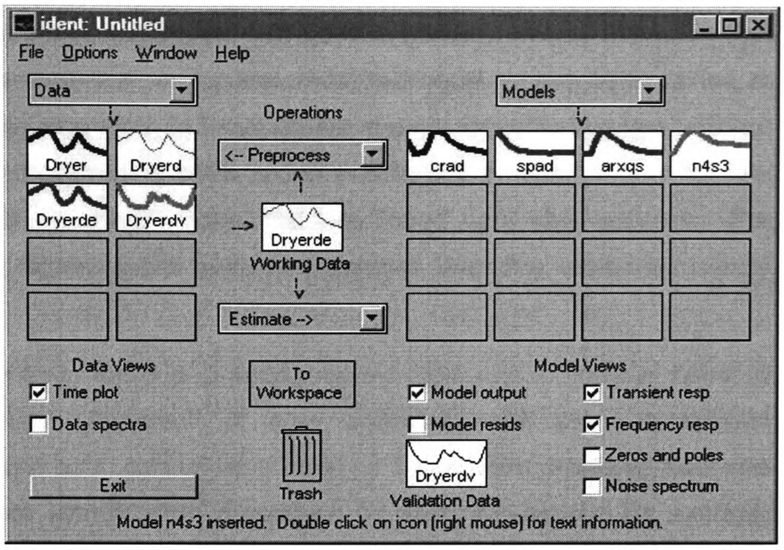


Figure 4-12. The GUI of the System Identification Toolbox (Ljung 1995a)

Furthermore it is important to note that the realisation of GUI paradigms does not simply involve putting old structures into nice windows. Rather, it means a complete redesign of the existing command driven CACSD software, which has to be task-oriented in functionality for ease of use. Also data structures for CACSD systems and simulation programs have to be reviewed in this sense, because GUI design and functionality is closely linked to the object-oriented paradigm (see for example Jobling 1991, Jobling *et al.* 1994 or Varsamidis *et al.* 1996), which enables many developments that would be unthinkable using conventional approaches – not least because of complexity and maintenance issues.

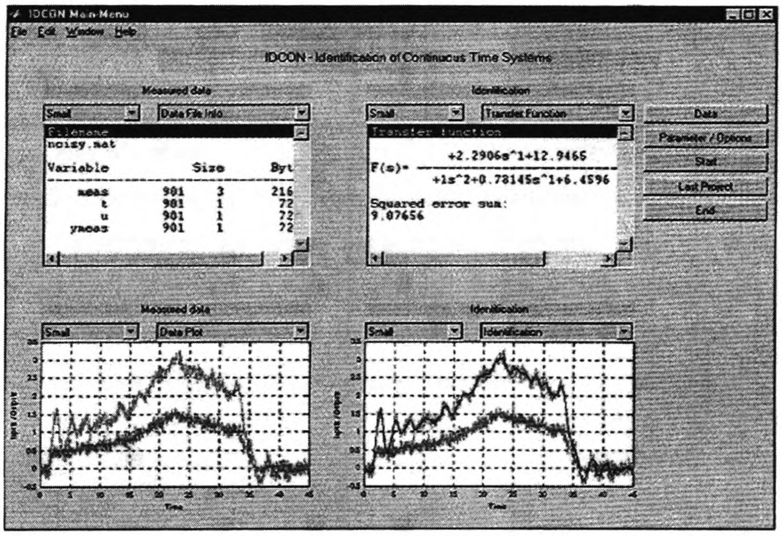


Figure 4-13. GUI of the IDCON toolbox (described in News Server 1998)

To sum up, a self-explanatory GUI is best used if it helps to avoid questions as far as possible and enables the user to feel through the system identification procedure.

4.9 Conclusion for this Chapter

The above overview of identification support (Section 4.8) gave a general idea of the work that has been done so far and the further potential. In general, most realisations have been aimed at control engineering experts in industry who want to focus on the system identification task without the requirement of both extensive programming skills and a substantial amount of time for low level interactions. However, there is an overall issue that has found very little attention. The support of potential users that do require more individual support than the industrial control engineer.

In the light of present day requirements, there is a strong economical and ecological need to replace the prevailing *'rule of thumb'* approach of area engineers with more systematic methodologies. The expert system approach cannot be utilised as it has been practised because it fixes the complexity of the system identification procedure as seen by experts for experts. Instead it is necessary to rethink the available identification methods and their interplay with respect to area engineers.

In this sense it is beneficial to provide a relatively small collection of simple methods being robust in application and which can be applied to a wide range of problems yielding sensible results. This way mathematical methods can help to reduce the software complexity – complexity, which is arduous for the programmer and also for the user.

Therefore the next chapters challenge identification in the light of CACSD for industrial use. At first concepts for an industrial CACSD procedure and a standardised nonlinear identification procedure are derived to provide an alternative for the oversimplified control design and identification strategies often being applied in the process industry. Additionally specifically suitable methods for the identification of static characteristics and linear dynamics are proposed, which have been modified for simplified use. Finally a prototype realisation is presented to validate the approach presented.

5 Proposal for an Industrial CACSD Scheme

From the requirement analysis (Chapter 2) it was concluded that an approach to nonlinear multi-variable control system design is needed which is intuitively understandable for industrial users who are not control experts and who need an efficient CACSD tool for commissioning. In this chapter an industrial CACSD scheme is proposed that sets the starting point for the development of the "*structured approach to identification techniques for the analysis of industrial processes*", which follows the pragmatic engineering approach of "*starting simple and adding complexity only if necessary*".

In order to enlarge the design transparency for industrial users a stepwise approach had to be designed, which reflects the users' thinking by increasing the complexity of the process model successively and which is described in the next sections. This industrial approach has been standardised for control systems based on advanced industrial PID controllers thus avoiding heuristic controller design. As an example for the efforts to formalise the industrial control system design procedure a prototype realisation utilising a multi-variable pilot plant is described in Section 5.2.

5.1 Structure of the ICACSD Scheme

In Section 3.1 it has already been noted that standard control schemes are available for some multi-variable processes. However, if these standard control schemes are not accessible or valid any more, for example if a process is redesigned, a new control system design must be elaborated. Mostly the first step in practical control design is based on the simplified assumption that it is sufficient to split the multi-variable process into independent single-variable subprocesses by associating each process output to be controlled to the process input with the greatest influence on it. For each of these independent main SISO paths a separate PID controller is designed on the basis of rather rudimentary process information (Schumann *et al.* 1996). Only if this approach fails due to unacceptable control performance a deeper process analysis is undertaken. Then observed changes in process gains and time constants as well as coupling effects between SISO subsystems are additionally considered for the control design to cope with the observed effects. Over time quite complicated industrial control schemes might develop that could become standard for specific processes.

This pragmatic design procedure has been utilised in order to avoid heuristic control system design and to ease the support for process personnel. The resulting ICACSD (industrial CACSD) scheme depicted in Figure 5-1 is based on two main principles:

- the straightforward *model evolution scheme*, which describes the complexity of advanced industrial PID controller structures based on block oriented standard models.
- the simplified *standardised CACSD procedure*, which is worked through in each phase of the model evolution scheme.

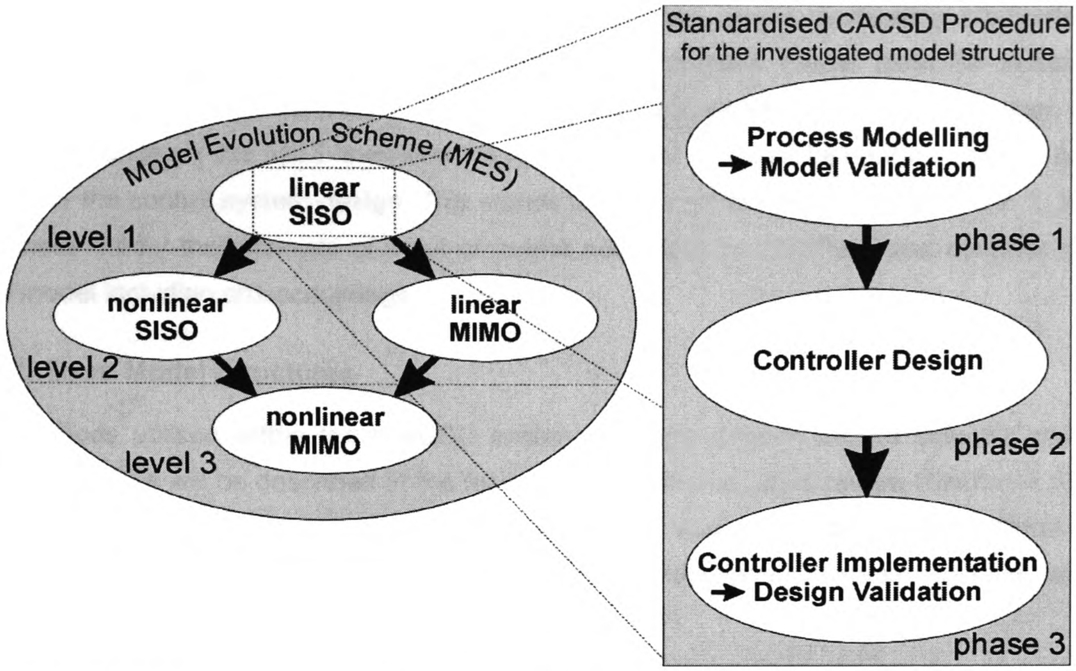


Figure 5-1. ICACSD scheme for multi-variable processes

5.1.1 The Model Evolution Scheme (*MES*)

This part of the ICACSD scheme is called *model evolution scheme (MES)* because it describes the complexity of the model, on which the control system design is based. In this sense it describes more the control system complexity than the actual process model complexity itself. Advanced PID control has been selected because in Section 3.2 it was shown that this is the only control technique that displays a gradient in the performance-effort-area implying that a better process model can lead to better control performance. This is a desirable property for a subsequent approach that can be easily handled by industrial users. Therefore it was decided for this research project to investigate and develop identification strategies that suits industrial purposes best as a basis for 'advanced PID' control.

The *MES* consists of three levels with increasing complexity. The first level within the *MES*, namely the *linear SISO*, stands for the control system design which is based on the separation of a multi-variable process into independent single-variable subprocesses that associate each process output to be controlled to that process input with the greatest influence on it. If the individually designed PID controllers do not work satisfactorily, the next step in practice would be directed at a compensation of the observed effects as outlined above. In contradiction to this heuristic approach, the second level within the model evolution scheme distinguishes the *nonlinear SISO* approach, which is particularly aimed at the compensation of the nonlinearities, or the *linear MIMO* approach, which is solely aimed at taking into account the linear dynamic crosscouplings. This means that if nonlinearities affect the model quality significantly nonlinear elements will be added to the control system or, if the process exhibits linear behaviour in the working range but couplings affect the control performance, linear decoupling elements will complement the control system design. Of course, the process model must be sufficiently complex in each stage of the scheme to represent a suitable base for control system design. If the control performance on this *MES*-level fails the most complex structure being the *nonlinear MIMO* is applied for the control system design. This stands for PID control system design which is based on a process model that consists of input or output nonlinearities and the linear dynamic multi-variable model including crosscouplings.

5.1.2 Utilised Model Structures

As the methods utilised within the ICACSD scheme strongly depend on the selected process model structure this will be described in the following. From the literature review (Section 4.5) and the discussion of the practical requirements (Section 2.3) it was shown that simple Wiener- and Hammerstein-models are especially suited to system identification aimed at industrial control system design.

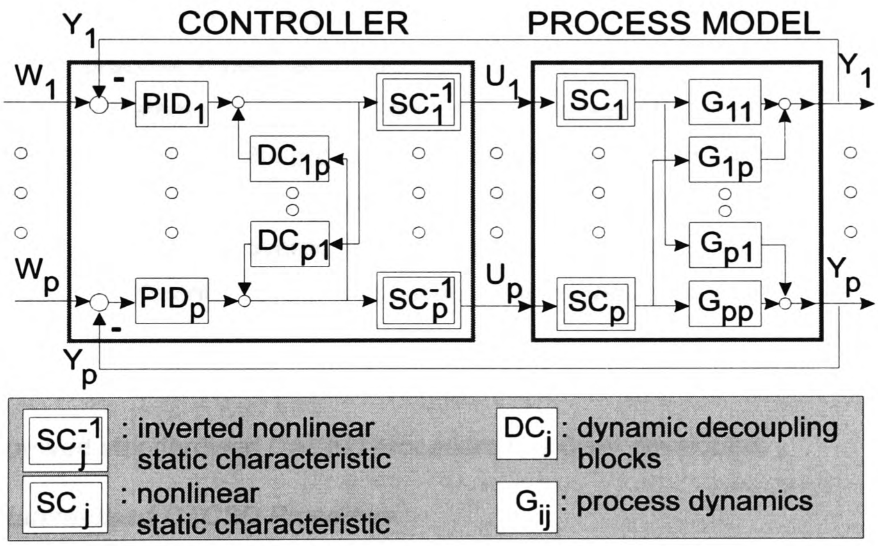


Figure 5-2. Hammerstein-model structure and controller

Therefore a key element of the ICACSD scheme is to compose the process model from linear dynamic single-variable blocks and, if necessary, nonlinear static blocks combined as Wiener- or Hammerstein-models. Accordingly the structures of the complementary control systems range from linear SISO to nonlinear MIMO depending on the utilised model complexity, which is necessary to achieve satisfactory control performance. An example for a control structure of level 3 resulting from the *MES* for a process represented by a Hammerstein-model structure with separate SISO-SCs is depicted in Figure 5-2.

5.1.3 The Standardised CACSD Procedure

Another important difference to the intuitive, heuristic approach in industry is the application of a control system design procedure, which has been standardised for the class of model structures outlined in Section 5.1.2. This facilitates the use of simply-applicable methods for identification and control system design. Naturally, a standardised and therefore simplified CACSD procedure – based on simple process models - cannot guarantee an optimal result which may be possible with more degrees of freedom. However, for the commissioning of industrial processes a good control design following standard design paths may be the only result achievable based on a background of limited expertise and little time available. Furthermore the standardised CACSD procedure provides good reproducibility of the gained results by restricting the variety of possible solutions. Within this work a basic form of the standardised CACSD procedure was developed at first that had to be applied for each level of the *MES*.

The Basic Standardised CACSD Procedure

The so-called basic standardised CACSD procedure consists of the three main CACSD phases already outlined in Figure 3-2. It is applied for each level in the *MES* serving as the main vehicle to produce models and corresponding control structures. This procedure is rather inflexible and especially aimed at process personnel as described by Körner *et al.* (1996). After the process identification the control system performance can be predicted by simulation of the complete control system with process model and controller. If the simulated control performance is satisfactory the control system can be tested on the real process. If the control performance on the real process is not satisfactory the procedure has to be repeated on the next level of the *MES* until the last level is reached. Naturally one must consider that the control system has been designed only for those effects covered by the process model used and that the controlled process might behave differently to the simulation. This means that the agreement between simulated and actual control behaviour on the first two levels of the model evolution scheme can be quite poor.

Therefore an improved standardised CACSD procedure has been developed.

The Improved Standardised CACSD Procedure

The improved standardised CACSD procedure is based on the same set of model structures but it utilises an advanced identification scheme within phase 1, namely the standardised nonlinear identification procedure called *SNIP,* explained in Chapter 6. The *SNIP* supports the process modelling for linear, Wiener- or Hammerstein-models. Hence the control design is based on a better process model and phase 1 of the standardised CACSD procedure is only done once. Therefore it is not necessary any more to execute all CACSD phases on each level of the *MES*. As the process is analysed within the *SNIP* with respect to the *MES* the controller can also be designed for the detected process model complexity directly. However, it might be helpful to undertake the control design step by step for each level of the *MES* up to the detected process model complexity, because then it is possible to check and visualise the stepwise increase in control performance in order to select the necessary control equipment. Additionally the simulation results are more trustworthy than those of the basic standardised CACSD procedure because of the improved process model quality.

5.2 Prototype Realisation of the ICACSD Scheme

The ICACSD scheme was tested in many stages of its development and refined where necessary. Experimental work was undertaken through simulations as well as on various multi-variable pilot plants (see also Section 9.5.1). These include an air conditioning process with humidity and temperature as control variables (described by Syska *et al.* 1999), a 3-tank process with the pressure to be controlled and a process with water level and pressure as control variables. The latter is described below as an example of the efforts that were undertaken to formalise the industrial control system design procedure. In the following the original ICACSD scheme is described utilising the basic standardised CACSD procedure (see also Schumann *et al.* 1996).

5.2.1 Nonlinear MIMO Pilot Plant

The main component of the nonlinear multi-variable pilot plant (Figure 5-3) is a semi-closed water tank filled with water by the water-pump (No.1 in Figure 5-3) and with air by the air-pump (No.3 in Figure 5-3). Two valves, one for water level and one for air pressure (No.2 and 4) allow the adjustment of the operating point of this plant. Valves 5 and 6 can be used to generate disturbances for the water level Y_w and air pressure Y_a, which are controlled by the voltages U_w for the water-pump and U_a for the air-pump. The plant is clearly crosscoupled in the sense that the water-pump does not only influence the water level, but also the air pressure and vice versa the air pump also influences the water level. This process has been used to assess the applicability of the ICACSD scheme and also to test various available CACSD tools.

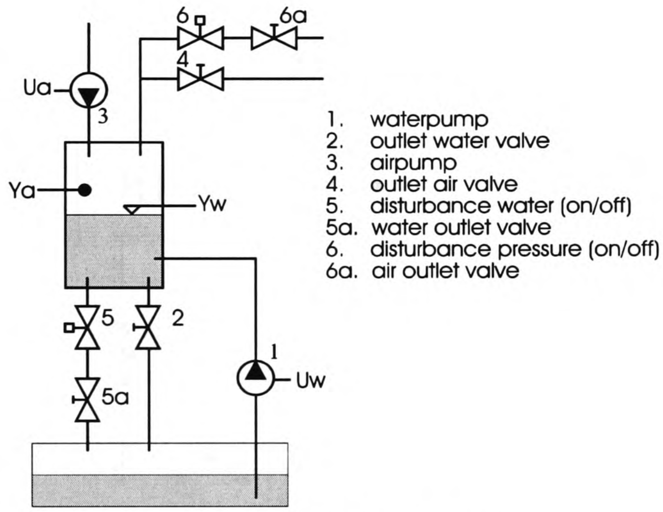


Figure 5-3. Nonlinear MIMO pilot plant

Table 5-1. Utilised tools for the prototype realisation of the ICACSD approach

Task within Standardised CACSD Procedure	CACSD-Tool	Functionality
identification of static characteristics	*EasyStat* by FH Hanover	automatic experiments in open or closed loop, suitable for static single and multidimensional characteristics
identification of linear dynamics	CADACS™ by University of Bochum	appropriate real-time module, different identification methods available
controller design	DORA 5.1™/DORA-Fuzzy by University of Dortmund	offers integrated and simple optimisation facilities for PID-control design
controller implementation	TCS™ with Loopdraw by EUROTHERM	typical industrial controller, block oriented configuration SW 'Loopdraw'

5.2.2 Utilised Tools for a Prototype Realisation of the Standardised CACSD Procedure

Considering the objectives of the ICACSD scheme as outlined in Section 5.1 several CACSD programs were tested with respect to a prototype realisation of the standardised CACSD procedure. Finally those programs shown in Table 5-1 were selected to experimentally test the whole scheme. Additionally the program *EasyStat* (see Appendix A) has been developed, because no commercial tool was found that supported the modelling of multi-dimensional static characteristics satisfactorily.

5.2.3 Experimental Results

In the prototype run discussed in the following the ICACSD scheme was tested with respect to set point changes and disturbances. Among the many results a ramped set point change with positive and negative flank (as often applied in the process industry) on the water level was selected to validate the overall control performance. For this purpose the process input variables water and air pump voltage U_w and U_a, as well as the process output variables water level Y_w and air pressure Y_a of the controlled process were recorded (see Figure 5-4).

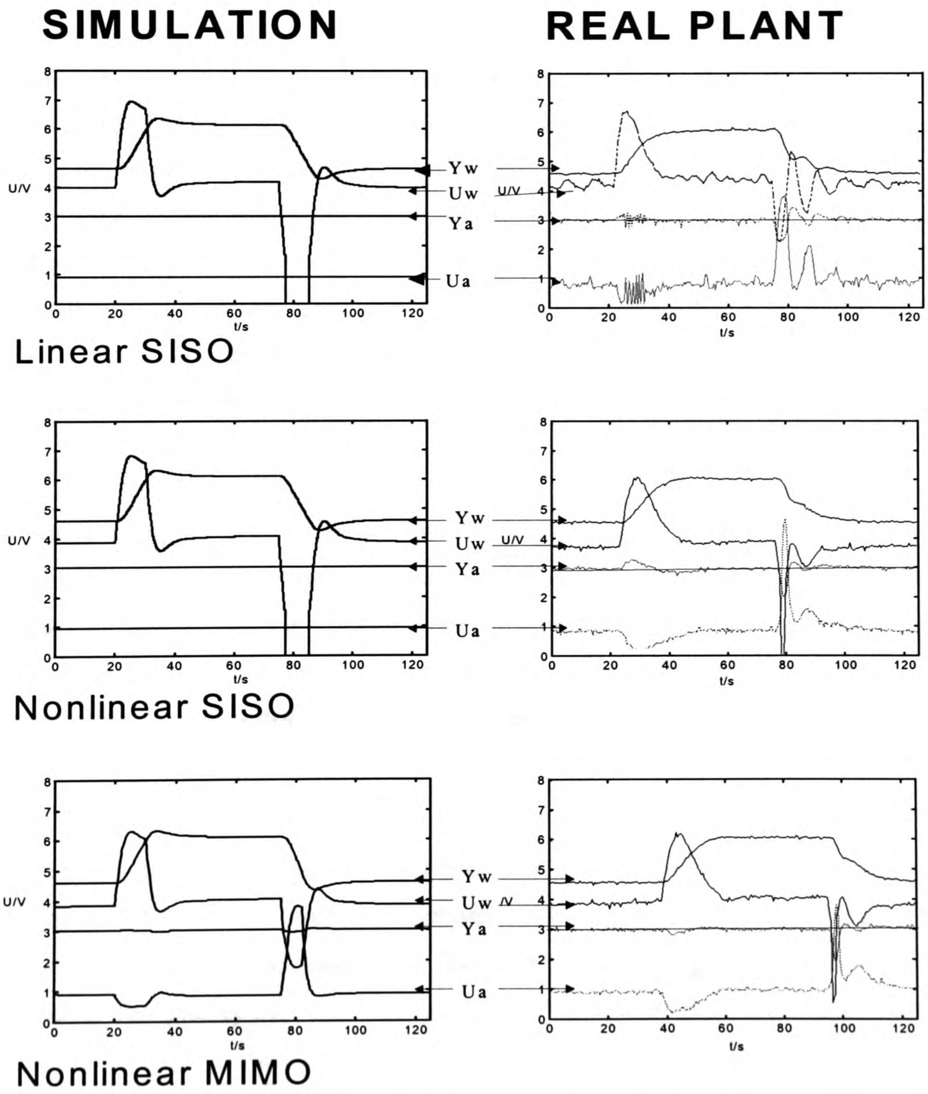


Figure 5-4. Comparison of simulations and experimental results of the controlled process

Linear SISO

According to the model evolution scheme (Figure 5-1) the experiments started on the linear SISO level. Firstly the separated linear dynamic SISO models for water level and air pressure were identified. Then the separate SISO controllers have been designed and the control loops were simulated. As the simulations showed a satisfactory control behaviour the controllers were implemented at the process. Figure 5-4 shows the results of the simulation compared to the real experiments at the controlled process. The control behaviour on the real plant shows unacceptable oscillations and coupling effects. Absolutely, the simulation does not show any of these effects, because the simulated model does not cover any nonlinear or coupling effect and the controller is designed just for the modelled effects. This first experiment displays the sub-optimal performance that can be often met in industry where the final control design often results in excessive wear of the actuators. Therefore the next model complexity level was required.

Nonlinear SISO

In order to reduce the observed oscillations and to improve the control performance it was decided to consider the nonlinearities but not the couplings at this level. The static characteristics were modelled for each main I/O path with *EasyStat* and the linear dynamics were identified. Validation tests revealed Hammerstein-like behaviour, such that the static characteristics containing the varying process gain were set in front of the normalised linear dynamics. The control system structure was extended accordingly. Based on the improved process model the control performance on the real plant became much better with respect to the oscillation effects observed in the first attempt, see Figure 5-4. Naturally the simulated system behaviour remained identical to the linear SISO case, because the nonlinear effects were compensated. However, the coupling effects were still considered to be too strong so that the last complexity level of the ICACSD scheme was required.

Nonlinear MIMO

This is the most complex approach of the ICACSD scheme as outlined in Section 5.1. The process model was complemented with the identified coupling paths and a linear decoupling network was implemented into the control system. Now the crosscoupling effects could be clearly reduced and the comparison of the control system simulation and the real-time control experiments shows a much better fit than on the previous levels (see Figure 5-4).

5.3 Conclusion for this Chapter

An ICACSD scheme was proposed consisting of a standardised CACSD procedure and a model evolution scheme. The model evolution scheme reflects the traditional way of doing control system design in a systematic way. The standardised CACSD procedure only supports a constrained model complexity that can easily be handled by inexperienced users and it provides a good reproducibility of the gained results by restricting the variety of possible solutions.

A basic and an improved standardised CACSD procedure have been introduced and the advantages of the improved procedure have been discussed. However, the improved procedure necessitates a standardised nonlinear identification procedure, which is introduced in the next chapter.

It has been experienced by students in the control laboratory that the stepwise increase of control performance following the model evolution scheme is quite helpful to get a feeling for the process characteristics and the possibilities of advanced PID control system design.

The experimental results that were gained utilising four selected CACSD tools demonstrated the feasibility of assisting the user with computer aids following the ICACSD scheme. However, as the utilised CACSD programs were not designed for the support of this methodology, industrial users cannot apply these with reasonable effort. Furthermore the use of different proprietary programs causes problems like:

- Interfacing problems concerning file format, data structures and meaning of parameters. The data must be rearranged and adapted.
- Different user interfaces and different utilisation of the software. The user has to keep track of all programs and underlying program philosophies.

These problems are addressed by the prototype development currently being carried out at the University of Glamorgan and the Fachhochschule in Hannover. The first part of this ICACSD prototype is the *ICAI* toolbox outlined in Chapter 9, which utilised the *SNIP* as a main prerequisite for the application of the improved standardised CACSD procedure.

6 Proposal for an Identification Procedure for Nonlinear Processes

In Sections 4.5 and 4.6 the advantages of Wiener- and Hammerstein-model structures with respect to industrial needs have already been discussed and different possibilities for Wiener- and Hammerstein-modelling have been summarised. These structures need a reasonable number of model parameters and are easy to understand. Not least they can be implemented in most block-oriented industrial controllers.

Within this chapter the concept for a *standardised nonlinear identification procedure (SNIP)* is developed having following properties:

- One pragmatic transparent procedure covering linear and nonlinear modelling.
- Reflection of concepts that the industrial user applies naturally.
- Simplified experiment set up.
- Guided tour through the partly automated procedure.

6.1 Assumptions about the Process

Industrial processes in the process industry with continuous-time behaviour often have the following characteristics (Section 4.1):

- Nonlinear for practical operating ranges.
- Deadtimes.
- Time-varying dynamics.

For sensible identification results within the *SNIP* these characteristics have to be considered properly. This can be done if some assumptions on the working points of the process are made, namely:

- The process is operated in the neighbourhood of a limited set of working points.
- At each working point the process dynamics are time-invariant or the change is very slow.
- At each working point the disturbances are considered to be stationary.
- The outputs to be controlled can be easily measured.
- The process can be sufficiently described by Hammerstein- or Wiener-models.

On the basis of these assumptions it is reasonable to concentrate on the identification of time-invariant, finite-dimensional models of single- and multi-variable industrial processes.

6.2 The Standardised Nonlinear Identification Procedure (*SNIP*)

From the considerations elaborated in Section 4.6 a standardised nonlinear identification procedure (*SNIP*) has been developed. It is based on the SC-LD procedure because this is not restricted to polynomial SC and it is straightforward in industrial application. Figure 6-1 presents the flowchart of the *SNIP*, which starts with some pre-experiments within the operating range of the process under investigation and ends with an analysis tailored to the *MES* (Section 5.1.1). In the following the steps within the *SNIP* are explained.

6.2.1 *SNIP*-Experiments

If only poor a-priori knowledge is available it is advantageous to perform a series of preliminary experiments (*pre-experiments*) before the *final identification*.

Pre-Experiments

Before the pre-experiments start the inputs and outputs have to be selected.

- *Selection of inputs and outputs.* Generally process outputs are those variables to be controlled or to be predicted (simulated). If the outputs are easily measurable the next step is to determine those inputs that have a strong influence on the outputs and that can be reliably manipulated for control purposes. With respect to the input selection it may not be possible to manipulate an input, although it can be measured. This input can therefore be used in the design of a feedforward compensator. With respect to the output selection it may be too difficult or costly to measure certain outputs to be controlled. Then other measurable outputs have to be selected, if possible, in order to determine the behaviour of the controlled outputs.

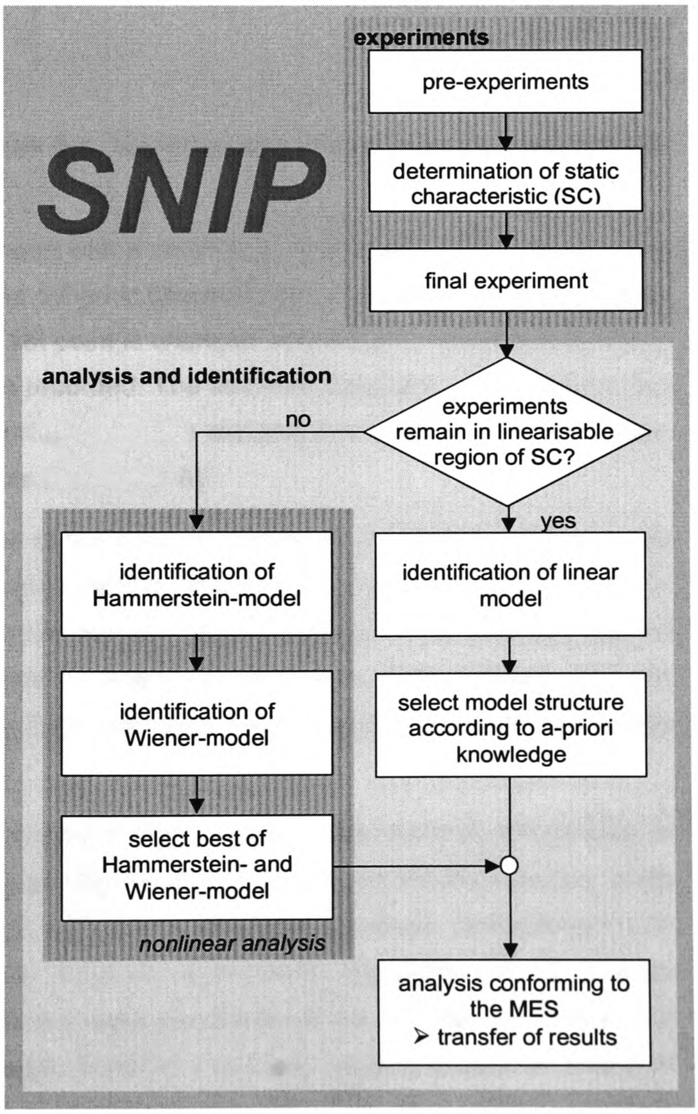


Figure 6-1. SNIP flowchart

- *Process pre-experiments.* The process observation is a technically straightforward experiment. The process is run in open loop to allow the assessment of the pure process. Simple test signals like step or impulses with small amplitude can be applied and recorded within the operating range to get a feel for the process behaviour. When the intervention of the operator is kept to a minimum rough information can be gained about the process behaviour concerning the characteristics of the output changes, the amplitude range and the frequency spectra. For more details please refer to Zhu and Backx (1993).

Determination of the Static Characteristic

The static characteristic and thereby the degree of nonlinearity of the process in the working range is determined by a steady-state experiment. A practical test signal for steady-state experiments has the shape of a staircase (Figure 6-2). It is important that the length of the stairs allows the process to settle and that the step input changes are small in those regions with a big gain compared to those with a small gain. The steady-state test can also be used to yield valuable information from the dynamic characteristics between the steady-states. If the process is operated in closed-loop the set point is changed according to the staircase shape and the steady-states of input and output are recorded. The software *EasyStat* - developed within this project - is a first prototype tool to support steady-state experiments in open and closed-loop for processes up to two inputs and two outputs (Appendix A).

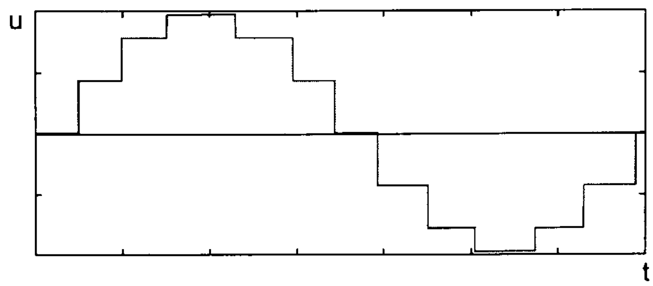


Figure 6-2. Test input signal for steady-state experiment

By doing steady-state experiments the shape of the static characteristic is gained. Thereby it can be determined if the SC is linearisable in some intervals. Furthermore the static gains are found as the gradient of the SC. Also the deadtimes and the largest relevant time constant can be roughly estimated from open-loop experiments, such that the experiment duration can be determined by rules of thumb (Section 4.1.1). If the test signal has positive and negative flank also hysteresis effects are detected.

If this kind of experiment is not permitted due to technical or economical reasons, simple step or impulse responses should be used in this phase (Section 4.1.1). These are mostly accepted when process operators use these to regulate their processes. Naturally, the static characteristic can only be obtained if multiple step or impulse response experiments are performed at different operating points. For MIMO processes different representations of the SC for the Wiener- or Hammerstein-model are possible. For example, SISO-SCs at every input or output or a field of SCs (MIMO-SC) in front or behind the linear dynamics can be used. The different characteristics of these realisations are discussed in Section 4.5.2.

Final Experiment for the Determination of the Linear Dynamic Process Model

The final experiment is the last experiment of the *SNIP* (Figure 6-1) and it is based on the results of the pre-experiments and available a-priori knowledge. Section 4.1 provides practical guidelines for performing a good final experiment. Only if the whole identification procedure is not successful it might be necessary to repeat the experiment as outlined in Section 5.1. In the literature the topics about experiment design and input design are mostly related to this final experiment.

The design of the final experiment within the *SNIP* procedure depends on the a-priori knowledge about the position of the SC in the sense of a Wiener- or Hammerstein-model structure.

- If the position of the SC within the model structure is known in advance then the final experiment should be carried out in the *linear working range* of the process, if possible. For example if the nonlinearity is contained in a valve at the process input with a time constant very small compared to that of the process under investigation then the Hammerstein-structure is most suitable and an experiment in the linear region of the SC can be carried out. This procedure is advantageous because it allows the direct application of standard methods for linear identification and often only small excitations of the process are necessary.
- If the position of the SC is not known a-priori then the experiment has to be carried out in a *nonlinear working range* of the process simply because the process structure can only be evaluated from measured nonlinear effects. It is advantageous if the experiment stays in a region where the SC is invertible (Section 4.6.1). Otherwise the parameter estimation of the linear dynamic part of the Wiener-model has to be based on optimisation techniques. The position of the static characteristic can be finally analysed as outlined in the next section.

6.2.2 *SNIP*-Analysis and -Identification

The *SNIP* enables the use of a wide range of well tested algorithms for linear dynamic identification because it treats nonlinear static and linear dynamic characteristics separately. Based on the final experiment the main part of the *SNIP*, which is the *analysis and identification*, can be undertaken, resulting in an evaluation of the results in conformance with the *MES*.

Identification

If the final experiments remained in a linearisable region of the static characteristic and if the position of the SC within the process model structure is known then the conventional linear dynamic identification can be performed according to Section 4.2. A specifically capable identification method is presented in Chapter 8. The model structure is selected according to the a-priori knowledge before the process analysis is performed.

If the position of the SC within the process model structure is not known and if the final experiment has been carried out in a nonlinear working range then both, Wiener- and Hammerstein-model, are identified in order to select the superior result finally.

- *Identification of the Hammerstein-model.* At first the static characteristic is used to compensate for the nonlinearity of the Hammerstein-model so as to prepare the data for a conventional linear dynamic identification.
- *Identification of the Wiener-model.* If the static characteristic is invertible then it is also used to compensate for the nonlinearity of the Wiener-model followed by the conventional linear dynamic identification. Otherwise optimisation methods have to be applied for the Wiener-model structure (like the MRAS explained in Section 4.4) that tune the model parameters of the linear dynamic model according to a specific loss function.
- *Model Structure Selection.* The *SNIP* includes a pragmatic selection strategy. This is supported by a very expressive and simple validation criterion - the output error (OE) and a graphical comparison of simulated and experimental process behaviour. Based on the OE a model structure is proposed to the user, who has to judge as to whether the Wiener- or Hammerstein-model structure is superior. However, it might happen that the output errors for Wiener- and Hammerstein-model structure are not significantly different. If the identified SC is nonlinear then either preceding steps have to be repeated or further tests on the process have to be carried out. As the final selection criterion is the control performance, it is also possible to design controllers assuming Wiener- and Hammerstein-model structures in order to determine the best control structure on the process thus implicitly defining the model structure.

SNIP Interacting with the MES

Figure 6-3 depicts the improved ICACSD scheme which includes the improved standardised CACSD procedure already outlined in Section 5.1.3. Phase one of the basic standardised CACSD procedure has been replaced by the *SNIP*, which finally delivers the process model conforming to the *MES*. This analysis evaluates influences of the nonlinear static characteristic and the linear couplings (in case of MIMO processes) and determines the first relevant level of the *MES*, for which a controller has to be designed.

Table 6-1. SNIP process models analysed for the MES

<table>
<tr><th colspan="2">SNIP process model</th><th>SNIP-SISO
process model</th><th colspan="2">SNIP-MIMO
process model</th></tr>
<tr><td rowspan="3">controller structure within the MES</td><td>level 1</td><td>linear SISO</td><td colspan="2">linear SISO</td></tr>
<tr><td>level 2
complexity</td><td>static characteristic and linear SISO dynamic</td><td>static characteristic and linear SISO dynamics</td><td>linear decoupling network (MIMO)</td></tr>
<tr><td>level 3</td><td></td><td colspan="2">linear decoupling network (MIMO) and static characteristic</td></tr>
</table>

This interdependence is shown in Table 6-1. For example if a MIMO process exhibits a significantly nonlinear behaviour then the modelling of the most simple controller on level 1 is skipped and level 2 of the *MES* is accessed at once. Similarly it is possible that the modelling of a control system according to level 3 will be initiated at once if nonlinear influences and couplings are a significant part of the process model.

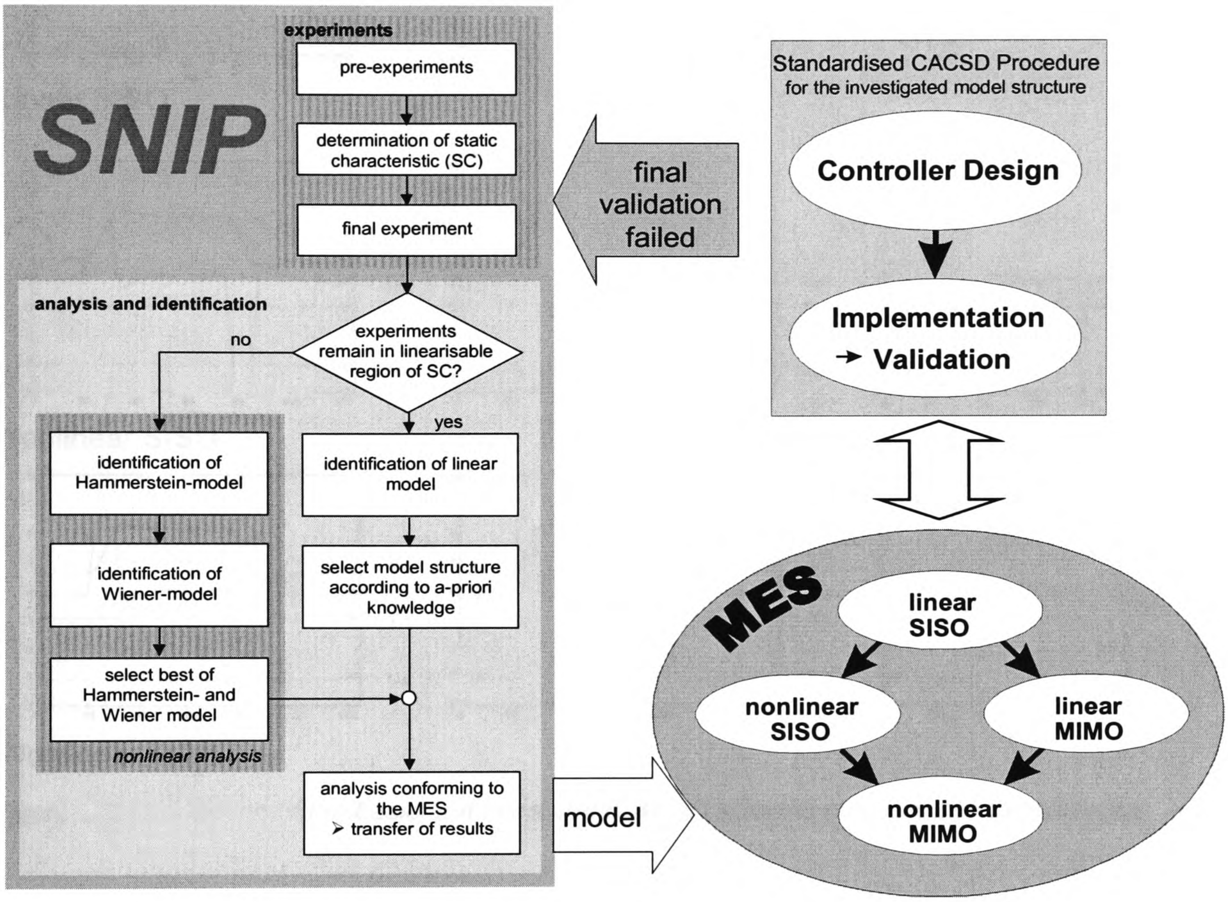


Figure 6-3. Improved ICACSD scheme

As explained in Section 5.1 it might be sensible to start with the simplest level of the *MES* although a more complex process model exists. For example, it might be sensible to check if simple PID controllers work sufficiently at the complex process model and to add complexity only if necessary. Compared to the initial ICACSD procedure the simulation results are more reliable because the simulations include the complete model identified by the *SNIP*.

In Figure 6-4 the simulation results of the basic and improved CACSD procedure are compared. Obviously the increased model quality utilising the *SNIP* improves the quality of the simulation results, which now comprise the coupling effects. The linear SISO control design already exhibits oscillations in the simulation based on the *SNIP* model. This clearly indicates that it would not be necessary to test this sub-optimal control design and that it is necessary to proceed with the next level of the *MES*.

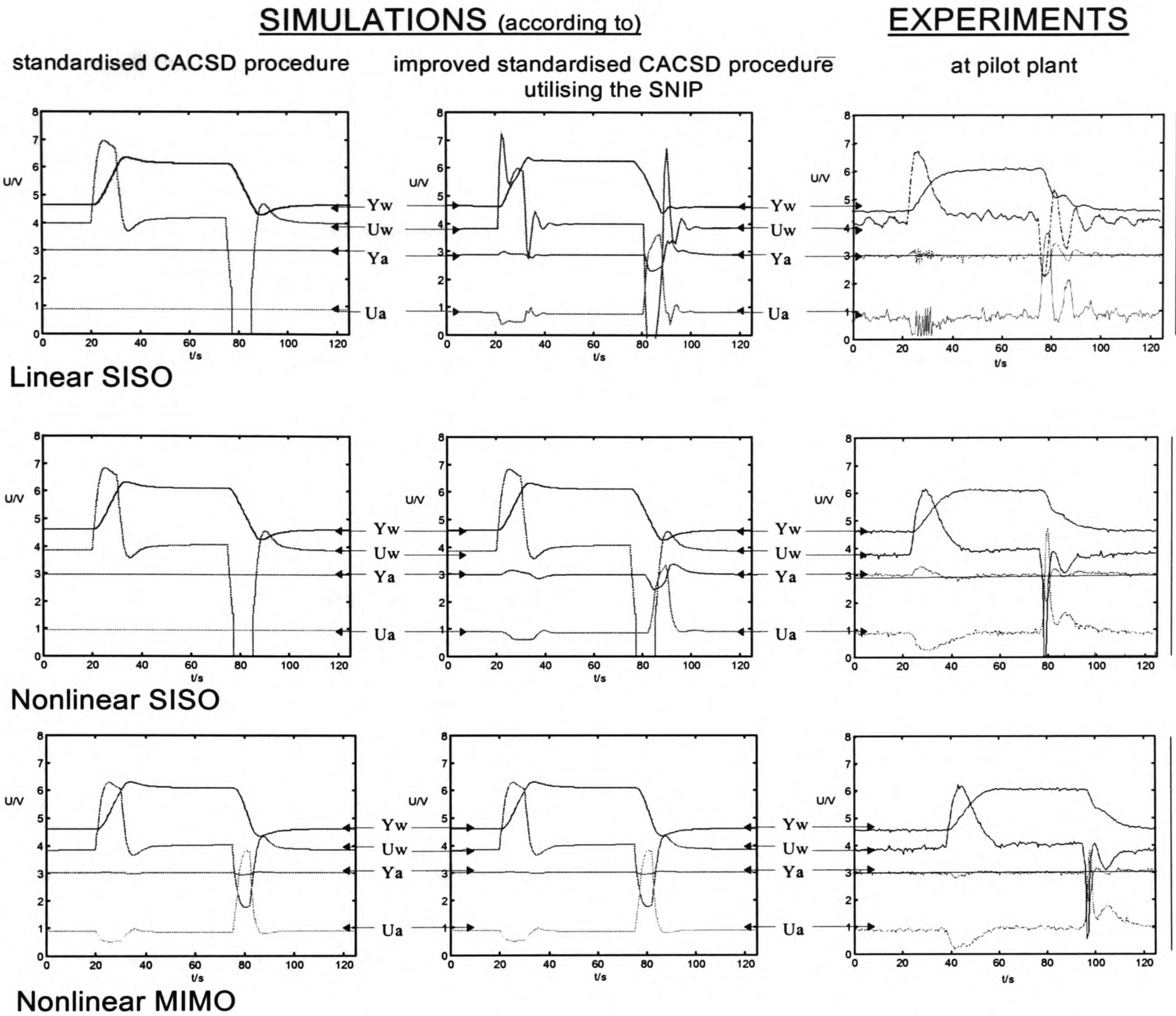


Figure 6-4. Simulation of the basic and improved ICACSD scheme compared to experimental results

6.3 Conclusions for this Chapter

A standardised identification procedure has been developed for Wiener- and Hammerstein-model structures. The pragmatic procedure offers a transparent method of experimental modelling for industrial users and starts with simple pre-experiments, which allow the determination of the static characteristic. This static characteristic is utilised to determine a suitable region for the final identification experiment.

If the experiment leaves the linear working range of the process then the static characteristic is used to compensate for the nonlinear effects. Therefore the procedure can be directed to the usage of conventional identification methods for linear dynamic characteristics. The identification procedure has been also linked to the ICACSD scheme evaluating nonlinearities and couplings of the process model with respect to the model evolution scheme (*MES*).

7 Development of an Effective Method for the Approximation of Static Characteristics

For the modelling of the nonlinear static part of the Wiener or Hammerstein-model within the *SNIP* procedure a reliable method was sought that was suitable for approximating multi-dimensional static characteristics, while being simple in application for static characteristics without memory. The method described in the following has some appealing properties:

- capable of interpolation and approximation (depending on one single 'form parameter')
- predictable fitting properties
- algorithmic simplicity
- simple applicability
- easily extendible to multi-dimensional problems for arbitrary distributed data points

This method, which has been derived from the generalised weighted mean, is called modified generalised weighted mean. It will be compared with the standard methods described in Section 4.7 and its relation to fuzzy and neural net approaches will be shown. Finally some practical aspects for the application are presented.

7.1 The Generalised Weighted Mean (GWM) Method

The *Generalised Weighted Mean (GWM)* has been introduced by Preuß (1994). As the name implies the idea for the GWM is based on the weighted mean. With the data points $P(x_i,y_i)$ and the individual weights g_i the weighted mean y is defined as:

$$y = \frac{g_1 y_1 + g_2 y_2 + \ldots + g_N y_N}{g_1 + g_2 + \ldots + g_N} = \frac{\sum_{i=1}^{N} g_i y_i}{\sum_{i=1}^{N} g_i} \qquad (7\text{-}1)$$

Therefore y represents the mean of the y-values, if $g_i = 1$ for $i = 1 \ldots N$. The bigger a single g_i the more weight is put on the i^{th} value, i.e.

$$\lim_{g_i \to \infty} y = y_i \, . \qquad (7\text{-}2)$$

In order to allow the description of a curve approximating the data points the individual weights must be replaced by a function that introduces the independent variable x. Considering equation (7-2) this function should reach its maximum at the exact data point, such that $y(x_i)$ is close to the measured y_i. A very simple function that could be used as initial guess is:

$$g_i(x) = \frac{c}{|x - x_i|}, \quad i = 1, \ldots, N \qquad (7\text{-}3)$$

However, this function has some disadvantages. Firstly it is not defined for $x=x_i$ and secondly it has a very steep slope such that the weight on the data y_i is rather big, resulting in an interpolation.

A much better function for this purpose is the Gaussian bell curve. Applied to the weights g_i it results:

$$g_i(x) = e^{-c(x-x_i)^2}, \quad i = 1,...,N \tag{7-4}$$

Being widely used in probability theory x_i can be interpreted as the mean of the corresponding curve. Here the parameter c determines the shape (breadth) of the curve and is therefore called the *form parameter*. Figure 7-1 illustrates the bell curve's dependency of $g_i(\Delta x)$ from c with $\Delta x=x-x_i$. The influence of the individual points decreases with the horizontal distance from the location x. According to McLain (1974) the choice of the bell curve is not entirely arbitrary. He showed for approximations utilising cubic splines that the effect of remote data points decreases approximately exponentially with the distance (though not with the square of the distance).

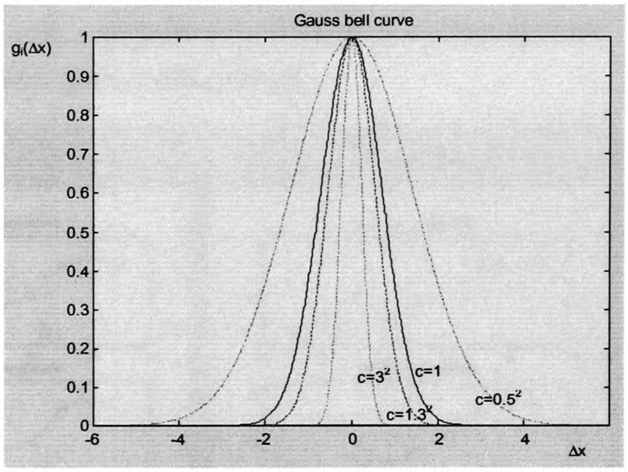


Figure 7-1. Influence of the form factor on the shape of the 'bell curves' with different

Inserting g_i into the weighted mean (7-1) one gets the equation for the GWM (Preuß 1994):

$$\boxed{y(x) = \frac{\sum_{i=1}^{N} e^{-c(x-x_i)^2} \cdot y_i}{\sum_{i=1}^{N} e^{-c(x-x_i)^2}}} \tag{7-5}$$

It is possible to determine the stiffness of the fit (i.e. the degree of approximation) by changing the form parameter c. The bigger c the stronger the weight on the data points. Therefore this simple function can realise an interpolation or approximation depending on c. However, Preuß (1994) does not provide guidelines for the selection of a good form parameter. Practically, the form parameter has to be iteratively changed as long as the fit of the resulting curve is not satisfactory. For example, if this intuitive procedure has to be used for interpolation, it might be possible that the data points are not met when the chosen form parameter is too small.

7.1.1 Interpolation

The capabilities of the GWM compared to linear-, spline- and Lagrange-interpolation are depicted in Figure 7-2. Besides the linear fit, a *3rd* order spline interpolation has been chosen as well as a polynomial (Lagrange-) interpolation with an order equal to the number of data points. Figure 7-2 contains three different data sets to clarify the main characteristics of the GWM. Data set 1 shows the effect of an outlier. Data set 2 describes a static characteristic by equally spaced data points and data set 3 is based on data set 2 but three data points have been cancelled such that the data points are not equally spaced any more.

The results of linear and GWM interpolation look quite satisfactory for the evenly spaced data set 1 of Figure 7-2. For data set 2 the spline method also gives a good result.

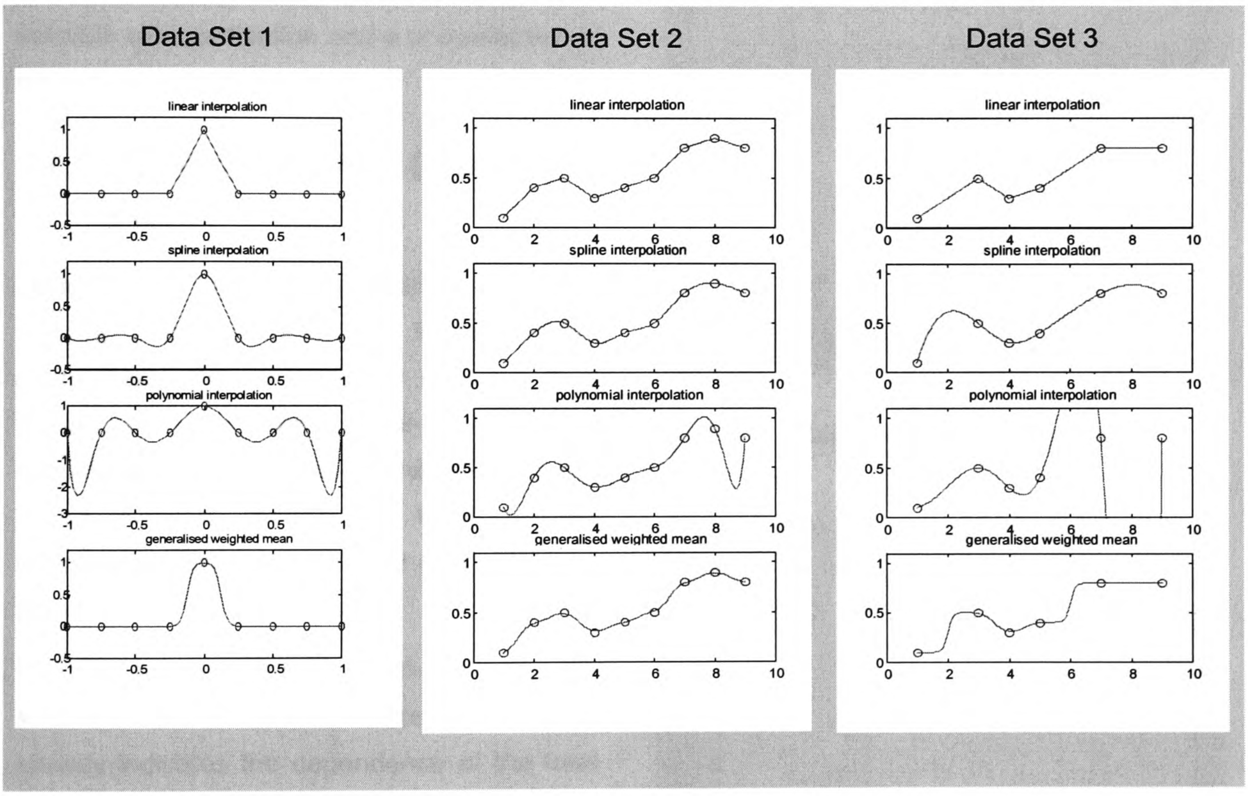


Figure 7-2. Comparison of different interpolation techniques with the GWM

However, data set 3 shows the shortcoming of the GWM, which still does not exhibit any oscillations as the Lagrange interpolation does but it has a tendency to build up plateaux around those data points in the neighbourhood of the cancelled data points. Altogether the result of the GWM for data set 3 is not as satisfactory as the result of the spline interpolation because the smoothness of the GWM result considerably varies for differently spaced data points.

7.1.2 Approximation

If the form parameter is decreased the GWM approximates the data points. Data set 3 was utilised for the evaluation of the GWM. Figure 7-3 compares the result of a *4th* order polynomial approximation and a distance-weighted least-squares approximation techniques developed by McLain (1974) with the result directly gained by the GWM method. Obviously the two curves resulting from the distance-weighted least-squares and GWM are quite close to the one that would be drawn by hand.

It is a main disadvantage of the GWM that the result cannot be gained in one single step. If the form parameter is too small the approximated line will be too straight. If the form parameter is too large plateaux will develop between the interpolation points. Therefore the next section presents a modification of the GWM that allows direct application utilising a sensible standardisation and a pre-selection of the form parameter.

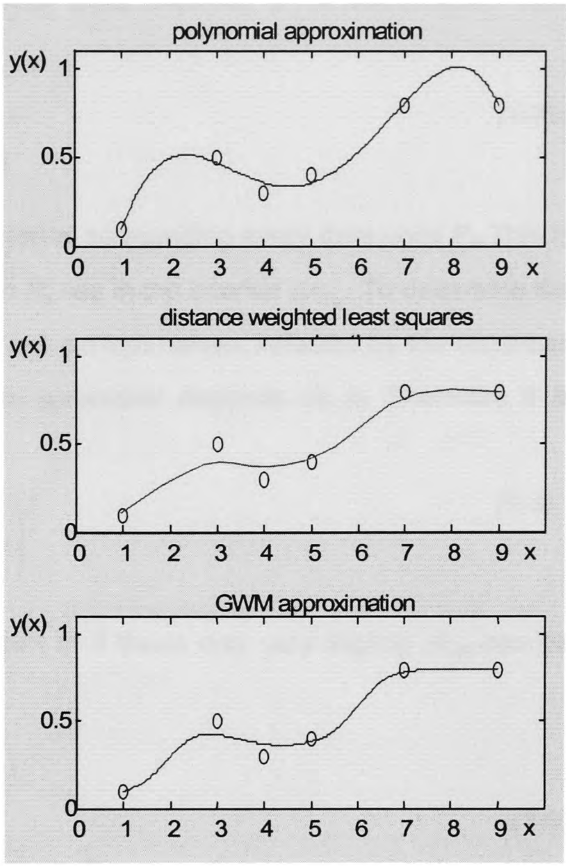


Figure 7-3. Approximation for data set 3

7.2 The Modified GWM (MGWM) Method

Up to now the main drawback of the GWM was that the form parameter had to be adapted interactively. For example for data set 3 of Figure 7-2 the form factor had to be considerably increased compared to data set 2 in an iterative manner to keep up the interpolation capabilities for all data points of the new data set.

It can be seen from data set 2 that the fit is the smoother the closer the data points. This already indicates the dependence of the form parameter *c* from $\Delta x_i = x_i - x_{i-1}$. Hence it is straightforward to modify the algorithm (7-5) respecting the varying distances between the data points.

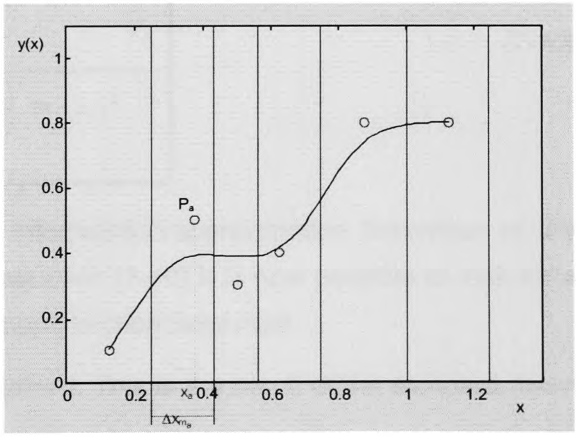


Figure 7-4. Determination of Δx_{m_i}

7.2.1 Modifications of the GWM Algorithm

Preuß (1994) only regards a form parameter *c* in the exponent of the bell curve (7-4). However, it is preferred to have this parameter squared, which is also better suited to describe the shape of the bell curve (Bronstein *et al.* 1995):

$$g_i(x) = e^{-a^2(x-x_i)^2}, \quad i = 1,\dots,N \tag{7-6}$$

Then the degree of approximation at the data points (x_i, y_i) depends on a standardised form parameter c_n divided by Δx_{m_i} and therefore:

$$a = \frac{c_n}{\Delta x_{m_i}} \tag{7-7}$$

The denumerator Δx_{m_i} represents the width of the interval surrounding every data point P_i. This is depicted in Figure 7-4, where the example data point P_a lies in the interval Δx_{m_a}. To determine the limits of Δx_{m_i} the distances between the data points are cut into halves. Affected by the distances between the data points the influence of the form parameter depends on x_i. Therefore it is sensible to replace c by:

$$c(x_i) = \left(\frac{c_n}{\Delta x_{m_i}} \right)^2 \tag{7-8}$$

If the distances between the data points are constant or if these only vary slightly Δx_{m_i} can be replaced by the mean of Δx_i, i.e. $\overline{\Delta x_i}$.

$$c = \left(\frac{c_n}{\overline{\overline{\Delta x_{m_i}}}} \right)^2 \tag{7-9}$$

Utilising this simplification the equation for the MGWM is:

$$\boxed{y(x) = \frac{\sum_{i=1}^{N} e^{-\left(\frac{c_n}{\overline{\overline{\Delta x_i}}} \right)^2 (x - x_i)^2} \cdot y_i}{\sum_{i=1}^{N} e^{-\left(\frac{c_n}{\overline{\overline{\Delta x_i}}} \right)^2 (x - x_i)^2}}} \tag{7-10}$$

Figure 7-5 illustrates the influence of c_n on the interpolation/approximation behaviour of the MGWM method on data set 3 of Figure 7-2. With equation (7-10) it is now possible to look for a default form parameter, which guarantees a good approximation behaviour.

For c_n=0 a straight line parallel to the abscissa is gained. This is the result of the standard mean $\overline{y} = \frac{\sum_{i=1}^{N} y_i}{N}$. For c_n=4 a satisfactory interpolation is only performed in the region where Δx is small, whereas for large Δx unacceptably sharp edges occur. The other curves lie in between. For practical application a form parameter has been chosen as outlined in Section 7.2.4. A way to consider varying Δx_{m_i} is discussed with the introduction of the adaptive form parameter in the next section.

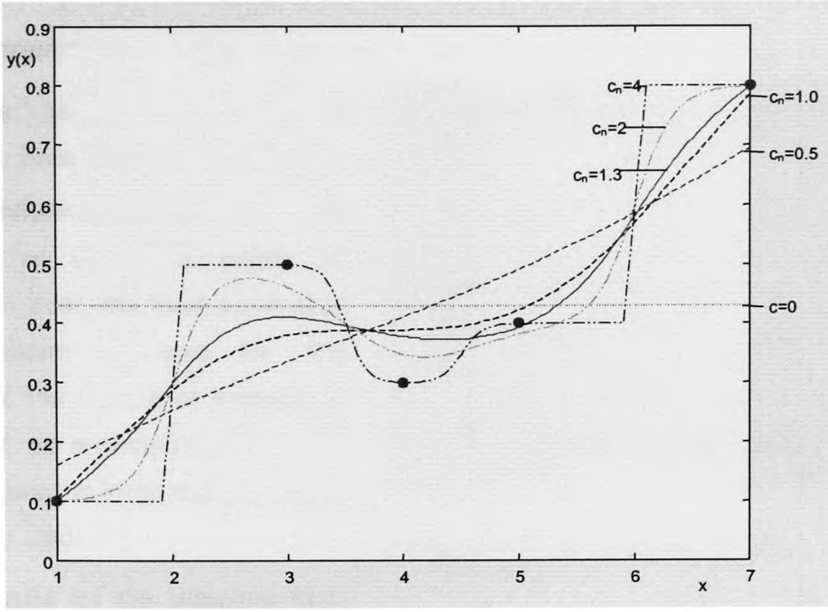


Figure 7-5. Influence of c_n on the interpolation / approximation behaviour

7.2.2 Regarding Varying Distances between the Data Points

In the previous section it has been already shown that the result of the MGWM interpolation still relies on the distance between the data points. Two strategies have been tested that can be applied in order to reduce the influences of differing Δx_i. The first one is a two-step method that performs a linear interpolation before the MGWM is executed, whereas the second one utilises a form parameter that depends on Δx_{m_i}.

Linear Interpolation before MGWM

The linear interpolation is especially appealing because of its simplicity. As a first step it can be applied to the data set in order to get evenly spaced data. Then the MGWM method is applied. The result of this combination of interpolation and MGWM is shown in Figure 7-6, where it is also compared to the MGWM without interpolation. This method is especially sensible for missing data points but not for arbitrarily varying Δx_i as often met in practical applications. Therefore it is superior to adapt the form parameter at Δx_{m_i} as explained in the following.

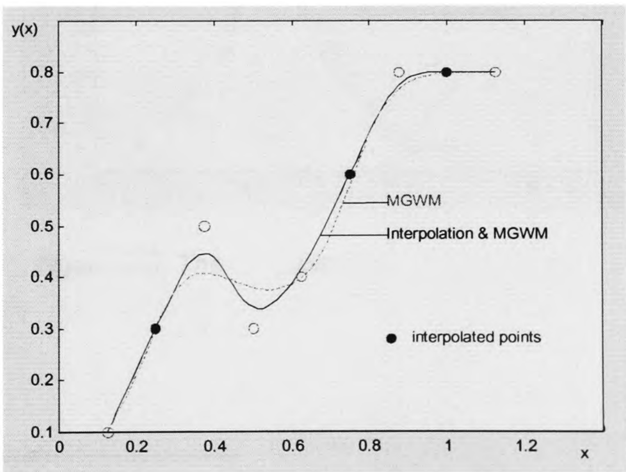


Figure 7-6. Influence of linear interpolation before application of the MGWM

Adaptive Form Parameter

The idea is to reduce the influence of the form parameter, where the Δx_{m_i} are big and to increase the influence otherwise. This is already stated for the data points in equation(7-8). However, the form parameter has to be determined also for the intermediate abscissa values of interest. A practical way to do so is shown in Figure 7-7 for data set 3, where the intermediate values have been interpolated.

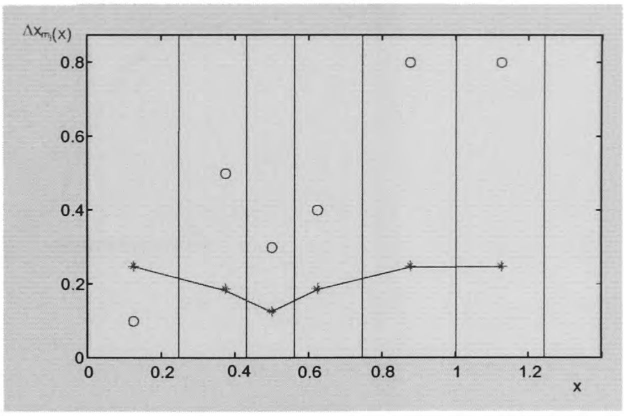


Figure 7-7. Interpolation of Δx_{m_i}

These are the basis for the adaptive form parameter called $c_a(x)$ which is calculated as follows:

$$\boxed{c_a(x) = \frac{c_n}{\Delta x_m(x)}} \tag{7-11}$$

The resulting adaptive form parameter depends on x. The intermediate points can be calculated utilising a simple linear interpolation (shown in Figure 7-7). The result for the adaptive form parameter is depicted in Figure 7-8 as solid line. Additionally Figure 7-8 shows the MGWM approximation of the adaptive form parameter as an alternative to the linear interpolation. The advantages of the latter method are outlined below. Utilising the adaptive form parameter $c_a(x)$ the equation for the MGWM becomes:

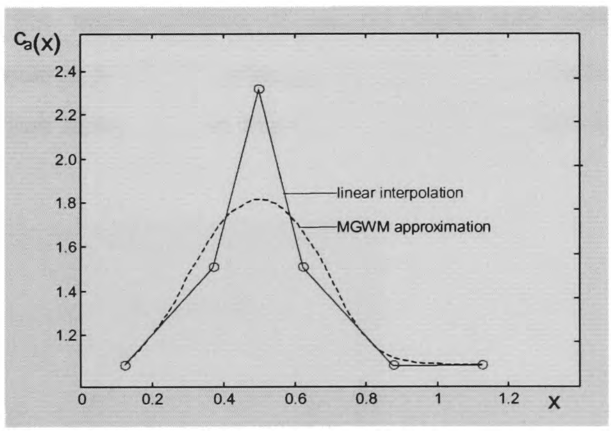


Figure 7-8. The adaptive form parameter c_a

$$\boxed{y(x) = \frac{\sum_{i=1}^{N} e^{-c_a^2(x)(x-x_i)^2} \cdot y_i}{\sum_{i=1}^{N} e^{-c_a^2(x)(x-x_i)^2}}} \tag{7-12}$$

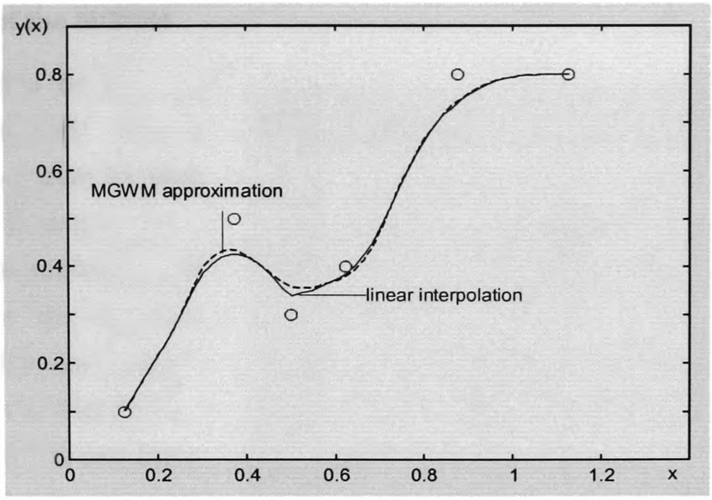


Figure 7-9. Interpolation / approximation of the varying Δx_i

Figure 7-9 (solid line) shows the result if equation (7-12) is applied to data set 3 utilising the linear interpolation. However, the result is not completely satisfactory because of the tiny edges in the resulting curve $y(x)$ that occur with a rapid change of the first derivative of $\Delta x_{m_i}(x)$. Therefore the MGWM has also been used to approximate $c_a(x)$, which leads to better results (dashed line of Figure 7-8 and Figure 7-9)

Depending on the form parameter used for the approximation of $\Delta x_{m_i}(x)$ more (big form parameter) or less (small form parameter) emphasis is put on the changes of Δx_{m_i}. A reasonable value for c_n that works practically well and that has been used in this work for approximation is presented in Section 7.2.4.

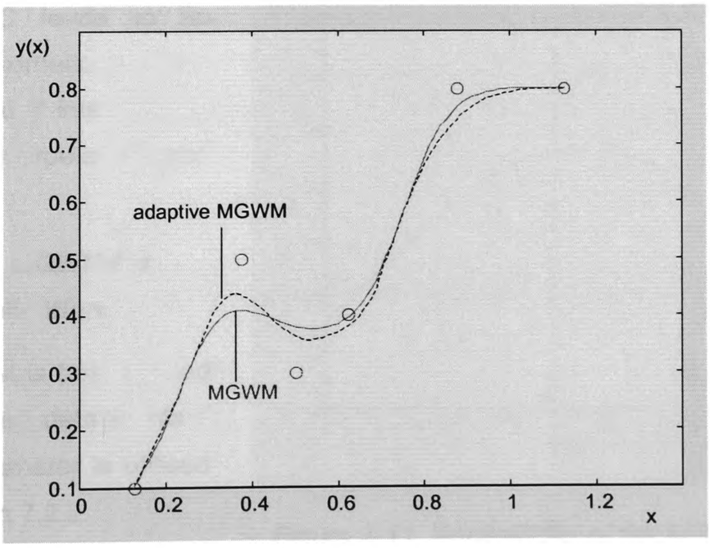


Figure 7-10. Influence of adaptation

Figure 7-10 finally compares the results that have been gained with the standard MGWM outlined in the preceding section and the adaptive MGWM. It can be clearly seen that the influences of differing Δx_i are reduced and that the result could be improved considerably.

7.2.3 Applicability of the MGWM

If the MGWM is applied as outlined above good approximations are yielded as for example shown for data set A of Figure 7-11. However, there still is a limitation of the applicability of the MGWM (and also the standard GWM) if the data points are strongly unequally distributed in *x*. Figure 7-11 displays the possible effects that occur if data set A is extended by one additional data point that is either too close to one data point (data set B of Figure 7-11) or that is far apart from the remaining data points (data set C of Figure 7-11). In both cases the adaptive form parameter is not able to satisfactorily improve the approximation. While in data set B mean values are approximated between the points close data points, which still might be satisfactory for some application, the additional data point in data set C leads to an unacceptable approximation. This could be circumvented if intermediate points would be interpolated and added.

7.2.4 The Adaptive MGWM for Practical Application

Altogether, the MGWM is best applied for unequally distributed data points if an adaptive form parameter is utilised as explained in Section 7.2.2.

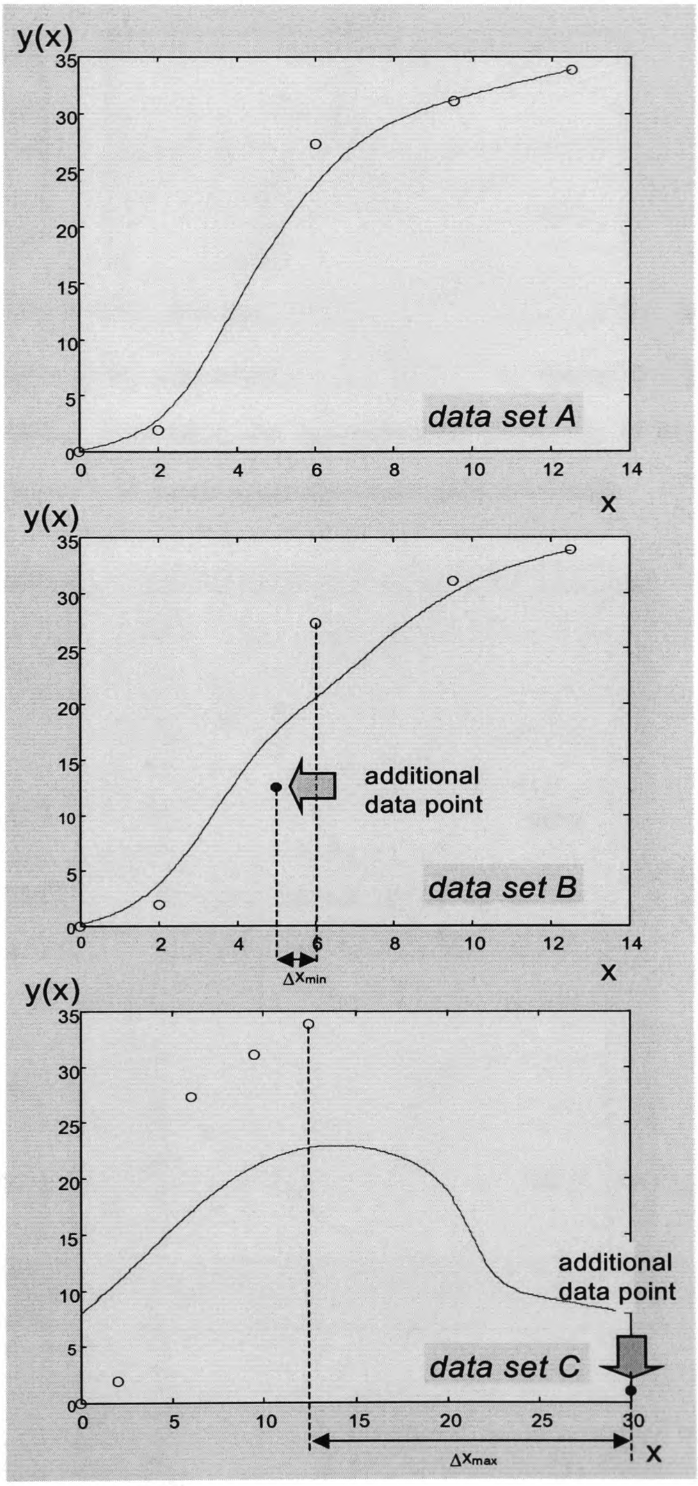


Figure 7-11. Applicability of the MGWM

Firstly the adaptive form parameter c_a is calculated from equation (7-10) applied to (7-11). This results is utilised for the MGWM. For implementation these equations result:

$$\boxed{y(x) = \frac{\sum_{i=1}^{N} e^{-c_a^2(x)(x-x_i)^2} \cdot y_i}{\sum_{i=1}^{N} e^{-c_a^2(x)(x-x_i)^2}}}$$

$$\boxed{c_a(x) = \frac{c_{n_2} \sum_{i=1}^{N} e^{-\left(\frac{c_{n_1}}{\Delta x_i}\right)^2 (x-x_i)^2}}{\sum_{i=1}^{N} e^{-\left(\frac{c_{n_1}}{\Delta x_i}\right)^2 (x-x_i)^2} \cdot \Delta x_{m_i}}}$$

In this work a form parameter was sought that exhibits a good approximation behaviour that is neither too stiff nor too straight. For practical work the standard form parameters c_{n_1} (determining the adaptation of the varying distances) and c_{n_2} (regulating the approximation behaviour of the data points) have been both set to 1.3. In Figure 7-12 some application examples are shown.

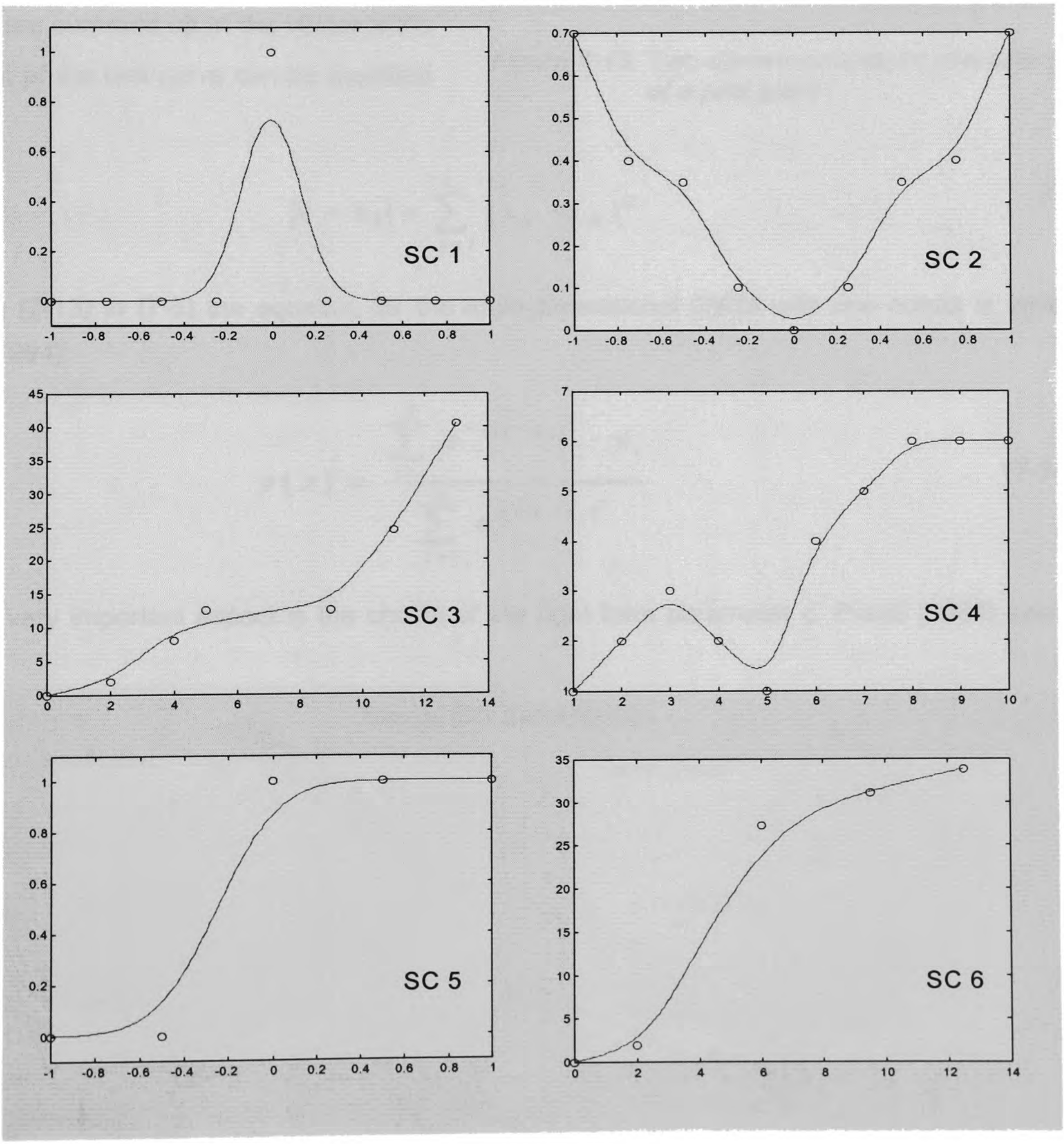


Figure 7-12. Application examples of the MGWM

7.3 Multi-Dimensional MGWM

In the previous sections the one-dimensional MGWM has been discussed in detail. Nevertheless many practical applications have more than one input and more than one output. An example for the MGWM approximation of a two-dimensional static characteristic that has been recorded at the pilot plant described in Section 5.2 is shown in Figure 7-13.

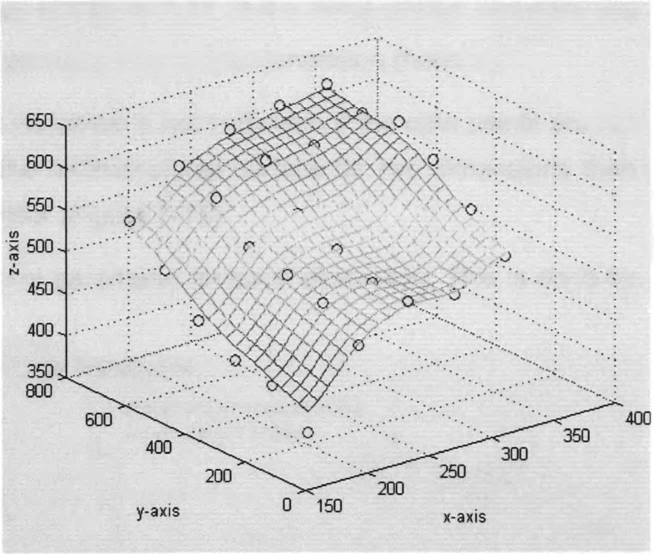


Figure 7-13. Two-dimensional static characteristic of a pilot plant

It is a main advantage of the GWM that the multi-dimensional case can be directly derived from equation (7-5). If the n_x inputs $x_1,...,x_{n_x}$ are summed up in the vector $\mathbf{x}$ the exponent of the bell curve can be modified by:

$$|\mathbf{x} - \mathbf{x}_i| = \sum_{j=1}^{n_x} (x_j - x_{ji})^2 \qquad (7\text{-}13)$$

Inserting (7-13) in (7-5) the equation for the multi-dimensional GWM with one output is yielded (Preuß 1994):

$$y(x) = \frac{\sum_{i=1}^{N} e^{-c|\mathbf{x}-\mathbf{x}_i|^2} \cdot y_i}{\sum_{i=1}^{N} e^{-c|\mathbf{x}-\mathbf{x}_i|^2}} \qquad (7\text{-}14)$$

Again a very important aspect is the choice of the right form parameter *c*. Preuß (1994) applies

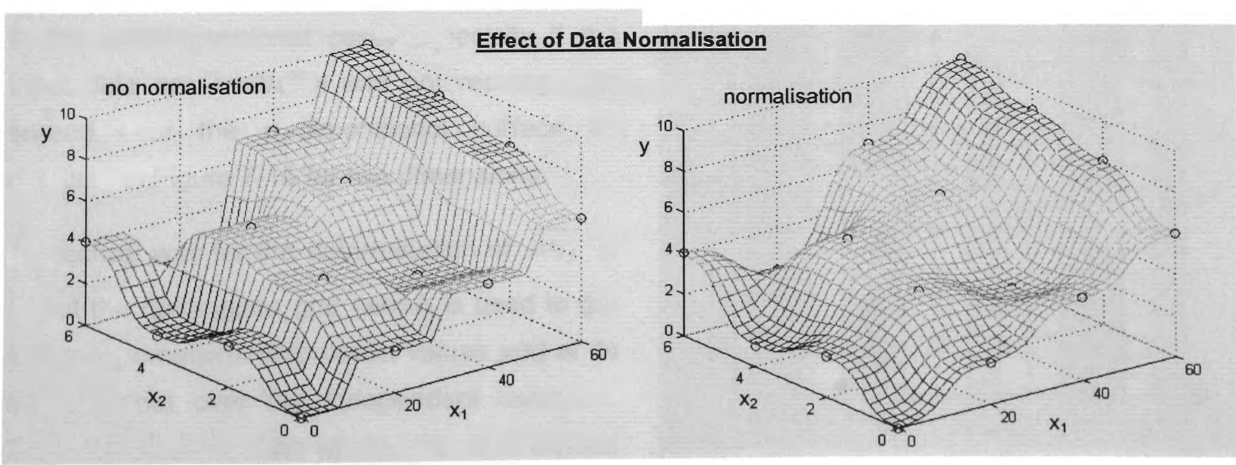


Figure 7-14. Effect of data normalisation utilising the multi-dimensional GWM

the same form parameter for all dimensions. However, this only works, if the data points are equally spaced in all dimensions (being rarely the case in practice) and if the data are normalised. The effect of normalisation of the data is shown in Figure 7-14. If the independent variables are not normalised then the approximation is done properly only in one dimension (here: x_2).

However, the normalisation of the independent variables is not sufficient. If the data points are not equally spread as shown in Figure 7-15 over the n-dimensional surface for two dimensions then the approximation behaviour can significantly differ (Figure 7-15).

Therefore it is necessary to select a suitable form parameter for each dimension. This is done by

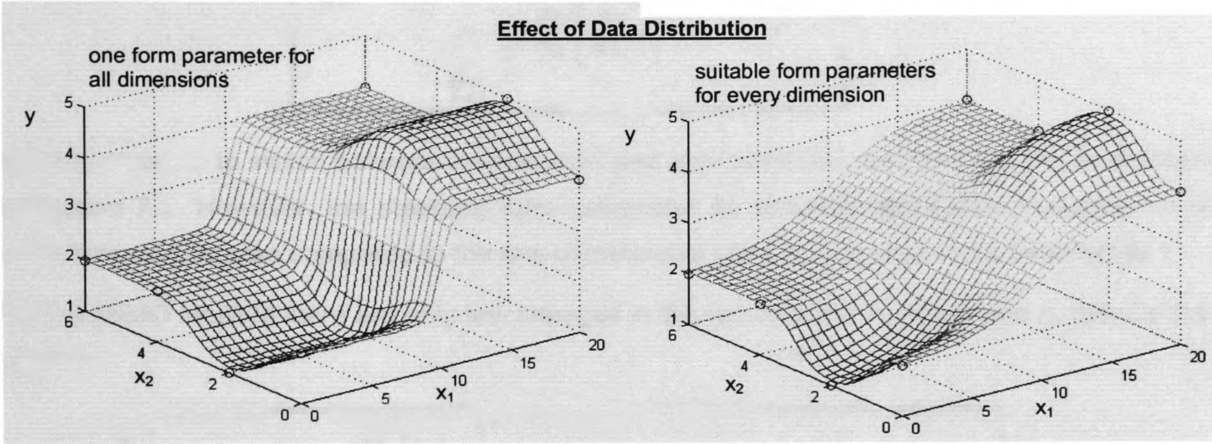


Figure 7-15. Effect of Data Distribution for Normalised Data

normalising the form parameter c_{n_j} for each dimension. Following equation (7-9) the normalised form parameter for the j^{th} dimension is:

$$c_{n_j} = \left(\frac{c_n}{\overline{\Delta x_{ij}}} \right)^2 \tag{7-15}$$

This means that the segmentation of the independent variables has to be considered. The determination of $\overline{\Delta x_{ij}}$ is more problematic than in the one-dimensional case, especially if the input data are arbitrarily and not rectangularly spread over the n_x-dimensional surface as depicted in Figure 7-16 for two dimensions.

A practical way for the determination of $\overline{\Delta x_{ij}}$ is to put the rectangular grid (which is used to get the approximation/interpolation values $y(\mathbf{x})$ at its intersections) over the independent variables. Then the breadth of the whole interval is divided by the number r of those segments that contain

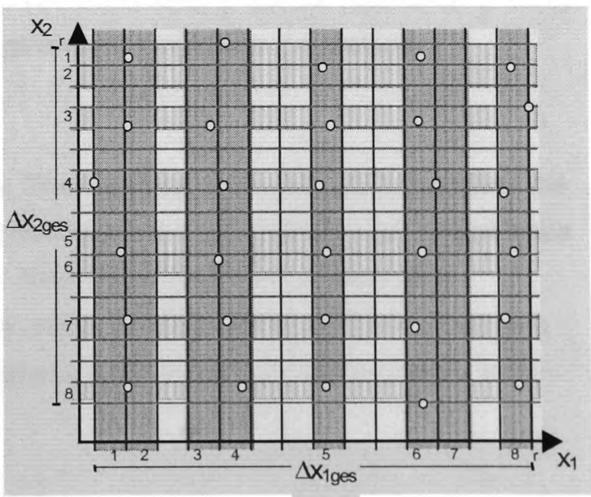


Figure 7-16. Segmentation of data points

variables associated with the data points (coloured and numbered in Figure 7-16)

$$\overline{\Delta x_{ij}} = \frac{\Delta x_{j\,ges}}{r} \tag{7-16}$$

This must be done for all n_x dimensions.

Herewith the equation of the multi-dimensional MGWM results:

$$\boxed{y(\mathbf{x}) = \frac{\sum_{i=1}^{N} e^{-\sum_{j=1}^{n_x}\left(\frac{c_n}{\overline{\Delta x_{ij}}}\right)^2 (x_j - x_{ji})^2} \cdot y_i}{\sum_{i=1}^{N} e^{-\sum_{j=1}^{n_x}\left(\frac{c_n}{\overline{\Delta x_{ij}}}\right)^2 (x_j - x_{ji})^2}}} \tag{7-17}$$

Still it is possible to switch between interpolation and approximation, just by tuning c_n as outlined in Section 7.1. However, the standard form parameter for practical application changes in the multi-dimensional case compared to the one-dimensional case. In this work it has been set to 1.

For the MIMO MGWM there are only few changes in the formula. For n_x inputs and n_y outputs the result is:

$$\boxed{y_k(\mathbf{x}) = \frac{\sum_{i=1}^{N} e^{-\sum_{j=1}^{n_x}\left(\frac{c_n}{\overline{\Delta x_{ij}}}\right)^2 (x_j - x_{ji})^2} \cdot y_{ik}}{\sum_{i=1}^{N} e^{-\sum_{j=1}^{n_x}\left(\frac{c_n}{\overline{\Delta x_{ij}}}\right)^2 (x_j - x_{ji})^2}} \quad \textit{for} \quad k = 1,\ldots, n_y} \tag{7-18}$$

7.4 Comparison with Fuzzy and Neural Net Approximations

It can be shown that it is possible to interpret the MGWM method as a special realisation of fuzzy and neural net approximations, which is valuable to gain more insight into the method.

7.4.1 Fuzzy Method and MGWM

Fuzzy methods are based on linguistic *'if–then'* rules. The conditioning if-part fuzzily describes the operating conditions under which the then-part must be executed. These if-then rules can be used to describe a control strategy or a system behaviour. Membership-functions translate the qualitative input into quantitative measures. The basic steps for the application of fuzzy logic are depicted in Figure 7-17. For a general introduction, please refer to Kahlert (1995a) and for more details to Zimmermann (1993).

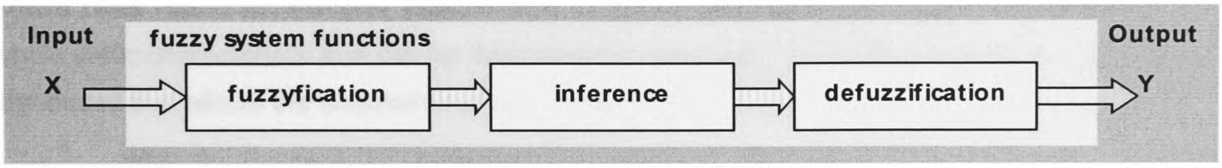


Figure 7-17. Basic fuzzy system functions

A detailed representation of these steps is illustrated by Figure 7-18. Assuming that N points $(P_1,...,P_N)$ describe a static one-dimensional relationship, then N rules can be derived:

$$\text{IF } (x = x_i) \text{ THEN } (y = y_i), \text{ for } i = 1,...,N \tag{7-19}$$

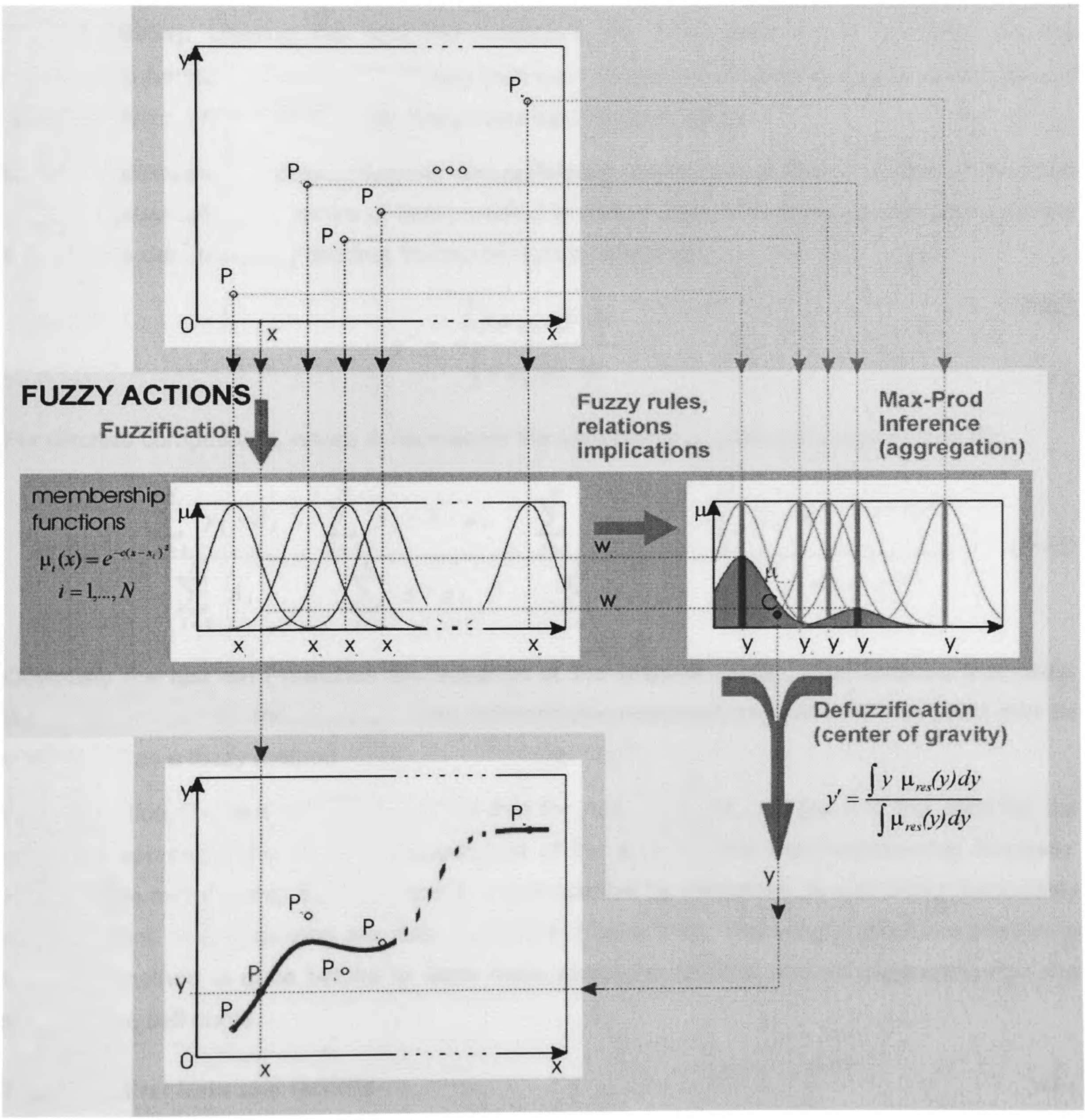


Figure 7-18. Fuzzy interpretation of the modified weighted mean

These rules represent discrete relationships so-called singletons. In order to approximate the whole static characteristic that can be described by these rules, a membership function μ_i must be introduced that allows the calculation of intermediate data points.

$$\text{IF } (x = \mu_i\, x_i) \text{ THEN } (y = \mu_i\, y_i), \text{ for } i = 1,\ldots,N \tag{7-20}$$

With μ_i being Gauss's bell curve:

$$\mu_i(x) = e^{-a^2(x-x_i)^2} \tag{7-21}$$

the first relation to the GWM has been realised. Having designed the membership functions the inference method has to be selected. The *Max-Prod-Inference* is a widely used inference method (Kahlert 1995a). Utilising the Max-Prod-Inference the truth value w_i is multiplied by the membership function of the corresponding then-part. In the one-dimensional case w_i equals μ_i. It can be seen from Figure 7-18 which fuzzy rules fire and their effects.

In order to calculate any desired $y(x)$ the defuzzyfication method has still to be chosen. The centre of gravity (also called the centre of area) method is widely applied to calculate the distinct output value that is determined by dividing the momentum by the area:

$$y' = \frac{\int y \mu_{res}(y)\ dy}{\int \mu_{res}(y)\ dy} \tag{7-22}$$

For discrete computation, where A_i represents the area below μ_{res} following equation holds:

$$y(x) = \frac{\sum_{i=1}^{N} y_i \cdot A_i}{\sum_{i=1}^{N} A_i} = \frac{\sum_{i=1}^{N} y_i \cdot A \cdot \mu_i}{\sum_{i=1}^{N} A \cdot \mu_i} = \frac{\sum_{i=1}^{N} y_i \cdot \mu_i}{\sum_{i=1}^{N} \mu_i} = \frac{\sum_{i=1}^{N} y_i \cdot e^{-a^2(x-x_i)^2}}{\sum_{i=1}^{N} e^{-a^2(x-x_i)^2}} \tag{7-23}$$

Obviously the last term matches the equation of the MGWM (7-10). This confirms that under certain conditions for membership functions, inference and defuzzyfication the MGWM can be understood as a fuzzy method.

From equation (7-23) it can also be seen that for $A_i = A$ for $i=1,\ldots,N$ (as it is the case for the proposed approach) the result is independent of the areas of the then-membership functions. Hence these membership functions could be represented by singletons as well being numerically more efficient. The singletons are also depicted in Figure 7-18. The insight about the relation to the fuzzy method is quite helpful to learn more about the MGWM and its dependence on the shape of the bell curve.

7.4.2 Neural Nets and MGWM

Neural nets are very capable of describing multi-dimensional static characteristics. They have the potential to perform interpolation and approximation and it is therefore interesting to investigate the relation between the MGWM and neural net methods.

In the following it will be shown that the MGWM method can be interpreted as a special form of Radial-Basis Function (RBF) networks. The RBF is a static feed-forward net as it is depicted in Figure 7-19.

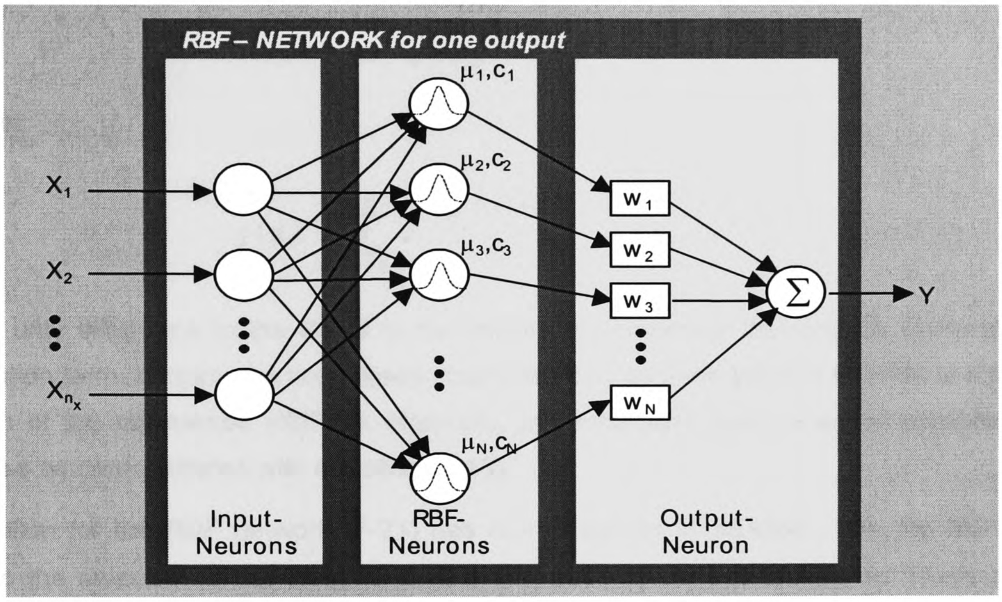


Figure 7-19. Structure of an RBF net with one output

The input-neurons only serve for the distribution of information to the neurons and the output is directly calculated from the transfer function $f_i(x)$ that is called Radial Basis Function having the form of a multi-dimensional Gaussian curve. The i^{th} inner neuron is defined by (Kahlert 1995a):

$$f_i(\mathbf{x}) = e^{-\sum_{j=1}^{n_x} \frac{(x_j-\mu_{ji})^2}{2\sigma_{ji}^2}} \quad for \quad i = 1,\ldots,N \tag{7-24}$$

Here the parameters μ_{ij} and σ_{ij} determine the shape of the Gaussian curve related to the i^{th} data point of the j^{th} input (n_x being the number of inputs). Here μ_{ij} can be interpreted as the mean of the corresponding curve and σ_{ij} as standard deviation being the shaping element of the curve. An RBF net contains therefore only two degrees of freedom per input. Finally the linear output neurons provide the weights w_i for the transfer functions as depicted in Figure 7-19. The equation for the whole RBF is therefore:

$$y = \sum_{i=1}^{N} w_i f_i(\mathbf{x})$$

$$y = \sum_{i=1}^{N} w_i e^{-\sum_{j=1}^{n_x} \frac{(x_j-\mu_{ji})^2}{2\sigma_{ji}^2}} \tag{7-25}$$

The agreement with the basic GWM equation (7-5) becomes clear with following replacements:

$$\begin{aligned} w_i &= y_i \\ c_{ij} &= \frac{1}{2\sigma_{ij}^{2}} \\ x_{ij} &= \mu_{ij} \end{aligned} \tag{7-26}$$

resulting in:

$$y(\mathbf{x}) = \sum_{i=1}^{N} e^{-\sum_{j=1}^{n_x} c_{ji}(x_j - x_{ji})^2} \cdot y_i \tag{7-27}$$

Then the only difference to the GWM is the missing denumerator that can be interpreted as a normalisation term. Utilising the normalised structure it can be seen that the MGWM is a particular realisation of the normalised RBF-net. Naturally, this evaluation can be easily extended to the MIMO case as demonstrated with equation (7-18).

The equation for the RBF network (7-21) has more degrees of freedom than the MGWM with respect to the selection of the weights w_i and the mean of the bell curves μ_{ij}. There are quite simple learning rules for the adaptation of the parameters describing the bell curves but the weights of the output neurons have to be determined utilising time-consuming optimisation and learning methods.

7.4.3 Conclusion of the Comparison

In the preceding sections the similarities between the MGWM and particular realisations of a fuzzy as well as an RBF method have been demonstrated. Of course, this also indicates that Fuzzy and RBF methods can be transformed into one another, which has already been shown by Jang and Sun (1993). All methods including the GWM exhibit a strong local behaviour. This is reasoned in the shape of the bell curve. The thinner the curve the less the influence of the inputs. This local behaviour is also responsible for the simple learning rules in the RBF networks. All methods are universal approximators utilising scaled bell curves. Although derived from different origins and with different interpretations of the data processing the three methods can yield functionally equivalent models. The MGWM method is superior to the RBF network and the Fuzzy realisation in a sense that it neither needs time consuming learning rules nor the fuzzy framework. The MGWM it is not as flexible as the other methods that provide more degrees of freedom. However, this is advantageous for direct application without user interaction.

7.5 Aspects of Application

There are many possibilities for adapting the MGWM method to special needs, some of which are discussed in the following. For the sake of readability all aspects are explained for the one-dimensional MGWM.

7.5.1 Margin Correction

A simple method for the correction of the values at the margin that works very well for the one-dimensional MGWM is to add two data points - one at each margin being a linear extrapolation of the next 2 data points . All results presented in this chapter have been achieved utilising margin correction.

7.5.2 Improving Numerical Efficiency

It could be already assumed from the fuzzy interpretation shown in Figure 7-18 that it is not necessary to calculate each bell curve for all x, but to calculate only those values x in the region of x_i that contribute sufficiently to the result. The number of these values depends directly on the shape of the bell curve. Therefore if $g(\Delta x)<\varepsilon$ for very small ε, as seen in Figure 7-1, these need not be considered. For example it is sufficient for $c = 1.3^2$ to consider $|\Delta x| < 2$.
If the adaptive MGWM (7-12) is used with the MGWM approximation for the form parameter then this approximation can be directly implemented into the algorithm such that the Gaussian functions must be calculated only once.

7.6 Conclusion for this Chapter

It has been shown that the generalised weighted mean (GWM) method provides sensible results for interpolation and approximation of static characteristics if user interaction is provided. The GWM has been modified for practical application in order to allow simple application and to gain sensible results without trial and error. The Modified GWM (MGWM) does not need user assistance and still is numerically simple even in case of multi-dimensional static characteristics.

Additionally the MGWM method was discussed as a special realisation of fuzzy and neural network methods. Finally some aspects of the practical application of the MGWM method have been explained in order to improve the applicability of the method. This method has been developed to serve as the main method for the approximation of static characteristics within the prototype realisation outlined in Section 9.2.

8 Development of a Reliable Method for the Identification of Linear Dynamic Models

It has been already mentioned in Section 4.4.1 that it can be beneficial to firstly identify a nonparametric model before the final parametric identification. This way a two-step identification of the linear dynamic model is performed. The nonparametric model can be generally estimated without assumptions about the model structure; it gives valuable information for the final parametric identification and is described by less data than the original data set thus concentrating the process information. This is especially useful for the determination of order and deadtime allowing the parametric estimator to work on reduced data sets, which can considerably decrease computing effort.

A specifically flexible and useful two-step identification method suitable for a wide range of input signals is based on auto- and cross-correlation functions, which is called correlation least-squares method (Cor-LS, Isermann *et al.* 1974). In the first step a nonparametric model is identified from correlation functions without the need for initial guesses concerning model structure and deadtime. In the second step the parametric model is identified from the correlation functions. The Cor-LS method has been simulatively compared to many other methods and it has been shown that the Cor-LS method is well suited to tackle practical identification tasks (Isermann 1992). The performances of LS, GLS, ELS, ML and Cor-LS method have been already compared (Table 4-2).

However, for reliable application it is necessary to improve the Cor-LS method with respect to numerical behaviour and function. Therefore the LS estimator - the second step of the Cor-LS method - has been reformulated introducing order recursive structures, which allow the parallel estimation of models from order 1 to n (n being the maximum order) without increasing the computational load. This is based on a combination of square root filtering using the QR matrix decomposition techniques with an order recursive estimation scheme, which has been proposed for the LS method by Niu and Fisher (1994) and called the multiple model least square (MMLS) estimation method.

In this chapter the combination of correlation techniques with the MMLS method is elaborated. The resulting new two-step method is called CorMMLS (correlation multiple model least squares Körner and Schumann 1998b) and yields unbiased estimates also in the presence of coloured noise. Furthermore modifications and extensions of the CorMMLS method are outlined to solve practical application problems.

8.1 Components for the Method

The two components of the CorMMLS method will be introduced next. It starts with a description of the Cor-LS method and its properties. Afterwards the MMLS method is explained before both methods are merged. For the sake of simplicity both methods will be introduced for single-variable processes without deadtime.

8.1.1 The Correlation Least Squares (Cor-LS) Method

Assuming an arbitrary stationary (pseudo-)random stochastic input test signal as discussed in Section 4.1.1 the following correlation functions can be defined. For the cross-correlation function of input- and output signal:

$$\phi_{uy}(\tau) = \lim_{N\to\infty} \frac{1}{N+1} \sum_{k=0}^{N} u(k-\tau)\, y(k) \tag{8-1}$$

and for the auto-correlation function of the input signal *u*:

$$\phi_{uu}(\tau) = \lim_{N\to\infty} \frac{1}{N+1} \sum_{k=0}^{N} u(k-\tau)\, u(k) \tag{8-2}$$

with τ being the discrete shift in time.

With *d* being the deadtime the difference equation of the process model can be written as:

$$\mathrm{y(k)} + a_1 \cdot \mathrm{y(k-1)} + \cdots + a_n \cdot \mathrm{y(k-n)} = b_1 \cdot \mathrm{u(k}-d-1) + \cdots + b_n \cdot \mathrm{u(k}-d-\mathrm{n)} + n(k) \tag{8-3}$$

y(k) and *u(k)* represent the process output and input signals respectively, *n(k)* is a coloured noise signal with zero mean, a_i, b_i *(for i = 1,...,n)* are the model parameters and *d* is the discrete deadtime. If this difference equation is multiplied by $u(k-\tau)$ and summed up over time, the expectations of the terms can be determined from the correlation functions:

$$E\{u(k-\tau)\, y(k)\} = \phi_{uy}(\tau) = \lim_{N\to\infty} \frac{1}{N+1} \sum_{k=0}^{N} u(k-\tau)\, y(k) \tag{8-4}$$

Herewith the process model parameters can be related to the correlation functions:

$$\phi_{uy}(\tau) + a_1\phi_{uy}(\tau-1) + \ldots + a_n\phi_{uy}(\tau-n) = b_1\phi_{uu}(\tau-d-1) + \ldots + b_n\phi_{uu}(\tau-d-n) + \phi_{un}(\tau) \tag{8-5}$$

For a finite number of samples the correlation functions can be estimated by (Isermann 1992):

$$\boxed{\hat{\phi}_{uy}(\tau) = \frac{1}{N+1} \sum_{k=0}^{N} u(k-\tau)\, y(k)} \tag{8-6}$$

$$\boxed{\hat{\phi}_{uu}(\tau) = \frac{1}{N+1} \sum_{k=0}^{N} u(k-\tau)\, u(k)} \tag{8-7}$$

As Equation (8-5) is also valid for the estimates as well:

$$\hat{\phi}_{uy}(\tau) + a_1\hat{\phi}_{uy}(\tau-1) + \ldots + a_n\hat{\phi}_{uy}(\tau-n) = b_1\hat{\phi}_{uu}(\tau-d-1) + \ldots + b_n\hat{\phi}_{uu}(\tau-d-n) + \hat{\phi}_{un}(\tau) \tag{8-8}$$

Compared to the standard least squares method this equation contains the estimated correlation functions instead of the data vector. In this context correlation can be interpreted as a special prefiltering technique.

As this is the only difference, the estimation can be directly applied utilising the new data vector:

$$\boxed{\psi_c(\tau) = [-\hat{\phi}_{uy}(\tau-1), -\hat{\phi}_{uy}(\tau-2), \ldots, -\hat{\phi}_{uy}(\tau-n), \hat{\phi}_{uu}(\tau-d-1), \hat{\phi}_{uu}(\tau-d-2), \ldots, \hat{\phi}_{uu}(\tau-d-n)]^T} \tag{8-9}$$

in order to estimate the parameter vector:

$$\boxed{\hat{\boldsymbol{\theta}} = [\boldsymbol{\Psi}_c^{\,T}\boldsymbol{\Psi}_c]^{-1}\boldsymbol{\Psi}_c^{\,T}\hat{\boldsymbol{\varphi}}_{uy}} \tag{8-10}$$

Convergence

The last term of Equation (8-8) represents the estimated correlation of input signal and noise. Assuming that the input signal *u* is independent of the noise signal *n* the following holds:

$$\lim_{N\to\infty} E\{\hat{\phi}_{un}(\tau)\} = 0 \tag{8-11}$$

as long as:

$$E\{n(k)\} = 0 \text{ and } E\{u(k-\tau)n(k)\} = 0 \tag{8-12}$$

This means that the influence of the corresponding estimated cross-correlation function in Equation (8-8) disappears with increasing *N*. This effect has been demonstrated in many simulation examples by Isermann (1992). If the model represents the process structure and order, then also the equation error converges to zero:

$$\lim_{N\to\infty} E\{\mathbf{e}\} = \lim_{N\to\infty} E\{\hat{\phi}_{uy} - [\hat{\phi}_{uy}]_M\} = 0 \tag{8-13}$$

Data Reduction

The information necessary for the estimation procedure is contained in those correlation values that significantly differ from zero. Therefore the selection of relevant correlation function values reduces the data vector while increasing the efficiency of the estimation.

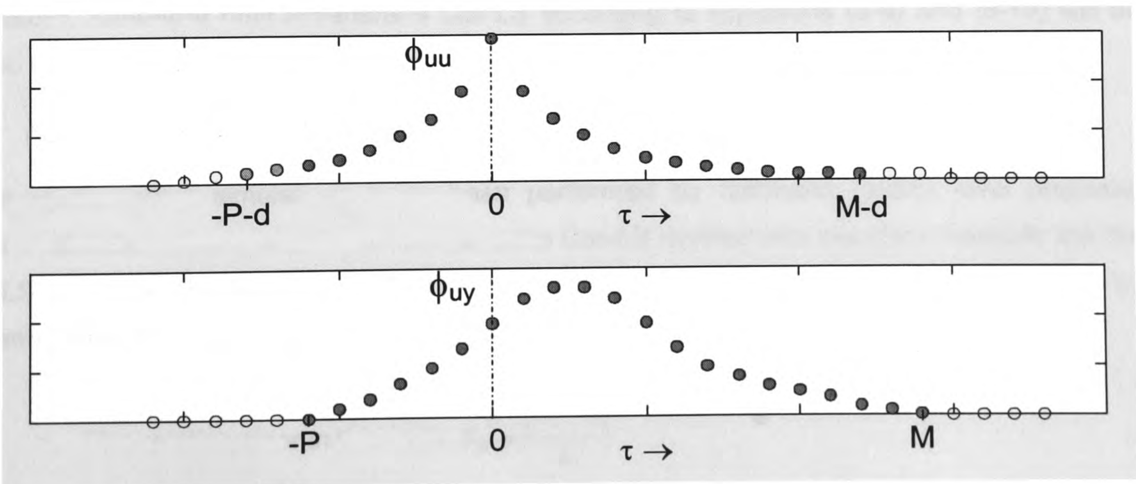


Figure 8-1. Limits P and M for auto- and cross-correlation function

Due to the distinct shape of correlation functions as shown in Figure 8-1 (here for a coloured noise input signal) two limits *-P* and *M* (with $M \geq \tau \geq -P$) can be determined that are valid for the cross-correlation and autocorrelation function determining the relevant data for the estimation. Outside the interval $M \geq \tau \geq -P$ the cross-correlation functions are close to zero.

White Noise as Input Signal

All the preceding considerations are true for arbitrary stationary coloured stochastic input signals with sufficiently high order such that $\det[\psi_c{}^T \psi_c] \neq 0$. However, the preceding equations become simpler if the input signal is white noise with following autocorrelation function:

$$\hat{\phi}_{uu}(\tau) = 0 \text{ for } \tau \neq 0 \text{ and } \hat{\phi}_{uu}(0) \neq 0 \tag{8-14}$$

In this case the cross-correlation function is a scaled weighting function and therefore

$$\hat{\phi}_{uy}(\tau) = 0 \text{ for } \tau < 0 \text{ and}$$

$$\hat{g}(\tau) = \frac{1}{\hat{\varphi}_{uu}(0)} \hat{\varphi}_{uy}(\tau) \tag{8-15}$$

Utilising Equations (8-14) and (8-15) the matrix Ψ_g becomes:

$$\Psi_g = \begin{bmatrix} -g(d) & \cdots & -g(d-n) & 1 & 0 & \cdots & 0 \\ -g(1+d) & \cdots & -g(1+d-n) & 0 & 1 & & 0 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots \\ -g(M+d-1) & \cdots & -g(M+d-n) & 0 & 0 & 0 & 1 \end{bmatrix} \tag{8-16}$$

and therefore:

$$\hat{\boldsymbol{\theta}} = [\boldsymbol{\Psi}_g{}^T \boldsymbol{\Psi}_g]^{-1} \boldsymbol{\Psi}_g{}^T \mathbf{g} \tag{8-17}$$

Although theoretically interesting the latter considerations are too restrictive for an estimation method aimed at industrial users because a white noise input signal cannot be always guaranteed. Therefore only the general Cor-LS according to Equations (8-9) and (8-10) will be realised in a numerically improved fashion as outlined in Section 8.1.2.

Practical Considerations and Application

Numerous numerical simulations have been performed by Isermann (1992), who proposed different validation criteria in order to compare the Cor-LS method with standard methods like the LS, GLS, ELS, IV and ML (Isermann 1973). The following quality measures were checked by Isermann, while the computing effort was also assessed:

- The relative error in the gain: $\delta_K = \dfrac{|\hat{K} - K|}{K}$

- The relative error in each single parameter: $\delta_{\theta_i} = \frac{\Delta\theta_i}{\theta_i} = \frac{\left|\hat{\theta}_i - \theta_i\right|}{\theta_i}$
- The mean squared error in the parameters: $\delta_\theta = \sqrt{\sum_{i=1}^{p}\left(\frac{\Delta\theta_i}{\theta_i}\right)^2}$

  with p being the number of estimated parameters

Furthermore Isermann (1992) provides many practical examples for successful application of the Cor-LS method, which has been found superior for various processes in the process industry. For example different heat exchangers, a climate process and industrial dryers have been identified with the purpose of control utilising the Cor-LS method.

Altogether Isermann (1992) showed that the Cor-LS method:

- generally leads to a good model quality compared to the other methods,
- is applicable for a broad range of disturbances,
- needs only small computing effort,
- is insensitive with respect to Y_0 as long a the mean of $u(k)$ equals zero

with the disadvantage that the disturbance is not modelled. However, this is not necessary for this work as discussed in Section 4.4.3 that also provides a table for direct comparison of the different methods for identification.

8.1.2 The Multiple Model LS (MMLS) Method

The LS estimation method has been dominating the field of parameter estimation for many years because of its theoretical simplicity and convenient applicability. However, especially if the model is over-parameterised the numerical performance becomes poor. Naturally, also the results of the Cor-LS method become poor for over-parameterised models and ill-conditioned covariance matrices, which occur for small excitation of the process through the input signal.

Therefore a method was sought that improves the numerical properties of the LS estimator. Square-root filtering is often utilised to improve the numerical behaviour by reducing the size of the elements in the covariance matrix (Isermann 1992). However, Niu (1994) proposed an estimation method with an order recursive structure, which simultaneously produces multiple models from order 1 to a pre-specified order n. Therefore it has been called the multiple model least squares method (MMLS). The MMLS method utilises a factored structure, which makes it less vulnerable to numerical problems. Niu and Fisher (1994) demonstrated that the multiple model structure makes it possible to avoid over-parameterisation and to detect underparameterisation. They proved that numerical problems in the higher order models do not affect lower order models, which is superior to the standard LS estimators where numerical problems affect the entire covariance matrix. The application of the MMLS method is demonstrated in the following.

Assume that the process under investigation can be represented by following linear discrete time difference equation:

$$y(k) + a_1 \cdot y(k-1) + \cdots + a_n \cdot y(k-n) = b_1 \cdot u(k-1) + \cdots + b_n \cdot u(k-n) + n(k) \tag{8-18}$$

The key to the MMLS method lies in the reformulation of the data regression vector.

$$\boxed{\psi_a(k) = [-y(k-n), u(k-n), .., -y(k-1), u(k-1), -y(k)]^T} \tag{8-19}$$

Obviously this data vector differs from the conventional LS data vector with its grouped elements *{y(·),u(·)}* and the inclusion of the current process output *y(k)*. Therefore $\psi_a(k)$ is called the augmented data vector. All realisations of the MMLS method depend on the augmented data matrix Ψ_a.

$$\boxed{\Psi_a = \begin{bmatrix} \psi_a^T(1) \\ \psi_a^T(2) \\ \vdots \\ \psi_a^T(N) \end{bmatrix}} \tag{8-20}$$

From the augmented data matrix Ψ_a the augmented covariance matrix **C** or its inverse, the augmented information matrix **S,** is generated:

$$\mathbf{C} = [\Psi_a^T \Psi_a]^{-1} \quad \text{and} \quad \mathbf{S} = [\Psi_a^T \Psi_a] \tag{8-21}$$

By decomposition of the augmented covariance or information matrix the parameter matrix **P** including an accompanying loss function matrix **L** are gained, which contain the parameters and losses respectively (for all models from order 1 to *n*). The implementation of the MMLS can be performed in three different ways (Niu 1994):

1. The augmented data matrix Ψ_a can be directly decomposed with a QR type decomposition, which is well known for its superior numerical performance (Niu *et al.* 1996) and thus is the recommended technique for batch implementation.
2. In case of batch implementation the augmented information matrix **S** is easily available, which can be decomposed utilising Cholesky / LU / LDU / LDL^T decomposition techniques to produce the parameter and loss function matrix.
3. The augmented covariance matrix **C** is constructed and decomposed to produce the parameter and loss function matrix utilising the UDU^T factorisation method, which is also convenient for recursive implementation of Bierman's UD-factorisation technique.

The different implementations are summarised in Table 8-1. Detailed descriptions of the methods and simulation examples are provided by Niu (1994). Particularly the QDU decomposition method has been found to represent a good compromise between numerical performance and algorithmic complexity for batch identification (Niu *et al.* 1995).

Table 8-1. Decomposition methods for MMLS

Method	Based on	Basic Formula	Parameter Matrix **P**	Loss Function Matrix **L**
QR	data matrix Ψ_a	Ψ_a = **QR**	**P** = **R**$^{-1}$ diag(**R**)	**L** = [diag(**R**)]^2
QDU		Ψ_a = **QDU**	**P** = **U**$^{-1}$	**L** = **D**^{2}
LU	information matrix **S**	**S** = **LU**	**P** = **L**$^{-T}$	**L** = diag(**U**)
LDL^T		**S** = **LDL**^{T}	**P** = **L**$^{-T}$	**L** = **D**
Cholesky		**S** = **GG**^{T}	**P** = **G**$^{-T}$ diag(**G**)	**L** =[diag(**G**)]^2
UDU^T	covariance matrix **C**	**C** = **UDU**^{T}	**P** = **U**	**L** = **D**$^{-1}$

Therefore its implementation will be explained in more detail leading to the specific parameter and loss function matrices shown in Figure 8-2.

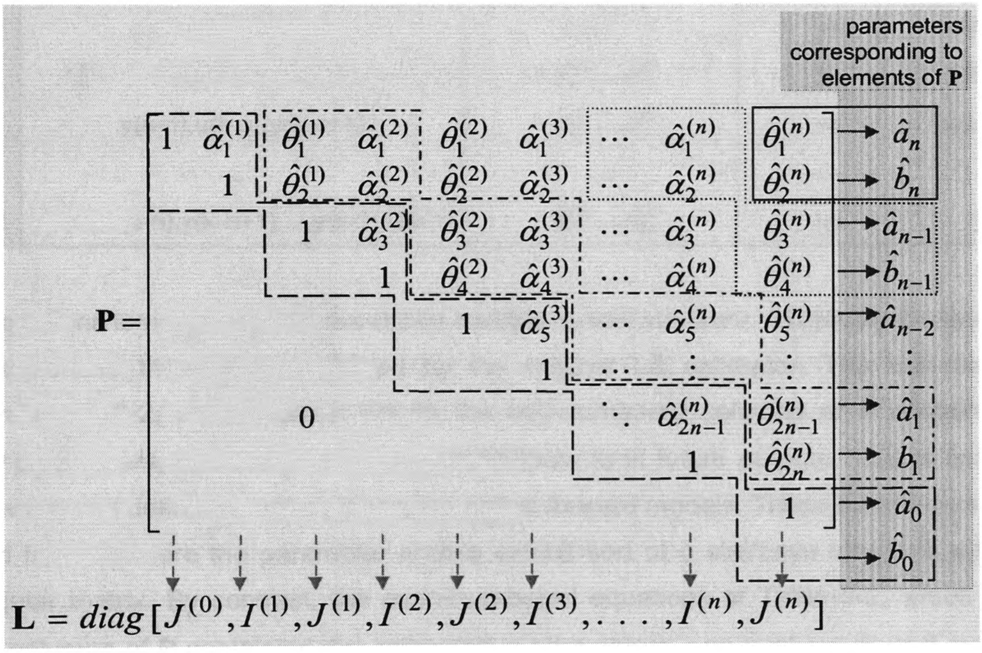


*Figure 8-2. Parameter matrix **P** in unit-upper-triangular form and corresponding loss matrix **L***

The QDU decomposition can be derived from a standard QR decomposition of the augmented data matrix:

$$\Psi_a = \mathbf{QR} = \mathbf{QDU} \tag{8-22}$$

Q becomes an orthogonal matrix and **R** is upper triangular. The diagonal **D** and the unit upper triangular **U** matrices are obtained by further decomposition of **R** utilising the Housholder transformation. For more details please refer to Niu *et al.* (1996). Then the following two matrices:

$$\mathbf{P} = \mathbf{U}^{-1} \text{ and } \mathbf{L} = \mathbf{D}^2 \tag{8-23}$$

provide the parameter and loss function matrices respectively for the multiple models.

The unit-upper-triangular parameter matrix **P** contains the parameters of the models from order *1 to n*, while the loss function matrix **L** contains the corresponding losses (Figure 8-2).

*Parameter Matrix **P***

With the augmented data vector multiplied by the parameter matrix **P** (Figure 8-2):

$$\psi_a^{\ T}(k)\mathbf{P} = \lambda(k)$$

the variables $\lambda(k)$ can be fitted or predicted utilising a linear combination of the measurements and the estimated model parameters as shown in Table 8-2.

Table 8-2. Interpretation of $\psi_a^{\ T}(k)\mathbf{P} = \lambda(k)$

index	measurements utilised for a linear combination with ...	...the estimated model parameters to...	...fit (→) or predict (⇒) ...	...variable λ
1	0	1	⇒	y(k-n)
2	y(k-n)	$1, \hat{\alpha}_1^{(0)}$	→	u(k-n)
3	y(k-n),u(k-n)	$1, \hat{\theta}_1^{(1)}, ..., \hat{\theta}_2^{(1)}$	⇒	y(k-n+1)
4	y(k-n),u(k-n),y(k-n+1)	$1, \hat{\alpha}_1^{(1)}, ..., \hat{\alpha}_3^{(1)}$	→	u(k-n+1)
...	...	...	...	...
2n +1	y(k-n),u(k-n),y(k-n+1),...,y(k-1),u(k-1)	$1, \hat{\theta}_1^{(n)}, ..., \hat{\theta}_{2n}^{(n)}$	⇒	y(k)

The *2n+1* equations lumped together are called *multiple model structure*. Obviously the last row of Table 8-2 describes the equation utilised by the original LS estimator. The odd-numbered equations in Table 8-2, which correspond to the odd-numbered columns of **P** containing the estimated θ parameters, use the past inputs and outputs to fit future outputs. This is consistent with the conventional definition and so these are called *forward* models. These parameters can be directly read from **P** where the parameter entries exhibit sort of a staircase structure within the upper triangular matrix. By contrast, the even-numbered equations of Table 8-2, which corres-pond to the columns of **P** containing the estimated α parameters, use past inputs and outputs to fit the inputs. These models are called *backward* models. Niu *et al.* (1995) showed that if output feedback affects a system this is modelled by the backward models. Here the order of numerator and denumerator can be equal because of the feedback. The parameter b_0 copes with this situation. In case of feedback control the controller can be modelled (Bowyer and Clarke 1996).

*Loss Function Matrix **L***

The loss matrix L of Figure 8-2 contains the corresponding losses and has a diagonal form:

$$\mathbf{L} = diag[J^{(0)}, I^{(1)}, J^{(1)}, ..., I^{(n)}, J^{(n)}] \tag{8-25}$$

All its diagonal elements contain the losses of the corresponding models. The $J^{(i)}$ elements correspond to $\theta^{(i)}$ forward model parameter estimates of i^{th} order, while the $I^{(i)}$ elements correspond to the $\alpha^{(i)}$ backward model parameters.

Forward Models

In most practical applications the forward models are the ones of interest. Therefore these will be explained in more detail. The parameters $\theta_i^{(n)}$ $(i=1,...,2n)$ represent the parameters of the n^{th}-order model for following n difference equations (as shown in Figure 8-2):

model equation	loss
$y(k)+\hat{a}_1 y(k-1)=\hat{b}_1 u(k-1)$	$\rightarrow J^{(1)}$
$y(k)+\hat{a}_1 y(k-1)+\hat{a}_2 y(k-2)=\hat{b}_1 u(k-1)+\hat{b}_2 u(k-2)$	$\rightarrow J^{(2)}$
$\vdots$	$\vdots$
$y(k)+\hat{a}_1 y(k-1)+...+\hat{a}_n y(k-n)=\hat{b}_1 u(k-1)+...+\hat{b}_n u(k-n)$	$\rightarrow J^{(n)}$

In the parameter matrix a_0 is already normalised to 1 for the forward models. The corresponding losses are contained in the odd-numbered columns of the loss function matrix, i.e. $J^{(i)}$ (for $i=1,...,n$) For example the 3^{rd} column contains the first order model:

$$1\cdot y(k)+\hat{a}_1 y(k-1)=\hat{b}_1 u(k-1)$$

The corresponding loss contained in the 3^{rd} column of the loss function matrix is $J^{(1)}$.

Order Recursive Structure of the MMLS Method

The special structure of the augmented data vector is the basis of the MMLS approach allowing its order recursive nature, which means that the calculation of models starts with the lowest order model and continues by calculating the higher order models successively. This has been investigated by Niu and Fisher (1994), who proved that numerical problems with higher order models do not affect lower order models. Therefore also matrix singularity (corresponding to zero elements in the loss function matrix) occurs in the forward and/or backward models of higher order first. This observation can be utilised to quickly determine a suitable model structure as outlined in Section 8.3.2.

All the information on the parameters and loss functions for all $(2n+1)$ models are implicitly contained in the augmented information matrix. The computational effort needed for calculating all $(2n+1)$ models and corresponding losses is nearly the same as for the single model calculated in the traditional way. However, the latter extracts only the information of the highest (n^{th}) order process model. The MMLS method manipulates on an augmented covariance matrix of dimension $p\times p$ with $p=2n+1$ (while $p=2n$ for the covariance matrix of the traditional least-squares estimator). The QDU realisation of the MMLS method represents an especially good compromise between numerical performance and algorithmic complexity for batch identification and has been therefore utilised for the correlation multiple model least squares method described next.

8.2 The Correlation MMLS (CorMMLS) Method

In the preceding sections it has been shown that the Cor-LS is a suitable method for system identification of disturbed systems and that the MMLS is a numerically appealing method. The symbiosis of the MMLS and the Cor-LS method results in the new CorMMLS method, which not only improves the numerical properties of the Cor-LS but also calculates all interesting models from order *1 to n* simultaneously.

The construction of an *augmented correlation function vector* for the CorMMLS is straightforward. It has the structure of the augmented data vector but contains the correlation function values (Körner and Schumann 1998b):

$$\boxed{\psi_{a_c}(\tau) = [-\hat{\phi}_{uy}(\tau-n), \hat{\phi}_{uu}(\tau-n), \ldots, -\hat{\phi}_{uy}(\tau-1), \hat{\phi}_{uu}(\tau-1), -\hat{\phi}_{uy}(\tau)]^T} \tag{8-26}$$

Again, the augmented correlation function matrix forms the basis for the simultaneous estimation of multiple orders. It only contains those elements of the correlation functions that significantly differ from zero, i.e. that lie in the interval $M \geq \tau \geq -P$ (Figure 8-1).

Hence:

$$\boxed{\Psi_a = \begin{bmatrix} \psi_{a_c}{}^T(1) \\ \psi_{a_c}{}^T(2) \\ \vdots \\ \psi_{a_c}{}^T(N) \end{bmatrix}} \tag{8-27}$$

As this matrix corresponds to the augmented data matrix as outlined above in (Equation (8-20)) the MMLS method can be directly applied utilising correlation functions instead of measurement data. Of course, the CorMMLS method can be realised utilising all the different techniques outlined in Table 8-1. Therefore the CorMMLS method is as simple and versatile in application as the MMLS but it incorporates the advantages of the Cor-LS for the estimation of parametric discrete time models derived in Section 8.1.1, namely:

- The number of data used for the parametric estimation is considerably reduced. Instead of N measurement values only $P+M+1$ correlation values are necessary. This property considerably decreases the computational effort and is advantageous, for example, for a repeated estimation with different deadtimes (explained in Section 8.3.2) and for the identification of multi-variable systems. Naturally, the augmented information matrix composed from the augmented correlation function vectors still has the same dimension as in the MMLS case.
- If the input signal is an arbitrary coloured noise signal the estimates are consistent also for output signals affected by arbitrary stationary disturbances. Further there is no need to estimate disturbance filter parameters as for the IV or ELS method.
- If the process input signal $u(k)$ is offset free, i.e. $E\{u(k)\}=0$, it can be shown (see below) that the offset of the output does not affect the estimate for a sufficiently high number of measurements.

8.3 Modifications and Extensions

For the sake of simplicity the previous sections introduced the CorMMLS method for open loop identification for single-variable processes without deadtime. In practical applications, however, it might be necessary to cope with multi-variable processes and various deadtimes. Quite often the processes must be operated in closed-loop to guarantee stability. Additionally, measured input and output signals are not as ideal as desired (Section 4.1). The signals may contain offsets which have to be cancelled before the application of the estimation algorithm. In the following some solutions for the described problems are outlined which support the practical application of the CorMMLS method considerably.

8.3.1 Offset Cancellation

In practice it is necessary to take account of the offsets of the measured absolute input *U(k)* and output *Y(k)* signals. The offsets are defined as U_{00} and Y_{00} respectively. Therefore:

$$Y(k) = y(k) + Y_0 \quad and \quad U(k) = u(k) + U_0 \tag{8-28}$$

The following methods are specifically useful for offset cancellation:

- Most commonly, the offsets are calculated as the means of the signals ($Y_0 = \overline{Y}$ and $U_0 = \overline{U}$), which are finally subtracted from the measured data sets to gain *y(k)* and *u(k)*. This procedure is applicable for time-invariant processes with stationary excitation signals and zero-mean disturbances.
- Another way to take account of the offsets is to calculate the first difference of the measured *I/O* signals:

$$\begin{aligned}\Delta Y(k) &= Y(k) - Y(k-1) \\ &= [y(k) + Y_0] - [y(k-1) + Y_0] \\ &= \Delta y(k)\end{aligned} \qquad \begin{aligned}\Delta U(k) &= U(k) - U(k-1) \\ &= [u(k) + U_0] - [u(k-1) + U_0] \\ &= \Delta u(k)\end{aligned} \tag{8-29}$$

  and to use the first differences of input and output signal for the identification procedure. This is possible as the process model (8-3) holds also for the first differences and for the corresponding correlation functions. However, this procedure increases the amplitudes of high-frequency disturbances, which decreases the signal to noise ratio.

- An implicit estimation is also possible if the offset is added to the process model as offset constant K_0:

$$Y(k) + a_1 \cdot Y(k-1) + \cdots + a_n \cdot Y(k-n) + K_0 = b_1 \cdot U(k-d-1) + \cdots + b_n \cdot U(k-d-n) \cdot \tag{8-30}$$

  Then K_0 can be determined by setting (8-28) into (8-3):

$$K_0 = Y_0(1 + a_1 + \ldots + a_n) - U_0(b_1 + \ldots + b_n) \tag{8-31}$$

The correlation functions become:

$$\hat{\phi}_{UY}(\tau) = \frac{1}{N+1}\sum_{k=0}^{N} U(k-\tau)\,Y(k) \qquad (8\text{-}32)$$

$$\hat{\phi}_{UU}(\tau) = \frac{1}{N+1}\sum_{k=0}^{N} U(k-\tau)\,U(k) \qquad (8\text{-}33)$$

and the augmented correlation function vector is supplemented by one:

$$\psi_{a_c}(\tau) = [1, -\hat{\phi}_{UY}(\tau-n), \hat{\phi}_{UU}(\tau-d-n), \ldots, -\hat{\phi}_{UY}(\tau-1), \hat{\phi}_{UU}(\tau-d-1), -\hat{\phi}_{UY}(\tau)]^T \qquad (8\text{-}34)$$

Then K_0 is automatically incorporated into the parameter matrix in form of a new first row containing the constants for the different models. This increases the parameter matrix by one in both dimensions.

- As long as it is guaranteed that $U_0=E\{u(k)\}=0$ it can be demonstrated that Y_0 does not influence the estimation for large N by inserting (8-28) into (8-3) to determine Equation (8-8) (Isermann 1992).

8.3.2 Model Structure Determination

Most often the real process structure is not known in advance. However, the CorMMLS method is specifically suitable for the determination of model order and deadtime for different reasons:

- An approximate estimate for the deadtime can be gained from the correlation analysis in case of PRBS or white noise input signals (Isermann *et al.* 1974).
- The process models of subsequent orders are reliably calculated including the losses (Section 8.1.2).
- The reduced data vector allows a rapid calculation of alternative models.

In the following a pragmatic method for the determination of model order n and deadtime d is presented especially suitable for interactive application. It has been found that the equation error already delivered by the CorMMLS method gives a good initial guess of the model structure. With respect to the deadtime it is necessary to estimate the model parameters for different deadtimes. That means that the input signal of Equation (8-3) is shifted by d discrete time steps. Naturally, it is beneficial to have an initial estimate d_i for the deadtime such that the estimation can be directed to the region around d_i. Utilising the CorMMLS this is gained from the correlation analysis. The following estimations for $d_{min}<d_i<d_{max}$ are done with small computing effort because of the simple generation of the multiple models that are based on the reduced data vector. The fit generally improves with higher order because the equation error decreases with increasing degree of freedom of the model (Figure 8-3).

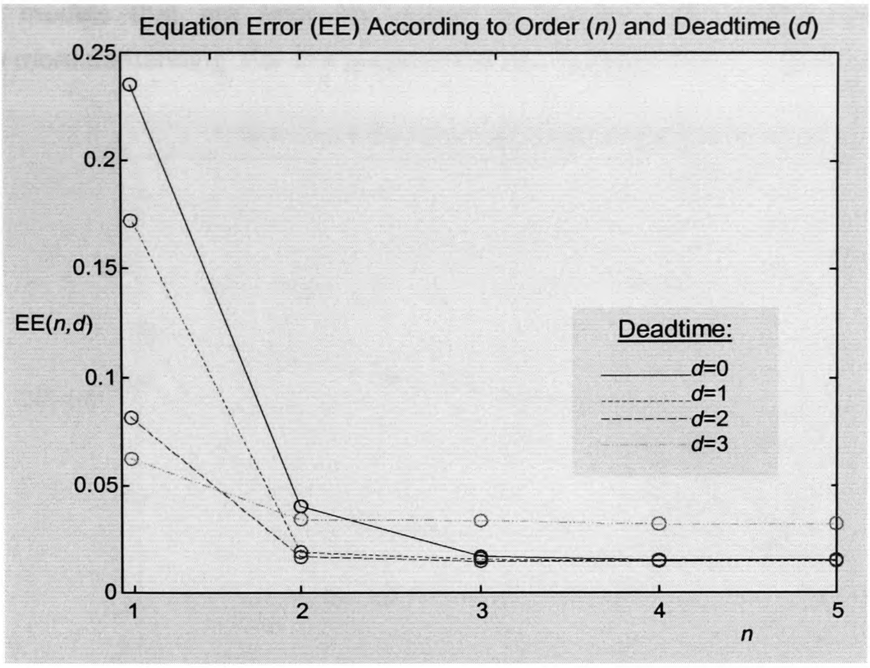


Figure 8-3. Equation error according to order and deadtime

In simulations the equation error (*EE*) can become zero under specific conditions, for example, if no disturbances are assumed. However, in practice the equation error will be small if an appropriate structure has been found. Then the equation error will not significantly change even if the model order is increased (over-parameterisation). This means that for a fixed deadtime d_f the criterion ΔEE:

$$\Delta EE(n+1,d_f) = EE(n,d_f) - EE(n+1,d_f) \qquad (8\text{-}35)$$

can be utilised to find:

$$\Delta EE(\hat{n}+1,d_f) << \Delta EE(\hat{n},d_f) \qquad (8\text{-}36)$$

or the ratio *R*:

$$R(n) = \frac{\Delta EE(\hat{n}+1,d_f)}{\Delta EE(\hat{n},d_f)} \qquad (8\text{-}37)$$

where the improvement in the equation error is very small. A practically useful rule is:

if $R(n) < 0.1$ *then* $\hat{n}$ *is the estimated order.*

According to Figure 8-3 it seems that the example process (the laboratory climate plant described in Section 9.5.1) could be described by models of order 2 with a deadtime of 1 or 2. Also a process model of *3rd* order with no deadtime could be utilised.

However, for final validation it is beneficial to utilise the errors between the measured output and the simulated output of the process model reacting to the same input just because the equation error only considers the fit of the correlations. Therefore the equation error is useful to preselect those process models that are later on utilised to generate the output error, which is computationally more demanding. For this purpose the results presented in Figure 8-4 are used.

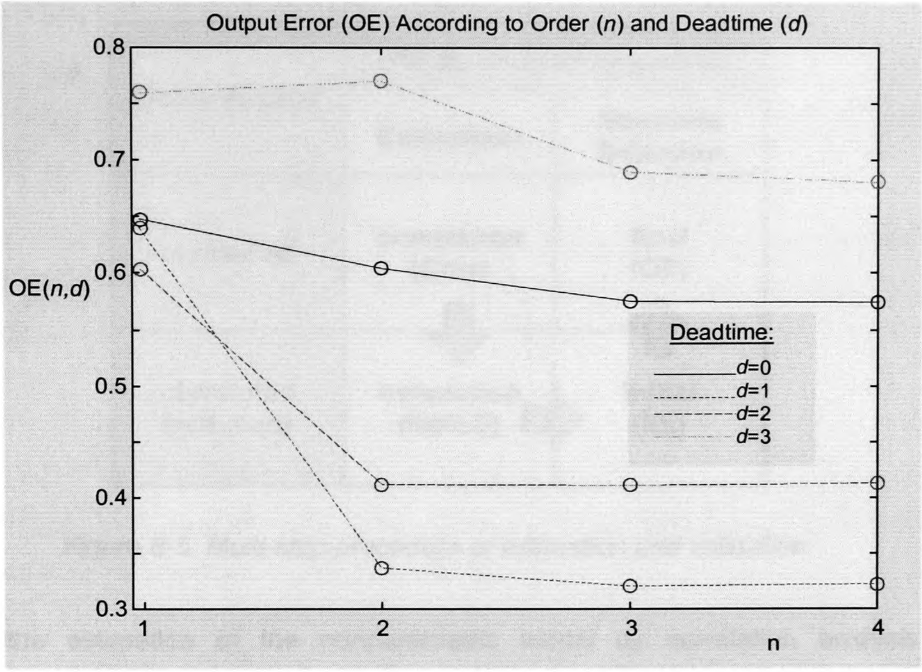


Figure 8-4. Output error according to order and deadtime

With the error between measured and simulated output being:

$$e(k,n,d) = y(k) - \psi^T(k,n,d)\hat{\theta}(n,d) \quad \text{for } k = 0,\dots,N \tag{8-38}$$

The output error becomes:

$$OE(n,d) = \frac{\|\hat{\mathbf{y}}(n,d) - \mathbf{y}\|_2}{\|\mathbf{y}\|_2} = \frac{\sqrt{\sum_{k=1}^{N} e^2(k,n,d)}}{\sqrt{\sum_{k=1}^{N} y^2(k)}} \tag{8-39}$$

The rating of the errors is equivalent to Equations (8-35) to (8-37) setting *OE* instead of *EE*.

From Figure 8-4 it can be seen that the example process should be described by a second order model with deadtime one as a good compromise between model complexity and fit.

The complete multi-step procedure of estimation and order selection for open loop identification is summarised in Figure 8-5.

Data Vector	Steps in Identification	
	Estimation	Structure Selection
measured	correlation (Corr)	final (OE)
correlated (reduced)	estimation (MMLS)	initial (EE)

Figure 8-5. Multi-step procedure of estimation and validation

It starts with the estimation of the nonparametric model by correlation analysis of the measurements. Herewith presumption about the process characteristics are gained and the data vector for the parametric estimation is reduced selecting all the elements of the correlations that significantly differ from zero. The MMLS estimation delivers a number of models from order 1 to n and the accompanying equation errors on which the initial model structure selection (initial validation) is performed. The final structure selection is based on the output error criterion, which is again based on the measurements.

For an automated procedure, it is also sensible to utilise extra stochastic criteria to determine a statistically significant change of fit between models with different orders and deadtimes. Possible criteria are the χ^2*-test* (,which is comparable to the F-test that describes the static case) or the *final prediction error* and the *average information criterion*. Söderström and Stoica (1989) demonstrated the asymptotic equivalence of all three criteria. Different realisations of these criteria with respect to the MMLS method have been realised by Niu (1994). Further tests that can be implemented are the pole-zero cancellation or the residual test described by Isermann (1992).

8.3.3 Estimation of MIMO Models

Mostly the processes in the process industry are multivariable. Therefore the estimation of MIMO models plays an important role. Most of the algorithms presented in the previous sections can be extended to multi-variable processes if the process inputs are not correlated, which has to be checked carefully. The p-canonic MIMO system is a straightforward extension of the SISO system, in which the system input becomes a n_u-dimensional vector $\mathbf{u}(k)$ and the output becomes a n_y-dimensional vector $\mathbf{y}(k)$. In the following the MMLS estimator will be extended to MIMO systems. Thereafter the multi-variable CorMMLS is derived.

The p-canonic MIMO process can be described by following discrete time difference equation:

$$\boxed{\mathbf{A}_0\mathbf{y}(k) + \mathbf{A}_1 \cdot \mathbf{y}(k-1) + \cdots + \mathbf{A}_n \cdot \mathbf{y}(k-n) = \mathbf{B}_1 \cdot \mathbf{u}(k-1) + \cdots + \mathbf{B}_n \cdot \mathbf{u}(k-n) + \mathbf{n}(k)} \tag{8-40}$$

with:

$$\mathbf{y}(k) = \begin{bmatrix} y_1(k) \\ y_2(k) \\ \vdots \\ y_{n_y}(k) \end{bmatrix} \text{ and } \mathbf{u}(k) = \begin{bmatrix} u_1(k) \\ u_2(k) \\ \vdots \\ u_{n_u}(k) \end{bmatrix} \tag{8-41}$$

The process model defined by (8-40) is described as having n_y subsystems corresponding to the n_y outputs. Each of the subsystems has n_u inputs. The number of outputs, inputs and the model order determine the model structure. The coefficient matrix for the MIMO MMLS evolving from (8-40) is defined by:

$$\boxed{\mathbf{\Theta} = [\mathbf{A}_n, \mathbf{B}_n, \ldots, \mathbf{A}_1, \mathbf{B}_1, \mathbf{A}_0]^T} \tag{8-42}$$

It has the dimension *ixj*, where $i = n_y$ and $j = [(n_y+n_u)\ n + n_y]$. Similar to the SISO case, the augmented data vector for the MIMO case is defined as:

$$\boxed{\mathbf{\psi}_a(k) = \begin{bmatrix} -\mathbf{y}(k-n) \\ \mathbf{u}(k-n) \\ \vdots \\ -\mathbf{y}(k-1) \\ \mathbf{u}(k-1) \\ -\mathbf{y}(k) \end{bmatrix}}, \tag{8-43}$$

where ***y(k)*** and ***u(k)*** are defined by (8-41). The augmented covariance matrix (of dimension *jxj*) is defined and decomposed in the same way as described in Section 8.1.2. Similar to the SISO case all models from order *1* to *n* for every subsystem plus the corresponding loss function are calculated. Again, the latter can be used to determine a suitable model structure. Naturally, the MIMO parameter and loss function matrix have to be interpreted differently.

Figure 8-6 depicts the MIMO parameter and the loss function matrix for a MIMO process.

- The MIMO *parameter matrix* contains *2n+1* models composed from *n+1* forward and *n* backward models (explained in Section 8.1.2). These are marked as blocks in Figure 8-6 and labelled accordingly. The forward models consist of n_y columns each, while the backwards models consist of n_x columns each. Each column represents a sub-model (exemplary cases are depicted in Figure 8-6).
- The accompanying *loss function matrix* helps to separately determine the order for each of the n_y forward MISO sub-models. In this way it is possible to construct the MIMO model from MISO models with different orders. The extraction of the model parameters is done as described in Section 8.1.2 and pictured in Figure 8-6.

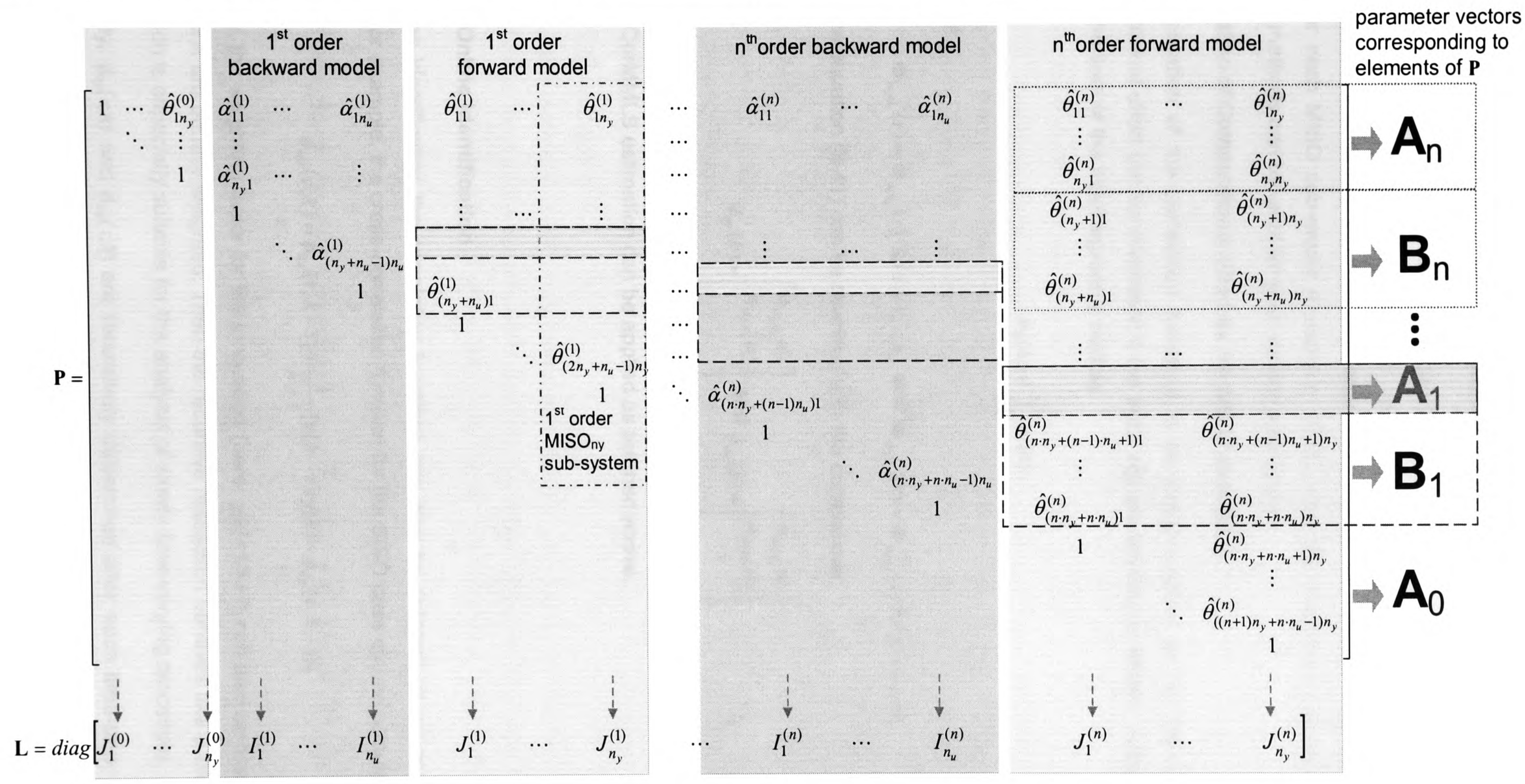


*Figure 8-6. MIMO parameter matrix **P** in unit-upper-triangular form and corresponding MIMO loss matrix **L***

In practical application it is advisable to do pure MISO modelling (n_y=1) for each output separately. This decreases the number of estimated parameters and therefore increases numerical robustness. Furthermore this approach enables the utilisation of different sampling times for each MISO sub-model (Diekmann 1983). Also the deadtimes can be eliminated and specific methods can be used for time-variant sub-models.

Minimisation of Computational Effort for the MIMO CorMMLS

The application of the correlation functions is as straightforward as for the SISO case. The computational effort can be minimised if the input signals are not correlated. According to Hensel (1987) the sum of the input signals is built as:

$$u_\Sigma(k) = \sum_{i=1}^{n_u} u_i(k)\,, \tag{8-44}$$

such that

$$\mathbf{\Phi}_{u_\Sigma u_i}(\tau) = \mathbf{\Phi}_{u_i u_i}(\tau) \text{ for i} = 1,\ldots,n_u \text{ and } \mathbf{\Phi}_{u_\Sigma y_j}(\tau) = \mathbf{\Phi}_{u_i y_j}(\tau) \text{ for } j = 1,\ldots,n_y \tag{8-45}$$

Therefore equation (8-41) can be rewritten with the correlations:

$$\phi_{uy}(k) = \begin{bmatrix} \Phi_{u_\Sigma y_1}(k) \\ \Phi_{u_\Sigma y_2}(k) \\ \vdots \\ \Phi_{u_\Sigma y_{n_y}}(k) \end{bmatrix} \text{ and } \phi_{uu}(k) = \begin{bmatrix} \Phi_{u_\Sigma u_1}(k) \\ \Phi_{u_\Sigma u_2}(k) \\ \vdots \\ \Phi_{u_\Sigma u_{n_u}}(k) \end{bmatrix} \tag{8-46}$$

and the CorMMLS estimation can be applied as outlined above.

8.3.4 Online Identification

For online identification the correlation functions can also be determined recursively (Isermann 1992). For example, the cross-correlation function for the SISO case can be calculated:

$$\hat{\phi}_{uy}(\tau,k) = \hat{\phi}_{uy}(\tau,k-1) + \frac{1}{k+1}[u(k-\tau)\,y(k) - \hat{\phi}_{uy}(\tau,k-1)] \tag{8-47}$$

Naturally, the weighting factor for the innovation (here: $\gamma(k)=1/k+1$) can also be fixed such that the innovations are equally weighted. Then the recursive realisation behaves like a discrete low-pass filter, which is especially suitable for the analysis of slowly time-varying processes.

Practically, $\phi_{uy}(\tau,k)$ and $\phi_{uu}(\tau,k)$ are recursively determined after each time-step. Afterwards the parameter vector can be estimated as outlined above after each time-step or after longer periods of time.

8.3.5 Closed-Loop Identification

Gevers (1993) reports that a huge gap developed at the end of the 1980s between control design and identification although these should be seen as two parts of a joint design problem. Classical process identification spends enormous numerical effort on characterisation of the random errors in the model resulting from disturbances although their value for control purpose is questionable (Ljung 1991). Identification for control has to consider the most essential features of the process from the view of control. This is advantageous because closed-loop identification considers the influence of the controller in the model structure, which shifts the frequency band relevant for identification while it allows the identification during regular closed-loop operation of the process.

Usually, *direct* and *indirect* identification methods are distinguished. For direct identification input and output signals in the process are directly utilised. By contrast, the indirect identification determines a model of the entire closed-loop, from which the process model is extracted this being possible only if the controller structure is known. Because of the model complexity (big number of parameters to estimate) the indirect method converges quite slowly. The direct identification is advantageous because the complexity of the estimated model is minor.

For closed-loop identification without additional excitation two identifiability conditions must be met:

1. The order and structure of the process model must be known a-priori.
2. The order of the controller must be sufficiently high.

If an additional test signal u_T (being statistically independent from n) is fed to the process input in addition to the controller output then it is sufficient for the direct identification if only the first identifiability condition is met.

For this case Isermann (1992) has shown that the Cor-LS method can be applied with similar results also for closed-loop identification and that the same properties hold with respect to noise elimination as discussed for the open loop case. These results are also valid for the CorMMLS as well.

8.4 Conclusion for this Chapter

Within this chapter two well-tested methods for parameter estimation have been combined. The combination of the practically frequently applied Cor-LS technique with the MMLS method results in the two-step CorMMLS method for the estimation of parametric discrete-time models. As the MMLS method is numerically more reliable, more efficient and never worse than the standard least squares method (Niu 1994) it has been utilised to replace the least squares method within the CorMMLS identification method.

The concurrent estimation of multiple models makes this algorithm especially appealing if order and deadtime of the model are unknown, which is practically mostly the case. The combination with correlation technique leads to unbiased estimates also in the presence of a coloured zero mean noise signal as long as the input test signal applied is statistically independent of the noise. Besides, the two-step method has the advantage of generating valuable a-priori information within the first step (the correlation) and the parametric estimation is performed on a reduced data vector. This is an especially appealing property in case of MIMO identification, where the effects of utilising a reduced data vector are remarkable.

Furthermore a multi-step procedure has been elaborated to estimate and validate the process models based on the CorMMLS method. This is initially based on the equation error delivered by the CorMMLS and finally validated by the OE criterion. This multi-step procedure can be used either interactively or by an automated procedure. It is summarised in Figure 8-5 and specifically suitable for the software development outlined in the following chapter.

9 Prototype Realisation and Validation

In Chapter 2 it has been shown that modern identification techniques are rarely used in the process industry for many reasons. In order to promote the use of system identification techniques numerous interviews in industry were carried out, from which a concept for industrial CACSD has been developed (Chapter 5) and a standardised nonlinear identification procedure (*SNIP*) has been proposed (Chapter 6). For the realisation of the *SNIP*, specifically suitable methods have been selected and improved for simplified application (Chapters 7 and 8).

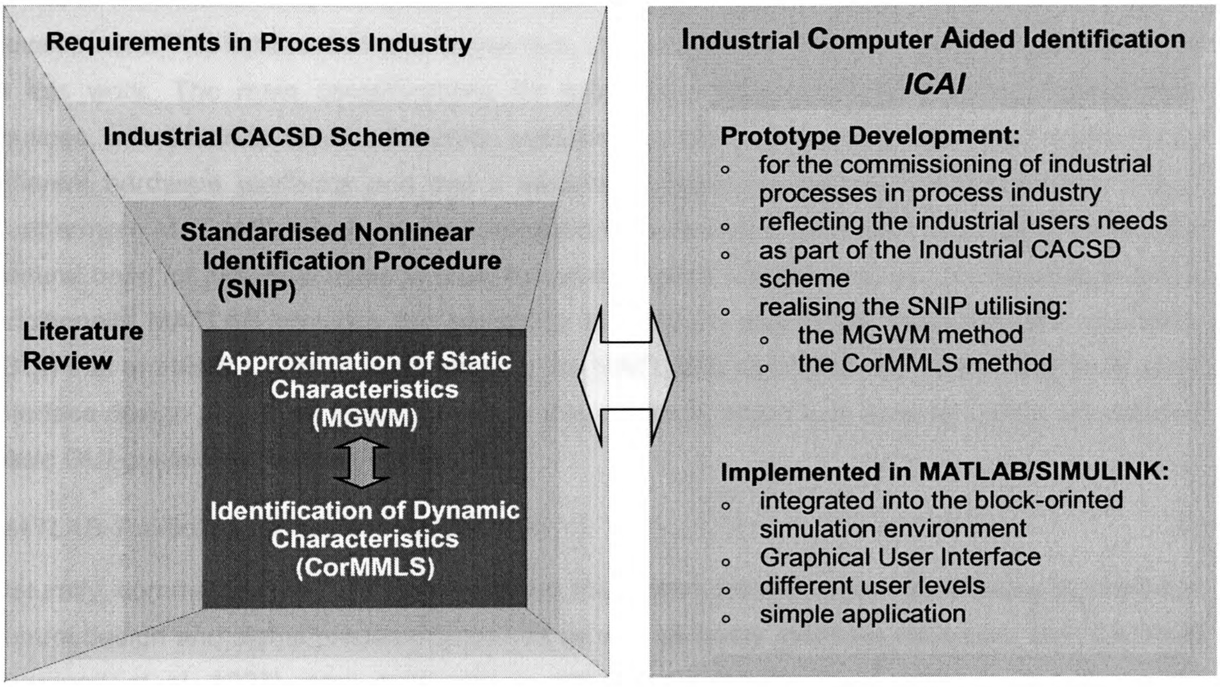


Figure 9-1. The groundwork necessary for the software development

However useful this work depicted in the left half of Figure 9-1, it does not say anything about its applicability. Therefore a prototype software has been developed that makes accessible these procedures and methods. The prototype software also provides a platform to discuss the implemented methods and strategies. While this prototype has been developed and tested the understanding for practical problems increased considerably and many valuable ideas developed. This iterative process is discussed in more detail in Chapter 10.

The following sections explain the main ideas of this prototype development. At first some general considerations for the prototype development are discussed. Then the prototype realisation is described and technical details for the practical realisation are summarised. Finally the prototype is viewed in the context of surrounding future developments.

9.1 General Considerations for the Prototype Development

Before a software prototype can be developed some side conditions must be clarified. A suitable software platform for the prototype development has to be selected, an appropriate requirements analysis has to be performed and the user interface has to be defined following some basic project guidelines, while every step within the prototype development is influenced by the customer's needs. In the following the general considerations will be detailed.

9.1.1 Decision on the Software

MATLAB (The MathWorks 1993) has been selected as an appropriate tool to perform a prototype development. This entails means to implement, try and test the schemes and methods elaborated in this work. The main considerations for this choice were MATLAB's tested mathematical routines, orientation to control, flexibility, suitability for rapid prototyping purposes, availability for different hardware platforms and that it became a quasi-standard in CACSD (Section 3.3.3). Furthermore MATLAB's block-oriented simulation environment SIMULINK is suited to serve as a general base for industrial users in order to solve practical identification and control design tasks. Additionally MATLAB provides the possibility to program proprietary graphical user interfaces (GUI) that enable the user to avoid learning the MATLAB's command-line syntax. Basically, good interface design played an important role in this research, which was assured by the utilisation of basic GUI-guidelines (Section 9.1.3).

MATLAB Toolboxes for System Identification

Naturally, commercial MATLAB toolboxes like the system identification, neural network, nonlinear control design and signal processing toolbox as well as freely available toolboxes, like NNSYSID (Nørgaad *et al.* 1997) were evaluated in simulation experiments in order to investigate the requirements for the ongoing research. The system identification toolbox (SITB 4.0 by Ljung, 1995a) offers, in particular, a variety of sophisticated algorithms. It provides a GUI aimed at specialists who need a tool that keeps track of the history of the data and that allows to compare multiple estimation results in parallel in various ways.

Handling of the System Identification Toolbox

While the SITB is a great help for experts it is not ideal for industrial users inexperienced in identification. Many difficulties have been observed when uninitiated users have tried to utilise the SITB the first time, in particular:

- The main window has not been found to be self-explanatory, so that the users had to ask for help. There is no guided tour to identification for industrial users.
- The operations on the process data and the estimations on the working data had to be carefully parameterised. Misunderstood parameterisation lead to poor results.
- The provided *'quick start'* proved to be the only way for practitioners to work with the SITB. However, the *'quick start'* is not self-explanatory and needs careful introduction. Further

problems occurred when the identified discrete-time models had to be transferred to SIMULINK, namely:

- the selection of a suitable SIMULINK block.
- the format of the polynomials.
- the errors doing the manual data transfer.

These experiences directly influenced the design of the *ICAI* toolbox and the formulation of the design principles for the prototype realisation.

9.1.2 Main Design Principles

Some basic design principles for ICACSD tools have already been formulated in Chapter 2. However, for software development for system identification the statements had to be refined. In the preceding section some problems that accompany up to date identification software became obvious. Following conclusions can be drawn with respect to the design principles for an industrially suitable identification software:

1. It must be tailored to industrial needs, such that even inexperienced users can intuitively use and understand the identification routine. This means:
   - The user must be guided through standard design paths (like the *SNIP*).
   - The use of advanced methods must be simplified utilising sensible defaults.
   - Even nonlinear and MIMO identification should be supported in a simple, transparent, reliable and reproducible procedure.
   - By default, all the results should be represented in time domain through easily understandable graphs, whenever possible.
2. A primary requirement is the integration of the identification task into a block-oriented simulation environment, because block-oriented simulation environments are frequently used to investigate industrial processes. Thus the user does not have to switch between different programs, when utilising an integrated solution.

A prerequisite for the realisation of these design principles is the proper design of a graphical user interface (GUI) that provides a guide through the whole identification procedure.

9.1.3 GUI Assistance for this Project

As pointed out in Section 4.8 several ways have been tried to assist the user in interacting with software programs. The most successful approach is the use of GUIs, which have been established as very helpful means to guide the user through a design procedure and to provide a context dependent assistance whenever necessary. However, building good GUIs is challenging and it needs some time and experience to assemble user-friendly interfaces that provide sensible access to the necessary functions. Although GUI design is a rather vital field in computer science, the universal qualities of good design have remained unchanged. Hundreds of articles have been written about design and although this work is not about design, the software realisation has to cope with this subject, just because good design helps the user considerably, or in other words:

"... it can be extremely comforting to tackle a GUI design problem and find centuries of wisdom waiting patiently to help you." (The MathWorks 1996).

Principles for the ICAI - GUI Design

It is the sheer endless freedom that makes GUI design so ambitious. There are numerous interactive graphical aids like buttons, radio buttons, check boxes, pop-ups, sliders and there is an even bigger number of graphical extras like grid lines, colours, numbers and so on, all of which can considerably help if put into the right context. However, these can be equally confusing, for example if too many functions are provided simultaneously.

The three main ideas of building GUIs are (sorted by importance):

1. Simplicity (guidance)
2. Consistency (handling)
3. Readability (form, colour)

Furthermore a fourth point has to be regarded being important not only from the programmers point of view. It is:

4. Reliability (robustness, speed)

With respect to 1) it is aimed at a clean look of the GUI that provides only the functionality necessary for the next step within the design procedure. Furthermore graphical representations should be reduced as far as possible to concentrate the user's attention. For example there have been some prototype GUIs within this project carrying too much graphical information. These were considerably confusing the users, such that the GUIs had to be simplified. Additionally it is helpful to provide graphical input, whenever possible instead of numeric input. For example, it is sensible to specify limits, offsets and so on directly from within a plot of a data set by mouse-clicking the meaningful points or regions instead of numeric input, which is prone to errors.

With respect to 2) it is targeted at a consistent user strategy. For example the standardised placement of interactive GUI elements is one important layout characteristic. In this GUI development interactive elements have been set on the right hand side of a figure grouped inside a frame, while the colour of the frame corresponds to the importance of the functionality provided. This is shown, for example, in Figure 9-6.

The latter aspect is already part of point 3). Utilising special colours for special functionality the readability of the GUI is enhanced, thus becoming quickly familiar to the user. This concept has been found so important that some guidelines were developed for this project (see below).

With respect to 4) it is also necessary to consider the dynamic aspects of GUI design. No matter how excellent the design, it will be only accepted if it triggers a quick and reliable action. This means that it should interact with the user smoothly. Furthermore it is advantageous if an action or a series of actions can be reverted.

Another important detail is the right wording in GUI design. In this work the wording in the messages and windows has been improved iteratively with the help of test-users. However, further refinements with potential users might be necessary to eliminate possible misunderstandings.

Naturally this is only a selection of the most important aspects that have been considered while doing the GUI design. For more details please refer to Percoco and Sarti (1996) or Grams (1998).

Finally, if the GUI design is finished, it's success can be measured in two ways:

- The time needed to perform a GUI-supported task *the first time.*
- The time needed to perform a GUI-supported task *once the interface is familiar.*

Of course, the aim is to minimise these metrics as far as possible. Nevertheless it is a tedious procedure and needs a continuous feedback from the user already in the design process.

The GUI Design Process

Before a GUI can be designed its task and functionality must be clearly defined. This has been explained above. Then the GUI design process can be separated in the design and the implementation phase as depicted in Figure 9-2. It is important to complete the iterative design phase as perfectly as possible before the implementation phase starts. Otherwise it takes a very long time to gain satisfactory results, because changes in the design phase are done much quicker on paper than in the implementation phase, where code is written for the GUI and the associated functionality. Furthermore it is a long step back from the test of the coded GUI to the GUI design.

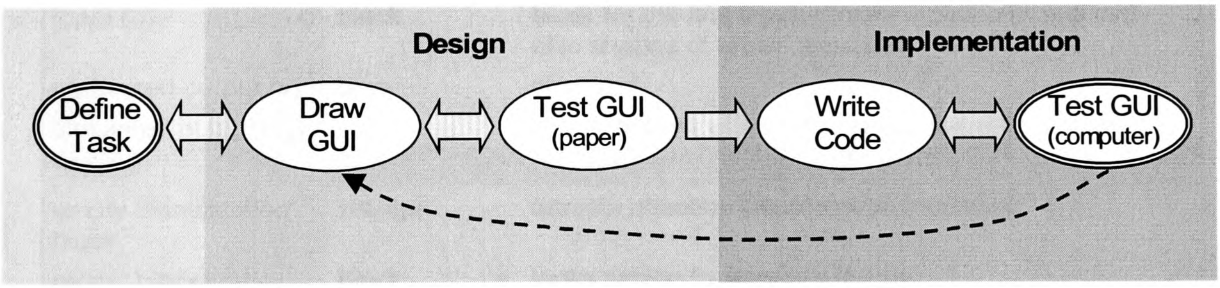


Figure 9-2. GUI design process

The GUI Colour Concept

Colours affect human thinking and behaviour similarly. Therefore it is worthwhile to consider their effects while programming GUIs for industrial users. Charwat (1996) investigated aesthetic aspects and remarks that the aesthetic solution preferred by the individual is not necessarily an appropriate/functional one: *'Better functional than beautiful'*. It is sensible keep this in mind when designing a GUI.

Through the quasi-windows-standardisation that has taken place in recent years many useful colour concepts have been realised without urging programmers to think about the effects of colours. In former times black and blue were popular background colours satisfying aesthetic feelings. Now grey is the standard for windows because this is the background colour which facilitates recognition of the highest number of different foreground colours (Charwat 1996).

Grouping by colour is an efficient help for the user to recognise similar functionality. It should be used together with another coding like shape or direction to further reduce the recognition time of the user. To attract attention for one particular object it is sensible to increase the contrast and the luminosity of its colour and to decrease the number of luminous objects. If important or even hazardous events occur also sound can be used (for example if the process leaves its limits) to emphasise that user interaction is needed. According to these considerations those colours outlined in Table 9-1 have been defined for the design of the GUI.

Table 9-1. Colour conventions

Object	Colour	Remark
background		
figure (main window)	light green	homogenous background without pattern or picture with similar properties as grey but allowing a corporate design of the toolbox
main tasks	bright grey	beneath important interactive elements to attract attention, not framed
general tasks	dark grey	beneath less important interactive elements, not framed
help text	shade of grey	beneath standard help
action field	white	to attract the user's attention to this field
diagrams	grey	grids (if used) should not be dominant
message box	grey	standard
lines		
input (u)	black	black for the first input, if more inputs are depicted also shades of brown are used
measured output (y)	blue	standard
simulated output (y_s)	red	good contrast with blue to recognise degree of fit
blocks		
empty identification block	orange	attracts attention (must still be modelled)
model block	black	looks similar to standard blocks

9.1.4 Influence of the User-Profile on the Prototype Realisation

In Chapter 2 the importance of user-oriented software support has been detailed. Consequently, it is necessary to base a software development on the users needs. To specify the necessary functionality for the identification toolbox a user-profile matrix has been elaborated, which determines the tasks that are necessary for users with different expertise. Table 9-2 contains the most important characteristics that should be met by identification software aimed at the different types of users. Naturally, such standardisation cannot meet the needs of each individual user. However, it allows the design of a transparent software concept and clearly indicates that the identification support for process personnel must be excessively automated, while control experts wish more freedom.

As the project aim had to be oriented at industrial reality the prototype development was focussed on the process personnel requirements. However, to respect the various needs of area engineers and control specialists in industry different user levels have been implemented into the prototype realisation, which is described in Section 9.2.

Table 9-2. Software functionality matched to user-profile

functionality	process personnel	area engineer	control expert
general			
transfer of results to simulation environment	√	√	√
password protection		√	√
library support	√	√	√
control strategy dependent identification		√	√
help system	√	√	√
linear dynamic identification			
parametric	√	√	√
nonparametric			√
time domain	√	√	√
frequency domain		√	√
state space			√
discrete		√	√
continuous	√	√	√
selection of different identification methods		√	√
modelling of disturbance			√
specification of a-priori knowledge (deadtime, dominating time constant,...)	√	√	√
interactive data pretreatment (manual offset cancellation, prefiltering, outlier,...)		√	√
interactive model structure determination		√	√
selection of validation criteria			√
multivariable modelling	√	√	√
input signal design	√	√	√
direct access of SITB (System Identification Toolbox)			√
nonlinear identification			
selection of different identification methods for static characteristics	√	√	√
automatic steady state recognition	√	√	√
support of Wiener- and Hammerstein modelling	√	√	√

9.2 MATLAB Toolbox Description

The MATLAB toolbox has been designed to satisfy the design principles of Section 9.1.2 by:

1. addressing the industrial user's needs
2. integrating the identification task into MATLAB's block-oriented simulation environment SIMULINK.

In the following the toolbox functionality is described highlighting these design principles.

9.2.1 The *ICAI* (Industrial Computer Aided Identification) Blockset

The *ICAI*[7] (Industrial Computer Aided Identification) toolbox has been realised as a SIMULINK blockset (Figure 9-3). This allows the user to solve modelling, identification and control design including simulation in one single graphical environment without the arduous need to convert models and data according to the needs of separate programs or to use MATLAB's command line.

The formerly presented *ICAI* toolbox (Körner *et al.* 1996 and 1997) was completely revised to include GUI standards, different user profiles and improved data storage and retrieval (Körner 1998; Körner and Schumann 1998c). The *ICAI* blockset contains *ICAI* ID (identification) blocks which are simple in use providing a guided tour to identification and making use of a unified GUI that controls a standardised identification procedure for each block (Körner and Schumann 1996a).

An ID block represents a black box which models an unknown part of the process model and which allows easy access to identification methods within SIMULINK.

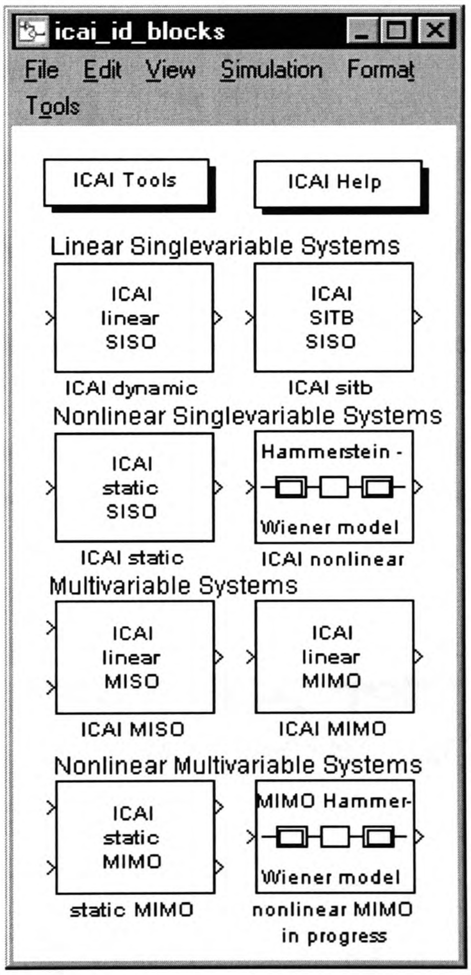


Figure 9-3. ICAI ID blocks

[7] After establishing the abbreviation *ICAI* (industrial computer aided identification) it was recognised that the abbreviation *ICAI* is already used in the field of artificial intelligence applied to education. Here *ICAI* stands for *Intelligent Computer Aided Instructions* (Percoco and Sarti 1996). This is a surprising coincidence because the guided tour which is provided by this software for identification is a sort of intelligent instruction set based on previous results within the identification procedure. Without being aware of this coincidence some basic thoughts of Percoco's and Sarti's paper were essential for this work as well, like:

- How can we fully exploit the communication capabilities of graphics through careful use of colour, shape, fonts etc.?
- How can we set the graphic properties of the presentation, which best support the achievement of our (learning) goals?

In the second statement the original aim of *ICAI* in education, i.e. learning, has been set into parenthesis, because the toolbox presented here is not aimed at learning but at producing quick and reliable solutions for the identification task. However, learning effects are also possible, when users understand the system identification procedure supported by the *ICAI* prototype.

According to Figure 9-3 the *ICAI* ID blockset comprises:

SISO blocks

- *ICAI static SISO:* A static SISO ID block for the modelling of one-dimensional static characteristics.
- *ICAI linear SISO:* A linear dynamic SISO ID block for the identification of single-variable linear dynamics.
- *ICAI SITB SISO: A* linear dynamic SISO ID block for the identification of single-variable linear dynamics utilising the SITB (Ljung 1995a).
- *ICAI nonlinear SISO:* A nonlinear dynamic SISO ID block for the identification of nonlinear single-variable dynamics composed from a one-dimensional static characteristic and linear dynamics in the sense of Wiener- or Hammerstein- models.

MIMO blocks

- *ICAI static MIMO:* A static MIMO ID block for the modelling of multi-dimensional static characteristics with two inputs and two outputs.
- *ICAI linear MISO*: A linear dynamic MISO ID block for the identification of multi-variable linear dynamics with multiple inputs and one single output.
- *ICAI linear MIMO:* A linear dynamic MIMO ID block for the identification of multi-variable linear dynamics with multiple inputs and multiple outputs.

It is valuable to provide a software functionality oriented at different user levels (Section 9.1.4). Therefore the *ICAI* ID blocks support three different user levels and provide the means to specify sensible side conditions in advance as outlined next.

9.2.2 *ICAI* Project

The *ICAI project* sets the frame for the *ICAI* identification task. The project settings (Figure 9-4) must be specified for each block diagram comprising *ICAI* ID block(s). The project settings require information about the user profile, process type and the aim of the project, information that is used by *ICAI* or by the complementary control system design blockset currently under development (see Section 9.6).

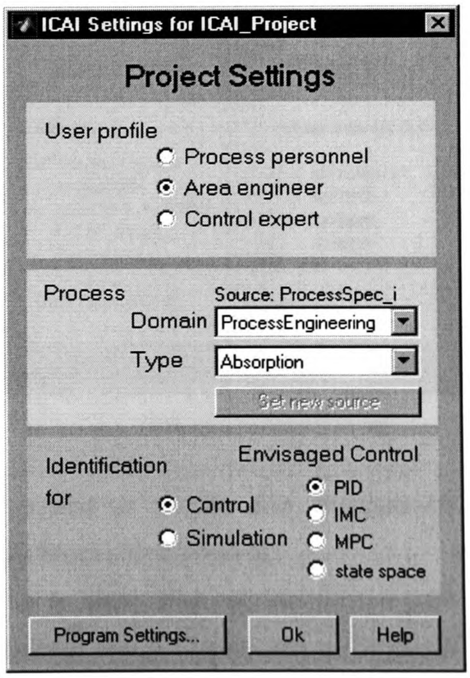


Figure 9-4. ICAI project

The password protected *user profile* controls the complexity of the graphical user interfaces of the *ICAI* ID blocks for the user levels *process personnel, area engineer* and *control expert*. This way a control expert will get more degrees of freedom and more sophisticated presentations of the identification results than area engineers or process personnel, in order not to confuse the inexperienced user. While the choice among alternative design methods is provided on the area engineer level, there are only few choices on the process personnel level, on which pre-parameterised default methods are utilised.

The *process specification by domain and type* allows the integration of SIMULINK block diagrams, which can be used as process model libraries to store available process knowledge. A sensible library approach including a concept of granularity that can be invoked by *ICAI* has been described by Ravn and Szymkat (1997). Additionally the envisaged control strategy can be selected respecting the appropriate user level.

At any time it is possible to change the project settings with the appropriate password. This feature, for example, enables the control expert to access and make use of the data and results that have been gained on lower user levels beforehand, now utilising all available methods.

9.2.3 Handling of *ICAI* ID blocks

ICAI ID blocks represent just another type of SIMULINK blocks. The use of *ICAI* ID blocks can be compared to the use of standard SIMULINK blocks. The differences:

- A double-click does not only open a window for editing vectors or matrixes - moreover it offers a guided tour to identification through different *ICAI* windows.
- When the identification has been successful the *ICAI ID block* becomes an *ICAI model block* and it's appearance changes in colour and label. Herewith unmodelled ID blocks can be quickly identified even in bigger projects and distinguished from the readily modelled *ICAI* model block. Furthermore a warning message occurs if unmodelled ID blocks are utilised within a SIMULINK block diagram for simulation.

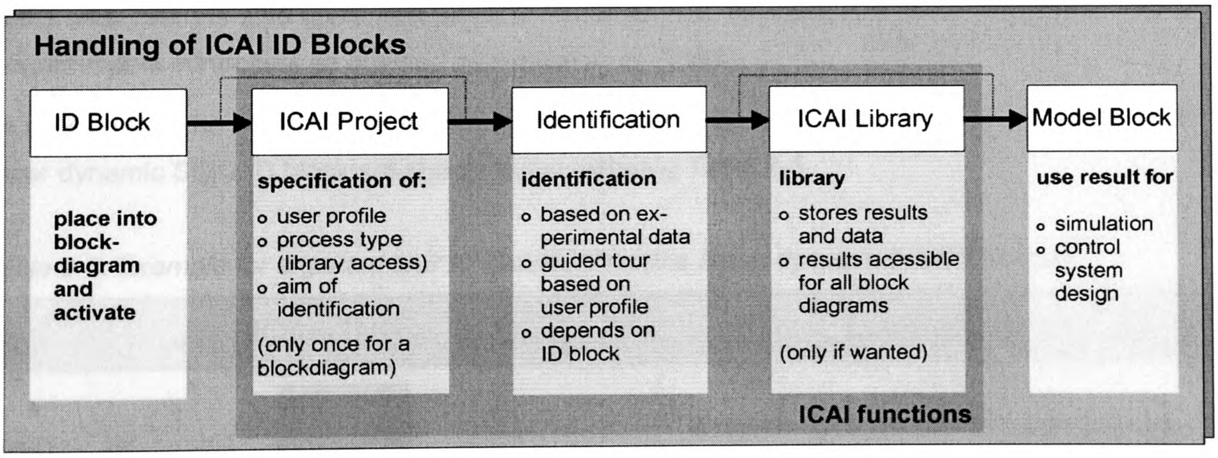


Figure 9-5. Handling of ICAI ID blocks in 5 phases

During the whole development of the *ICAI* toolbox it was aimed at simple and transparent handling of the ID blocks[8]. Figure 9-5 depicts the utilisation of ID blocks in 5 phases:

1. The *ID block* is placed into the SIMULINK block diagram as a 'black box' representing a part of the process model or the complete process model. Herewith the ID block can already be

[8] For better readability the *ICAI ID blocks* will be called *ID blocks* in the following.

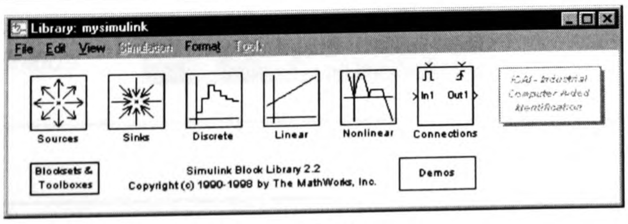
connected with other (sub-)models. It can also be connected with the envisaged controller in order to perform the control system design when the ID block is identified.

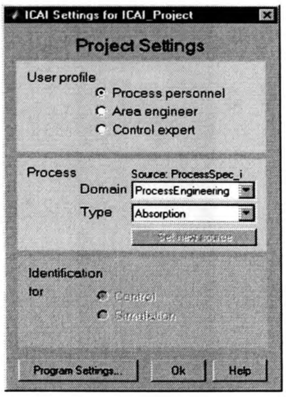
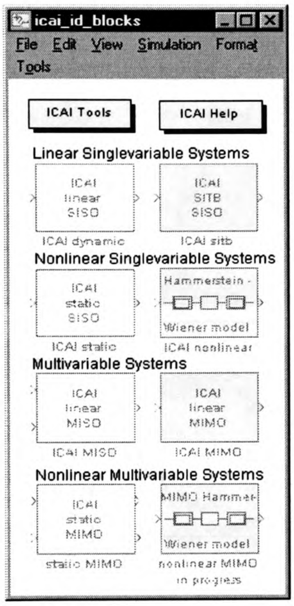
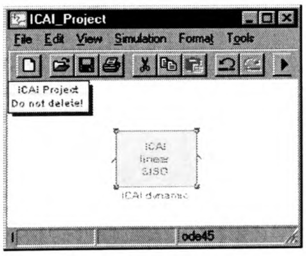
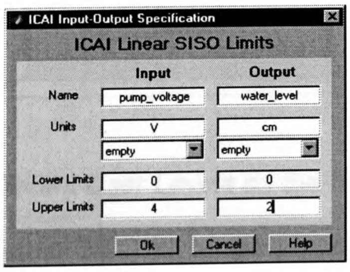
2. When the ID block is activated the *ICAI project* conditions have to be specified once for each SIMULINK block diagram (Section 9.2.2)
3. The *identification* is based on experimental data, which must be gained from suitable experiments (Section 4.1.1). A guided tour is started based on the user profile and depending on the functionality of the ID block itself. The *ICAI* GUI offers context sensitive and user specific functionality for every step within the identification procedure. The input of the measurement data is preferably done interactively from a MATLAB-file utilising an *ICAI* tool (*data import* of Table 9-3). In some cases it is sensible to use *ICAI* Draw, which allows a manual data input utilising graphical aids. Relevant a-priori knowledge can be incorporated if available and used to structure and parameterise the identification procedure. Only few reliable identification methods are offered with extensive pre-parameterisation, such that the user has to deal with only few transparent design decisions. The validation of the results is done by graphical comparison of the simulated data from the identified model against the measurement data. The *ICAI model block* contains the identified process model and is linked to the a-priori knowledge and the experimental data (Section 9.3.1).
4. The model block can be stored in the *ICAI* Library that allows a quick retrieval of identified models.
5. Finally the model block can be used for simulation or control system design within SIMULINK like any other standard SIMULINK block.

As *ICAI* ID blocks are based on standard SIMULINK blocks the 'real-time workshop' (The Mathworks 1993) is also applicable on *ICAI* model blocks. This allows a quick design and test of programmable controllers as outlined by Pfeiffer *et al.* (1998).

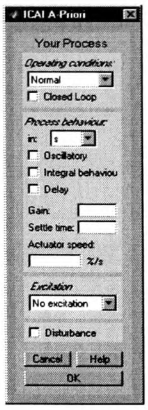
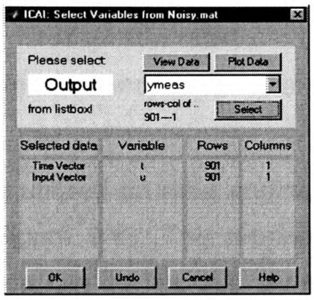
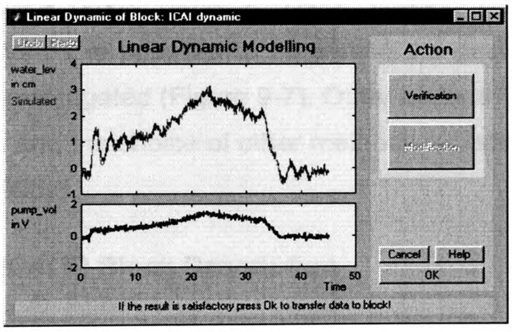
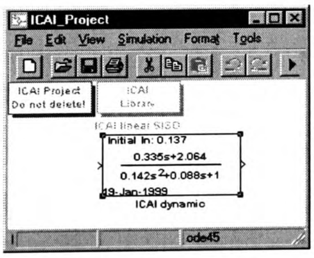
As an example the steps for the identification of a linear dynamic process model utilising *ICAI*'s linear dynamic SISO ID block are shown in the following Table 9-3.

Table 9-3. Example for a guided tour to identification (the linear dynamic SISO ID block)

ICAI GUI	***Description***
ICAI windows icon 	A double-click on the *ICAI*-icon from the windows operating system
 *extended SIMULINK with ICAI*	directly starts SIMULINK extended with *ICAI*. This way all SIMULINK functions are available plus *ICAI*. If *ICAI* is activated

	ICAI GUI	***Description***
ICAI project		a fresh block diagram is opened and the *ICAI* project has to be specified.
ICAI ID block library		Then the *ICAI* block library is opened, from which
ICAI ID block & new block diagram		the linear dynamic SISO ID block is copied into the new block diagram[9].
ICAI I/O definition		With the activation of the linear dynamic SISO ID block a GUI for the I/O specification is displayed.

[9] If the linear dynamic SISO ID block is copied into another block diagram the *ICAI* project has to be specified for the new block diagram directly after the ID block is activated.

<table>
<tr><th>ICAI GUI</th><th>Description</th></tr>
<tr><td>ICAI
a-priori
knowledge

</td><td>Then the a-priori knowledge is collected by a GUI-questionnaire.</td></tr>
<tr><td>data import

</td><td>Afterwards the measurement data are specified interactively and imported.</td></tr>
<tr><td>ICAI main window

</td><td>Automatically the first identification result is calculated, which is ready for validation. This is best done on a fresh data set.</td></tr>
<tr><td>ICAI
model

</td><td>Finally the ID block becomes the ICAI model block containing a continuous time model ready for simulation and controller design (not shown here).</td></tr>
</table>

Influence of Different User Levels on the Linear Dynamic SISO ID Block

The linear dynamic SISO ID block supports the modelling of linear dynamic processes. On the *process personnel* level the *CorMMLS* method elaborated in Chapter 8 is applied as default. A set of process models is generated and the best model is selected as outlined in Section 8.3.2. The block automatically handles signal offsets and drifts, determines the deadtime and checks the identification results against available a-priori knowledge.

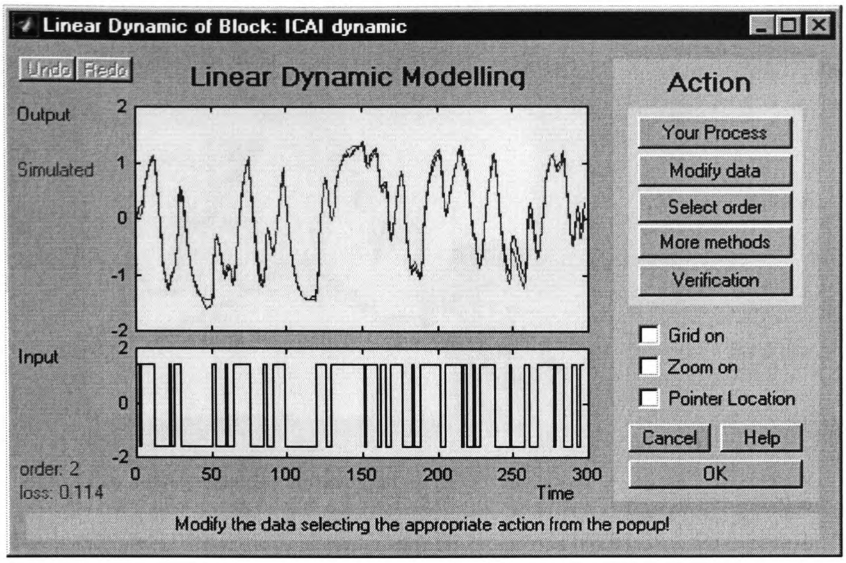


Figure 9-6. Main window of linear dynamic ID block

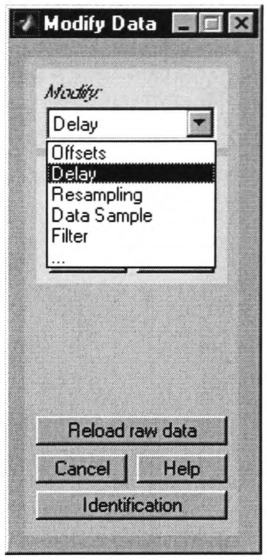


Figure 9-7. Data pre-treatment

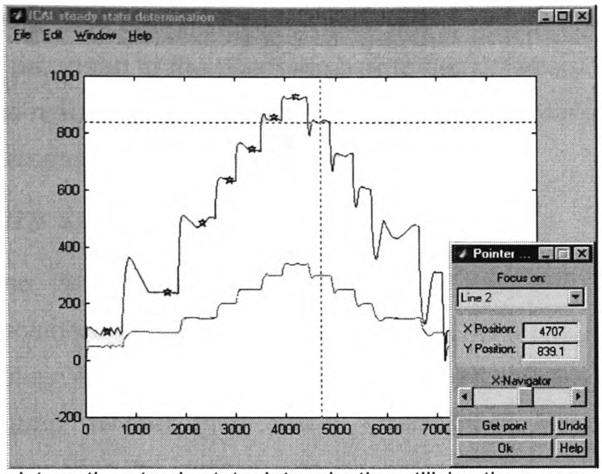
If the ID block is utilised on area engineer level or higher more actions are accessible from the main window (Figure 9-6). The a-priori knowledge can be changed from this level and the measurement data can be interactively modified before the identification is performed. Standard actions, like offset removal, linear filtering and resampling can interactively be performed. Furthermore it is possible to determine the deadtime and to select the part of the data set that shall be investigated (Figure 9-7). Other actions from the main window comprise the manual order selection and the choice of other methods including the access of the system identification toolbox (Ljung 1995a).

9.2.4 *ICAI* ID Block Description

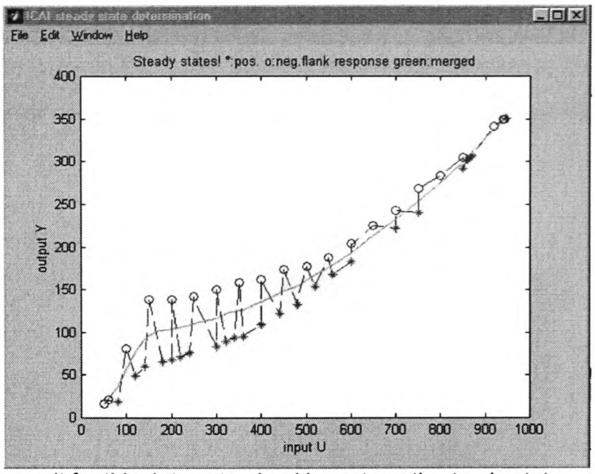
In the following a brief description of the remaining *ICAI* ID blocks is provided to clarify the principal *ICAI* prototype functionality. A more detailed description of the *ICAI* functions is available as *ICAI* online help (Section 9.2.5).

Static SISO ID Block

The *static SISO ID block* generates a static nonlinear function from steady state data by interpolation or approximation. The steady state data can be directly loaded from a mat.-file or by graphical input. It is also possible to extract the steady state data from dynamic measurements automatically or manually provided that steady states are included in the measurements. Figure 9-8 shows an example for interactive and automatic determination of the static characteristic for the air pressure depending on the valve position of an absorption column pilot plant. In the example hysteresis effects have been detected. However, as the *SNIP* is not capable of handling these effects, the static characteristic has been averaged out. The MGWM method outlined in Chapter 7 is used as default to fit the data. Polynomial or linear fits are also provided on area engineer level and higher. Finally, the modelled static characteristic is stored as a table in the *ICAI* model block, which enables the simulation of this block through linear interpolation of the table values.



interactive steady state determination utilising the cursor



result for this data set gained by automatic steady state recognition (hysteresis effects are averaged out)

Figure 9-8. ICAI steady state determination

Nonlinear SISO ID Block

The *nonlinear SISO ID block* supports the identification of simplified Wiener- and Hammerstein-models. This ID block represents a macro block that controls the successive modelling of the static characteristic and the linear dynamic as outlined in the *SNIP* (Chapter 6). For this purpose the macro block utilises the static and the linear SISO ID block (Figure 9-9). Therefore the standard identification procedure is to model the static characteristic first utilising the functions of the static SISO ID block. Then the linear dynamic part of the model is identified based on the dynamic measurement data pre-processed in order to compensate for the static nonlinearity. If an inversion of the static characteristic is impossible only a Hammerstein-model is identified.

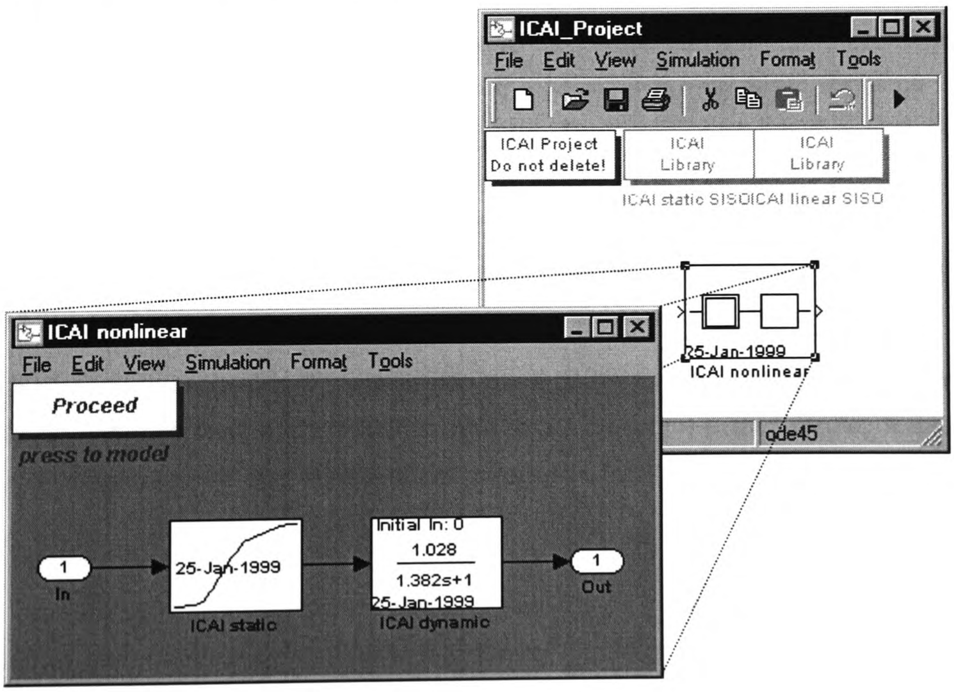


Figure 9-9. Nonlinear SISO ID block

Otherwise also a Wiener-model is estimated. The nonlinear model with the minimum output error is proposed to the user who has to decide about the final model. Figure 9-9 shows an example for the resulting sub-model of the identification procedure and the change in the representation of the successfully identified *ICAI* model block.

SITB SISO ID Block

The *SITB (System Identification Toolbox) SISO ID block* considerably differs from the other *ICAI* ID blocks because it solely is an interface block that allows access to the SITB (Ljung 1995a) from within SIMULINK. This block is only accessible for control experts because it needs considerable control expertise to properly get acquainted to the advanced features of the system identification toolbox and the various graphical representations. If a SITB SISO ID block is activated it is possible to utilise the SITB *directly* or to *import* an SITB-file:

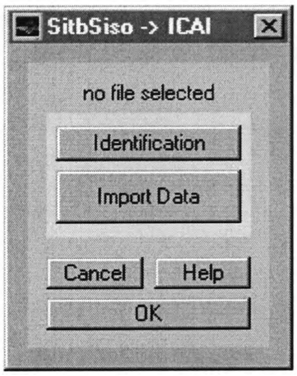


Figure 9-10. SITB interface

- If the SITB is *directly* used, the identification session is accessed through the SITB SISO ID block. Afterwards the result has to be stored as SITB-file first before the results for the model block are accessible. Then the SITB-file is evaluated for the data transfer into the SIMULINK environment.
- If the SITB-file is *imported* the ID block provides access to all stored measurement data and accompanying process models, from which the best can be chosen to fill the final model block. Furthermore it is possible to work on the data utilising all the functions provided by the SITB.

Linear Dynamic MISO ID Block

The *linear dynamic MISO ID block* has been designed for the identification of linear multi-variable process models. It follows a similar procedure as the linear dynamic SISO ID block but it can handle in addition an arbitrary number of input signals. The functionality is similar to that of the linear SISO ID block. However, only the process personnel level has been realised in the *ICAI* prototype and the data pre-treatment is not as elaborated compared to the linear SISO ID block. Nevertheless, this block sufficiently supports the inexperienced user to tackle even multi-variable processes and it is the basis for the linear dynamic MIMO ID block.

The CorMMLS method described in Chapter 8 has been implemented as standard method. Figure 9-11 illustrates an example for the identification of a three-input one-output model. It can be seen that the final identification result is a MISO model composed from SISO transfer functions.

Main GUI of ICAI MISO **ID Block** Resulting ICAI MISO **Model Block**

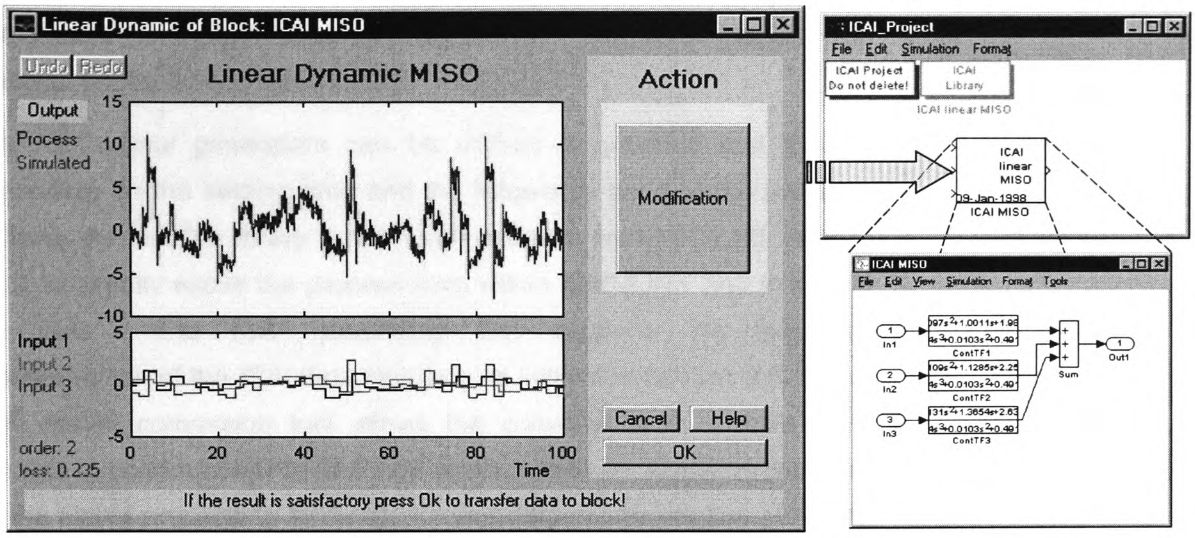


Figure 9-11. GUI of linear dynamic MISO ID block and structure of the MISO model block

Linear Dynamic MIMO ID Block

The *linear dynamic MIMO ID block* is composed from MISO sub-models utilising another macro block which controls as many MISO ID blocks as outputs are selected. These MISO ID blocks are worked through one after the other having the same properties as outlined above. An application example is provided in Section 9.5.3.

Static MIMO ID Block

The *static MIMO ID block* has been designed to fit multi-dimensional static characteristics. Similarly to the static SISO ID block the process data must contain a sufficient number of steady states to enable a satisfactory approximation. The standard method utilised is the multi-dimensional MGWM (Section 7.3). The GUI is currently restricted to models with two inputs and two outputs. This block offers a convenient way to fit measurement data to static characteristics as shown in Figure 9-12.

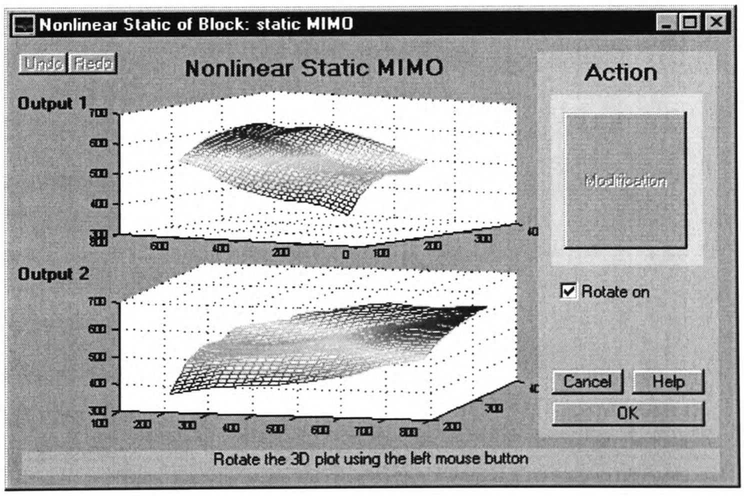


Figure 9-12. GUI of static MIMO ID block

ICAI Tools

Some tools have been developed for the *ICAI* toolbox to increase it's practical applicability (Figure 9-13).

The *RBS-signal* generators can be utilised to produce test signals depending on the settling time and the frequency band of the process. Utilising the *AD/DA library* it has been possible with MATLAB version 4.2.1 to directly excite the process from within SIMULINK and to record the data[10]. The *ICAI structuring* tool supports the graphical representation of the *ICAI* data structure as utilised in Section 9.3.1. The *ICAI model conversion* tool allows the conversions of models from discrete to continuous time and vice versa (based on functions provided by the signal processing toolbox). A more detailed description including some more pictures can be found in the *ICAI* help system.

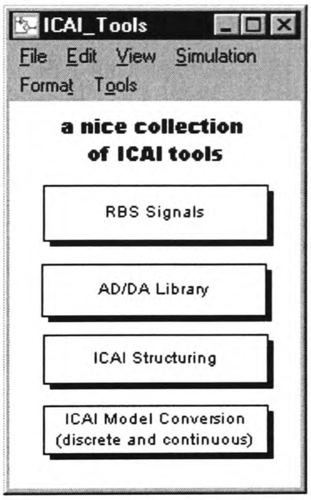


Figure 9-13. ICAI tools

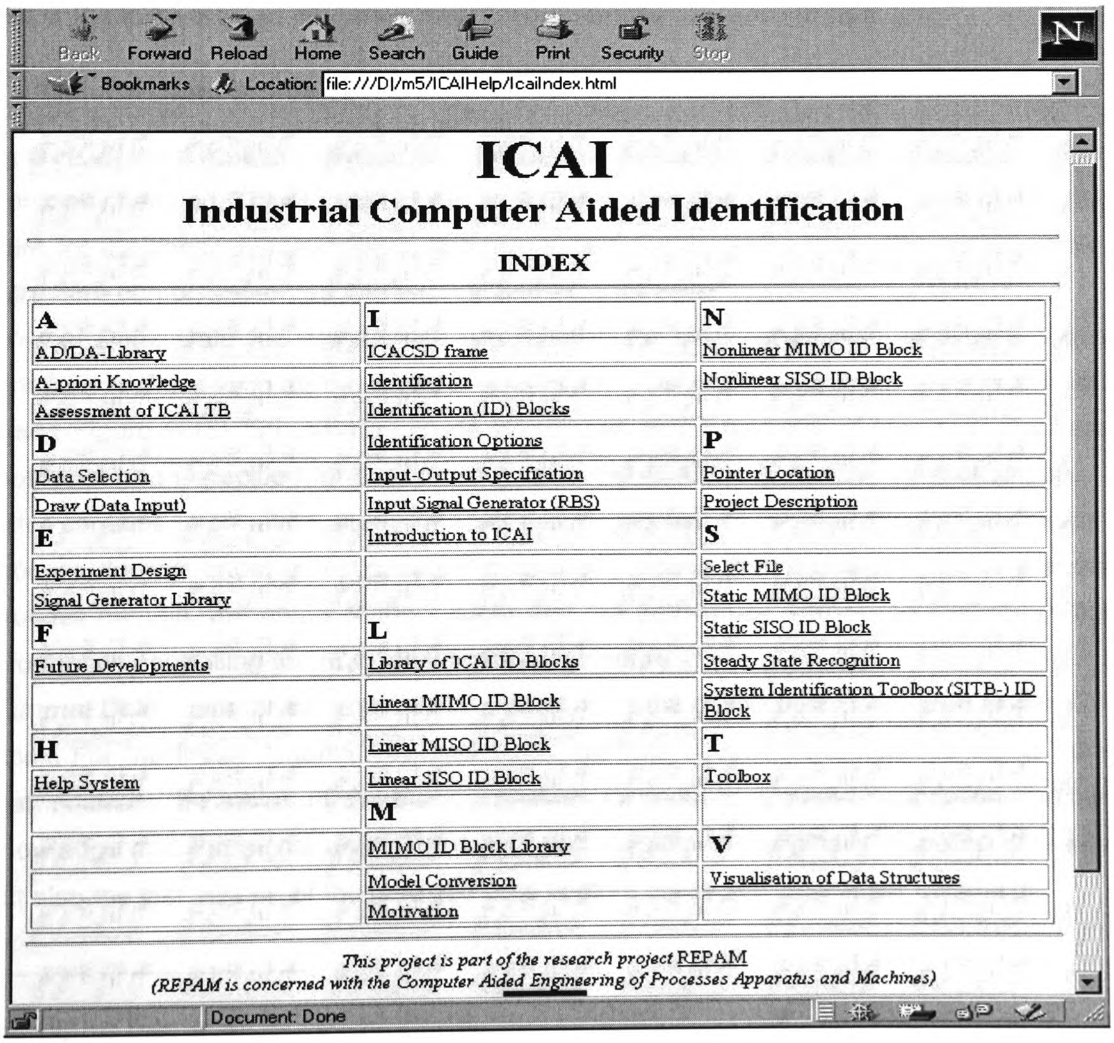


Figure 9-14. Index of ICAI help system

[10] This library was designed for an AD/DA card (PCI 20428w) and is currently updated for MATLAB 5.

9.2.5 *ICAI* Help System

The *ICAI* help system has been realised in html-format, which is easily extendible and versatile through the use of hyperlinks. It is accessible through standard browsers like *Netscape* or the *Microsoft Internet Explorer*. Furthermore it has been possible to publish the *ICAI* help system on the internet, such that updated versions were easily available (Körner 1997).

Currently, the help comprises approximately thirty html-documents, which contain about one hundred pictures to clarify the *ICAI* functionality[11]. Figure 9-14 shows the index for the *ICAI* help systems in order to provide an idea of the help system's contents. Most of the files are linked amongst themselves providing a *learning path* that introduces the uninitiated user to the *ICAI* philosophy and functionality.

9.3 Some Aspects of the Prototype Realisation

The functionality of the *ICAI* toolbox described above could only be achieved by utilising a capable data structure and a flexible program structure. The main aspects of this groundwork are outlined in the following and the limitations of the prototype are summarised.

9.3.1 The *ICAI* Data Structure

During an *ICAI* identification session much information in the form of data is generated, which has to be properly handled. These data are called *'session user data (SeUd)'*. An exemplar of the data structure for the linear dynamic SISO ID block is depicted in Figure 9-15. The entries within the data structure can be separated into *internal* and *external* data[12].

- *Internal Data:* Data for internal use are relevant for the management of program specific functions and are valid for the whole block diagram. Data of this category are labelled by *d* or *f* (see Figure 9-15) The *d*-data include information about the path of the block, the settings that have been specified for the currently activated *ICAI* block and the Project settings. The latter contains information about the user profile, the aim of the identification procedure, the process domain under investigation, the type of process, the standard units for a specified process domain, the name of the system and the pathcodes. The *f*-data (flag-data) include information controlling the *ICAI* identification procedure.
- *External Data:* Data for external use are relevant for the characteristics of each single *ICAI* ID block. For the linear dynamic SISO ID block the data structure comprises a status variable that indicates the status of the block, variables for input and output description, for a-priori knowledge and the signal data. The signal data contain the originally recorded data signals but also the last version of the modified data vectors utilised for parameter estimation.

---

[11] As the help has been written for a prototype development the wording is not always academic. The passive has been avoided to directly address the reader in order to motivate her/him to send a feedback and to incorporate some new ideas into the *ICAI* development.

[12] The structured labels of these data are composed from the names shown in Figure 9-15 starting with the name of the most upper hierarchy level (SuUd) to which the other names are added (separated by a full stop '.') until the last level is reached, which is finally specified. For example, the name of the output signal variable of the firstly generated linear SISO block within a SIMULINK block diagram is: *SeUd.LinSiso{1}.Data.Output*

The variables containing the a-priori knowledge comprise user knowledge that has been collected within the identification procedure. For example, the process behaviour (integral, oscillatory), the mode of operation (closed-loop), gain and settling time and the disturbance characteristics can be recorded and utilised for identification but also for validation. For the many details and the data formats, please see Appendix B.

Besides the clear structure, a primary advantage of this data structure (being linked to the *ICAI*-Project block) is that all data within the structure can easily be saved and loaded with the associated SIMULINK block diagram. The session user data are linked to the active SIMULINK block diagram, from where the *ICAI* actions are executed.

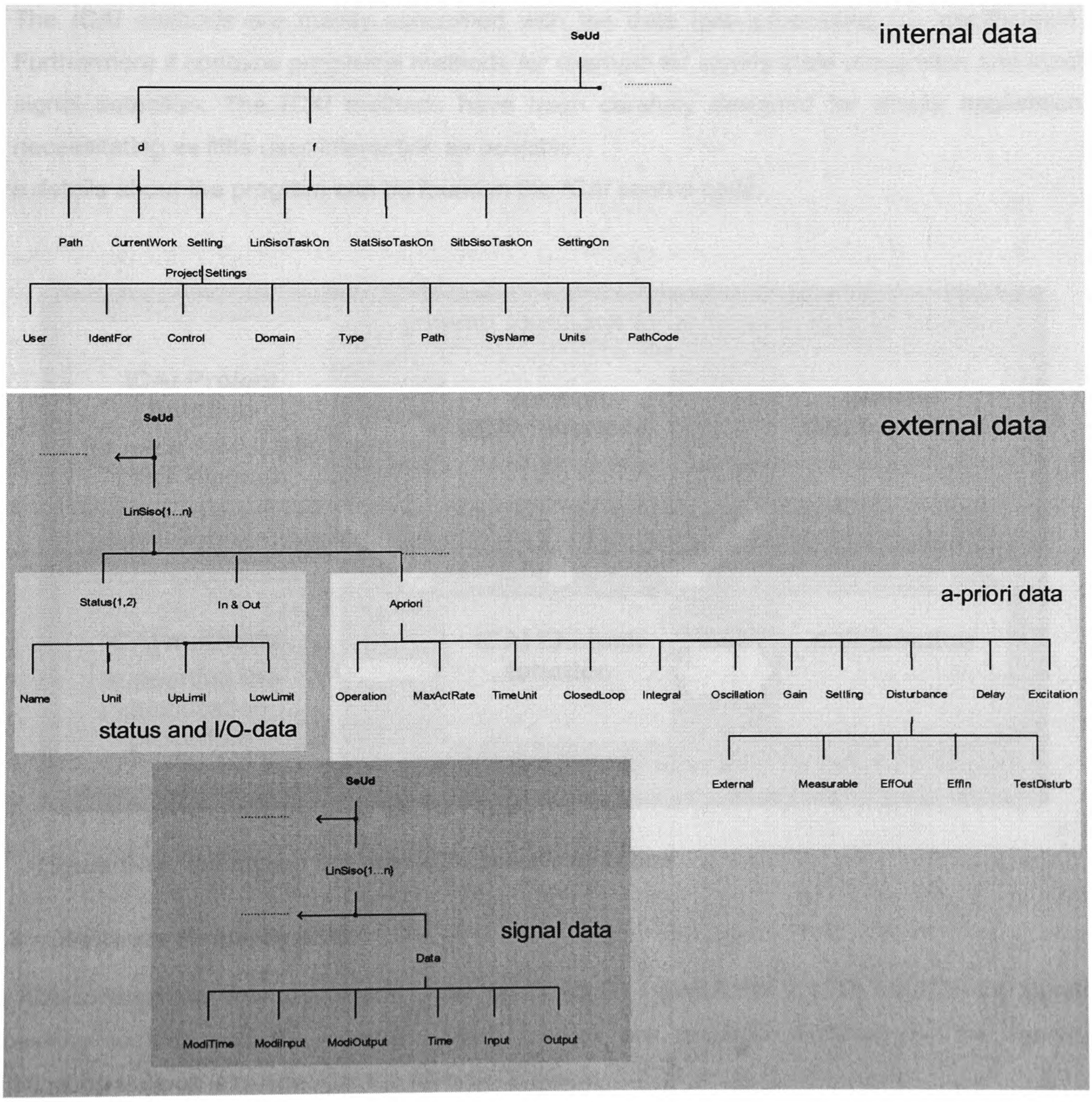


Figure 9-15. Structure of ICAI session data for the linear dynamic SISO ID block

9.3.2 The *ICAI* Program Structure

The *ICAI* program mainly consists of six types of functions (*m*-files), all of which are programmed in MATLAB. The *ICAI* functions can be separated into *general* and *block specific* functions and the *ICAI methods*. Figure 9-16 displays the interplay between these functions[13].

- The *general functions* are utilised by all types of *ICAI* blocks and comprise the *ICAI* project function that controls the *ICAI* Project (Section 9.2.2), the general *ICAI* functions that direct functions of all *ICAI* blocks (like the library facility) and the general GUI functions, which can be called by every *ICAI* ID block.
- The *block specific* functions are valid for one type of *ICAI* block only and consist of one *ICAI* ID block function and it's associated GUI function, with which it closely interacts. The *ICAI* ID block function controls the whole identification procedure utilising each type of *ICAI* function, when appropriate.
- The *ICAI methods* are mainly concerned with the data (pre-)processing for identification. Furthermore it contains peripheral methods for example for steady state recognition and input signal detection. The *ICAI* methods have been carefully designed for simple application necessitating as little user-interaction as possible.

More details about the program can be found in the *ICAI* source code.

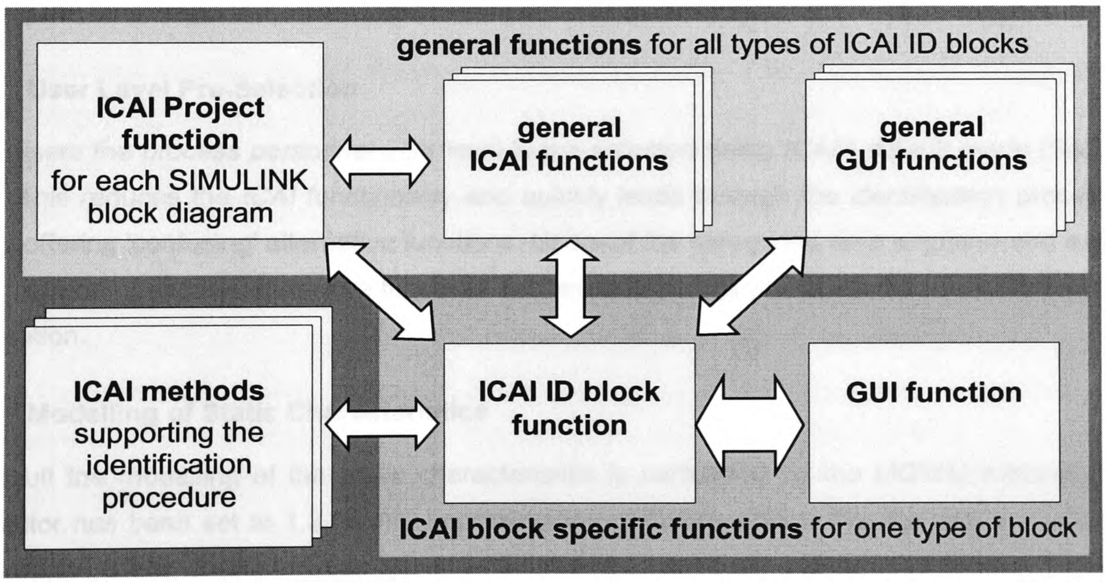


Figure 9-16. Interaction between ICAI specific functions

9.3.3 Software Requirements

The *ICAI* toolbox has been developed under MATLAB 5.1 / SIMULINK 2.1. To run *ICAI* the signal processing toolbox and the control system toolbox are required. Furthermore the system identification toolbox is needed for the *ICAI* SITB block.

[13] All *ICAI* functions are programmed following the switchyard concept.

9.3.4 Prototype Limitations

Naturally, there are some limitations within a prototype realisation. First of all the *ICAI* blocks have the same limitations as SIMULINK blocks because they are based on them. This means that the *ICAI* prototype realisation is restricted to the MATLAB/SIMULINK environment. Furthermore, only the process personnel level has completely been realised for the multi-variable *ICAI* ID blocks. Besides, there still is some potential for the improvement of the Wiener-Hammerstein blocks to provide user-level-specific guidance and to supply an identification method for the Wiener-model identification with non-invertible static characteristics. Also the reversion of actions ('undo'-button) have not been implemented completely.

Currently, SIMULINK cannot be operated 'hardware-in-the-loop' directly. It is impossible to run a SIMULINK block diagram with direct access to the pilot plant although this would be very beneficial for direct application of the toolbox. However, it is possible to generate C-code from SIMULINK diagrams utilising the MATLAB's real-time workshop for real-time applications.

9.4 *ICAI* Toolbox Standards

It has already been discussed that the applicability of the *ICAI* toolbox relies on the usage of sensible default and standard procedures in order to relieve the industrial user. This enables the user to apply identification with only very few application specific adjustments, if any. In the following the most important *ICAI* toolbox standards are summarised.

9.4.1 User Level Pre-Selection

For all users the *process personnel* user level is pre-selected being *ICAI*'s default mode (Section 9.2.2). This reduces the *ICAI* functionality and quickly leads through the identification procedure without offering 'confusing' alternative functions. Users of the categories *area engineer* and *expert* are authorised to access alternative functions for flexibility in data pre-treatment, identification and visualisation.

9.4.2 Modelling of Static Characteristics

By default the modelling of the static characteristics is performed by the MGWM method. The form factor has been set to 1.3 for the approximation of steady states. For the approximation of multi-variable static characteristics the MGWM method has been extended. For more details, please refer to Chapter 7.

9.4.3 Identification of Linear Dynamic Characteristics

The identification of linear dynamic characteristics has already been discussed in Chapter 4 providing practical guidelines for the main identification phases. With respect to the *ICAI* realisation some additional information is provided concerning standard and default settings in the following.

A-Priori Knowledge

As outlined in Section 9.3.1 the *ICAI's* data structure is prepared to carry a-priori information. If provided, this is utilised to pre-parameterise the identification methods (for example, the settling time is utilised to determine the sampling rate of the experiment) and to check the identification result against the a-priori knowledge (for example, the gain and the settling time of the final model are checked against the a-priori knowledge).

Experiment Set up

The basic experimental set up is outlined in Figure 9-17 and follows the guidelines given in Section 4.1.1 Generally, the system is excited by the D/A-output of the computer utilising a test signal that excites the relevant frequency range. The process response *y(t)* is filtered in order to get rid of aliasing effects. Via the A/D converter the signal is passed to the computer. After the experiment the data are available for evaluation and identification within ICAI.

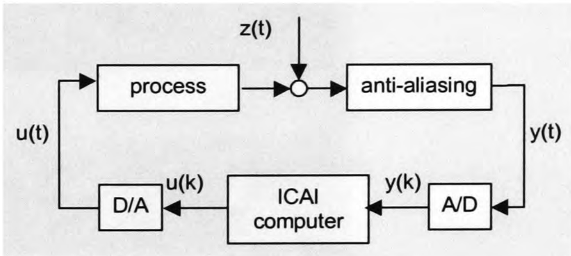


Figure 9-17. Experimental set up

Identification

At first the signal data are pre-processed. *ICAI* automatically detects if step, impulse, random or random binary signals are utilised as input signal and selects appropriate methods for the subsequent actions. Offsets and trends as well as outliers are automatically removed from the signal data *u(k)* and *y(k)*. Then input and output signals are normalised for proper weighting. For a-periodic step responses a numerical optimisation (Hankel and Reiner 1992) method has been selected that is specifically suitable because it is directly possible to determine an approximate deadtime (Körner and Schumann 1998a). For stochastic signals the CorMMLS method is applied. Utilising the CorMMLS method the generally separate phases of model structure selection and parameter estimation are combined. Thus the equation error for multiple models is calculated in parallel in order to determine the model structure from a set of models. For the final model validation the output error criterion is used comparing the simulated with the process output (see Section 8.3.2).

9.4.4 Standard Representation of Linear Dynamic Models

The properties of continuous- and discrete-time model representations have been already discussed in Section 4.3.1. Herewith aspects of computer implementation have especially been considered exhibiting several advantages of discrete-time identification. However, in the context of user interaction the criteria for the selection of an appropriate model representation have to be reconsidered because it is essential that the user understands the representation of the process model.

Especially if the model serves as a base for controller design in frequency domain - being mostly the case in the process industry - it is advantageous to provide continuous-time models because the use of different model representations in parallel could confuse the user. Furthermore most industrial users are familiar with continuous-time representations (Chapter 2). Therefore it is better to provide continuous-time models for user interaction.

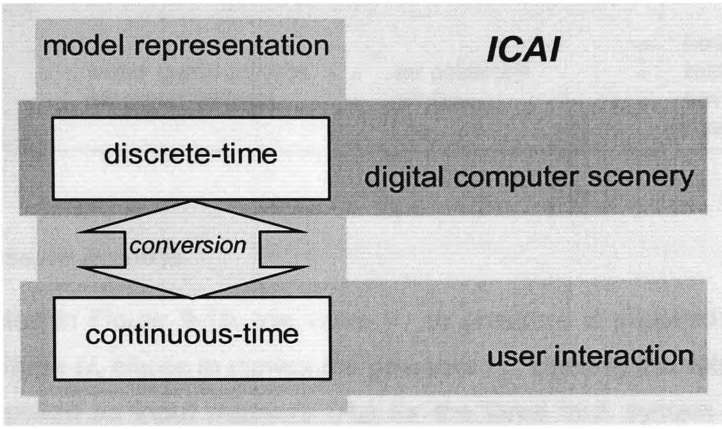


Figure 9-18. ICAI model representation

The final realisation of model representations within *ICAI* is shown in Figure 9-18. Discrete-time models are used internally in order to apply methods that are close to the computer scenery. For user interaction, however, continuous-time models are utilised that are close to the understanding of industrial users. Therefore a model conversion is necessary that transforms the internally utilised discrete-time model into a continuous-time model. This has been realised for the *ICAI* toolbox utilising MATLAB standard methods from the control design toolbox. The drawback of the conversion is a possible loss in model accuracy. However, this is recognised and the user is warned. Only the *ICAI* SITB ID block does not fit into the scheme (Figure 9-18) as it is the only block that is represented in discrete-time. However, it must be recognised that this block can be utilised by experts only.

9.5 Validation of the Approach

This section gives a brief summary of the tests carried out in order to verify and validate the proposed novel approach. The *novel approach*, in this context, refers to the procedures and methods that have been implemented into the *ICAI* prototype as outlined in the beginning of Chapter 9.

9.5.1 Description of the Test Processes

Various tests have been carried out in the laboratory with the laboratory staff and students. The investigated processes are summarised in Table 9-4. The UML has been already described in Section 5.2 (see Figure 5-3). The 3T (3-tank laboratory pressure process) and the LKR (laboratory air conditioning process) will be described below.

Table 9-4. Application examples

Process	UML (universal laboratory process)	3T (3-tank laboratory pressure process)	LKR (laboratory air conditioning process)
Output (Control) Variables	o water level o air pressure	o air pressure at every tank	o temperature o humidity
Input Variables	o water pump voltage o air pump voltage	o air pressure o air flow	o heating voltage o humidifier voltage o fan voltage (air flow)

3 tank Laboratory Pressure Process

This process is depicted in Figure 9-19. Via valve V_1 air pressure is supplied for this laboratory process. The input voltage U_v allows to control the pressure P_v. Utilising the valve V_2 the pressure P_v can be directly supplied as input pressure (P_E) for the three tank system or it can be send through the valve which is controlled by U_s in order to control the flow rate. It is possible to reduce the complexity of the process by switching off the last two tanks of the three tank system utilising valves V_7 and V_8. The pressure within the tanks is converted into a current used as output signal (I_1, ...,I_3). The valves V_3, V_4, V_5 can simulate disturbances.

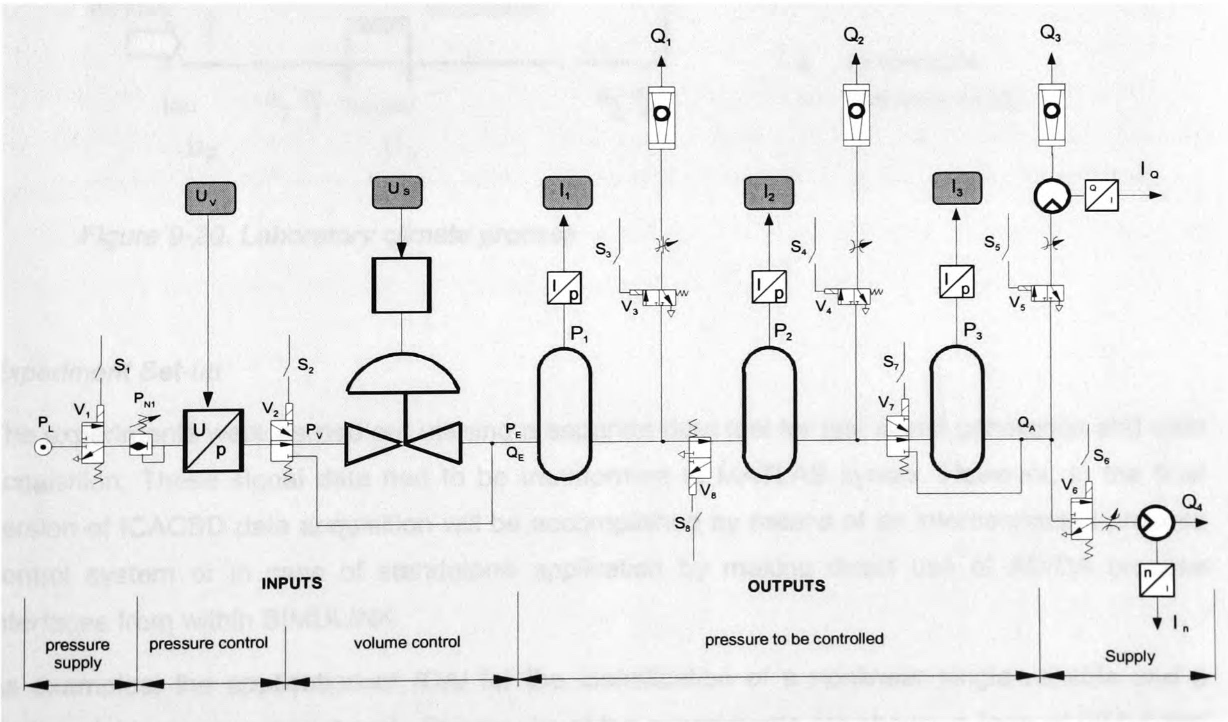


Figure 9-19. 3 tank laboratory pressure process

Laboratory Air-conditioning Process

The laboratory air conditioning process is depicted in Figure 9-20. It consists of a curved air channel that contains a fan at its inlet followed by a heater and a humidifier. The air channel ends with a mixing chamber at its outlet. This is the spot where relative humidity and temperature have to be controlled. Input variables are the voltage at the fan u_F affecting the air flow, the heater voltage u_ϑ affecting the air temperature ϑ and the voltage at the humidifier u_φ affecting the relative humidity φ. Two additional measurement devices are installed for the determination of the temperature and relative humidity within the air channel (however these have not been used for the described experiments). Naturally, this process exhibits strong couplings between temperature and relative humidity. An increase of the temperature in the mixing chamber (Figure 9-20) will decrease the relative humidity and vice versa an increase of humidity will decrease the temperature.

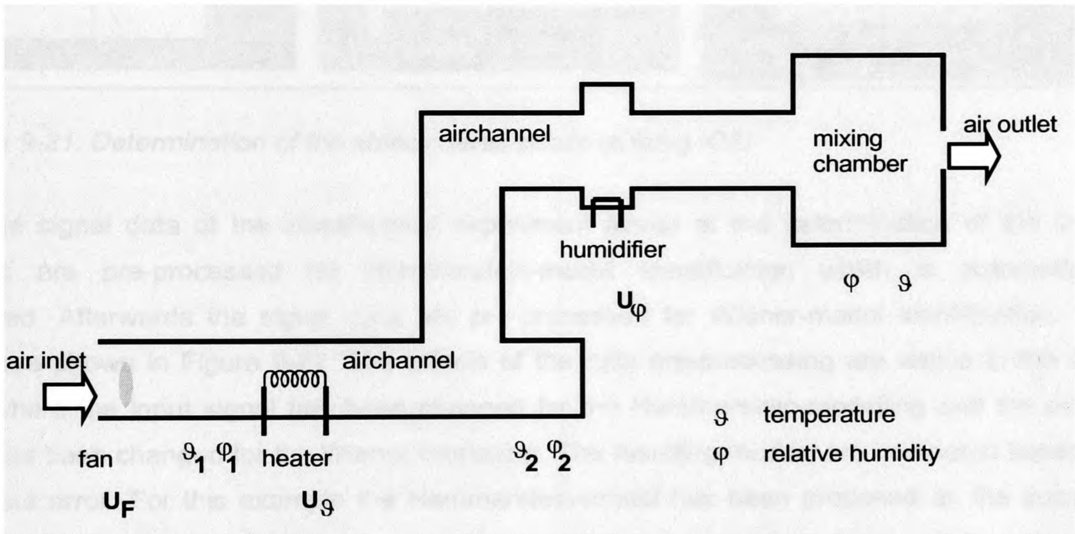


Figure 9-20. Laboratory climate process

Experiment Set-up

The experiments were carried out utilising a separate data tool for test signal generation and data acquisition. These signal data had to be transformed to MATLAB syntax. However, in the final version of ICACSD data acquisition will be accomplished by means of an interconnected process control system or in case of standalone application by making direct use of AD/DA process interfaces from within SIMULINK.

As examples, the application of *ICAI* for the identification of a nonlinear single-variable and a multi-variable process are shown. The results of the experiments are shown in form of SIMULINK and *ICAI* windows.

9.5.2 Identification of a Nonlinear Single-Variable Process

This example has been realised at the universal laboratory process (UML of Section 5.2) utilising *ICAI's* nonlinear SISO ID block (Section 9.2.4) and herewith the implemented *SNIP*. Only the airpressure controlled by the airpump voltage has been regarded while the water level was fixed in the operating point. At first the data from the steady state experiment are evaluated (interactively or automatically) to model the static characteristic (Figure 9-21).

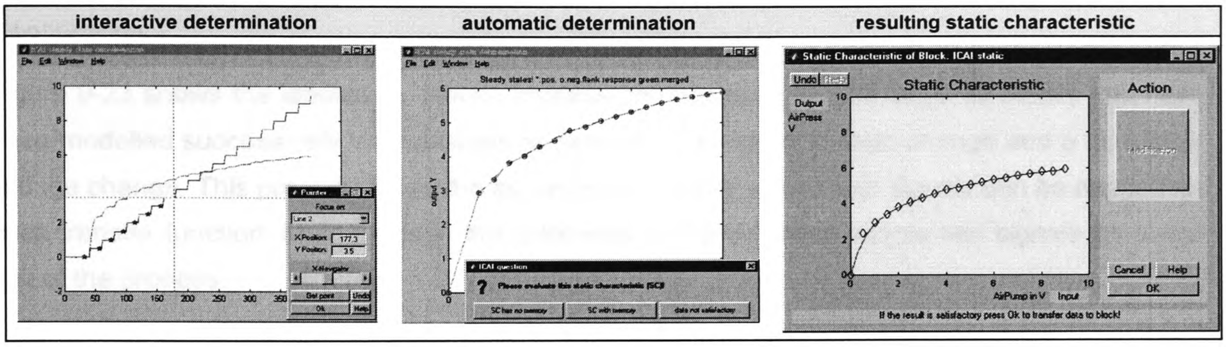


Figure 9-21. Determination of the static characteristic utilising ICAI

Then the signal data of the identification experiment aimed at the determination of the linear dynamic are pre-processed for Hammerstein-model identification which is automatically performed. Afterwards the signal data are pre-processed for Wiener-model identification. The results are shown in Figure 9-22. The effects of the data pre-processing are visible in the *ICAI* GUIs, where the input signal has been changed for the Hammerstein-modelling and the output signal has been changed for the Wiener-modelling. The resulting models are compared based on the output error. For this example the Hammerstein-model has been proposed as the superior model structure by *ICAI*. Actually, it comes close to the real model structure of the pressure system of the universal laboratory process because the nonlinear effect is introduced by the air pump, which has a very small time constant compared to the time constant of the main process.

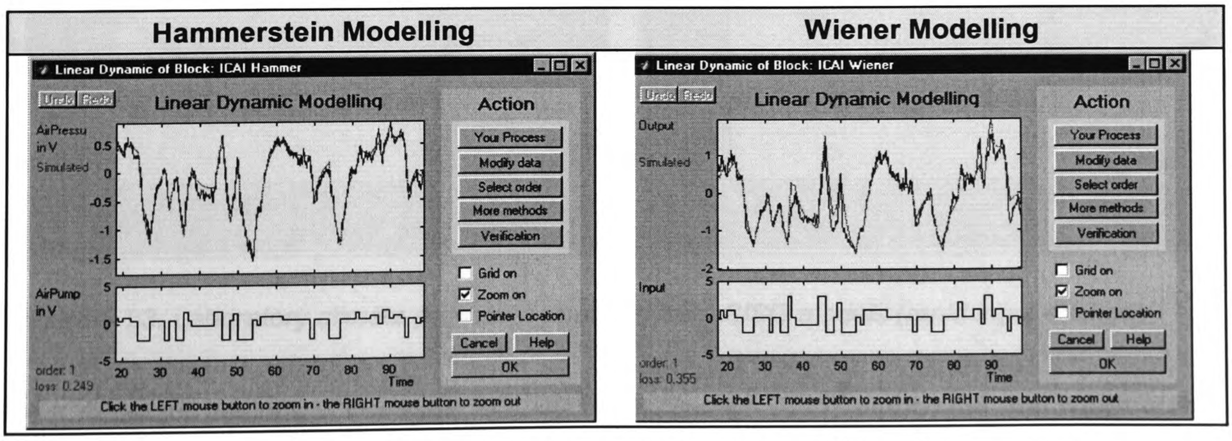


Figure 9-22. Hammerstein- and Wiener-modelling for the universal laboratory process

9.5.3 Identification of a Multi-Variable Process

The laboratory climate process (described in Section 9.5.1) has been selected as application example for the MIMO identification because of the strong couplings between temperature and humidity that necessitate MIMO modelling.

The experiments at the process have been performed in a linearisable region. The air flow was fixed such that the two inputs, heater voltage and humidifier voltage, influenced temperature and humidity respectively. Step response as well as random binary signals were utilised for identification.

Figure 9-23 shows the laboratory climate process composed from *ICAI* SISO ID blocks that have been modelled successively from process responses on a heater voltage change and a humidifier voltage change. This procedure has the advantage that very simple test signals can be applied for each transfer function. Most users in the laboratory preferred these simple test signals for a first test of the process.

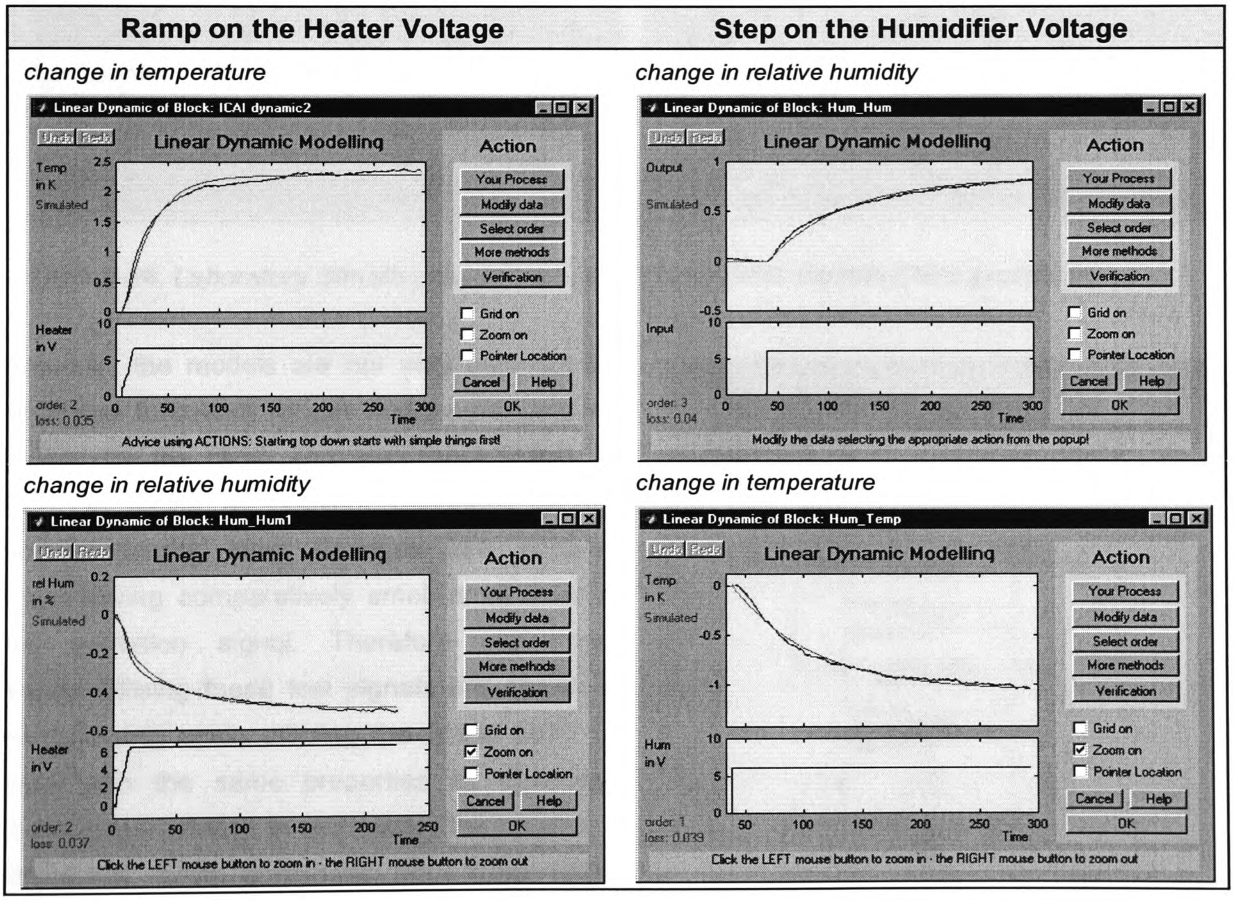


Figure 9-23. Laboratory climate process composed from SISO models (rapid input change)

The *ICAI* default settings of the linear SISO ID block (Section 9.2.3) support the user considerably providing input signal detection, offset correction, deadtime recognition and order determination. This way satisfactory results have been gained easily.

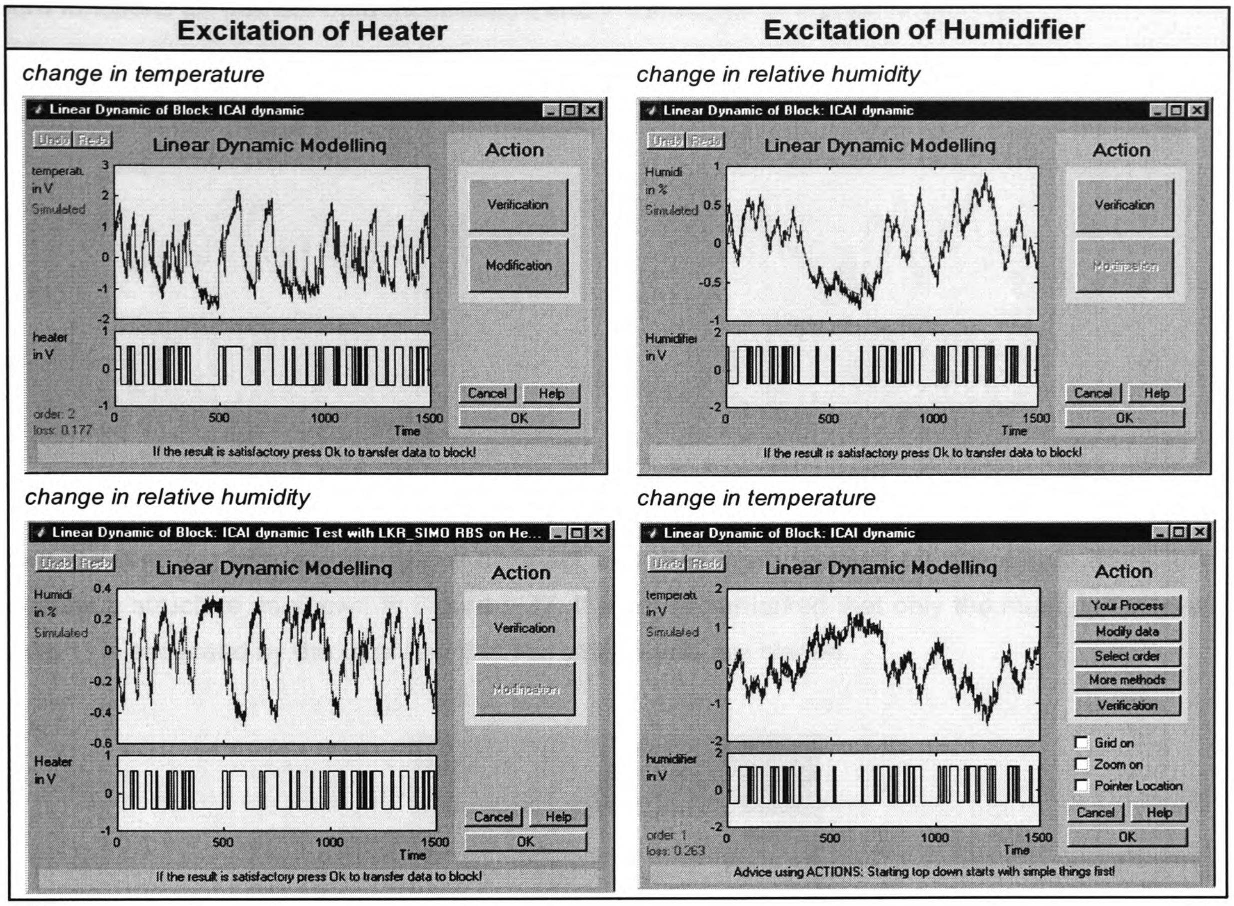


Figure 9-24. Laboratory climate process composed from SISO models (RBS excitation)

However, the models are not very accurate for the higher frequency ranges that are not properly excited by the ramp and step input signal. In order to excite all relevant frequency ranges random binary input changes are superior necessitating comparatively small amplitudes in the excitation signal. Therefore experiment results utilising these test signals are shown in the following again utilising the linear SISO ID block with the same properties as described above (Figure 9-24). In both cases the result is a SIMULINK model composed from linear SISO model blocks as depicted in Figure 9-25.

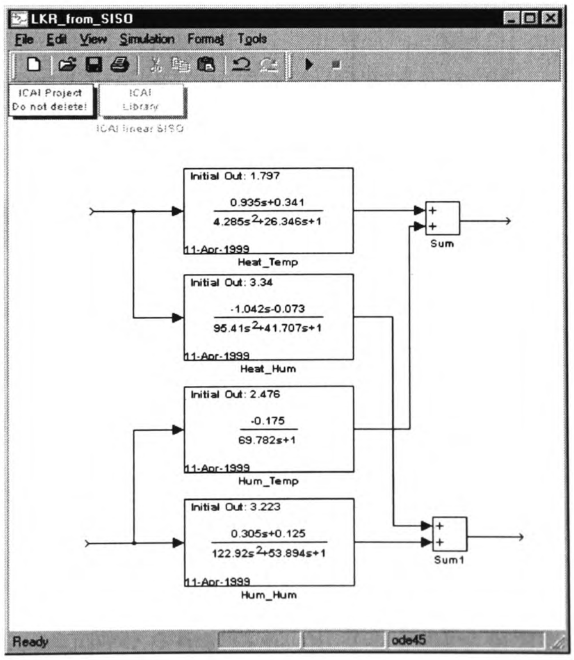


Figure 9-25. SIMULINK block diagram composed from SISO models

It is also possible to perform MISO modelling as supported by the linear MIMO ID block (Figure 9-26). However, this block is not as automated as the linear SISO block. It does not provide functions like automatic offset compensation, extra functions for manual data modification and the inclusion of a-priori knowledge.

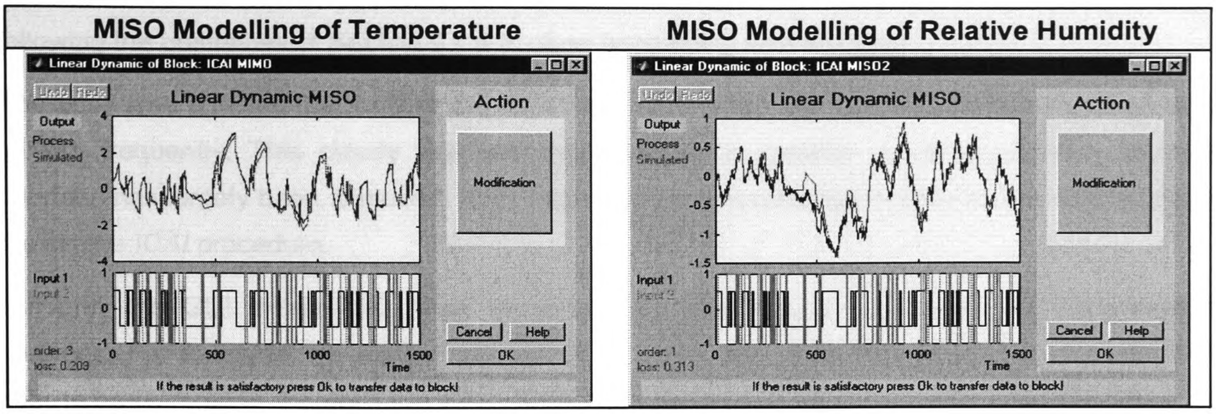


Figure 9-26. Laboratory climate process composed from ICAI MISO models

Here the resulting model looks more compact as it has been realised by one single block that contains a structure as shown in Figure 9-27. It must be remarked that only the most upper level (level 1) is accessed by the user and that the other levels are hidden.

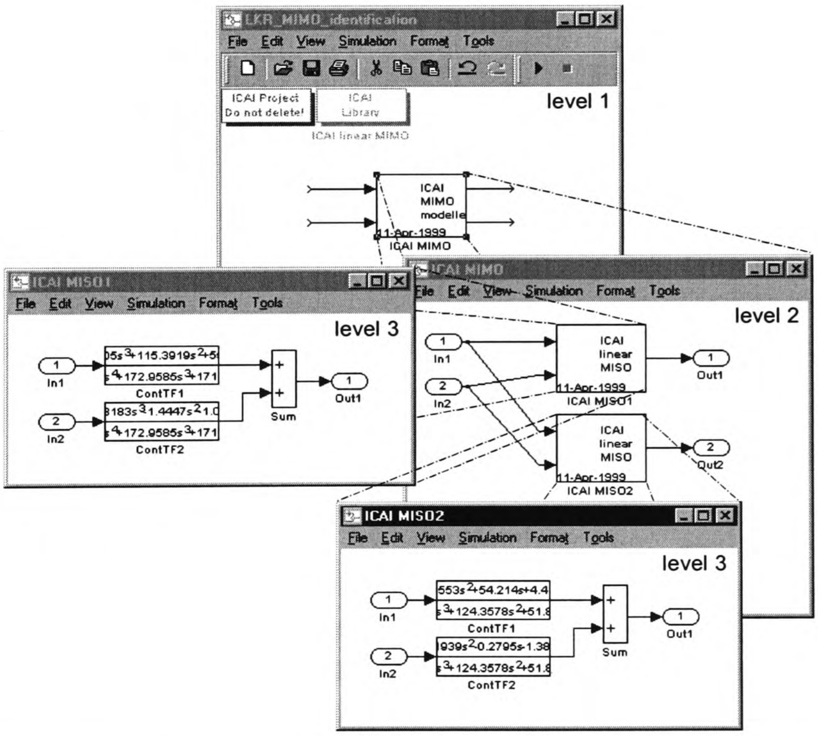


Figure 9-27. SIMULINK representation of the process composed from MISO models

9.5.4 Test of *ICAI* by Uninitiated Users

The application of the *ICAI* prototype by uninitiated users gave a feedback on the quality of the GUI that was utilised to iteratively increase the applicability of *ICAI*. This was an essential part of the GUI design process outlined in Figure 9-2 and considerably increased the applicability of *ICAI*.

For example, Figure 9-28 shows the initial user interface of the main window already presented in 1996 (Körner *et al.* 1996). Obviously there has been a significant improvement in the GUI design following the guidelines of Section 9.1.3 in close agreement with the users.

The tests also showed that neither the *ICAI* help buttons nor the 'right-button-help'-function was utilised frequently. This clearly indicates that the aim to develop a self-explanatory intuitive interface has largely been achieved. Even users inexperienced in identification were able to follow the simple *ICAI* procedure.

Not only the GUI design has been improved but also the applicability of the implemented methods. For example, the implementation of an automated input signal detection allowed the user to be relieved of the selection of appropriate methods, which is now done automatically with default values.

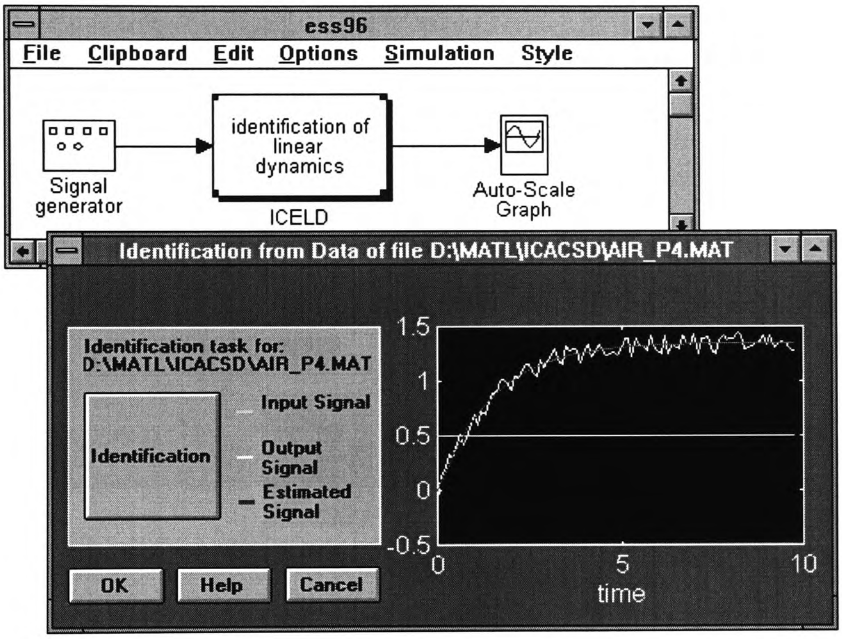


Figure 9-28. Former version of the linear dynamic SISO ID block (tested at the 3-tank system)

As a whole, the acceptance of the *ICAI* prototype was very satisfactory and the rate of successful identifications was much higher than those performed with other identification tools. The main reason for this success is the guided tour provided by simple self-explanatory user interfaces and the default methods that have been implemented.

9.5.5 Tests in Industrial Environments

It was intended to test the approach ultimately in an industrial environment. The test in industry, however, could not be carried out yet, mainly because there was no industrial partnership prepared to provide the significant involvement of industrial employees. However during this project the prototype development has frequently been discussed with industrial users based on the work in the laboratory.

Currently, tests at an absorption column mini plant (Figure 9-29) are carried out. These tests proceed in exactly the same way as the testing done in the laboratory based on real and simulated process data as outlined before. However, the mini plant absorption column has already industrial scale necessitating carefully planned testing.

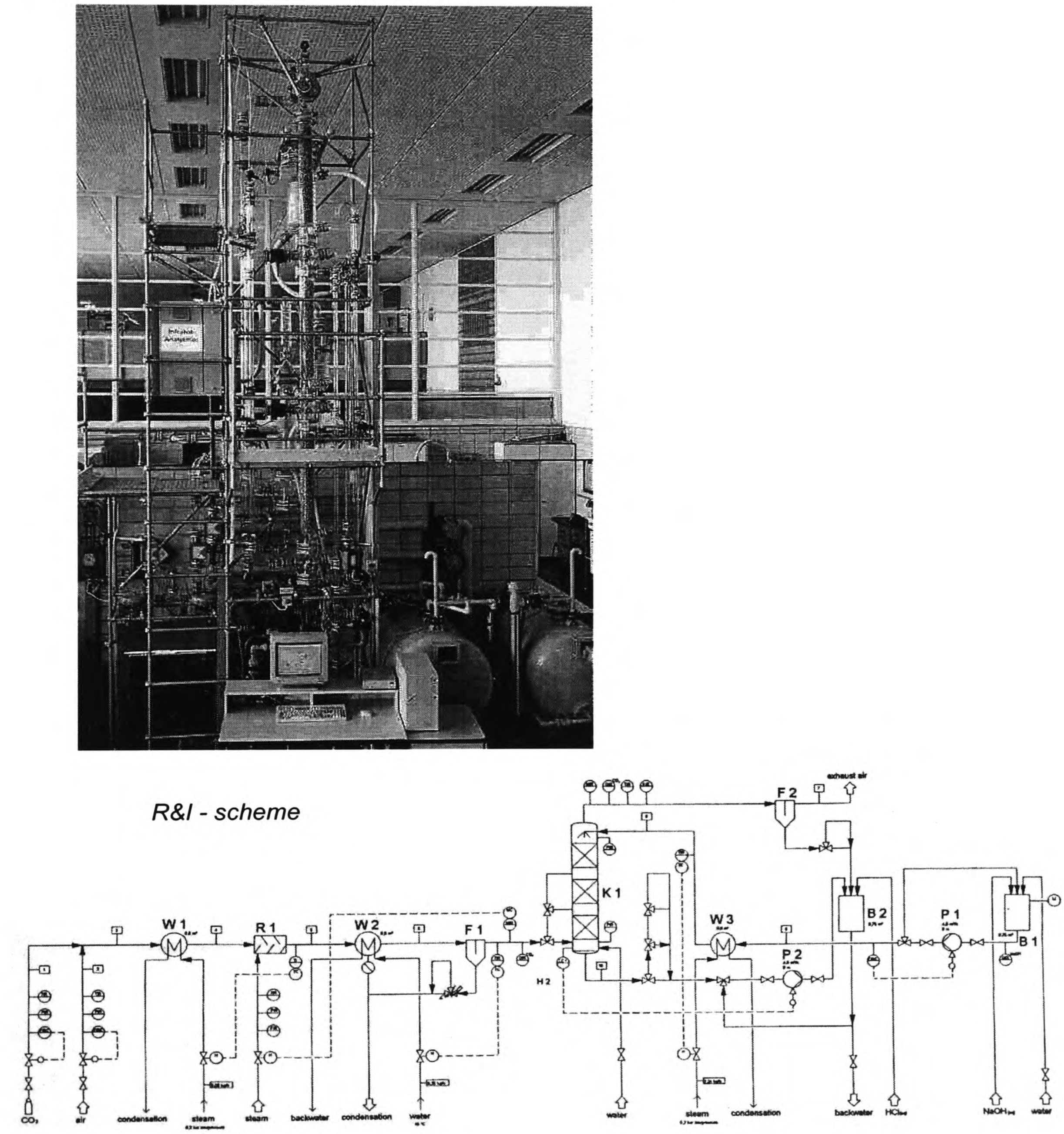


Figure 9-29. Absorption column mini plant

The mini plant is also utilised to test and develop the last bit of the envisaged ICACSD approach outlined in Section 5.1, the control design toolbox. As soon as the automatic control design toolbox is readily validated it is possible to test the whole ICACSD approach in an industrial environment with a satisfactory outcome also for the partner in industry. Therefore the *ICAI* toolbox will later on be applied in industrial environments, when the tools for the ICACSD approach are complete.

9.6 The Outcome of this Work in the Context of the Collaborative Project

In the context of the collaborative research project, this work provides an industrial CACSD scheme (Chapter 5) and an appropriate approach to identification that can even be applied by inexperienced users. These developments are utilised by the control system design module ICAC (Industrial Computer Aided Control) being currently under development (Syska *et al.* 1999). In this context the ICACSD scheme provides the strategy for the computer based commissioning system and the *ICAI* toolbox provides the process model for the ICAC toolbox as shown in Figure 9-30.

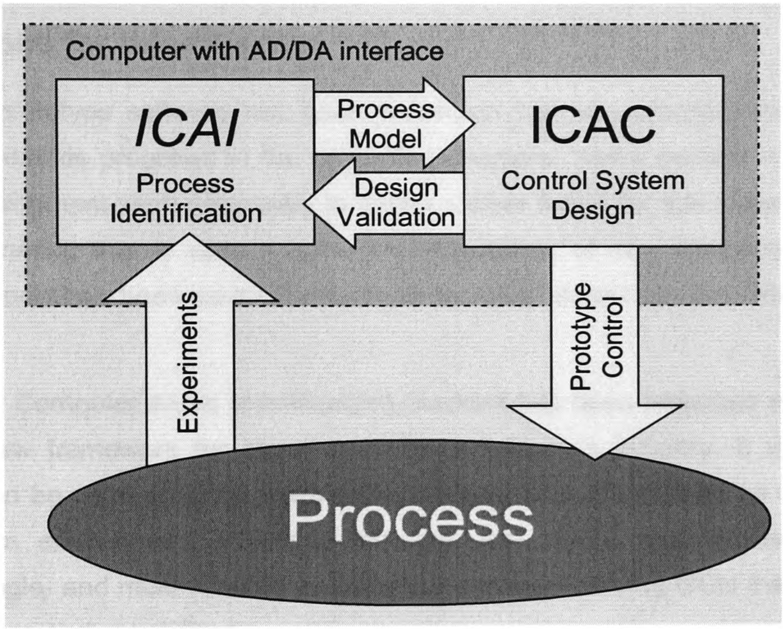


Figure 9-30. Interplay of ICAI and ICAC

This figure can well be compared to the improved ICACSD scheme (Figure 6-3). The *SNIP* of Figure 6-3 is incorporated into the *ICAI* toolbox, while the standardised CACSD procedure and the MES are covered by ICAC. The steps for the generation of the *ICAI* process model have been thoroughly described above. Then the process model is utilised by the ICAC toolbox for control system design following the improved standardised CACSD procedure (Section 5.1.3). Afterwards, the resulting prototype control is tested at the process.

Depending on the control performance reached at the real process the process model is finally validated or falsified. In case of unsatisfactory performance the identification will repeated to produce a better process model, if possible.

Most of the information processed by *ICAI* can be utilised by ICAC. For example, the *ICAI* project settings (Section 9.2.2) similarly influence the design process of the ICAC toolbox with respect to the user profile. Besides, the gathered a-priori information about the process behaviour can also be used for the control design. Furthermore the relative influence of couplings occurring in multi-variable processes is analysed in order to prepare the process model for the control design with ICAC.

Of course, this information is also accessible for other modules easily, because all data necessary for the simulation of the process model is directly inserted into a standard SIMULINK block structure and further information is simply linked to the project block structure (Section 9.3.1).

Naturally, it is also possible to use the experimentally developed *ICAI* process models for simulation directly. Besides a test of the complete process model, also parts of it can be simulated, thus getting a feel for the nonlinear static or linear dynamic influences. This possibility could be used for further assessment of the process model quality and as a key to important decisions on the control system design.

9.7 Conclusions for this Chapter

In this chapter a prototype software has been presented that was programmed to validate the procedures and methods proposed in the preceding chapters. Some general considerations for the prototype development were necessary to define a clear frame for this project and to allow a structured programming that is open for the implementation of new concepts. The MATLAB prototype development has been created, influenced by many ideas from industrial engineers and practitioners.

An *ICAI* (Industrial Computer Aided Identification) blockset has been proposed and implemented incorporating a new framework for identification in the process industry. It is integrated into SIMULINK and can be used even by inexperienced users who are capable of utilising a block-oriented simulation environment. Providing different user levels, standardised paths to the identification of single- and multi-variable processes are offered utilising GUIs that are adapted to the user's knowledge and capabilities.

Some application examples with different laboratory processes prove the applicability of the software and the incorporated methods. It must be recognised that the design of the prototype was an integral part of the research and an appropriate tool to validate the practical aspects of it. In order to clarify the research method that was utilised the next chapter explains the timely aspects and the strong interplay between the research in theory and practice.

10 General Discussion and Reconsideration

This chapter gives a brief review of the research methodology applied in the different stages of this work, from the broad starting point to the specific definition of this particular project and its realisation. Naturally, this has been an iterative procedure, which derives from the knowledge and experience that was gained during the project.

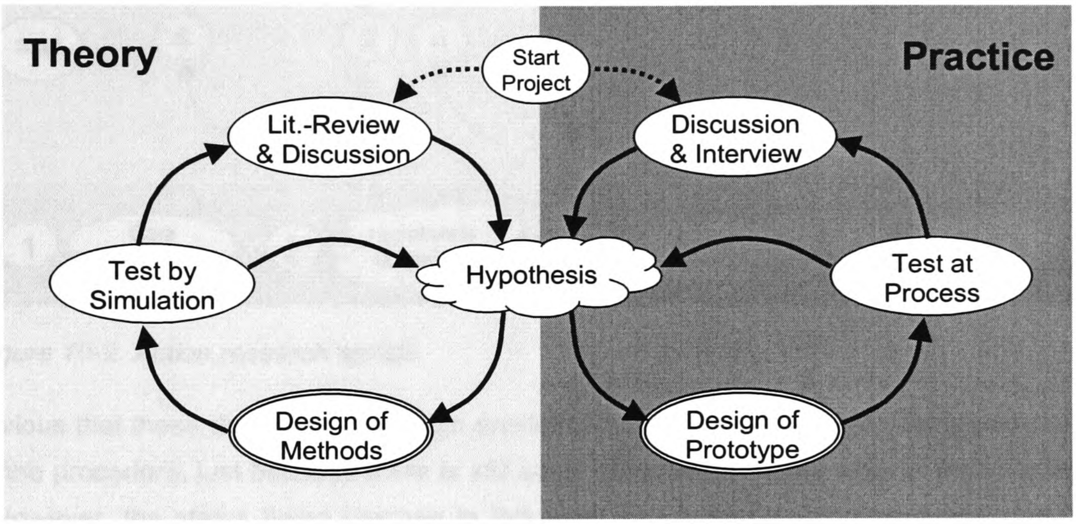


Figure 10-1. Research methodology combining theory and practice

Figure 10-1 depicts this in a simplified form emphasising that this work has been influenced by theory and practice. The diverse requirements of these domains with respect to CACSD system development have been already pointed out in Chapter 2. Therefore it is necessary to explore practice and theory differently. On the one hand theory is provided extensively by literature, which can be discussed with experts and implemented in more or less complex simulation or CACSD programs. On the other hand there are real processes and industrial users, who need usable tools for complex tasks, especially within the commissioning phase. From both domains conclusions must be drawn, which affect each other in the hypothesis, which is located in the centre of Figure 10-1.

Figure 10-1 also displays the activities within both domains. These activities are part of loops, which are worked through respecting the current intention of the research project. This way the loops have been worked through many times to develop practically useable schemes and mathematical methods, each time with increasing knowledge. Therefore the loops can be also viewed as spirals. This proceeding is very similar to *'action research'* being widely used in evaluation research depicted in Figure 10-2 (Lewin 1985). In fact, with each iteration (no matter in which field) a piece of knowledge and experience has been gained affecting the hypothesis being the heart of this thesis concerning project aims, methods, prototype development and even methodology.

This is the position where aim and current state have been compared and redirected. Each time a 'higher level' within this research has been reached, which allowed a critical assessment of the efforts undertaken so far. As a result the assessed efforts could be intensified or new directions could be taken.

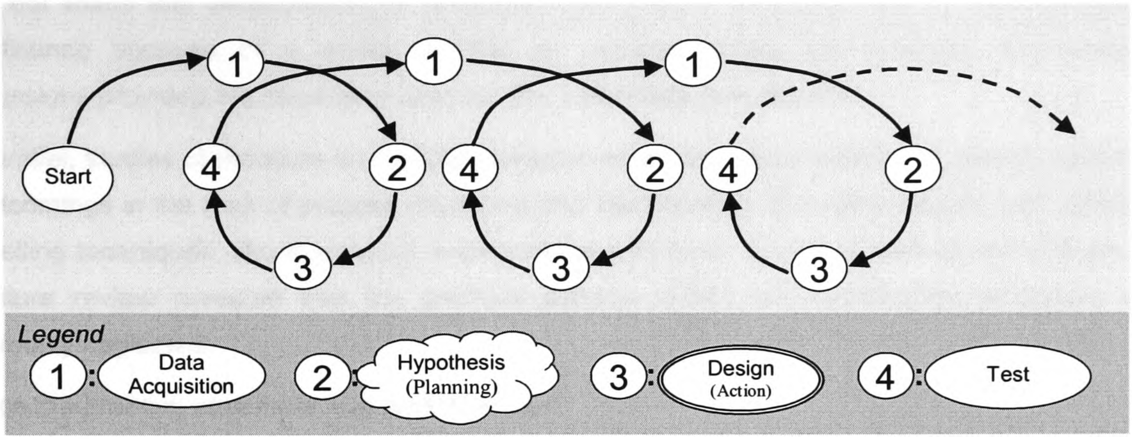


Figure 10-2. Action research spirals

It is obvious that these diagrams – although providing starting conditions – do not provide a clear end of the procedure, just because there is still some potential for future work in the investigated area. However, the status being reached in this work concerning methodical development and realisation of a prototype for process identification represents a good base for future developments (Chapter 11).

Within the following sections Figure 10-1 will be explained step by step following the path which this research took, while all major design decisions and results are reconsidered and discussed on the basis of the experiences gained.

10.1 Decision on the Aim of this Project

The start for the collaborative research project between the University of Glamorgan and the Fachhochschule of Hannover was based on a very broad overall problem statement[14]:

"None of the existing CACSD programs addresses the need of area engineers in industry!"

Figure 10-1 shows that this starting point triggered activities in practice and theory and a variety of methodologies has been applied to get a clearer picture of this project within the overall aim of developing a user-friendly CACSD tool.

With respect to the practical part a 'hands-on' overview of existing CACSD programs was performed initially. The VDI-workshop (Schumann 1994) provided the opportunity to view and test a variety of about 40 software-tools altogether covering the field of CACSD thoroughly.

[14] This starting point was formulated by Professor Schumann based on his expertise as chairman of the VDI-workshops 'Regelungstechnische Programmpakete (Control Engineering Programs)'; 1989, 1991 and 1993 in Düsseldorf and his reviews of the field (see e.g. Schumann 1989 and 1994)

Furthermore a series of meetings with representatives of industrial companies started in order to get a clear view for their needs with respect to practical CACSD. However, a complete statistical evaluation of the results of these meetings was impossible not only because of the relatively small number of interviews but also because of the different applications. Besides, a questionnaire was sent out within this collaboration by Strickrodt (1997) but it could not help to reach statistical significance because of a limited number of useable replies. Nevertheless, the resulting discussions provided the necessary base for this industrially oriented work.

In parallel, studies of literature on CACSD developments have been carried out, clearly exhibiting shortcomings in the area of process modelling and identification. The familiarisation with different modelling techniques, like theoretical, experimental and library based modelling and a thorough literature review revealed that the practical aspects within the identification procedure are commonly neglected.

Hence the *problem statement* was redefined as:

Generally, the current CACSD programs are not geared to the requirements and skill level of area engineers in industry. Especially the support of the commissioning phase is neglected. In particular, there is no practically suitable approach existing, which allows area engineers to build a structured process model from experiments suitable for systematic control design.

Consequently the *need* was formulated as a directive for this project:

A practical approach to system identification techniques must be developed that is based on the everyday work expertise of area engineers in industry and their knowledge about the process behaviour. The resulting process model must be suitable for industrial control system design.

On this basis the decision to pursue the *Structured Approach to Identification Techniques for the Analysis of Industrial Processes* was made. The detailed subject, need and aim of this work have been described in Chapter 1. From the present point of view, after completing the project, all considerations that led to the decision for this emphasis of the work are still valid. Fortunately also other researchers discovered this valuable practically oriented field lately and there have been some few contributions in literature in the domain of experimental modelling to explicitly support the industrial user (Hahn and Nöth, 1997).

10.2 Research and Development of an Industrially Suitable CACSD Scheme

The original idea of building an industrially suitable CACSD scheme triggered this very first part of the research, strongly emphasising the practical aspects in the commissioning phase (right half of Figure 10-1). In interviews with technicians and engineers it was detected that, especially for complex MIMO processes, structural design support is needed and first layouts of a practical MIMO design procedure were drawn. The following literature review (Chapters 3 and 4) revealed that no industrially suitable scheme had been developed before. Orienting at the industrial engineers way of '*start simple - add complexity only if necessary*' the ICACSD scheme has been designed (Chapter 5). To keep the scheme as simple as possible a type of process model was sought that is able to describe nonlinear processes and that can be incorporated into a stepwise procedure oriented at industrial control system design. From the literature review, and considering the practical needs of control design in the process industry (Chapter 2), it was decided to reduce the variety of possible models to Hammerstein- and Wiener-like models (Section 4.5).

In order to test the applicability of the ICACSD scheme simulations were performed (left half of Figure 10-1). Necessary changes within the procedure were detected, new hypothesises were built and the scheme was revised for the next test. When the scheme reached a satisfactory stage experiments on pilot plants were performed (right half of Figure 10-1). Besides the handling, also the applicability and the commercial software and hardware support have been evaluated within a prototype realisation (Section 5.2). No commercially available software tool was found that satisfactorily supports non-expert users supporting all CACSD design phases, especially in case of multi-variable processes. Therefore a variety of different tools was utilised for different stages within the ICACSD scheme with all the accompanying drawbacks.

After tests on different pilot plants the stepwise increase of the process model complexity was reconsidered and it was decided that the procedure should start based on a model being as good as possible. Herewith the modelling phase of the *improved standardised CACSD procedure* has only to be performed once for the subsequent control system design stages (Section 5.1.3).

Overall the ICACSD scheme proved to be so useful that still most experimental work in the students' control laboratory is based on this simple scheme. Of course, there is still scope to develop more advanced schemes utilising alternative process model structures (and corresponding controller structures) that are closer to the real process structure, thereby fulfilling theoretical conditions. For example, it would be sensible to develop specific schemes for specific applications as lately realised in the field of power plants by Rode and Krüger (1998). Due to the temporal constraints of this project and aiming at a broad applicability, this work has focused solely on the identification phase of the general ICACSD scheme.

10.3 Research and Development of a Standardised Procedure for Nonlinear Identification

The process modelling is the first phase of the *standardised CACSD procedure* within the ICACSD scheme and serves as the base for control system design. Therefore the quality of the model directly influences the quality of the control system design. However, there is hardly any time for extensive modelling and identification in the commissioning phase and industrial users are not aware of all the facilities in identification theory. Consequently a reliable systematic standard identification procedure tailored to the industrial user's needs is required. Considering the kind of process models utilised within the model evolution scheme, only two different models, namely static and dynamic models, had to be investigated. With respect to Figure 10-1 this information is part of the hypothesis containing all the knowledge accumulated within the preceding phases. This knowledge directs the complexity and aim of the standardised nonlinear identification procedure (*SNIP*). Although all the steps within this procedure are well known in literature it is the combination of simple steps to yield an understandable nonlinear model which makes the procedure so useful (Chapter 6). Naturally, a standardised and therefore simplified procedure based on simple models cannot guarantee an optimal result which may be possible with more degrees of freedom. However, for an industrial user a good identification result following standard design paths may be the only possible result achievable on the basis of limited expertise and little time available.

After simulation tests the procedure could be implemented into the *ICAI* prototype for Wiener and Hammerstein-modelling and checked on pilot plants. Also the identification of both models in one step has been tested (Section 4.5). However, the proposed two-step procedure is clearer in application, not only because the experiments are simpler but also because the steps within the procedure are very transparent, which was appreciated by uninitiated users. Therefore research into simply applicable methods for the modelling of static and dynamic characteristics was intensified.

10.4 Research into Methods for the Modelling of Static Characteristics

The starting point for this more theoretically oriented part of the work (left half of Figure 10-1) was the necessity to find a robust and simple method for the modelling of single and multi-dimensional static characteristics from data. The well known difficulties in the application of standard methods for interpolation and approximation, particularly in case of multi-dimensional static characteristics utilising polynomials or splines, prompted investigation into alternative methods like neural net and fuzzy modelling. These methods were tested with respect to performance and simplicity. Special attention was directed to the number of free parameters and the number of iterations necessary for good results. Many tests have been performed with MATLAB toolboxes and some self-coded programs.

Additionally a final year student project was arranged and supervised with the aim of supporting the user during the modelling of the static characteristics. A simple tool was programmed called *EasyStat* that was able to perform experiment design, hardware in the loop experiments and the evaluation of one- and two-dimensional static characteristics. For a short description see Appendix A.

Within the literature search a surprisingly simple method was found that has been based on the generalisation of the weighted mean method and that is able to perform interpolation and approximation dependent on one single parameter. However, test implementations revealed that excessive trial and error was necessary to determine this parameter. Therefore this method was modified for sensible practical application. At first SISO problems were tackled and the applicability of the new modifications was tested by simulation. After the results for the one-dimensional static characteristic were satisfactory the method was extended to multi-dimensional problems. Furthermore some extensions were developed that improved the numerical efficiency of this method and it was shown that the modified weighted mean method could be viewed as a special case of fuzzy and neural network methods (Chapter 7). Finally the modified weighted mean method was implemented in the *ICAI* prototype.

10.5 Research into Methods for the Identification of Linear Dynamics

In the literature review it has been well acknowledged that the identification of linear dynamics has been a very active research field for many decades. Many successful applications of system identification are published in the literature. However, most of these applications are not directed at practical use. Therefore this project has set the focus on the development of well manageable identification methods. This ambitious task was directed by specific side conditions, for example to aim at industrial PID control and not at minimum variance or stochastic control (which can be seen as the influence of the hypothesis of Figure 10-1). Therefore only general transfer functions with deterministic model structure have been investigated, ignoring noise models. Thorough investigations of this field doing literature research and simulation tests resulted in a matrix rating different identification methods (see Section 4.4.3). From this matrix the promising correlation least squares (CorrLS) method was selected. However simulations revealed that the method was not always as reliable as required. Hence the numerical efficiency had to be improved. Good results were achieved by the integration of a robust and well tested multiple model least squares method (MMLS) (see Section 8.1.2). Many practically important aspects had to be studied for a proper application of this method within the identification procedure as well, for example the data pre-treatment and input signal design because this identification method should be utilised as part of the *SNIP*.

10.6 Software Aspects and Implementation

A main part of this work has been the implementation of the derived results into software. On the one hand (left half of Figure 10-1) methods had to be developed and implemented into a software environment for tests. On the other hand (right half of Figure 10-1) practical aspects like the *SNIP* had to be realised supported by a GUI that guides the user through the identification task. It needed numerous tests and iterations in theory and practice until satisfactory results were gained. The main aspects of the software development cycles are described in Chapter 9.

In the following some aspects of the *ICAI* toolbox development are outlined:

- It was marginally impossible to build up sensible data structures as long as MATLAB was restricted to one and two dimensional arrays but when MATLAB provided multi-dimensional and structure arrays this proved to be so useful that parts of the program had to be rethought and reprogrammed.
- New program concepts could be realised following the idea of putting all relevant data in a special structure outlined in Figure 9-15 (see also Appendix B). Furthermore the realisation of the *ICAI* Project with different user levels became possible and new internal standards could be introduced.
- When the *ICAI* development started with MATLAB it was not possible to do object-oriented programming. This drawback was accepted and the procedural programming of functions commenced. Now, as MATLAB allows the implementation of object-oriented paradigms it is worthwhile to think about recoding to benefit from these techniques.

10.7 The Overall Results Matched against Expectations

Overall, the expectations of this work being outlined in Chapter 1 have been fulfilled and even somewhat exceeded. With respect to the broad scope of the thesis it was necessary to consider various aspects within the field of CACSD with a clear emphasis on process identification. As a result the thesis comprises five major contributions:

1. The ICACSD scheme was developed and tested setting the main frame for the project.
2. A standardised identification procedure for nonlinear processes has been elaborated as part of the ICACSD scheme. This procedure was equipped with two improved algorithms (contributions 3 and 4).
3. The generalised weighted mean method (for the approximation of single and multi-dimensional static characteristics) was massively modified for simplified application.
4. The correlation multiple model least squares methods for the identification of discrete-time linear dynamic models was elaborated by combining two successful standard methods.
5. A prototype tool was developed, which lays ground for the integration of the whole ICACSD scheme into a block-oriented simulation environment.

The tests of this prototype have been very satisfactory. The experiences gained thereby were valuable to reconsider the work and to introduce further improvements, which have either already been implemented or are proposed as future extensions in Chapter 11. However, the implementation efforts have been underestimated. The main reason lies in the varying backgrounds of potential users and the wide gap between *'thinking at university'* and the *'action in practice'*. The prototype was frequently discussed, not only with colleagues but also at international conferences. However, pushing the prototype to industrial usability by testing the software on industrial processes still requires some minor refinements. Therefore it is advisable to carry out further tests in industry and to bring together abstraction and reality more closely.

Furthermore it was experienced that the varying experiences of different users are problematic, because what one user understands easily could confuse the other due to the differing backgrounds. Therefore three user levels have been introduced in the *ICAI* prototype to provide the means to address different groups of users adequately.

The generally good match between the goals of the project and its results show that the original expectations were justified. Besides it has been shown that further work in this direction is still promising.

11 Conclusions and Recommendations

This work presents a systematic approach to identification, which is aimed at the controller design during the commissioning phase of industrial processes in the process industry. It concentrates on the standardised development of black-box models aimed at PID control design and has been verified in a prototype implementation. In the next sections the conclusions are drawn and recommendations for future work are provided.

11.1 Conclusions

Major emphasis of this research was put on interviews in industry in order to extract the main requirements for the new approach and on the practical examination of available software tools. It turned out that the identification of nonlinear or multivariable processes is specifically difficult for the average industrial user concerned with the commissioning particularly because currently available identification software is difficult to use requiring specific knowledge and offering a wealth of functions. The efforts to bridge the gap between academic equipment and practical needs by offering special training for non-expert users have not succeeded due to the lack of time in industry to get familiar with the software tools provided; in other words: *The more advanced identification becomes, the more advanced must be the human operator.*

Hence it was found that an easy to follow, intuitive and stepwise identification procedure for industrial processes is missing, which is aimed at control and designed to satisfy industrial needs. Three main requirements were formulated:

1. The user should be guided through transparent standard identification paths especially in case of nonlinear or multivariable process identification tasks.
2. Only few identification methods with easy parameterisation and a simplified experiment set up should be provided.
3. The identification task should be integrated into a block-oriented simulation environment, such that the user can utilise one environment for identification, simulation and controller design.

Many steps were necessary towards the realisation of a software prototype for identification that fits to these requirements. The *main novelties* that resulted from this contribution are:

- An *Industrial CACSD (ICACSD) scheme* has been proposed for the solution of practical controller design tasks in the process industry, which consists of a *model evolution scheme* and a *standardised CACSD procedure* oriented at the industrial users thinking of *start simple, add complexity only if necessary*. The model evolution scheme reflects the traditional way of doing control system design in a systematic way. The standardised CACSD procedure only supports a constrained model complexity (utilising simple Wiener- and Hammerstein-model structures) that can be handled easily and it provides a good reproducibility of the gained results by restricting the variety of possible solutions. The feasibility of the proposed ICACSD approach was tested at different pilot plants.

- For the identification of Wiener- and Hammerstein-models a *standardised nonlinear identification procedure (SNIP)* has been proposed. The pragmatic procedure offers a transparent approach to experimental modelling for industrial users. It starts with simple pre-experiments at the process, which provide insight into the process behaviour and are useful to determine the static characteristics. The latter is utilised to determine a suitable region for the final identification experiment and to compensate for the nonlinear effects. Therefore it enables the usage of conventional identification methods for linear dynamic characteristics. The SNIP has been also linked to the ICACSD scheme evaluating nonlinearities and couplings of the process model with respect to the model evolution scheme.
- For the modelling of nonlinear static characteristics within the SNIP the generalised weighted mean method has been modified in order to allow simple application and to gain sensible results without trial and error. The *modified generalised weighted mean (MGWM)* method does not need user assistance and still is numerically simple, while it is able to provide sensible results for interpolation and approximation including the case of multi-dimensional static characteristics.
- For the identification of linear dynamic characteristics within the SNIP two well-tested methods for parameter estimation have been combined. The combination of the frequently applied correlation least squares (Cor-LS) with the multiple model least squares (MMLS) method results in the two-step CorMMLS method for the estimation of parametric discrete-time models. The concurrent estimation of multiple models makes this algorithm especially appealing if order and deadtime of the model are unknown, which happens often in practice. The combination with correlation technique leads to unbiased estimates also in the presence of a coloured zero mean noise signal as long as the input test signal applied is statistically independent of the noise. Besides, the two-step method has the advantage that it can generate valuable a-priori information within the first step (the correlation) and that the parametric estimation is performed on a reduced (correlated) data vector, which is especially advantageous in case of MIMO identification, where the effects of utilising a reduced data vector are remarkable.

In order to validate the work all the proposed '*novelties*' have been made accessible in form of a software prototype with an ergonomically designed graphical user interface allowing easy application of the developed methods. The prototype realisation for *Industrial Computer Aided Identification (ICAI)* supplies the new *structured approach to identification techniques for the analysis of industrial processes* from within a block-oriented simulation environment.

11.2 Recommendations

Essentially, the main issues of the requirements analysis have been addressed within this work. Therefore the realised software prototype *ICAI* can be directly applied even by inexperienced users, who look for quick and efficient solutions as a basis for controller design. However, the field of system identification for industrial users is still wide and worth to discover. Its unused potential in industry is still enormous. Particularly, it is necessary to intensify research in the field of identification for blockoriented process models, which is extremely useful for application in process industry.

This final section points the way to possible extensions of this work and its use for future developments:

- *Improved support of nonlinear multi-variable processes.* The support for the identification of nonlinear multi-variable processes has not been implemented completely. Following the *SNIP* it is possible to easily implement new *ICAI* blocks that support the identification of multivariable Wiener- and Hammerstein- models comparable to the nonlinear dynamic *ICAI* ID block. This means that the nonlinear static MIMO ID block and the linear dynamic MIMO ID block should be interfaced.
- *Ease of use.* The prototype software can still be improved with respect to the aim of being self-explanatory. For example, the utilisation of tooltips and the display of images on pushbuttons offer possibilities to better assist the user. ICAI provides a library for (intermediate) identification results. However, it is not possible to automatically compare the gathered results. Therefore a data base support would be beneficial that cares for the data history and an automatic comparison of the identification results.
- *Completion of an ICACSD system.* This work presents a substantial part of the collaborative research project between the University of Glamorgan and the Fachhochschule Hannover. Overall, the collaborative project aims at making the ICACSD approach accessible for engineers with little or no experience in control engineering in order to replace inefficient control strategies thus leading to a better use of resources. Therefore it is necessary to interface the ICAI toolbox with other modules in order to provide an integrated CACSD system. Besides the control design task there are some general concepts that should be addressed:
  - *Interface drivers to industrial controllers.* Once a control design strategy has been elaborated it is necessary to transfer the data into a suitable controller. In order to support this procedure interface drivers to industrial controllers are necessary that allow to download the data.
  - *Integration into process control systems.* The integration of an ICACSD tool within a process control system would extremely simplify the application of ICACSD because the interconnections to the process are already provided and can be directly used for experiments. Additionally the final control strategy can be directly implemented and tested, which is valuable to speed up the commissioning phase.

References

Al-Duwaish, H. and M. Nazmul Karim (1997). A new method for the identification of Hammerstein-model, *Automatica*, vol. 33, no. 10, pp. 1871-1875

Anderson, E. *et al.* (1995). LAPACK users' guide (second edition). *Society for industrial and applied mathematics*, Philadelphia

Andersson, M. (1989). An object-oriented language for model representation, *IEEE Workshop on CACSD*, Danish Automation Society, pp. 8-15

Andersson, M., S.E. Mattsson and B. Nielsson (1991). On the architecture of CACE environments, *IFAC Computer Aided Design in Control Systems*, pp. 41-46

Åström, K. J. *et al.* (1993). Automatic tuning and adaption for PID controllers, A survey, *Control Engineering Practice*, vol. 1, no. 4, pp. 699-714

Atkinson P. and L.I. Williams (1978). Correspondence: Identification of a class of nonlinear systems using correlation analysis, *IEE Proceedings - D*, vol. 125, no. 11, pp. 1307-1308

Ayoubi, M. (1998). Comparison between the dynamic multi-layered perceptron and the generalised Hammerstein-model for experimental identification of the loading process in diesel engines, *Control Engineering Practice* , vol. 6, no. 2, pp. 271-279

Bai, E.-W. (1998). An optimal two stage identification algorithm for Hammerstein-Wiener nonlinear systems, *Automatica* , vol. 34, no. 3, pp. 333-338

Barker, H. A. and C. Y. Linn (1979). A process control system based on a graphical language,*Trends in Online Control Systems,* IEE Proceedings, London, vol. 126, pp. 45-48

Barker, H.A., M. Chen, P.W. Grant, C.P. Jobling, A. Simon and P. Townsend (1988). The manipulation of graphical and symbolic models of dynamic systems, *IFAC-Conference on Analysis, Design and Evaluation of Man-Machine Systems*, no. 3, pp. 301-305

Barker, H.A., M. Chen, P.W. Grant, C.P. Jobling and P. Townsend (1989). A man-machine interface for computer-aided design and simulation of control systems, *Automatica*, vol. 25, no. 2, pp. 311-316

Barker, H. A., M. Chen, P.W. Grant, I.T. Harvay, C.P. Jobling and P. Townsend (1990). The impact of recent developments in user-interface specification, design and management on the computer-aided design of control systems, *11th IFAC World Congress*, Tallinn, Estonia, vol. 5, pp. 393-398

Barker, H. A. (Ed.) (1991). *5th IFAC Symposium on Computer Aided Design in Control Systems*, Beijing, PRC, Swansea, UK

Barker, H. A., M. Chen, P.W. Grant, C.P. Jobling and P. Townsend (1993). Open architecture for computer-aided control engineering, *IEEE Control Systems Magazine*, vol. 13, no. 2, pp. 17-27

Barker, H. A., P.W. Grant, L.T. Harvay, C.P. Jobling and P. Townsend (1996). An approach to project management in computer aided control engineering, *Control Engineering Practice*, vol. 4, no. 4, pp. 441-453

Benveniste, A. and K.J. Åström (1993). Meeting the challenge of computer science in industrial applications of control: An introductory discussion to the special issue, *Automatica*, vol. 29, no. 5, pp. 1169-1175

Bergeon, B., J.L. Ermine, A. Khaddad and M. Monsion (1991). Generic expert system for identification of industrial process, *9th IFAC Symposium on Identification and System Parameter Estimation*, Budapest, pp. 635-640

Betta, A. and D.A. Linkens (1990). Intelligent knowledge-based system for dynamic system identification, *IEE Proceedings – D*, vol. 137, no.1, pp. 1-12

Bieling, H.-H. (1997). Leitfaden zur Erstellung eines optimierten PLT-Konzeptes, *atp - Automatisierungstechnische Praxis*, vol. 39, no. 6, pp. 78-80

Billings, S.A. and S.Y. Fakhouri (1977). Identification of nonlinear systems using the Wiener-model, *Electronic Letters*, vol. 13, no. 17, pp. 502-504

Billings, S.A. and S.Y. Fakhouri (1978). Identification of a class of nonlinear systems using correlation analysis, *IEE Proceedings - D*, vol. 125, no. 7, pp. 691-697

Billings, S.A. (1980). Identification of nonlinear systems - a survey, *IEE Proceedings - D*, vol. 127, no. 6, pp. 272-285

Boutayeb, M., M. Darouach, H. Rafaralahy and G. Krzakala (1993). A new technique for identification of MISO Hammerstein-model, *American Control Conference*, San Francisco, vol. 2, pp. 1991-1992

Boutayeb, M. and M. Darouach (1995). Recursive identification method for MISO Wiener-Hammerstein-model, *IEEE Transactions on Automatic Control*, vol. 40, no. 2, pp. 287-291

Bowyer, R. and D. Clarke (1996). Estimation of multiple order models, *International Conference on Identification in Engineering Systems*, Swansea, pp. 205-213

Bretthauer, G., H.-M. Hanisch, R. Isermann, H. Kiendl, G. Schmidt and H.-H. Wilfert (1998). Eine gute Theorie findet ihre Bestätigung in der Praxis, *atp - Automatisierungstechnische Praxis*, vol. 40, no. 6, pp. 65-77

Bronstein, I. N., K.A. Semendjajew, G. Musiol and H. Mühlig (1995). *Taschenbuch der Mathematik*, Verlag Harri Deutsch, 3-8171-2001-X

Brownlie, I., R. Herzog and R. Simon (1997). Automatisierung mit Verteilten Systemen, *atp - Automatisierungstechnische Praxis*, vol. 39, no. 6, pp. 46-55

Brucker, A., T. Hauff and T. Tauchnitz (1997). ACHEMA 97: Prozeßleitsysteme, *atp - Automatisierungstechnische Praxis*, no. 11, pp. 13-32

Cellier, F.E. (1991). *Continuous system modeling*, Springer-Verlag,0-387-97502-0,

Chang, F.H.I. and R. Luus (1971). A noniterative method for identification using Hammerstein-model, *IEEE Transactions on Automation and Control,* vol. 16, pp. 464-468

Charwat, H. J. (1996). Farbkonzept für die Prozeßführung mit Bildschirmen, *atp - Automatisierungstechnische Praxis*, R. Oldenbourg Verlag, vol. 38, no. 5, pp. 50-53

Chen, M. (1991). *Graphical man-machine interface to CACSD,* PhD Thesis University of Wales, Swansea

Chen, H.-W. (1995). Modeling and identification of parallel nonlinear systems: structural classification and parameter estimation methods, *IEEE Proceedings*, vol. 83, no. 1, pp. 39-66

De Moor, B. and P. Van Overschee (1994). *Graphical user interface software for system identification*, ESAT-SISTA TR 1994/06, Dept. of EE K.U, Leuven

Diekmann, K. (1983). Identifikation von Mehrgrößensystemen; Teil 2: Parameterschätzverfahren und spezielle Probleme der Mehrgrößenidentifikation, *Regelungstechnik*, no. 12, pp. 398-402

Dongarra, J. J., J. R. Bunch, C. B. Moler and G. W. Stewart (1979). *LINPACK user's guide*, Society for industrial and applied mathematics, Philadelphia

Elmqvist, H. (1994). *DYMOLA user manual,* DynasimAB

Elmqvist, H. and S.E. Mattsson (1989). Simulator for dynamical systems using graphics and equations for modeling, *IEEE Control Systems Magazine*, pp.53-58

Elmqvist, H., S.E. Mattsson and M. Otter (1998). Modelica - An international effort to design an object-oriented modeling language, *Summer Computer Simulation Conference -98* , Reno, Nevada

Eskinat, E., S.H. Johnson and W.L. Luyben (1991). Use of Hammerstein-models in identification of nonlinear systems, *AIChE Journal,* vol. 37, pp. 255-268

Eykhoff, P. (1974). *System identification,* Wiley, London, ISBN 0-471-24980-7

Eykhoff, P. and P.C. Parks (1990). Identification and system parameter estimation; where do we stand now?, *Automatica*, vol. 26, no. 1, pp. 3-5

Flaton, K.A., S. Gehlen, M. Hormel, W. Konen, J. Kopecz (1995). Neuronale Netze: Grundlagen und Anwendungen, *in H.-J. Zimmermann (ed.) 'Neuro +Fuzzy '*, VDI Verlag, pp. 26-58

Foss, B.A., B. Lohmann and W. Marquardt (1998). A field study of the industrial modeling process, *Journal of Process Control*, vol. 8, no. 5/6, pp. 325-338

Frederick, D.K., C.J. Herget, R. Kool and M. Rimvall (1992). ELCS - The extended list of control software, *Dept. of Automatic Control*, Swiss Federal Institute of Technology (ETH), Mar., no. 5

Froese, T. (1995). Übergeordnete Regelung chemischer Verfahren mit Fuzzy Control und Künstlichen Neuronalen Netzen, *in H.-J. Zimmermann (ed.) 'Neuro +Fuzzy*, VDI-Verlag, pp. 102-118

Funk, G.P.E. (1994). Applying online computer-aided control system design to the manufacturing industry enhances control performance, *IEEE / IFAC Symposium on CADCS*, pp.429-434

Garbow, B.S., J.M. Boyle, J. J. Dongarra and C. B. Moler (1977). *Matrix Eigensystem Routines - EISPACK Guide Extensions,* Springer-Verlag, New York,

Gauss, K.F. (1809). *Theoria motus corporum celestrium in sectionibus conicus solem ambientium*, English Translation: Theory of the motion of heavenly bodies moving around the sun in conic sections, Dover, New York

Gawthrop, P.J. and J.W. Ponton (1996). Improved control using dynamic process models, *TransIChemE*, vol. 71, no. A1, pp. 63-70

Gentil, S., A.Y. Barraud and K. Szafnicki (1990). SEXI: An expert identification package, *Automatica*, vol. 26, no. 4, pp. 803-809

Gevers, M. (1993). Towards a joint design of identification and control, *Essays on Control: Perspectives in the Theory and its Applications*, Trentelman and Willems (ed.), Groningen, pp. 111-151

Godfrey, K. (Ed.) (1993). *Perturbation signals for system identification*, Prentice Hall, New York, ISBN 0-13-656414-3

Goldberg, S. and A. Durling (1971). A computational algorithm for the identification of nonlinear systems, *Journal of the Franklin Institute*, pp. 427-447

Grams, T. (1998). Bedienfehler und ihre Ursachen (Teil 1), *atp - Automatisierungstechnische Praxis*, vol. 40, no. 3, pp. 53-56

Grant, P.W., C.P. Jobling and C. Rezvani (1991). PRODYNSA - A package of tools for linear dynamic systems analysis written in PROLOG, *Control 91 - International Conference Publication*, The Institution of Electrical Engineers, vol. 1, no. 332, pp. 294-299

Gray, J. and Mattson, S.E. (Ed.) (1997). *7th IFAC Symposium on Computer Aided Control System Design*, Gent, Belgium, ISBN 0-08-0423833

Grübel, G. (1995). Regelungstechnische Entwurfsautomation, *at - Automatisierungstechnik*, vol. 43, no. 1, pp. 3-13

Grübel, G. and C.P. Jobling (1994). Simulation tool-abstractions based on the neutral dynamics-model description format DSblock, *IEE International Conference on Control*, pp. 1-6

Haber, R. and L. Keviczky (1978). Identification of nonlinear dynamic systems - a survey paper, *4th IFAC Symposium on Identification and System Parameter Estimation*, Tbilisi, pp. 79-126

Haest, M., G. Bastin, M. Gevers and V. Wertz (1990). ESPION: an expert system for system identification, *Automatica*, vol. 26, no. 1, pp. 85-95

Hahn, V. and G. Nöth (1997). Anforderungen an regelungstechnisches CAE in der chemischen Verfahrenstechnik, *VDI/VDE-GMA Jahrbuch*, VDI-Verlag, Düsseldorf, pp. 260-280.

Hankel, M. and A. Reiner (1992). Approximation der Sprungantwort von linearen aperiodischen Strecken, at - Automatisierungstechnik, vol. 40, no. 7, pp. 263-266

Heiss, M. (1995). Kennfelder in der Regelungstechnik, *at - Automatisierungstechnik*, vol. 43, no. 8, pp. 363-367

Helms, M. and H. Knoll (1995). *Rechnergestützte Aufnahme von Kennlinien bzw. Kennlinienfeldern*, Diplomarbeit, FH Hannover

Hensel, H. (1987). *Methoden des rechnergestützten Entwurfs und Echtzeitansatzes zeitdiskreter Mehrgrößenregelungen und ihre Realisierung in einem CAD-System*, Doctorate Dissertation, University of Darmstadt, VDI--Verlag, no. 4

Hoffmann, G. and M. Rimvall (1988). Knowledge representation in computer-aided control systems design packages, *Proc of the Euro. Simulation Multi-Conf.*, pp. 223-227

Hortsch, A. and K. Schlüter (1996). Matlab statt C und Fortran - Rechenkünstler, *mc extra*, pp. 36-41

Hvelplund, H. (1986). The ESPRIT Eurohelp project: an intelligent help system, *Proceedings of the International Conference on Knowledge Based Systems*, pp. 61-70

Isermann, R. (1973). Test-cases for evaluation and comparison of different identification and parameter-estimation methods using simulated processes, 3rd IFAC Symp. on Identification and Parameter Estimation, The Hague, Delft, pp. 1101-1102

Isermann, R., U. Baur and H. Kurz (1974). Identifikation dynamischer Prozesse mittels Korrelation und Parameterschätzung, *Regelungstechnik und Prozeßdatenverarbeitung*, vol. 22, no. 8, pp. 235-242

Isermann, R. (1988). *Digitale Regelsysteme*, Springer-Verlag, 3-540-16596-7

Isermann, R. (1991). *Digital Control Systems*, Vol.II, Springer-Verlag, 3-540-50997-6

Isermann, R. (1992). *Identifikation dynamischer Systeme 1+2*, Springer-Verlag, 2nd, ISBN-3-540-54924

Ivakhnenko, A.G. (1968). The group method of data handling - a rival of the method of stochastic approximation, *Soviet Automatic Control*, vol. 3, no. 13, pp. 43-55

James, G. (1992). *Modern engineering mathematics*, Addison-Wesley, 0-201-18054-5

Jang, J.-S. R. and C.-T. Sun (1993). Functional equivalence between radial basis function networks and fuzzy inference systems, *IEEE Trans. on Neural Networks*, pp. 156-159

Jategaonkar, R.V. (1985). Parametric identification of discontinuous nonlinearities, *IFAC Identification and System Parameter Estimation*, pp. 167-172

Jobling, C.P. (1991). Building better graphical user interfaces for CACSD - the case for object-oriented programming, *IFAC Computer Aided Design in Control Systems*, pp. 147-151

Jobling, C.P., P.W. Grant, H.A. Barker and P. Townsend (1994). Object-oriented programming in control system design: a survey, *Automatica*, vol. 30, no. 8, pp. 1221-1261

Johansen, T.A. and B.A. Foss (1997). Towards grey box modelling and identification - operating regimes and multiple models, *Eurosim - Simulation News in Europe*, no. 21, pp. 7-10

Kahlert, J. (1995a). *Fuzzy Control für Ingenieure*, Vieweg Verlag, 3-528-05460-3

Kahlert, J. (1995b). Globale vektorielle Optimierung mit Evolutionsstrategien, *at - Automatisierungstechnik*, vol.43, no. 3, pp. 118-126

Kalman, R.E. and T.S.A. Englar (1966). *User's Manual for the Automatic Synthesis Program*, NBS #: 70400375

Kiendl, T. *et al.* (1998). *DORA -Dortmunder Regelungstechnische Anwenderprogramme*, version 6.0, manual: http://esr.e-technik.uni-dortmund.de/esr/WinDORA/Handbuch/Startseite.htm

King, R.A. and J.O. Gray (1986). A methodology for the design and implementation of graphical man-machine interfaces, *Knowledge engineering and computer modelling in CAD*, Butterworths, pp. 419-25

Körner, S. and R. Schumann (1996a). Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE, *VDI/VDE- GMA Kongreß, Meß- und Automatisierungstechnik*, Baden-Baden, VDI-Bericht Nr. 1282, pp. 93-103

Körner, S. and R. Schumann (1996b). CAE hilft Prozessen schneller auf die Sprünge, *Chemische Industrie*, no. 9, pp. 38-39

Körner, S., R. Schumann and K.J. Baker (1996). New approach to identification for block-oriented simulation environments, *8th European Simulation Symposium*, Vol.2, Genua, pp. 245-246

Körner, S. (1997). *ICAI* – Industrial Computer Aided Identification – A new MATLAB toolbox for industrial users, Online, Internet, *http://www.repam.fh-hannover.de/automa/ICAIHelp/Intro.html*

Körner, S., R. Schumann and K.J. Baker (1997). *ICAI* - A MATLAB toolbox for Industrial Computer Aided Identification, *11th IFAC Symposium on System Identification*, SYSID, Kitakyushu, Japan, pp. 1611-1616

Körner, S. (1998). Industrietaugliche Identifikation mit *ICAI*, *GMA Kongreß'98 Meß- und Automatisierungstechnik*, Ludwigsburg, VDI Berichte 1397, pp. 119-128

Körner, S. and R. Schumann (1998a). Revisiting step response identification, *17th IASTED international conference on Modelling, Identification and Control*, Grindelwald, Switzerland, pp. 423-426

Körner, S. and R. Schumann (1998b). Robustified estimation of multiple order models, *17th IASTED international conference on Modelling, Identification and Control*, Grindelwald, Switzerland, pp. 427-429

Körner, S. and R. Schumann (1998c). A New Framework for Computer Aided Identification in Process Industry, *17th IASTED international conference on Modelling, Identification and Control*, pp. 437-439

Kortmann, M. (1988). *Die Identifikation nichtlinearer Ein- und Mehrgrößensysteme auf der Basis nichtlinearer Modellansätze*, Doctorate Dissertation, University of Bochum, VDI-Verlag, no.177

Kortmann, M. and H. Unbehauen (1988). Structure detection in the identification of nonlinear systems, *APII Automatique productique informatique industrielle*, vol. 22, no. 1, pp. 5-25

Kuhn, U. (1993). INTERKAMA 92: Prozeßregler, *atp - Automatisierungstechnische Praxis*, R. Oldenbourg Verlag, vol. 35, no. 2, pp. 98-104

Kurth, J. (1995). *Identifikation nichtlinearer Systeme mit komprimierten Volterrra-Reihen*, Doctorate Dissertation, University of Aachen, VDI--Verlag

Lachmann, K.H (1983). *Parameteradaptive Regelalgorithmen für bestimmte Klassen nichtlinearer Prozesse mit eindeutigen Nichtlinearitäten*, Doctorate Dissertation, University of Darmstadt, VDI-Verlag

Lechner, W. and N. Lohl (1990). *Analyse digitaler Signale*, Vieweg Verlag, 3-528-04727-5

Leontaritis, I.J. and S.A. Billings (1985). Input-output parametric models for nonlinear systems, *International Journal of Control*, vol. 41, pp. 303-344

Lewin, M.(1985). *Psychologische Forschung im Umriß*, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo

Li, P.H.M. and J.O. Gray (1993). Graphical system input for design, *CAD for Control Systems*, in D.A. Linkiens (ed.), Marcel Decker, NY, pp.531-559

Linkens, D.A. (Ed.) (1993). *CAD for Control Systems*, Marcel Dekker, New York

Ljung, L. (1987). *System identification; Theory for the user*, Prentice Hall, Englewool Cliff, NJ, ISBN 0-13-881640-9

Ljung, L. (1991). Issues on system identification, *IEEE Control Systems Magazine*, pp. 25-32

Ljung, L. (1995a). *System identification toolbox*, The MathWorks Inc.

Ljung, L. (1995b). *From data to model: a guided tour*, The MathWorks Inc.

MacFarlane, A.G.J., G. Grübel and J. Ackermann (1989). Future design environments for control engineering, *Automatica*, vol. 25, no. 2, pp. 165-176

Maffezzoni, C. and P.Rocco (1995). Robust tuning of PID regulators based on step-response identification, *European Control Conference*, Roma, pp. 2477-2482

Marenbach, P. and K.D. Bettenhausen (1995). Selbstorganisierende Generierung strukturierter Prozeßmodelle, *at - Automatisierungstechnik*, vol. 43, no. 6, pp. 277-288

Mattsson, S. E., M. Andersson and K.J. Aström (1993). Object-oriented modeling and simulation, *CAD for Control Systems*, Linkens, D.A. (ed.), Marcel Dekker, NY pp. 31-69

McLain, D. H. (1974). Drawing contours from arbitrary data points, *The Computer Journal*, vol. 17, pp. 318-324

Meier zu Farwig, H. and H. Unbehauen (1991). Knowledge-based system identification, *IFAC Symposium on Identification and System Parameter Estimation*, pp. 21-29

Meier zu Farwig, H. (1992). *Ein Beratungssystem zur Systemidentifikation*, Doctorate Dissertation, University of Bochum, VDI--Verlag

Moellenhoff, M. S. and K.T. Erickson (1992). A comparison of ARMA and unit pulse response identification of step response models, *American Control Conference*, Chicago, vol.2, pp. 1443-1444

Munro, N. and C.P. Jobling (1993). Ecstasy: A control system CAD environment, *CAD for Control Systems*, Linkens, D.A. (ed.), Marcel Dekker, NY, pp. 449-467

Nagy, P.A. and L. Ljung (1989). An intelligent tool for system identification, *IEEE Workshop on CACSD*, Danish Automation Society, pp. 58-63

Narendra, K.S. and P.G. Gallmann (1966). An iterative method for the identification of nonlinear systems using a Hammerstein-model, *IEEE Transactions on Automation and Control*, vol. 11, pp. 546-550

News Server (1998). *Das Magazin von Scientific Computers*, Ausgabe 2/98, p. 17

Niederlinski, A., J. Figwer and J. Kasprzyk (1991). EFPI - An integrated intelligent environment for system and signal identification, 9th IFAC / IFORS Symposium, Budapest, pp. 1203-1208

Niederlinski, A., J.Figwer and J. Kasprzyk (1994). Intelligent Identification: Decision trees or knowledge bases?, *IFAC Symposium on System Identification*, Danish Automation Society, vol. 2, pp. 471-476

Niu, S. (1994). *Augmented UD identification for process control*, PhD Thesis, University of Aberta, Canada

Niu, S. and D.G. Fisher (1994). A multiple model least-squares estimation method, *American Control Conference*, Baltimore, pp. 2231-2235

Niu, S., D. G. Fisher, L. Ljung and L.S. Sirish (1995). *Multiple model least squares and augmented UD identification - an overview*, draft paper available via ftp://prancer.eche.ualberta.ca/pub/outgoing/audi,Toronto, pp.1-17

Niu, S., L. Ljung and Å. Björck (1996). Decomposition methods for solving least-squares parameter estimation, *IEEE Transactions on Signal Processing*, vol. 44, no. 11, pp. 2847-2852

Nørgaard, M., N. Poulsen and O. Ravn (1997). NNSYSID & NNCTRL - Matlab tools for system identification and control with neural networks, *IFAC Symposium on System Identification*, Fukuoka, Japan

Norquay, S.J., A.N. Palazoglu and J.A. Ramagnoli (1996). Nonlinear model predictive control of pH neutralisation using Wiener-models, *13th IFAC World Congress*, San Francisco, vol. M, pp. 31-36

Nöth, G. and U. Keuchel (1996). Richtlinienentwurf Inbetriebnahmesysteme für Regelungen, *VDI/VDE- GMA Kongreß, Meß- und Automatisierungstechnik*, Baden-Baden, VDI-Bericht Nr. 1282, pp. 103-111

Nöth, G. (1998). Randbedingungen für den Einsatz regelungstechnischer Methoden, *GMA-Kongress'98, Mess- und Automatisierungstechnik*, VDI Berichte Nr. 1397, pp. 87-94

Otter, M. and P. Mostermann (1998). *The DSblock model interface*, version 4.0 - Incomplete Draft, http://www.Dynasim.se/Modelica/DSblock/index.html

Pajunen, G. (1982). Application of a model reference adaptive to the identification of simple Wiener type nonlinear systems, *Workshop on Adaptive Control*, Florence, pp. 311-332

Percoco, G. and L. Sarti (1996). On the usage of graphics in *ICAI* systems, *European Conference on Artificial Intelligence in Education*, Lisbon, Portugal, pp. 312-318

Perne, R. (1995). Erfahrungen mit einem regelungstechnisch orientierten Simulationswerkzeuges, in Schuler, H. (ed.), *Prozeßsimulation* , VCH Verlag, pp.291-348

Pfeiffer, B.-M., K.-H. Kirchberg and S. Bergold (1998). MATLAB als offene Systemplattform für 'Advanced Control' im Siemens Prozeßleitsystem PCS7, *GMA-Kongress'98 Mess- und Automatisierungstechnik*, VDI Berichte Nr. 1397, pp. 95-102

Piwinger, F. (1975). *Regelungstechnik für Praktiker*, VDI-Verlag, Düsseldorf, 3-18-400302-7

Polzer, K. (1998). Einheitliches Bedienen, Beobachten und Parametrieren von Prozeßgeräten, *atp - Automatisierungstechnische Praxis*, vol. 40, no. 3, pp. 57-64

Preuß, H.-P. (1994). Methoden der nichtlinearen Modellierung - vom Interpolationspolynom zum neuronalen Netz, *at - Automatisierungstechnik*, vol. 42, no. 10, pp. 449-457

Rake, H. and M. Enning (1996). INTERKAMA 95: Verfahren der Regelungstechnik, *atp - Automatisierungstechnische Praxis* , R. Oldenbourg Verlag, vol. 38, no. 3, pp. 53-57

Ravn, O. and M. Szymkat (1997). Modelling and calibration with mechatronic blockset for SIMULINK, *11th IFAC Symposium on System Identification*, SYSID, Kitakyushu, Japan, vol. 3, pp. 1643-1648

Reinsch, C. (1967). Smoothing by spline functions, *Numerical Mathematic*, vol. 10, pp. 177-183

Reuter, M. (1991). *Regelungstechnik für Ingenieure*, Vieweg, ISBN 3-528-74004-9

Richalet, J. (1993). Industrial applications of model based predictive control, *Automatica*, Pergamon Press Ltd, vol. 29, no. 5, pp. 1251-1274

Rimvall, M. (1988). Interactive environments for CACSD software, *4th IFAC Symposium on Computer Aided Design of Control Systems*, Beijing, pp. 17-26

Rimvall, M. and M. Kündig (1990). Intelligent help for CACE applications, *11th IFAC World Congress*, Tallinn, Estonia, vol. 5, pp. 367-372

Rode, M. and K. Krüger (1998). Toolunterstützte Inbetriebnahme von Regelungen in Kraftwerken, *GMA-Kongress'98, Mess- und Automatisierungstechnik*, VDI Berichte Nr. 1397, pp. 67-86

Rosenthal, B. (1985). *Ein Beitrag zur Identifikation einfacher nichtlinearer dynamischer Systeme in offener und geschlossener Kette*, Doctorate Dissertation, University of Ilmenau

Schmid, C. (1993). Verfügbare Softwaretools, *VDI/GMA-Workshop'1993 Regelungstechnische Programmpakete' ('Control Engineering Programs')*, Chair: Prof. Dr.-Ing. R. Schumann, Düsseldorf

Schmid, C. and R. Schumann (1994). Standardisierte GMA-Datenschnittstelle für regelungstechnische CAE-Werkzeuge, *atp - Automatisierungstechnische Praxis*, R. Oldenbourg Verlag, vol. 36, no. 4, pp. 55-56

Schmoll, J. (1997). Wird OLE for Process Control (OPC) ein neuer Industriestandard?, *atp - Automatisierungstechnische Praxis*, vol. 39, no. 5, pp. 11-17

Schuler, H. (1992). Was behindert den praktischen Einsatz moderner regelungstechnischer Methoden in der Prozeßindustrie?, *atp - Automatisierungstechnische Praxis* , R. Oldenbourg Verlag, vol. 34, no. 3, pp. 116-123

Schumann, R. (1989). CAE von Regelsystemen mit IBM-kompatiblen Personal Computern, *atp - Automatisierungstechnische Praxis*, R. Oldenbourg Verlag, vol. 31, no. 8, pp. 349-359

Schumann, R. (1994). CAE von Regelsystemen, *atp - Automatisierungstechnische Praxis*, R. Oldenbourg Verlag, vol. 36, no. 3, pp. 51-59

Schumann, R., S. Körner, K.J. Baker and M. Strickrodt (1996). Shaping CACSD for practical use in process industry, *IFAC World Congress*, San Francisco, vol. L, pp. 223-228

Sjöberg, J, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.Y. Glorennec, H. Hjalmarsson and A. Juditsky (1995). Nonlinear black-box modeling in system identification: A unified overview, *Automatica*, pp. 1691-1724

Söderström, T. and P. Stoica (1989). *System identification*, Prentice Hall, ISBN-0-13-881236-5

Sørlie, J.A. (1994). *On the interfacing of software for the identification of grey box models*, Automatic Control Department of Signals, Sensors and Systems - Royal Institute of Technology, Stockholm

Stoica, P. and T. Söderström (1982). Instrumental-variable methods for identification of Hammerstein systems, *International Journal of Control*, vol. 35, no. 3, pp. 459-476

Strickrodt, M., K.J. Baker and R. Schumann (1996). An integrated knowledge engineering approach to process modelling for CACSD, *13th IFAC World Congress,* San Francisco, vol. L, pp. 229-234

Strickrodt, M. (1997). *An integrated knowledge engineering approach to process modelling*, PhD Thesis University of Glamorgan, Pontypridd

Syska, B., S. Körner and R. Schumann (1999). User friendly CACSD for complex industrial processes, *14th IFAC World Congress,* Beijing, pp. 409-414

Takatsu H., T. Itoh and M. Araki (1998). Future needs for the control theory in industries - report and topics of the control technology survey in Japanese industry, *Journal of Process Control*, vol. 8, no. 5/6, pp. 369-374

Taylor, J.H. and M. Rimvall, H.A. Sutherland (1991). Future developments in modern environments for CADCS, *IFAC Computer Aided Design in Control Systems*, Swansea, UK, pp. 29-40

Taylor, J. *et al.* (1994). *Panel discussion on Open CACSD*, CACSD 94, CACSD newsletter, Online, Internet, *http://faith.swan.ac.uk/chris.html/OpenCACSD/archive/Docs/CACSD94-Panel.htm*

The MathWorks Inc. (1993). *MATLAB user's guide*

The MathWorks Inc. (1996). *Building GUI's with MATLAB*

Uecker, F. and U. Epple (1997). Die Bedeutung der Anlagen- und Prozeßüberwachung für die Prozeßführung, *atp - Automatisierungstechnische Praxis* , vol. 39, no. 6, pp. 56-65

Varsamidis, T., S. Hope and C.P. Jobling (1996). An object-oriented information model for computer-aided control engineering, *Control Engineering Practice*, vol. 4, no. 7, pp. 929-937

Wernstedt, J. (1989). *Experimentelle Prozeßanalyse*, R. Oldenbourg Verlag, 3-486-20869-1,

Wernstedt, J., P. Otto and F. Ross (1991). DIOPRAN-EXPERT - A consulting/expert system for experimental process analysis, *9th IFAC Symposium on Identification and System Parameter Estimation*, Budapest, pp. 641-645

Winter, M., R. Schockenhoff, H. Pu and H.A. Nour Eldin (1994). BLOCKSIM - application of nonlinear modelling and simulation in fuzzy control system design, *3rd IEEE Conference on Control Applications*, vol. 2, pp. 1337-1342

Wozny, G. and L. Jeromin (1991). Dynamische Prozeßsimulation in der industriellen Praxis, Chem.-Ing.-Tech., *VCH Verlagsgesellschaft*, vol. 63, no. 4, pp. 313-326

Zadeh, L. A. (1965). Fuzzy sets, *Info and Control*, no. 8, pp. 338-353

Zhen-Yu, C. (Ed.) (1988). *4th IFAC Symposium on Computer Aided Design in Control Systems*, Beijing, PRC

Zhu, Y. (1998). Multi-variable process identification for MPC: The asymptotic method and its applications, *Journal of Process Control*, vol. 8, no. 2, pp. 101-115

Zhu, Y.Z. and A.C.M.P. Backx (1993). *Identification of multi-variable industrial processes: For simulation, diagnosis and control*, Springer-Verlag, London

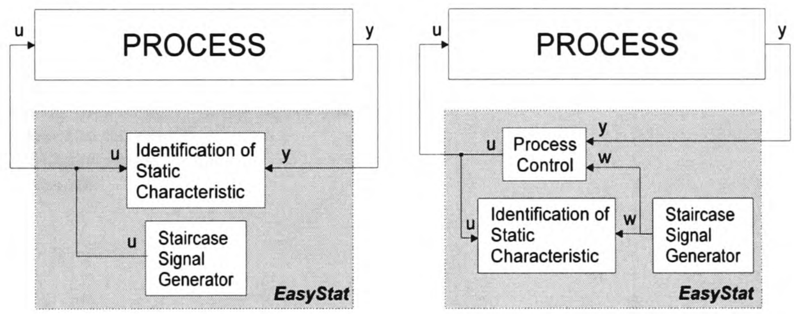
Zimmermann, H.-J. (1993). *Fuzzy Technologien*, VDI Verlag, ISBN 3-18-401269-7

Zi-Quiang, L. (1993). Controller design oriented model identification method for Hammerstein system, *Automatica*, vol. 29, no. 3, pp. 767-771

Appendix A: *EasyStat* – A Tool for the Evaluation of Static Process Characteristics

EasyStat was designed to simplify the possibly time-consuming task of measuring static characteristics or fields of static characteristics in open or closed loop. It is coded for personal computers with an AD/DA interface card (*Datalog PCL812PG*) and is currently restricted to processes with a maximum of *2* inputs and *2* outputs. *EasyStat* is programmed in object-oriented Borland C++ for MSWindows. Figure 1 shows the identification properties of *EasyStat*. Automatically designed uncoupled PID controllers can be used for the identification in closed loop to speed up the procedure (only if the controlled process stays stable). Then the signal generator controls the set point *w*, whereas in the open loop mode it controls the process input *u* directly. The output *y* is recorded automatically when a steady state is reached.

For the identification of multi-dimensional static characteristics inputs according to Figure 2 are applied. The range of the input signal depends on the limits of the plant. The length of plateaux of the signal (here: *w1*) depends on the saturation time, which can vary within the experiment. Data interfaces to STATISTICA and MATLAB were programmed. Therefore *ICAI* can also utilise the results of *EasyStat.* For example, Figure 9-13 shows the approximation of data points gained by *EasyStat* from the laboratory plant utilising the MGWM method.



Identification of Static Characteristic in Open Loop

Identification of Static Characteristic in Closed Loop

Figure 1.Identification Properties of EasyStat

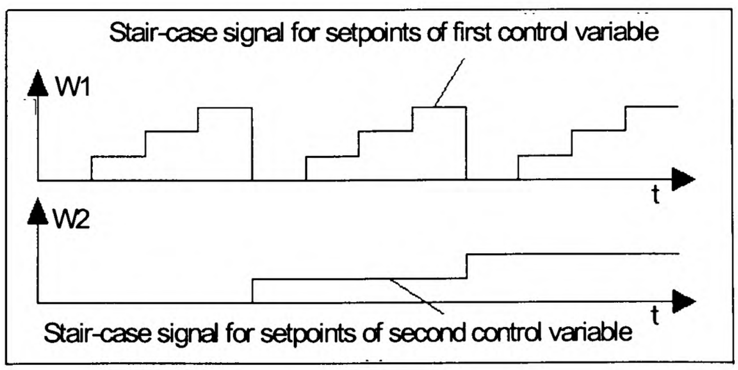


Figure 2. Input signals for the evaluation of a two-dimensional static characteristic

Further a *hypertext help system* has been implemented according to windows standard. Experiments at the laboratory plants described in Chapter 9.5.1 showed the applicability of *EasyStat*.

Appendix B: The Structure of Session User Data (SeUd)

In the following the structure of those data is shown that are generated during an ICAI identification session (see Chapter 9.3.1).

SeUd

d

used for more than one block

Path

Path of Block

SysName

Name of Simulink System

CurrentWork

Used to exchange data between program modules / delete before use !!
Is also used as default from one block to the next block
CurrentWork{1:8} reserved for I/Ospec (saved in Apriori)
CurrentWork{9}. Apriori (of LinSisoBlk)

Setting

i.e. Project Settings

User

Personal/ AreaEng/ Expert

IdentFor

Control/ Simulation

Control

PID/ IMC/ State Space/ Adaptive/ Simulation (although no control)

Domain

depends on process library loaded

Type

depends on process library loaded

Path

Path of current system

SysName

Name of current system

Units

Standard units plus units of library with respect to domain!

PathCodeArea

is current pathcode for area engineer

PathCodeExpert

is current pathcode for expert

f

f stands for flag - expressed by 0/1 (no/yes) ...

LinSisoTaskOn

Task is on (1) if a LinSiso block is currently utilised

StatSisoTaskOn

Task is on (1) if a StatSiso block is currently utilised

SitbSisoTaskOn
Task is on (1) if a SitbSiso block is currently utilised
NlSisoTaskOn
Task is on (1) if a NlSiso block is currently utilised
LinMisoTaskOn
Task is on (1) if a LinMiso block is currently utilised
LinMimoTaskOn
Task is on (1) if a LinMimo block is currently utilised
StatMimoTaskOn
Task is on (1) if a StatMimo block is currently utilised
SettingOn

StatSisoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

LinSisoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

SitbSisoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

NlSisoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

LinMisoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

LinMimoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

StatMimoIndex
Index corresponds to tagnumber (e.g. StatSisoBlk4)

StatSiso{1..n} *(up to n blocks can be managed)*
Status
Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (In&Out specification): 0/1
In
Name
Unit
UpLimit *upper limit of input*
LowLimit lower *limit of input*
Out
Name
Unit
UpLimit *upper limit of output*
LowLimit *lower limit of output*

LinSiso{1..n} *(up to n blocks can be managed)*
Status
Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (In&Out specification): 0/1
Status{3}: (Apriori specification): 0/1
In *(not Apriori for programming reasons)*
Name
Unit

UpLimit *upper limit of input*
LowLimit lower *limit of input*

Out

Name
Unit
UpLimit *upper limit of output*
LowLimit *lower limit of output*

Apriori

Operation
'Normal' 'Start-up' 'Shut-down' 'Emergency'
ClosedLoop 0/1
Integral 0/1
Oscill 0/1
Delay *check&allow 20% deviation*
variant or number(then delay is specified) / none=0
Excite
'no excitation' 'Single step' 'Series of steps' 'Impuls' 'RBS'
Disturb
External 0/1 *disturbance from extern*
Measurable 0/1 *can directly be measured*
TestDisturb 0/1 *disturbance can be used for identification*
EffOut 0/1 *disturbance affects output*
EffIn 0/1 *disturbance affects input*
Gain *check&allow 20% deviation*
Settling *check&allow 30% deviation*
MaxActRate *maximum actuator change rate*
TimeUnit
'ms' 's' 'min' 'h' 'd' 'a'

Data

Time *measurment vectors*
Input *measurment vectors*
Output *measurment vectors*
ModiTime *is modified by ModiSiso_i or ModiData_i (offsets, delay removed, etc.)*
ModiInput *is modified*
ModiOutput *is modified*

Struct

OE *output error (vector for orders 1:n)*
Method *CorrMMLS/ MMLS or others*
Order *model order*
Model *linear/ Wiener/ Hammer*

SitbSiso{1..n} *(up to n blocks can be managed)*

Status
Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (File specification): 0/1

File
File of System Identification Toolbox

NlSiso{1..n} *(up to n blocks can be managed)*

Status

Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (File specification): 0/1

StatIndex *is index of StatSisoBlk*

LinIndex *is index of LinSisoBlk*

Structure *Wiener/ Hammer*

LinMiso{1..n} *(up to n blocks can be managed)*

Status

Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (In&Out specification): 0/1
Status{3}: (Apriori specification): 0/1

Data

Time *measurment vectors*

Input *measurment vectors*

Output *measurment vectors*

ModiTime *is modified by ModiSiso_i or ModiData_i (offsets, delay removed)*

ModiInput *is modified*

ModiOutput *is modified*

Struct

OE *output error (vector for orders 1:n)*

Method *CorrMMLS/ MMLS or others*

Order

Model *linear (Wiener/ Hammer- in future versions)*

LinMimo{1..n} *(up to n blocks can be managed)*

Status

Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (File specification): 0/1

LinMisoIndex *is vector of indexes of LinMisoBlk*

StatMimo{1..n} *(up to n blocks can be managed)*

Status

Status{1}: (Blockstatus): empty/done/deleted(not used yet)
Status{2}: (In&Out specification): 0/1

Data

x *measurment vectors*

y *measurment vectors*

z *measurment vector output*

xm *is modified*

ym *is modified*

zm *is modified*

Publications

The below listed publications are appended in timely order after this page for reference.

- Schumann, R., S. Körner, K.J. Baker and M. Strickrodt (1996). Shaping CACSD for practical use in process industry, *IFAC World Congress*, San Francisco, vol. L, pp. 223-228
- Körner, S. and R. Schumann (1996a). Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE, *VDI/VDE- GMA Kongreß, Meß- und Automatisierungstechnik*, Baden-Baden, VDI-Bericht Nr. 1282, pp. 93-103
- Körner, S. and R. Schumann (1996b). CAE hilft Prozessen schneller auf die Sprünge, *Chemische Industrie*, no. 9, pp. 38-39
- Körner, S., R. Schumann and K.J. Baker (1996). New approach to identification for block-oriented simulation environments, *8th European Simulation Symposium*, Vol.2, Genua, pp. 245-246
- Körner, S., R. Schumann and K.J. Baker (1997). *ICAI* - A MATLAB toolbox for Industrial Computer Aided Identification, *11th IFAC Symposium on System Identification*, SYSID, Kitakyushu, Japan, pp. 1611-1616
- Körner, S. and R. Schumann (1998a). Revisiting step response identification, *17th IASTED international conference on Modelling, Identification and Control*, Grindelwald, Switzerland, pp. 423-426
- Körner, S. and R. Schumann (1998b). Robustified estimation of multiple order models, *17th IASTED international conference on Modelling, Identification and Control*, Grindelwald, Switzerland, pp. 427-429
- Körner, S. and R. Schumann (1998c). A New Framework for Computer Aided Identification in Process Industry, *17th IASTED international conference on Modelling, Identification and Control*, pp. 437-439
- Körner, S. (1998). Industrietaugliche Identifikation mit *ICAI*, *GMA Kongreß'98 Meß- und Automatisierungstechnik*, Ludwigsburg, VDI Berichte 1397, pp. 119-128
- Syska, B., S. Körner and R. Schumann (1999). User friendly CACSD for complex industrial processes, *14th IFAC World Congress,* Beijing, pp. 409-414

Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE

St. Körner, R. Schumann, Hannover

Zusammenfassung

Befragungen von industriellen Anwendern und Herstellern von Prozeßleitsystemen (PLS) haben ergeben, daß der bei Planung und Betrieb von PLS praktizierte CAE (Computer Aided Engineering) -Einsatz weder allgemein noch in bezug auf regelungstechnische Aufgabenstellungen die heute bereits denkbaren Möglichkeiten ausschöpft. Es kommen weitgehend isolierte CAE-Insellösungen zum Einsatz. Dieser Beitrag befaßt sich aus regelungstechnischer Sicht mit den potentiellen Möglichkeiten des CAE-Einsatzes bei Planung und Betrieb von Prozeßleitsystemen in der Verfahrenstechnik und beschreibt ein Konzept für ein industrietaugliches, regelungstechnisches CAE-System für die Inbetriebnahme.

Schlüsselworte: Regelungstechnisches CAE, PLS, Inbetriebnahme, CACE

EINLEITUNG

Prozeßleitsysteme sind ein wichtiger Bestandteil der Automatisierungstechnik. Aufgrund ihrer umfassenden Funktionalität tragen sie wesentlich zum sicheren, umweltverträglichen und rationellen Betreiben von verfahrenstechnischen Anlagen bei.

Eine besondere Stellung kommt der Prozeßleittechnik in der Inbetriebnahmephase der Anlage zu, in der sie die Komponenten unterschiedlicher Hersteller erstmalig zu einer Funktionseinheit verbindet. Diese umfangreiche und schwierige Aufgabe muß übergreifend bearbeitet werden, was normalerweise unter großem Zeitdruck geschieht, da zuvor verursachte Verzögerungen in der Inbetriebnahmephase wieder ausgeglichen werden sollen.

Durch diese obligatorische Zeitverknappung ist es oftmals nicht möglich, den Prozeß optimal einzufahren und gezielt neue Erkenntnisse über das Prozeßverhalten zu sammeln und zu nutzen. Vielmehr ist die Inbetriebnahme beendet, sobald die im Lastenheft festgesetzten Spezifikationen gerade eingehalten werden.

Weil zudem bei Planung und Betrieb von Prozeßleitsystemen (PLS) bisher nur regelungstechnische CAE-Insellösungen eingesetzt werden, ist eine durchgängige CAE-Unterstützung für die Planung und Inbetriebnahme von PLS wünschenswert, nicht nur um die Planungskosten zu senken, die Planungssicherheit zu erhöhen und die Anlage schneller und effizienter optimieren zu können, sondern auch weil die Kosten der Prozeßleittechnik inklusive der MSR-Komponenten bis zu 30% der Gesamtinvestition ausmachen können [1].

REGELUNGSTECHNISCHES CAE BEI PLANUNG UND BETRIEB VON PLS

Eine Übersicht über die Phasen, die bei der Planung eines PLS durchlaufen werden, zeigt Bild 1. Aufgabenstellungen, die mit regelungstechnischen CAE-Systemen bearbeitet werden könnten, ergeben sich dabei in vier Phasen:

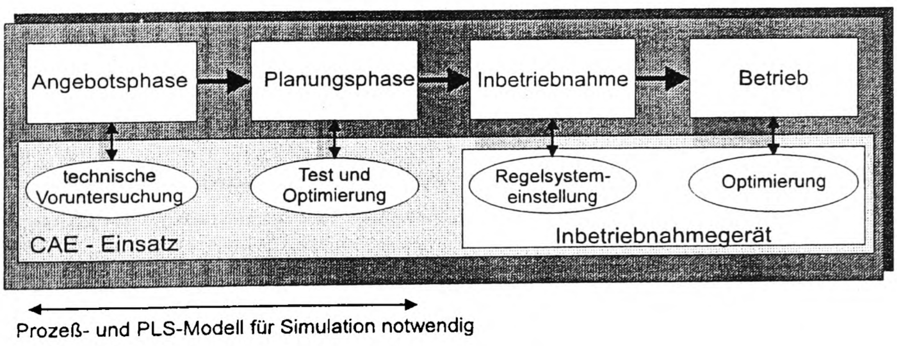


Bild 1. CAE-Einsatz bei Planung und Betrieb von PLS

1. ***ANGEBOTSPHASE*** *: CAE für die technische Voruntersuchung*
   In der Angebotsphase kann die simulierte Darstellung von regelungstechnischen Funktionen für das Angebot eines PLS-Herstellers wichtig werden, um dem Kunden die funktionellen und wirtschaftlichen Vorteile der vorgeschlagenen Lösung zu

veranschaulichen. Dazu sollte es mit vertretbarem Aufwand möglich sein, die herausgestellten Regelstrategien an einer (Teil-)Simulation des Prozesses zu demonstrieren. Der Einsatz eines regelungstechnischen CAE-Systems ist aber nur möglich, sofern dabei auf simulierbare (Teil-)Prozeßmodelle sowie simulierbare PLS-Funktionen zugegriffen werden kann.

2. ***PLANUNGSPHASE:*** *CAE für Test und Optimierung*
   In der Planungsphase können bei der Konfiguration des PLS mit Hilfe eines regelungstechnischen CAE-Systems die vorgesehenen Regelstrategien simulativ getestet und optimiert werden. Ein derartiges Vorgehen vergrößert die Planungssicherheit und minimiert die Inbetriebnahmezeit, da die zur Regelung notwendigen Regelstrategien ausgewählt, simuliert und parametriert werden. Auch dazu müssen Modelle von PLS-Funktionen und Prozeß vorhanden sein.
3. ***INBETRIEBNAHME:*** *CAE für Regelsystemeinstellung*
   In der Inbetriebnahmephase kann ein regelungstechnisches CAE-System die Einstellung und Modifikation von Regelschaltungen unterstützen, indem es Funktionen zur Prozeßidentifikation, Simulation und für den Regelsystementwurf zur Verfügung stellt. Ein derartiges Werkzeug kann den Inbetriebnahmeaufwand für regelungstechnische Funktionen erheblich verringern und damit auch Inbetriebnahmezeiten verkürzen. Voraussetzung dafür ist, daß es auf den industriellen Inbetriebnehmer zugeschnitten wird, der es einzusetzen hat.
4. ***BETRIEB:*** *CAE für Prozeßoptimierung*
   Im Betrieb kann ein regelungstechnisches CAE-System zur langfristigen Nachoptimierung sowie zur Entwicklung und Erprobung neuer Regelstrategien eingesetzt werden.

Als wichtige Voraussetzung für eine durchgängige CAE-Unterstützung aller Planungsphasen müssen insbesondere die folgenden Hindernisse beseitigt werden:

- ***Verfügbarkeit von Prozeß- und PLS-Modellen***
  In den ersten beiden Phasen ist der effektive Einsatz von CACSD-Werkzeugen direkt von der Verfügbarkeit von Prozeß- und PLS-Modellen abhängig. Während PLS-Funktionsmodelle zumindest prinzipiell aus dem realen PLS direkt ableitbar sind, ist der Zugriff auf Prozeßmodelle ein grundsätzliches Problem, da diese Modelle von den PLS-Herstellern vielfach noch nicht bereitgestellt werden können. Zum einen existieren für viele Prozesse keine allgemein verfügbaren, simulierbaren Modelle, zum anderen bildet

selbst bei theoretischen verfügbaren Prozeßmodellen der Aufwand für deren Simulation ein erhebliches Einsatzhindernis, das nur durch den systematischen Aufbau von allgemein zugänglichen, standardisierten Prozeßmodellkatalogen beseitigt werden kann. Derzeit bemüht sich der GMA-Ausschuß FA5.6 um einen Lösungsansatz zu diesem Thema.

- ***Ausbildung von Schnittstellenstandards***
  Der Einsatz regelungstechnischer CAE-Systeme und Inbetriebnahmehilfen wird durch Probleme bei der Ankopplung an CAE-Planungssysteme beziehungsweise an das PLS erschwert, was nur durch Ausbildung von Schnittstellenstandards erleichtert werden kann. So ist zum Beispiel die GMA Richtlinie VDI/VDE 3696 [2] auf der Grundlage der IEC1131-3 [3] ein wichtiger Schritt in diese Richtung, da sie nicht nur eine herstellerneutrale Konfigurationssprache auch für PLS darstellen, sondern ebenso den breiten Einsatz regelungstechnischer CAE-Systeme in der Prozeßleittechnik fördern kann. Auch der GMA-Ausschuß FA5.4 versucht mit einem Richtlinienentwurf für Inbetriebnahmesysteme den Weg zu einer Teillösung zu ebnen (VDI/VDE 3685, Teil 3).

Die durchgängige Unterstützung der PLS-Planung durch regelungstechnisches CAE wird durch die geschilderten Probleme zur Zeit noch behindert. Ein erster Schritt ist die industrielle Nutzung regelungstechnischer CAE-Systeme in der Inbetriebnahmephase, wie sie in den nächsten Abschnitten skizziert wird.

INDUSTRIELLE ANFORDERUNGEN AN REGELUNGSTECHNISCHES CAE

Ein Industrieingenieur benötigt eine *aufgabenorientierte CAE-Unterstützung* bei der Lösung praktischer Aufgabenstellungen, die einen überschaubaren Lösungsweg anbietet, keine tiefgreifenden theoretischen Kenntnisse voraussetzt und dennoch moderne, leistungsfähige Methoden der Regelungstechnik nutzt.

Derzeit ist eine große Zahl regelungstechnischer CAE-Systeme am Markt erhältlich [4,5], die den Entwurf von Regelstrategien teilweise oder in allen Entwurfsschritten unterstützen (siehe auch Bild 2). Dennoch sind diese Programme wenig für den industriellen Einsatz in der Prozeßleittechnik geeignet, da sich gezeigt hat, daß sie Planern, Inbetriebnehmern und

Betriebsleuten eine ungewohnte und nur mit regelungstechnischem Expertenwissen durchführbare Vorgehensweise aufzwingen. Das liegt daran, daß die überwiegende Mehrzahl der CAE-Systeme an Hochschulen von und für Regelungstechnik-Experten entwickelt worden ist und vorwiegend zur Implementierung und Erprobung von theoretischen Verfahren und Methoden dient. Diese Methoden versagen aber häufig in der Praxis, da ihre Anwendung schwierig ist. Daher werden in der industriellen Praxis normalerweise lediglich die von verschiedenen PLS-Herstellern angebotenen einfachen Inbetriebnahmehilfen für Eingrößenregelungen eingesetzt, die zumeist nur automatisierte Varianten alter Einstellverfahren sind.

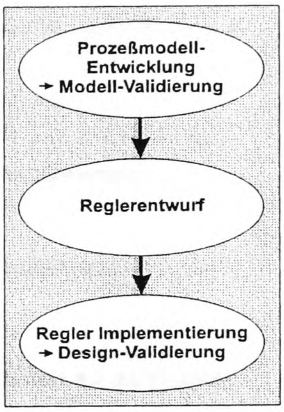


Bild 2. Regelungstechnische Entwurfsfolge

Grundlegend für die Lösung schwierigerer Aufgaben wie zum Beispiel der Regelung komplexer Mehrgrößenprozesse ist die Bereitstellung von verständlichen *Standardlösungswegen*, damit alle zur Lösung notwendigen Schritte nachvollziehbar werden. Besondere Bedeutung kommt dabei der gewählten mathematischen Modellbeschreibung zu. So sollten auch komplexe nichtlineare Mehrgrößenprozesse nach Möglichkeit als Verschaltung von linearen und einfachen nichtlinearen Blöcken modelliert werden, die für den Industriepraktiker verständlich sind.

Zudem ist es notwendig, regelungstechnische CAE-Methoden auf die *Kenntnisse und Arbeitsweise von Industrieingenieuren* zuzuschneiden. Die Methoden müssen durch leichtverständliche Parametrierung einfach anwendbar und zusätzlich robust sein. Wenige, leistungsfähige Methoden müssen einen weiten Einsatzbereich abdecken.

Besondere Beachtung verdient auch die *Gestaltung der Bedienoberfläche*, die Benutzerebenen je nach regelungstechnischem Wissen bereitstellen sollte und an die zu lösenden industriellen und nicht hochschulwissenschaftlichen Aufgabenstellungen angepaßt sein muß. Die Ergebnisse müssen dem Anwender entsprechend seines Kenntnisstandes präsentiert werden. Ein Benutzer mit geringem regelungstechnischen Wissen darf nicht mit abstrakten Maßzahlen und Parametern konfrontiert werden, vielmehr müssen ihm die Ergebnisse in intuitiv verständlichen Grafiken vorzugsweise im Zeitbereich präsentiert werden. Andere Darstellungsformen (Frequenzbereich, Zustandsraum etc.) sollten nur dem Spezialisten zugänglich sein, der zusätzliche Informationen über das Verhalten des Regelsystems benötigt und auch auswerten kann.

Auf Basis derartiger Überlegungen ist ein exemplarisches Konzept zum Einsatz eines regelungstechnischen CAE-Systems in der Inbetriebnahmephase entwickelt worden.

KONZEPT FÜR EIN INDUSTRIETAUGLICHES, REGELUNGSTECHNISCHES CAE-SYSTEM FÜR DIE INBETRIEBNAHMEPHASE

Das hier vorgestellte Konzept für ein regelungstechnisches CAE-System für die Inbetriebnahme orientiert sich an der Vorgehensweise industrieller Anwender. Regelungstechnische Aufgaben werden von Praktikern im allgemeinen nach einem einfachen Schema bearbeitet. Zunächst wird versucht, auch bei Mehrgrößenprozessen unabhängige PID-Einzelregler auszulegen (Bild 3). Normalerweise geschieht die Reglerauslegung nach praktischen Faustregeln. Sollte die erreichte Regelgüte dieser einfachsten Regelstruktur nicht ausreichend sein, werden bei nichtlinearen Einflüssen die Nichtlinearitäten kompensiert oder im Fall von störenden Kopplungen der Einzelregelkreise einfache Entkopplungsschaltungen implementiert. Bei Versagen dieser einfachen Vorgehensweise werden dann intuitive und individuelle Lösungen, zum Beispiel durch Einbau von nichtlinearen Gliedern wie Multiplizierern und Dividierern erstellt. Die so entstandenen Regelsysteme sind einer systematischen Analyse nur noch schwer zugänglich.

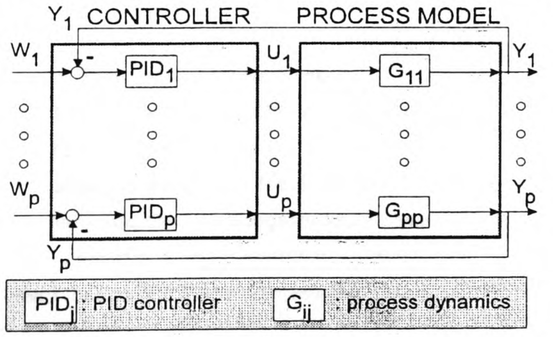


Bild 3. SISO Prozeßmodell mit zugehöriger Eingrößen-Reglerstruktur

Das hier vorgeschlagene Konzept für ein industrietaugliches, regelungstechnischen CAE-System beruht auf zwei Grundgedanken:

1. einem standardisierten Modellentwicklungs-Schema und
2. einer vereinfachten Vorgehensweise für den rechnergestützten Reglerentwurf.

Modellentwicklungs-Schema

Das Modellentwicklungs-Schema folgt dem oben beschriebenen Ingenieurgedanken „beginne einfach und erweitere die Komplexität nur soweit notwendig". Durch die Nutzung von Standardmodellen mit zugehörigen Regelungsstrukturen ist es so transparent, daß der Regelsystementwurf jederzeit nachvollziehbar bleibt.

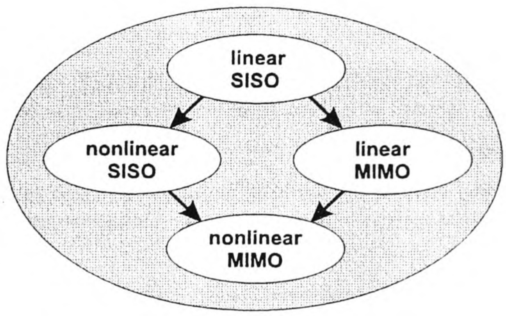


Bild 4. Modellentwicklungs-Schema

Die erste Stufe des Modellentwicklungs-Schemas (Bild 4) beruht auf der Auslegung von linearen Eingrößenregelkreisen (linear SISO: Single Input Single Output, Bild 3) und entspricht damit dem ersten Schritt des Industriepraktikers. Falls hierdurch keine zufriedenstellenden Ergebnisse erreicht werden können, wird die Regelungsstruktur entweder durch Nichtlinearitäten (nonlinear SISO) oder Kompensationsschaltungen (linear MIMO: Multiple Input Multiple Output) erweitert, je nachdem welche Einflüsse überwiegen. Dabei werden nichtlineare Zusammenhänge bewußt als einfache Kombination von statischer Kennlinie und linearer Dynamik in Form von Wiener oder Hammerstein-Modellen modelliert. Das Modell größtmöglicher Komplexität für einen Prozeß mit statischen Eingangsnichtlinearitäten ist im Bild 5 dargestellt (nonlinear MIMO).

Vereinfachte Vorgehensweise für den rechnergestützten Reglerentwurf

Die Vielfalt der möglichen Regelungsstrukturen wird schon durch die Anwendung dieses Modellentwicklungs-Schema grundsätzlich eingeschränkt, in dem immer ein Standardmodell und ein dazu komplementäres Regelschema angewendet wird. Die Anwendung der regelungstechnischen Entwurfsfolge in ihren drei Phasen (Bild 2) wird durch die folgenden Vorgaben weiter vereinfacht:

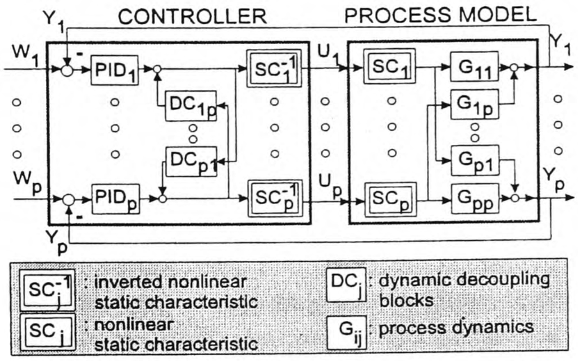


Bild 5. Nichtlineares Mehrgrößenmodell mit zugehöriger Reglerstruktur

- ***Prozeßmodellentwicklung:***
  Im Rahmen der Prozeßmodellentwicklung für die Inbetriebnahme stehen nur zwei parametrische Identifikationsmethoden zur Verfügung, von denen die eine zeitdiskrete und die andere zeitkontinuierliche Modelle liefert. Beide Methoden sind auf Robustheit überarbeitet und vorparametriert für eine einfache Handhabung. Nichtlineare statische Kennlinien werden durch ein einfaches Analyseprogramm automatisch bestimmt.
- ***Reglerentwurf***
  Der PID-Reglerentwurf basiert auf numerischer Optimierung, die über verständliche Vorgaben wie Ausregelzeit und Überschwingweite steuerbar ist. Eingangs- bzw. Ausgangs-Nichtlinearitäten, die als Kennlinien im Prozeßmodell vorliegen, werden im einfachsten Fall durch ihre Inversen kompensiert.
- ***Reglerimplementierung***
  Die Überprüfung des entwickelten Regelschemas geschieht durch Prototypenregelung auf dem regelungstechnischen Inbetriebnahmesystem oder mit Hilfe von industriellen Regelungssystemen. Wichtig ist, daß die dort verwendeten Regler und Funktionsblöcke bereits beim Reglerentwurf berücksichtigt werden.

Um unnötige Iterationen in der Entwurfsfolge zu vermeiden, wird in jeder Validierungsphase geprüft, ob die Ergebnisse für die nächsten Schritte ausreichend sind.

Eine weitere wichtige Komponente zur Vereinfachung der Handhabung ist die Verwendung einer aufgabenorientierten, graphischen Benutzeroberfläche, wie sie bereits oben beschrieben wurde.

STAND DER ENTWICKLUNG

Das skizzierte Konzept eines regelungstechnischen Inbetriebnahmesystems wurde in der ersten Projektphase mit verschiedenen regelungstechnischen CAE-Programmen an Laborprozessen mit Erfolg getestet [6]. Aufgrund der unterschiedlichen Stärken der eingesetzten Programme wurde für jede Phase des rechnergestützten Reglerentwurfs ein anderes Programm gewählt, wobei erwartungsgemäß folgende Probleme auftraten:

- verschiedenartige und schlechte Benutzerführung
- komplizierte Parametrierung und unzuverlässige Entwurfsergebnisse
- unterschiedliche Schnittstellen und Dateiformate

Um diese Probleme zu vermeiden, wird derzeit ein MATLAB™-Prototyp eines regelungstechnischen Inbetriebnahmesystems entwickelt, der von der Identifikation bis zum Reglerentwurf und -test alle Entwurfsphasen mit aufgabenorientierter, graphischer Benutzerführung gemäß dem vorgestellten Konzept unterstützt. Nach intensiven Tests im Labor und in der Industrie soll dieser Prototyp direkt programmiert werden.

AUSBLICK

Regelungstechnische CAE-Systeme und Inbetriebnahmehilfen besitzen für alle Planungs- und Betriebsphasen eines PLS beträchtliches Rationalisierungspotential. Sie können dabei folgende Verbesserungen bewirken:

- erhöhte Planungssicherheit für regelungstechnische Funktionen
- Verkürzung der Inbetriebnahmezeit für Regelungen

Voraussetzung für ihren industriellen Einsatz sind aber:

- effiziente Unterstützung aller regelungstechnischen Entwurfsphasen auch bei komplizierteren Prozessen
- reproduzierbare, verläßliche Entwurfsergebnisse
- Nutzbarkeit für industrielles Inbetriebnahme- und Betriebspersonal

Die heute erhältlichen regelungstechnischen CAE-Softwarepakete sind im wesentlichen auf akademische Nutzer zugeschnitten; industrietaugliche, regelungstechnische CAE-Systeme stehen erst am Beginn ihrer Entwicklung. Die Benutzerführung solcher Systeme muß auf die Lösung der industriellen Entwurfsaufgabe zugeschnitten sein. Dabei spielen standardisierte Lösungswege, die Verwendung vorparametrierter, robuster Methoden und eine aufgabenorientierte graphische Benutzeroberfläche eine entscheidende Rolle.

Das hier vorgestellte Konzept für ein industrietaugliches, regelungstechnischen Inbetriebnahmesystem ist ein erster Schritt in Richtung auf eine erweiterte Unterstützung von Planung und Betrieb von PLS durch regelungstechnische CAE-Systeme.

REFERENZEN

/1/ WEBER, K. H.: Inbetriebnahme verfahrenstechnischer Anlagen. VDI-Verlag, Düsseldorf, 1996, ISBN 3-18-401367-7.

/2/ VDI/VDE 3696 Herstellerneutrale Konfiguration von Prozeßleitsystemen. Beuth Verlag, Berlin, 1995.

/3/ IEC 1131-3: Deutsche Norm, Speicherprogrammierbare Steuerungen, Teil 3: Programmiersprachen. DIN EN 61131-3, Juli, 1993.

/4/ SCHUMANN, R.: CAE von Regelsystemen. atp - Automatisierungstechnische Praxis, R. Oldenbourg Verlag, 1994, vol. 36, no. 3, pp. 51-59.

/5/ FREDERICK, D. K.;HERGET, C. J.; KOOL, R.; RIMVALL, M.: ELCS - The extended list of control software. Dept. of Automatic Control, Swiss Federal Institute of Technology (ETH), 1992, no. 5.

/6/ SCHUMANN, R.; KÖRNER, S.; BAKER, K. J.; STRICKRODT, M.: Shaping CACSD for practical use in process industry. IFAC World Congress, San Francisco, 1996.

CAE hilft Prozessen schneller auf die Sprünge

Steffen Körner, Reimar Schumann. *Industrielle Denk- und Arbeitsweisen könnten regelungstechnische CAE-Werkzeuge für die Prozeßleittechnik effizienter machen und die zeitaufwendige Einstellung kritischer Regelungen verkürzen. Den angebotenen Werkzeugen fehlt vielfach der Bezug zur Praxis.*

Die Mehrzahl der regelungstechnischen Funktionen eines Prozeßleitsystems (PLS) werden in standardisierter Weise bereits bei der Planung festgelegt und, falls noch erforderlich, bei der Inbetriebnahme modifiziert und parametriert. Dabei bleiben nur besonders schwierige und zeitaufwendige regelungstechnische Aufgaben übrig, für die es keine Standardlösungen gibt oder bei denen die Einstellung der Regler problematisch ist. Regelungstechnische Computer-aided-engineering-Systeme (CAE-Systeme) können zu einer rationellen Lösung dieser Aufgaben beitragen. Ihre Einsatzmöglichkeiten bei Planung, Inbetriebnahme und Betrieb von Prozeßleitsystemen sind vielfältig (Abb. 1).

In der Angebotsphase können die Funktionen kritischer Anlagenteile zusammen mit der vorgesehenen PLS-Instrumentierung simulativ ausgelegt und bewertet werden. So kann schon zu diesem frühen Zeitpunkt die Leistungsfähigkeit der Anlage punktuell untersucht und Wirtschaftlichkeitsanalysen durchgeführt werden. Eine in der Planungsphase durchgeführte simulative Überprüfung aller konfigurierter PLS-Funktionen hilft, die Inbetriebnahme erheblich zu verkürzen. Allerdings müssen in beiden Phasen die zur Simulation benötigten Modelle von Prozeß und/oder PLS verfügbar sein, was heute noch selten der Fall ist. Während der Inbetriebnahme hingegen können regelungstechnische CAE-Systeme schon heute die schwierige und zeitaufwendige Einstellung kritischer Regelungen verkürzen helfen. Sie erlauben anschließend während des Betriebs eine weitere Untersuchung und Optimierung des Prozesses.

Erschwert wird der Einsatz regelungstechnischer CAE-Systeme und Inbetriebnahmehilfen durch Probleme bei der Ankopplung an CAE-Planungssysteme beziehungsweise an das PLS, was nur durch Ausbildung von Schnittstellenstandards erleichtert werden kann. So ist die IEC1131-3 ein wichtiger Schritt in diese Richtung, die nicht nur eine herstellerneutrale Konfigurationssprache auch für PLS darstellen, sondern ebenso den breiten Einsatz regelungstechnischer CAE-Systeme in der Prozeßleittechnik den Weg bereiten kann.

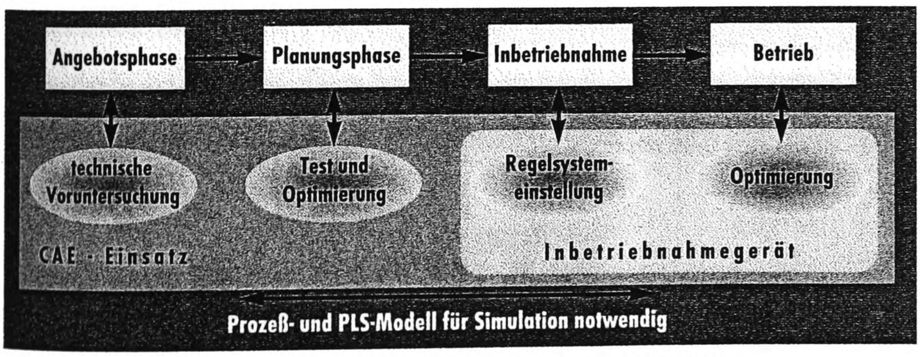


Abb. 1: Einsatz regelungstechnischer CAE bei Planung und Betrieb von PLS

Die von einem regelungstechnischen CAE-System angebotenen Funktionen, wie sie beispielsweise bei der Entwicklung einer einfachen Regelschaltung (Abb. 2) benötigt werden, unterstützen folgende Entwurfsschritte:

- Entwicklung des Prozeßmodells,
- Entwurf des Reglers sowie
- Implementierung des Reglers am realen Prozeß.

Diese Schritte sollen am Beispiel eines Inbetriebnahmesystems weiter erläutert werden, wobei hier vorausgesetzt wird, daß dieses Inbetriebnahmewerkzeug direkt auf die Prozeßsignale zugreifen kann. Die Entwicklung des Prozeßmodells erfolgt dann direkt mit Hilfe von Prozeßidentifikationsverfahren durch Analyse der gemessenen Ein- und Ausgangssignale des Prozesses. Im zweiten Schritt wird das so gewonnene Prozeßmodell genutzt, um einen passenden Regler zu entwerfen. Für einen PID-Regler beispielsweise können die Parameter durch simulative Optimierung sehr anschaulich bestimmt werden, während bei der Auslegung anderer, weniger gängiger Reglertypen spezielle Programme genutzt werden. Im dritten Schritt folgt die erste versuchsweise Implementierung des Reglers am realen Prozeß, bei der die im Inbetriebnahmesystem simulierte Regelschaltung mit „Hardware (Prozeß) in the Loop" anstelle des Prozeßmodelles betrieben wird. Je nach Güte des Prozeßmodells, auf dessen Basis der Regler entworfen wurde, wird sich nun ein reales Regelkreisverhalten ergeben, das dem zuvor simulierten um so näher kommt, je besser das simulierte Prozeßmodell ist.

Um regelungstechnische CAE-Programme nutzbringend einzusetzen, ist es notwendig, diese auf die Kenntnisse und Arbeitsweise von Ingenieuren zuzuschneiden. Besonders diese Aufgabe darf nicht unterschätzt werden, da die Vielfalt regelungstechnischer Methoden nahezu grenzenlos ist und nur von Experten beherrscht wird. Derzeit wird eine große Zahl regelungstechnischer CAE-Systeme angeboten, die den regelungstechnischen CAE-Entwurf teilweise oder in allen Entwurfsschritten unterstützen. Dennoch muß gefragt werden, inwieweit diese Programme für den industriellen Einsatz in der Prozeßleittechnik geeignet sind. Die überwiegende Mehrzahl ist an Hochschulen von und für Regelungstechnikexperten entwickelt worden und dient vorwiegend der Implementierung und Erprobung von theoretischen Verfahren und Methoden, die möglicherweise sehr effizient sind, aber häufig in der Praxis versagen, da die Anwendung zu schwierig ist.

Es gibt derzeit nur wenige industrielle Regelungstechnikexperten, insbesondere bei den großen PLS-Anwendern, die diese Programme gewinnbringend einsetzen können, und es werden eher weniger. Ein Praktiker hingegen, der nicht nur regelungstechnische Aufgaben zu erfüllen hat, verlangt eine funktionsorientierte CAE-Unterstützung, die einen überschaubaren Lösungsweg anbietet und keine tiefgreifenden theoretischen Kenntnisse voraussetzt. Das bisher fast immer angebotene Übermaß alternativer und zum Teil parameterintensiven Methoden verwirrt den Anwender weit mehr, als daß es ihm bei der Lösung praktischer Aufgaben hilft. Daher werden in der industriellen Praxis normalerweise lediglich die von verschiedenen PLS-Herstellern angebotenen einfachen Inbetriebnahmehilfen eingesetzt, die zumeist nur automatisierte Varianten alter, auf Tabellen basierender Einstellverfahren sind. Möchte man aber dem industriellen Anwender zusätzlich moderne, leistungsfähige Regelungstechnikmethoden zugänglich machen, so ist die Entwicklung praxistauglicher, industrieller CAE-Systeme notwendig.

Besondere Beachtung verdient dabei die Bedienoberfläche, die Benutzerebenen je nach regelungstechnischem Wissen bereitstellen und an die zu lösenden industriellen (und nicht hochschulwissenschaftlichen) Aufgabenstellungen angepaßt sein muß. Auch die Ergebnisse müssen dem Kenntnisstand des Anwenders entsprechend präsentiert werden. Das heißt, daß ein Benutzer mit geringem regelungstechnischen Wissen nicht mit abstrakten Maßzahlen und Parametern konfrontiert wird, sondern daß ihm die Ergebnisse in verständlichen Graphiken vorzugsweise im Zeitbereich präsentiert werden, zum Beispiel durch Darstellung der Sprungantworten von Prozeßmodellen und Regelsystemen einschließlich der dazugehörigen Zeitkenngrößen wie Anstiegs- oder Ausgleichszeit. Andere Darstellungsformen (Frequenzbereich, Zustandsraum und andere) sollten nur dem Spezialisten zugänglich sein, der zusätzliche Informationen über das Verhalten des Regelsystems benötigt und auch auswerten kann.

Regelungstechnische CAE-Systeme

Regelungstechnische CAE-Systeme und Inbetriebnahmehilfen bergen für alle Phasen des Engineerings eines Prozeßleitsystems ein beträchtliches Rationalisierungspotential in sich, wie zum Beispiel:
- Erhöhte Planungssicherheit für regelungstechnische Funktionen
- Kürzere Inbetriebnahme von Regelungen
- Effiziente Hilfe bei der Modellbildung, auch bei schwierigen Prozessen
- Erstellen reproduzierbarer und verläßlicher Entwürfe
- Anwendung auch durch reguläres Inbetriebnahme- und Betriebspersonal

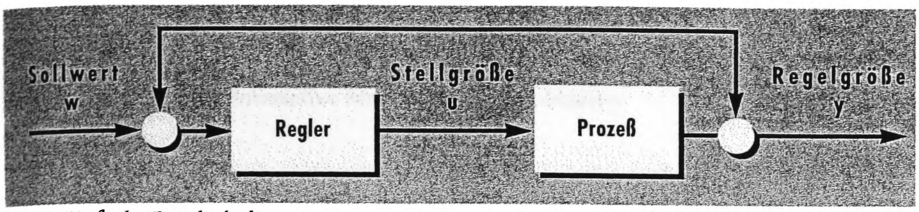


Abb. 2: Einfache Regelschaltung

Grundlegend für die Lösung schwieriger Aufgaben, wie zum Beispiel der Regelung komplexer Mehrgrößenprozesse, muß die Bereitstellung von verständlichen Standardlösungen sein, die alle notwendigen Schritte durchsichtig werden lassen. Eine hervorragende Bedeutung kommt dabei der gewählten, mathematischen Modellbeschreibung zu: Auch komplexe, nichtlineare Mehrgrößenprozesse sollten nach Möglichkeit als Verschaltung von linearen und einfachen nichtlinearen Blöcken modelliert werden, die auch noch für den Industriepraktiker verständlich sind. Auf Basis derartiger Überlegungen ist ein exemplarisches Konzept zum industriellen, rechnergestützten Reglerentwurf für Mehrgrößenprozesse entwickelt worden.

Die Nutzung der modernen CAE-Werkzeuge wird in der Planungsphase vor allem vom Mangel an simulierbaren Modellen behindert, während ihr Einsatz während der Inbetriebnahme und im Betrieb heute noch insbesondere an der Orientierung auf akademische Nutzer krankt, was aber durch die Entwicklung industrietauglicher Inbetriebnahmesysteme mit aufgabenorientierter Benutzerführung behoben werden kann.

Ferner sind für die zukünftige Entwicklung regelungstechnischer CAE industrielle Standards nötig. Die Entwicklung einer geeigneten Schnittstelle für den Datenaustausch mit anderen Systemen und die Einbindung der IEC1131-3 sind weitere wichtige Schritte auf dem Weg zum breiten Einsatz regelungstechnischer CAE in der Prozeßleittechnik.

Dipl-Ing. Steffen Körner, Prof. Dr.-Ing. Reimar Schumann, Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Tel. 0511/9296309.

New Approach to Identification in Blockoriented Simulation Environments

Steffen Körner*, Reimar Schumann* and Keith J. Baker**

* Forschungsschwerpunkt REPAM, FachhochschuleHannover, RicklingerStadtweg 120, 30459 Hannover, Germany, e-mail:koerner@repam.fh-hannover.de
**Department of Mechanical and Manufacturing Engineering, University of Glamorgan, Pontypridd, Mid Glamorgan, CF37 1DL, United Kingdom

KEYWORDS
Computer Aided Engineering (CAE), Industrial Engineering, System Identification, Interactive Modelling, Process Industry

ABSTRACT

This paper deals with the implementation of identification strategies into blockoriented simulation environments, which is particularly suitable for modelling and simulation tasks aimed at control. Although many tools for identification (i.e. experimental modelling) of processes are available nowadays most of these tools are implemented separately from the blockopriented simulation environment and strain the user by offering a wealth of sophisticated methods, which can only be understood and used by identification specialists. Generally, these tools do not address the blockoriented modelling of structured submodels and can not support the practical engineer sufficiently, who needs identification to solve a part of the design problem and not as a problem itself.

The new approach to integrated identification presented here allows the intuitive handling of the identification task within a simulation environment through an ergonomic user interface and is especially aimed at process industry. The development of a linear dynamic SISO (Single Input Single Output) and a nonlinear static SISO block in combination with the simplified use of efficient identification algorithms allows the application to a wide range of simulation and control design tasks. A prototype implementation in MATLAB/SIMULINK™ illustrates the application of this approach.

MODELLING AND IDENTIFICATION

Today, powerful blockoriented simulation environments are available, which allow a quick and also hierarchical generation of process models. These simulation environments are particularly suitable for control engineers, who are mostly familiar with block diagram representations (Jobling 1996). However the modelling of real processes often needs realtime experiments to finally adapt the process model that may have evolved from conventional theoretical modelling to the real process behaviour. The problem can be diverse. If the theoretical model is complete only parameters have to be adjusted; if some parts of the model are missing identification has to complement the theoretical model with identified submodels.

Even if no theoretical model is available identification can be used to get a black box model of the whole process. A specific advantage of black box identification is the relatively quick and inexpensive generation of process models compared to the effort needed to build theoretical models. Although identification is an important part of the modelling task especially in the context of control, currently available blockoriented simulation environments do generally not provide identification facilities that can be utilised even by inexperienced users. Besides available identification tools are usually a loose collection of algorithms, which mostly support different kinds of parameter estimation but they heavily rely on the expertise of the user as outlined by, for example, Eykhoff and Parks (1990) or Ljung (1987). Thus some of these tools are excellent test beds for identification methods, however, requiring considerable expert knowledge. It is left to the user to select appropriate methods and to tune them properly in order to get meaningful models, which at the end must be validated by comparison of the simulated and the real process behaviour. Mostly these tools do not address the blockoriented modelling of substructures and also neglect nonlinear models. In the end they cannot support the practical engineer sufficiently, who is no expert in identification.

To overcome the described problems identification must not be presented as a separated task for specialists but must become an integrated part of simulation environments. Only then even industrial users, who are inexperienced in identification, can make full use of properly identified models (Körner and Schumann, 1996).

REQUIREMENTS FOR PRACTICAL IDENTIFICATION

After interviews in process industry and tests at pilot plants general requirements were formulated that must be addressed for practical application. The primary requirement is to integrate the identification task into a blockoriented simulation environment. Further the integrated solution must:

- be tailored to industrial needs
- be intuitively to use and understand
- support nonlinear and MIMO designs
- guide the user through standard design paths
- simplifiy the use of advanced methods

The main goal of this industrial identification approach is to generate simple, reliable and reproducible solutions also for complex tasks. Therefore, one basic idea is to preselect only few identification methods which are preparametrised with default parameters and which are made robust for simplified use. This enables the user to apply identification with only very few application specific adjustments, if any.

Another key element is to restrict the model structure complexity to a set of standard model structures that can be applied to multivariable processes. The simplest structure is the linear SISO (single input single output) case that can be extended to the complex nonlinear MIMO (multiple input multiple output) case in a transparent procedure.

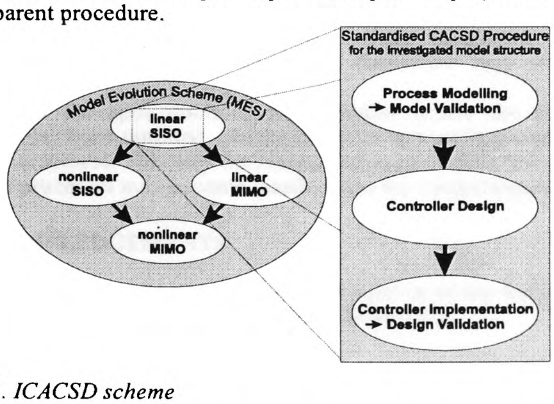


Fig. 1. ICACSD scheme

A general framework for such a simplified setup has been presented by Schumann et al. (1996), when introducing the ICACSD (Industrial Computer Aided Control System Design) scheme for the modelling of unknown processes (Fig. 1). The ICACSD scheme is suitable for a wide range of processes and consists of the Model Evolution Scheme (MES) and the Standardised CACSD Procedure, which is executed for each model structure of the MES under investigation. The models are composed of blocks with linear dynamics and static characteristic in form of Wiener- or Hammerstein- Models. If the identification results are not satisfactory for the actual model structure, the next more complex standard model structure is accessed according to the MES to improve the model quality until it is satisfactory.

PROTOTYPE IMPLEMENTATION

For the evaluation and first applications of the described identification approach a MATLAB/SIMULINK™ prototype has been programmed that integrates the identification task into the blockoriented SIMULINK environment. In the following the main features of the identification blocks are described as an example for the integrated implementation in blockoriented simulation environments. Then the application of the ICACSD scheme is outlined.

The Identification Block

While modelling the process from the SIMULINK block library identification blocks represent just an additional type of blocks that can be inserted, where necessary. The handling of an identification block is equal to that of other blocks, such that the user's modelling procedure is not affected. Only its colour is different in this stage to allow a quick recognition of the (unmodelled) identification blocks particularly within big projects.

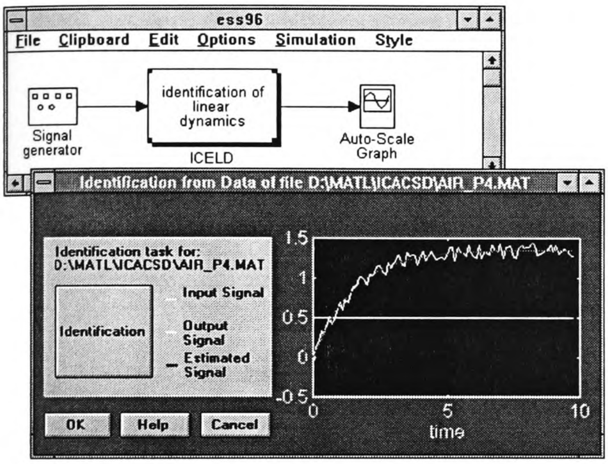


Fig. 2. Identification block for linear SISO dynamics

A double-click on the identification block starts a guided tour to identification utilising graphical user interfaces that request a minimum of information. Then the identification of the block model is done from input-output data, either available from a data file or from a realtime experiment at the process. The identification algorithm returns the estimated model that can be verified by graphical inspection (Fig. 2). After successful identification the block is filled with the identified model and switches to standard colour. Now the model can be investigated by simulation. In case the results are not satisfactory, the model can be refined by repeated identification. Up to now two identification block types have been programmed:

- the linear dynamic SISO model block and
- the nonlinear static SISO model block.

It is advantageous that through the integration of identification blocks within the simulation environment the usual interfacing problems are avoided that occur, if the identification results have to be transferred by hand from a separate identification program.

Identification within the ICACSD Scheme

For complex MIMO control problems a software tool is under development that generates automatically even complex process model structures following the ICACSD scheme and using the identification blocks described above This way a black box model is developed as a suitable basis for control design.

The ICACSD tool starts with a window showing the basic structure of the control system including the process model as shown in Fig. 3. At first the user has to specify the number of process inputs and process outputs. If some extra process knowledge can be provided, e.g. about process limits, couplings or nonlinearities, this will be considered when the model structure is generated. Thus the simplest model structure is generated automatically within SIMULINK composed from linear dynamic SISO and nonlinear static SISO blocks descried above. Then the user has to activate the identification blocks in order to get a model of the process step by step. The resulting process model is used afterwards for the design of the control system within the simulation environment (Fig. 4).

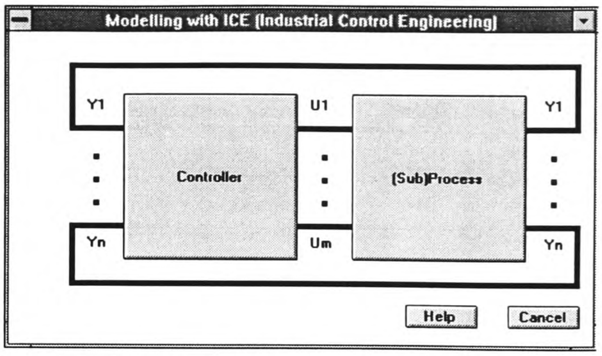


Fig. 3. Starting window to ICACSD

The control design procedure follows the ICACSD scheme and is iterated with a more complex model structure of the MES only if the control performance at the real process is not satisfactory.

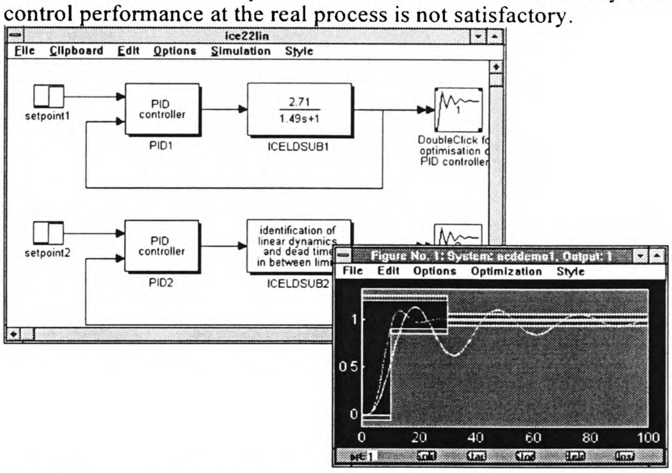


Fig. 4. Generated model structure with control design window

CONCLUSIONS

The need for an approach to integrated identification in blockoriented simulation environments has been discussed and the industrial requirements have been named. Based on this analysis integrated identification blocks have been developed for the MATLAB/ SIMULINK™ environment that allow easy access to identification.

Further the application of these blocks within the ICACSD scheme has been outlined, which allows an automatic generation of model structures. This way the identification of models for complex MIMO processes may become feasible also for inexperienced users.

ACKNOWLEDGEMENTS

The research project is carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

REFERENCES

Eykhoff, P.; P.C. Parks. 1990. „Identification and system parameter estimation; where do we stand now?".Automatica, Pergamon Press Ltd, vol. 26, no. 1: 3-5.

Jobling, C. P. 1996. Advances in computer aided control system designs." Computing and Control division, Colloquium about Advances in CACSD. (Mar.): 1/1-1/6.

Körner, S; R. Schumann. 1996 "Inbetriebnahmeunterstützungbei PLS durch regelungstechnisches CAE." VDI/VDE- GMA Kongreß, Meß- und Automatisierungstechnik, Baden-Baden, VDI-Bericht Nr. 1282: 93-103

Ljung, L. 1987. „System identification; Theory for the user." Prentice Hall, Englewool Cliffs, NJ.

Schumann, R; S. Körner; K.J. Baker; M. Strickrodt. 1996. „Shaping CACSD for practical use in process industry." IFAC World Congress, San Francisco, vol. L:223-228

ICAI - A MATLAB TOOLBOX FOR INDUSTRIAL COMPUTER AIDED IDENTIFICATION

Steffen Körner*, Reimar Schumann* and Keith J. Baker**

**Forschungsschwerpunkt REPAM, Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Germany, e-mail: koerner@repam.fh-hannover.de, FAX: +49-511-9296-111*

***Department of Mechanical and Manufacturing Engineering, University of Glamorgan, Pontypridd, Mid Glamorgan, CF37 1DL, United Kingdom*

Abstract: The Industrial Computer Aided Identification (ICAI) toolbox for MATLAB/ SIMULINK™ is especially aimed at industrial users like process engineers or commissioners who are not necessarily specialists in identification. In this paper the integration of the identification functionality into a blockoriented simulation environment by means of the ICAI toolbox is described. For SIMULINK™, i.e. MATLAB™'s simulation environment, ICAI identification blocks have been programmed, which can be handled intuitively like other SIMULINK™ function blocks supporting a guided tour to identification based on the simplified use of advanced identification methods and an ergonomically designed graphical user interface (GUI). Future extensions of the toolbox will allow identification of even nonlinear MIMO process models by making use of a set of standard structures automatically generated utilising ICAI MIMO identification blocks.

Keywords: System Identification, Computer Aided Engineering (CAE), Process Identification, Industry Automation, MIMO, User Interfaces, Interactive Approaches

1. INTRODUCTION

Nowadays various software programs for system identification are available. Most of these tools have been developed by academic experts at universities and are usually a comprehensive collection of algorithms that mostly support different kinds of parameter estimation. Therefore these tools are excellent test beds for identification methods but rely heavily on the expertise of the user as outlined by, for example, Eykhoff and Parks (1990) or Ljung (1991).

Ongoing efforts to simplify the access to identification methods led to developments like the system identification toolbox 4.0 (Ljung, 1995) for MATLAB™, which makes extensive use of a graphical user interface providing the means for book-keeping and comparison of different identification results. Nevertheless modern identification tools are rarely used in the process industry, a fact that motivated the design of the ICAI toolbox.

An essential starting point for this research work were interviews with potential identification users in the process industry. From these interviews practical problems were learnt that come along if process model identification must be practically applied to real size processes in the context of control system design. It was experienced that an identification tool is missing, which can be handled by those industrial users who are by no means identification experts.

Three main requirements were found for the design of an industrial identification tool:

- The identification task should be integrated into a blockoriented simulation environment where the user does not have to switch between different programs for identification, simulation and controller design.
- Only few identification methods with easy parametrisation and a simplified experiment setup should be provided.
- The user must be guided through standard identification paths especially in case of complex MIMO (Multiple Input Multiple Output) process identification tasks.

Of course, a standardised and therefore simplified identification procedure cannot guarantee an optimal identification result which may be possible with more degrees of freedom. However, for an industrial user a useable identification result following standard design paths may be the only result achievable on the background of limited expertise and little time available.

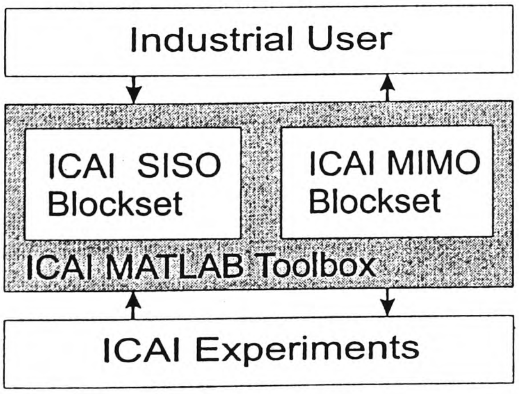


Fig. 1. ICAI MATLAB toolbox.

The new MATLAB™ ICAI (Industrial Computer Aided Identification) toolbox has been designed to address the industrial user's needs making available the power of modern identification algorithms to non-experts. It comprises a blockset of SISO ID (i.e. identification) blocks which integrate identification functionality into the SIMULINK™ environment. Furthermore MIMO ID blocks are under construction enabling industrial users to identify even nonlinear MIMO process models based on automatically generated standard block structures.

The paper is organised as follows: First the functionality of the ICAI toolbox is described. Then the use of the ICAI ID blocks in the SIMULINK™ environment is illustrated. Finally basic ideas of the ICAI MIMO ID blocks are outlined.

2. ICAI TOOLBOX DESCRIPTION

The ICAI toolbox is integrated in MATLAB™'s blockoriented simulation environment SIMULINK™ because a blockoriented simulation environment can serve as general base for industrial users to solve practical identification and control system design tasks.

The ICAI toolbox contains a SIMULINK™ blockset comprising various ID blocks for SISO (Single Input Single Output) models. Further MIMO ID blocks are under development for the simplified structuring and identification of MIMO processes.

According to Fig. 1 the functionality of the ICAI toolbox can be separated into two parts:

1. **ICAI SISO Identification blockset**: An ID block represents a black box which models an unknown part of the process model and which allows easy access to identification methods within SIMULINK™. At the moment the ICAI identification blockset comprises a static SISO ID block, a linear dynamic SISO ID block and a nonlinear SISO ID block for the identification of the respective models.

2. **ICAI MIMO Identification blockset**: MIMO ID blocks are under development being a generalisation of the SISO ID blocks. MIMO ID blocks support industrial users to structure and identify MIMO process models.

2.1. *Handling of the ICAI ID blocks*

ICAI ID blocks represent just an additional type of SIMULINK™ blocks. The handling of an ID block is identical to that of other blocks, only its colour is different to the other blocks and changes during the identification procedure. This allows easy recognition of the status of the ID block especially within a big model composed from several SIMULINK™ blocks. A double-click on the identification block starts a guided tour through identification utilising a graphical user interface and requesting only a minimum of information from the user.

he application of ICAI ID blocks is done in three ajor steps which are described in the following:

ep 1: ICAI identification experiment
he main prerequisite for the use of ICAI ID blocks the availability of process signal data sets haracterising the static and dynamic behaviour of e process component to be modelled by means of n ICAI ID block. It is well known that the setup of e identification experiment is a nontrivial task. In n industrial environment operating restrictions ave to be handled in addition. Nevertheless the on-expert industrial user must be provided with imple guidelines for the identification experiment. or the proper application of the ICAI ID blocks the ollowing minimum requirements must be met by e experiment:

- The recorded process input and output signals must cover the full operating range for which the process model is going to be designed.
- The process input and output signals must be changed between a sufficient number of stationary points such that it is possible to determine from the steady state data sets static characteristics and to identify the dynamic behaviour from the transfer behaviour of the signals.
- The experiments can be done in open or closed loop requiring systematic changes of the process input, respectively reference signals, in order to navigate the process to the different stationary points.
- It is assumed that all signal data are recorded using lowpass filtering according to industrial standards and that during the identification experiment neither periodic disturbances nor changing offsets are corrupting the signal data.

The experiment guideline for the industrial user will contain among others the above minimum requirements and in addition the advice to repeat and possibly intensify the identification experiment in case the identification results are unsatisfactory.

Step 2: Request of basic ID block information
After placing and connecting the ICAI ID block within the simulation environment the input and output signals of the ID block must be specified by the user such that signal data from the ICAI identification experiment can be referenced for evaluation. Optionally further characterisations of the block behaviour may be provided by the user like signal limits, nonlinear behaviour, couplings etc.. This additional information is evaluated if available to simplify the internal identification procedure carried out in the next step. In the future the systematic collection and evaluation of user knowledge by means of qualitative modelling methods is anticipated as applied in the *Modelling* approach (Strickrodt, 1997) providing useful a-priori information for the setup of the identification experiment and for the subsequent identification procedure.

Step 3: Model identification in the ICAI ID block
The identification itself is done directly in the ICAI ID block. Making use of robust standard identification methods the identification procedure is carried out mainly with default parameter settings requiring minimum interaction from the inexperienced user. The individual identification actions are described in the following for the different ICAI ID blocks.

2.2. *Static SISO ID block*

The ICAI static SISO ID block determines the static characteristic of a singlevariable process component from the steady state date sets recorded during the ICAI identification experiment. The static characteristic is approximated by default using a modified weighted mean method as default to fit the data (Fig. 2). Optionally also a polynomial or linear fit can be applied. For validation the identified static characteristic is plotted against the steady state data sets. After approval the static characteristic is written into the ICAI ID block, which then can be used like a standard SIMULINK™ block for a nonlinear static model.

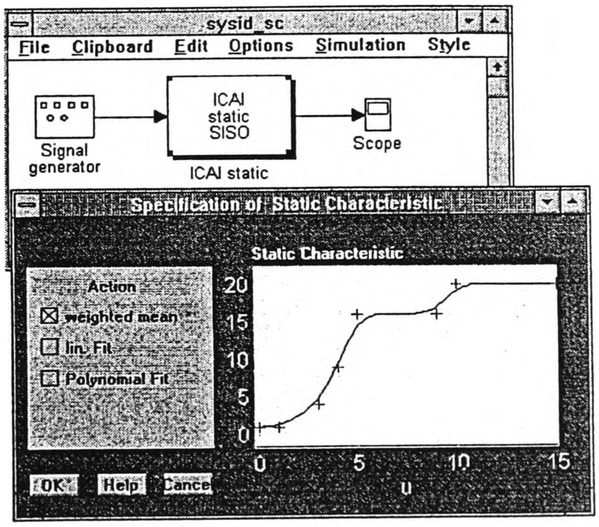


Fig. 2. ICAI nonlinear static SISO ID block.

2.3. *Linear dynamic SISO ID block.*

The ICAI linear dynamic SISO ID block (Fig. 3) identifies a linear dynamic model from signal data that have been recorded during the ICAI identification experiment. The underlying assumption is that the utilised data are not infected by nonlinear influences; i.e. either the process itself does not exhibit nonlinear behaviour or the signal data evaluated must be selected by the user from the neighbourhood of a single operating point.

For the identification procedure in the ID block a novel two-step identification method is applied that has been combined from a standard prefiltering procedure and the MMLS algorithm, the latter developed by Niu *et al.* (1992). This method produces a set of candidate models in an efficient and numerically reliable way. For all generated models the output error is calculated and a favourite model is proposed to the user who can decide about the final model by graphical comparison of model signal data with recorded signal data.

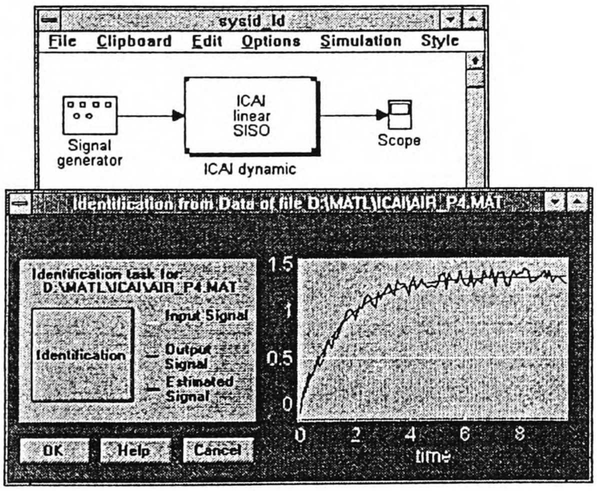


Fig. 3. ICAI linear dynamic SISO ID block.

At the end of the procedure the identified and approved model is written into the ID block which then can be used like a standard SIMULINK™ block for a linear dynamic SISO model (Fig. 4).

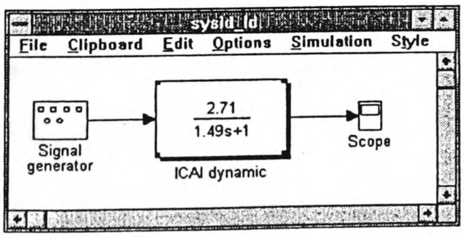


Fig. 4. Identified linear dynamic SISO model.

2.4. Nonlinear dynamic SISO ID block

The ICAI nonlinear SISO block (Fig. 5) allows to identify simplified Hammerstein or Wiener models as illustrated in Fig. 6. For this task process signal data sets from the ICAI identification experiment must be available comprising different steady states (for the static characteristic) and sufficiently excited transfer signals (for the linear dynamic model). The standard two-step procedure applied is to identify the static characteristic first from the steady state data sets and then based on these results the linear dynamic submodel.

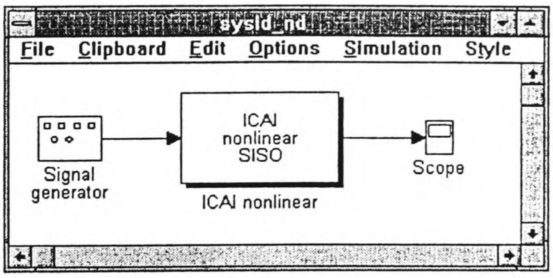


Fig. 5. ICAI nonlinear dynamic SISO ID block.

The identification of the linear dynamic part is handled as in the linear ID block. However, the recorded signal data characterising the behaviour of the complete nonlinear dynamic block are pre-processed in order to compensate for the nonlinear static characteristic already identified. The choice between Wiener and Hammerstein model is done simply by comparing the model quality of the results for the different models.

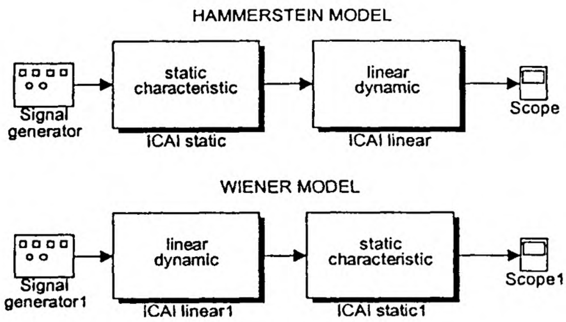


Fig. 6. Wiener and Hammerstein model structure.

2.5. The ICAI MIMO ID Blockset

For nonlinear MIMO black box identification the ICAI MIMO ID blockset is under development that identifies two-input two-output process models (see Fig. 7) with various internal structures. The ICAI MIMO modelling approach is organised according to Fig. 8 and follows in principal the same line as in the SISO case. Consequently the ICAI MIMO ID blockset may comprise a nonlinear static MIMO ID block, a linear dynamic MIMO ID block and a nonlinear dynamic MIMO ID block.

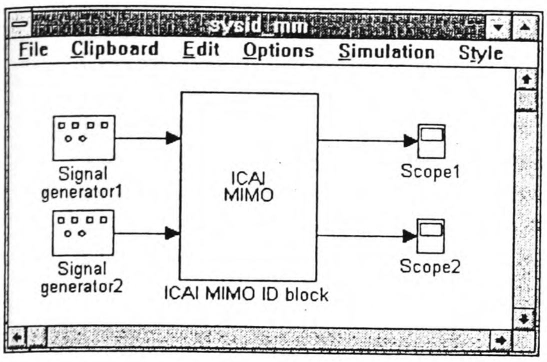


Fig. 7. ICAI MIMO ID block.

In the MIMO case the degree of freedom for the process modelling is much higher than in the SISO case thus complicating the identification procedure drastically. Therefore the systematic collection and evaluation of available user knowledge will play an important role in order to organise the ICAI identification experiment, to preselect appropriate MIMO model structures and to simplify the identification task. In this context the use of the *Modelling* approach is advisable (Strickrodt, 1997) as it allows an extensive qualitative and structured modelling of even nonlinear multivariate process components from the user's knowledge as far as possible providing helpful qualitative information especially about structural model properties.

Furthermore, a default preparatory evaluation of the multivariate static properties by means of the nonlinear static MIMO ID block will provide valuable information not only about the nonlinear effects to be taken into account in general but also about the internal couplings between the different inputs and outputs thus helping to preselect an appropriate MIMO model structure for the identification procedure

If couplings have to be modelled this will be done by multivariate static characteristics and/or by dynamic MISO models for every output resulting possibly in a substructured and complex MIMO model as shown in Fig. 9.

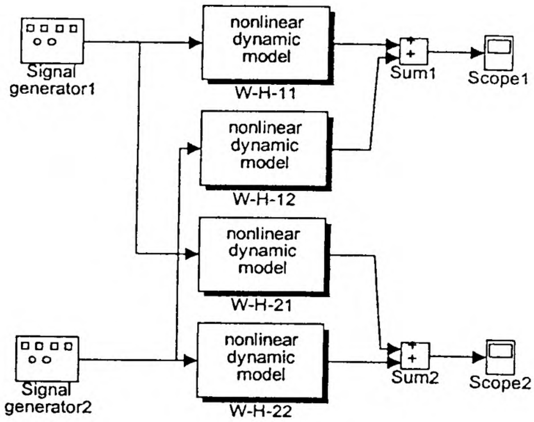


Fig. 9. Nonlinear p-canonic MIMO structure as possible result of ICAI MIMO identification.

At the end of the identification procedure the identified and approved MIMO model showing the selected submodel structure will be written into the MIMO ID block which then can be used like a standard SIMULINK™ block for simulation purposes.

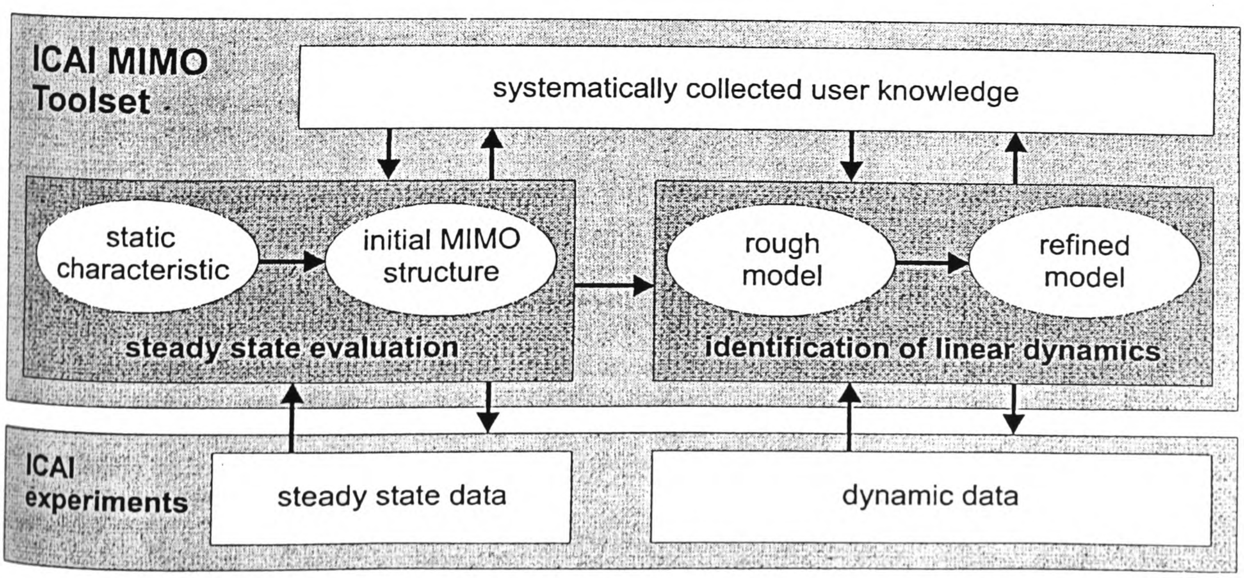


Fig. 8. ICAI MIMO modelling procedure.

3. ICAI TOOLBOX AND ICACSD SCHEME

The ICAI toolbox is part of a general framework for a simplified CAE approach to the design of control systems that can be used by non-expert engineers in industry. The ICACSD (Industrial Computer Aided Control System Design) scheme has been developed to allow the design of PID based control structures for MIMO processes that are as simple as possible and as good as required (see Schumann et al., 1996). A key element of the ICACSD scheme is to model a MIMO process by a restricted set of standard model structures used in a simple model evolution scheme, see Fig. 10. The used models are composed from linear dynamic SISO blocks and, if necessary, nonlinear static SISO blocks possibly combined to Wiener or Hammerstein models. Accordingly the structure of the complementary control systems generated within the ICACSD scheme ranges from linear SISO to nonlinear MIMO depending on the required model and control system complexity, which is necessary to achieve satisfactory control performance.

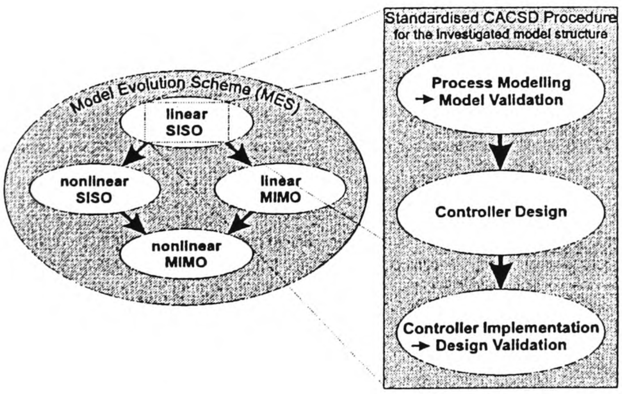


Fig. 10. ICACSD model evolution scheme and standardised CACSD procedure.

The ICAI toolbox supports the identification of all process models required within the ICACSD model evolution scheme from which the complementary control systems are designed. Moreover the ICAI toolbox will allow the simultaneous generation of all ICACSD models possibly avoiding repeated experiments.

One key element is the integration of the identification task into a blockoriented simulation environment. This is achieved with the ICAI toolbox for MATLAB/ SIMULINK™. The ICAI toolbox comprises ID blocks for linear and nonlinear SISO models which allow direct access to identification within SIMULINK™. The handling of the ICAI ID blocks is kept simple by standard identification paths, preselection of standard identification methods with default parametrisations and visual validation of the identification result using a graphical user interface.

The ICAI SISO ID blocks will be complemented in the future by ICAI MIMO ID blocks for MIMO identification supporting a simplified approach also to nonlinear MIMO modelling, which is particularly useful in the context of an industrial computer aided control system design scheme.

In the end the use of the ICAI toolbox may support the identification of simple SISO and complex MIMO process models in the context of control system design in a way such that CACSD becomes feasible even for inexperienced industrial users.

4. CONCLUSIONS

In order to promote the use of identification in industry a simplified approach to identification is required that takes into account the limited expertise and time available in industry for process modelling and control system design.

5. ACKNOWLEDGEMENTS

The research project is carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

6. REFERENCES

Eykhoff, P. and P.C. Parks (1990). Identification and system parameter estimation; where do we stand now?. Automatica, Pergamon Press Ltd, vol. 26, no. 1, pp. 3-5.

Körner, S. and R. Schumann (1996). Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE. *VDI/VDE- GMA Kongreß*, Meß- und Automatisierungstechnik, Baden-Baden, VDI-Bericht Nr. 1282, pp. 93-103.

Ljung, L. (1991). Issues on system identification, *IEEE Control Systems Magazine*, pp. 25-32.

Ljung, L. (1995). System identification toolbox user's guide, *The MathWorks Inc.*

Niu, S., D.G. Fisher and D. Xiao (1992). An augmented UD identification algorithm, *Intern. J. Control*, vol. 56, no. 1, pp. 193-211.

Schumann, R., S. Körner, K.J. Baker and M. Strickrodt (1996). Shaping CACSD for practical use in process industry. *IFAC World Congress*, San Francisco, vol. L, pp. 223-228.

Strickrodt, M. (1997). An integrated knowledge engineering approach to process modelling, doctorate thesis, *University of Glamorgan- Dept. of Mech. and Man. Engineering*, Wales

ICAI

ICAI - A MATLAB TOOLBOX FOR INDUSTRIAL COMPUTER AIDED IDENTIFICATION

Steffen Körner*, Reimar Schumann* and Keith J. Baker**

**Forschungsschwerpunkt REPAM, Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Germany, e-mail: koerner@repam.fh-hannover.de*
FAX: +49-511-9296-111
***Department of Mechanical and Manufacturing Engineering, University of Glamorgan, Pontypridd, Mid Glamorgan, CF37 1DL, United Kingdom*

Short description: The Industrial Computer Aided Identification (ICAI) toolbox is especially aimed at industrial users like process engineers or commissioners who are not necessarily specialists in identification because it was learned from interviews in process industry that no software tool is available supporting especially these users in a simple way. The ICAI toolbox has been programmed in MATLAB™ and integrates the identification functionality into MATLAB™'s blockoriented simulation environment SIMULINK™ through the ICAI identification blocks, which can be handled intuitively like other SIMULINK™ function blocks. Once activated the identification blocks support a guided tour to identification based on the simplified use of advanced identification methods and an ergonomically designed graphical user interface. At the moment SISO ICAI blocks are available for the identification of static characteristics and for the identification of linear and nonlinear dynamic systems.

Another part of this toolbox is under development that allows to identify even nonlinear MIMO process models by making use of a set of standard structures, which are automatically generated utilising the ICAI identification blocks.

Keywords: System Identification, Computer Aided Engineering (CAE), Process Identification, Industry Automation, MIMO, User Interfaces, Interactive Approaches

HARDWARE REQUIREMENTS

Computer Type:	Pentium PC
RAM (MB):	> 16 MB
Harddisk (MB):	> 500 MB
Graphics:	> 1 MB
Others:	

SOFTWARE REQUIREMENTS

Operating System:	Windows 95 or Windows NT 4.0
SW-Environment:	MATLAB™
Others:	Control System Toolbox

Proceedings of the 17th IASTED International Conference
Modelling, Identification and Control
February 18-20, 1998, Grindelwald, Switzerland

REVISITING STEP RESPONSE IDENTIFICATION

STEFFEN KÖRNER and REIMAR SCHUMANN

Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover
e-mail: koerner@repam.fh-hannover.de,
internet: www.repam.fh-hannover.de/automa

ABSTRACT

Step response identification is probably the most widely used method for process identification in industry due to its simple application and the intuitive picture that is provided about the process behaviour as a reaction to an input step change. Also the nature of step responses is of prime concern for the development of several predictive control strategies. In this paper nine methods for step response identification have been revisited and assessed with respect to the models produced, evaluation problems due to noise and computer implementation aspects.

Keywords: identification, step response, process automation

INTRODUCTION

Step response identification has been an active field of research in the 1950th and 1960th [1] and many methods have been developed that allow the determination of transfer functions from step responses. According to Ljung and Glad [2] transient (i.e. step response) analysis is probably the most widely used identification method in industry.

The advantages in short:

- o the process is excited with a simple input signal which may be even applied during normal process operation.
- o couplings can be detected directly in case that more than one output reacts on an input step change.
- o dominating time constants, static gains and to a certain degree also deadtime of the model can be easily detected.
- o based on the step response the process behaviour can be categorised (oscillatory, damped, aperiodic, etc.).
- o A good agreement of the model step response with the measured step response provides an intuitive confidence into the model.

There are also well known disadvantages:

- o many methods allow only the identification of simplified models by definition.
- o practical limits in the input step´s amplitude may cause a poor signal to noise ratio of the step response affecting the model´s accuracy.
- o step input changes excite especially low frequencies; information with respect to higher frequencies is less reliable.
- o during recording of the step response no offset changes or signal trends due to disturbances can be tolerated.

One of the best motivations for applying step response identification methods is the determination of valuable a-priori information for the application of more advanced identification methods. Hence it is worthwhile to revisit step response identification methods, to assess them with respect to the models produced and to discuss the aspects that are required for a successful computer implementation.

method number	category	model structure	noise problems	deadtime part	periodic responses	principle	references
1	table based methods	$G(s)=\frac{K}{(1+T\cdot s)^n}$	-	no	no	CVs: K, T_u,T_g	[3]
2		$G(s)=\frac{K}{(1+T\cdot s)^n}$	o	no	no	CVs: K, $T_{\%}$ (min. 2)	[4]
3		$G(s)=\frac{K}{(1+T\cdot s)\cdot(1+bT\cdot s)^{n-1}}$	o	no	no	CVs: K, $T_{\%}$ (min. 3)	[5]
4		$G(s)=\frac{K}{\prod_{k=1}^{n}(1+\frac{T}{k}\cdot s)^n}$	-	no	no	CVs: K, T_{63}, $y(T_{63}/2)$	[6]
5	numerical methods		+	no	no	0^{th} and 1^{st} momentum	
6		$G(s)=\frac{K}{\prod_{k=1}^{n}(1+\frac{T}{k}\cdot s)^n}\cdot e^{-T_d s}$	+	yes	no	linear regression	[7]
7		$G(s)=\frac{b_0+b_1 s+\ldots+b_m s}{1+a_1 s+\ldots+a_n s}$	+	no	yes	multiple integration	[1] [8]
8		$G(s)=\frac{b_0+b_1 s+\ldots+b_m s}{1+a_1 s+\ldots+a_n s}\cdot e^{-T_d s}$	+	yes	yes	optimi-sation	[4]
9		$G(s)=\frac{b_0+b_1 s+\ldots+b_m s}{1+a_1 s+\ldots+a_n s}$	o	no	yes	least squares	[2]

+: few o: medium -: many

Table 1. Selected methods for step response identification

COMPARISON OF SELECTED METHODS

In this paper nine step response identification methods are regarded, see Table 1. This table comprises four traditional methods which produce models by detecting characteristic values (CV) from the step response which are translated to the model parameters by means of specific tables – this is why these methods are called table based methods. The remaining five identification methods are based on numerical calculations for the determination of the model parameters – these methods are called numerical methods.

The most general model produced is the linear model extended with a deadtime part, i.e.

$$G(s) = \frac{b_0 + b_1 s + \ldots + b_m s}{1 + a_1 s + \ldots + a_n s} \cdot e^{-T_d s} \qquad (1)$$

with model parameters a_i, b_i, model deadtime T_d and model orders m and n. In Table 1 the models identified by the different methods are listed in column 3. The full model (1) is only identifiable with method 8, the methods 7 and 9 allow the detection of the linear part of the model. The other methods 1 to 6 are only able to identify simplified model structure with aperiodic behaviour and simplified structures based on a sequence of first order time lags.

Measured step responses are often corrupted by noise, see Fig. 1. Noise can create evaluation problems especially in the case of table based methods which use only few characteristic values directly taken from the measurements. This is why the table based methods in Table 1 are rated worse than the numerical methods with more or less inherent filtering properties due to the evaluation of numerous data points.

TABLE BASED METHODS

Table based methods are still applied manually in practice. In addition many of these methods are implemented in various industrial controllers and even in process control systems today. The methods 1-4 in Table 1 represent only a selection of the existing variety of table based methods utilising a small number of characteristic values (CVs). A CV common to all table based methods is the static gain K which is directly determined by the ratio of the steady state output change (Δy) to the input step change (Δu) applied, see Fig. 1. Deadtimes cannot be modelled explicitely by the listed methods – either they have to be modelled approximately by an increase of the model order or they have to be detected manually and split from the original step response.

The wellknown method 1 has been developed by *Streijc* [3] yielding the parameters for a n^{th} order model constructed as a sequence of first order time lags with identical time constants. From the step response tangent at the inflection point the time values T_u and T_g are taken as characteristic values (Fig. 1). The crucial task is the construction of the inflection point tangent especially if the data are corrupted by noise.

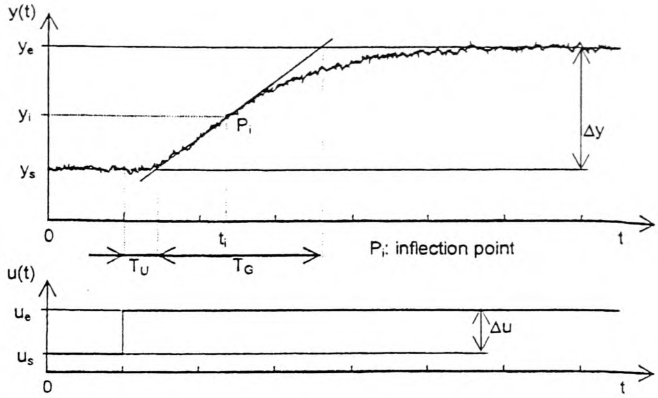


Fig. 1. Method 1 applied to step response data

Method 2 developed by *Schwarze* [4] uses the same model structure. The calculation of the model order n and the time constant T requires at least 2 characteristic time values directly taken from the step response, after which the step response has accomplished a specific percentage of the step response transient, e.g. T_{10} and T_{50} in Fig. 2.

Method 3 was also proposed by *Schwarze* [5] but produces a slightly more detailed model containing a second time constant. Here, however, at least 3 characteristic time values are required, e.g. T_{10}, T_{50} T_{90} in Fig. 2.

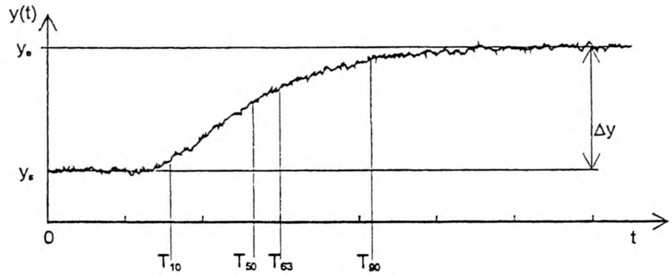


Fig. 2. Characteristic values for a step response

Method 4 uses according to *Radtke* [6] a practically relevant model that is described by a series of first order time lags with time constants developing as a harmonic row (ordered time constants). For the determination of the model order n and the time constant T the characteristic time value T_{63} and the step response value at $T_{63}/2$ are required, see again Fig. 2.

NUMERICAL METHODS

Numerical methods process a complete sequence of the input-output measurement values rather than to rely on few characteristic values thus being less sensitive to noise.

Method 5 was proposed by Radtke [6] as an alternative to produce the model of method 4. Here the model order n and the time constant T is determined by numerical calculation of the *0^{th} and 1^{st} momentum* of the step response.

Method 6 described in [7] uses again the model structure of ordered time constants as methods 4 and 5 but it includes also a separate deadtime part. The determination of the model parameters is done by fitting the mathematical step response function y(t)

$$y(t) = \Delta u \cdot K(1-e^{-(t-T_d)/T})^n \quad \text{for } t > T_d \tag{2}$$

to the step response measurements by a specific procedure including mathematical transformations and a *linear approximation* for different model orders n. By selecting the model with the best fit to the measurement data the model order n is determined, producing automatically the time constant T and deadtime T_d in addition. This method is less sensitive to disturbances if the measurements are taken from the region with a steep slope, practically between T_{10} and T_{80}.

All identification methods up to now are only suitable for the evaluation of aperiodic step responses. Of course identification methods that can handle arbitrary input signals may be used for step response evaluation as well not beeing restricted to aperiodic step responses.

Method 7 proposed by Streijc [8] finds model parameters and order from *repeated integration* of the step response and by minimisation of the difference between the model step response and the measured step response. Although this method is insensitive to high frequency noise due to the smoothing effect of the integration, low frequency inaccuracies are summed up with every integration. Therefore this method is practically only suitable for orders $n \leq 3$.

Method 8 stands for the use of *optimisation techniques* to fit model parameters to step response measurements by minimising the difference between the measured process step response and the simulated model step response. Optimisation is especially suitable if appropriate start values for the model parameters are provided for a local optimisation (fine tuning). Start parameters can be gained for example from the methods mentioned above. By comparing the model fit for different model orders and deadtime values the appropriate model order and deadtime may be determined.

Method 9 stands for the field of *least squares* identification as e.g. described in [4] which in principle can be applied in various different ways to identify linear continuous time models also from step responses. However, the excitation with a the single input step limits in many cases the direct application of least squares estimation due to numerical problems especially when noise is present. With the necessary precautions and sometimes also some numerical tricks, however, it is possible to get acceptable models from step responses.

By applying e.g. a modified MMLS method [9] as described in [10] model orders and parameters can be identified also from step response data with minimum computational effort. It is interesting to see that this method works especially well if the (noise corrupted) step response data are evaluated several times e.g. by using an extended data set including the mirrored step response as shown in Fig. 3.

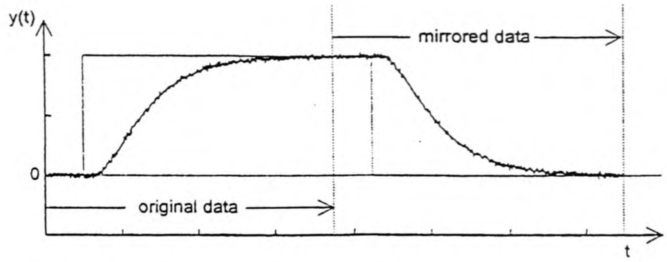


Fig. 3. Extended data set for modified LS

COMPUTATIONAL ASPECTS

There are two principal possibilities to implement the identification procedure in a computer program: An *automated identification procedure* will produce the model directly without user interaction, i.e. the user can only accept or reject the identified model. In an *interactive identification procedure* the user can support and influence the identification procedure also in intermediate states. In every case the identification procedure ends with the validation of the identified model by the user, who has to assess the agreement of the measured and simulated step response either graphically or by means of a quality index.

The computer implementation of all step response identification methods has to take into account that the step response measurements may be corrupted by noise or may contain offsets. Also, for all methods except methods 6 and 8, deadtimes contained in the process must be modelled in a separate procedure before the application of the step response identification method.

NOISE PROBLEMS

Table based methods have been developed for manual detection of the required characteristic values from the recorded step response. The excellent pattern recognition capabilities of human brain allows the determination of these characteristic values also in the presence of noise.

These human filtering capabilities can be used in an interactive implementation of the table based identification methods thus allowing e.g. a proper manual construction of the inflection point tangent for method 1 or the determination of other characteristic values even from noise corrupted step responses. For an automated implementation of table based methods however it is crucial to apply appropriate prefiltering methods in the presence of noise otherwise only poor characteristic values will be detected from the step response data resulting in poor models.

Numerical methods process a sequence of the input-output measurement values rather than to rely on few characteristic values thus becoming less sensitive to noise in principle. Nevertheless appropriate prefiltering of the step response data or the calculation of the mean step response from repeated experiments (carried out under the same conditions) will also improve the identification results of these methods in the presence of noise.

DEAD TIME DETERMINATION

It is always advantageous to determine process deadtimes directly from physical considerations (transport delays,...) and to model them in a separate deadtime model part. Only the methods 6 and 8 allow an automatic determination of the model deadtime. All other methods cannot handle deadtimes directly such that a separate procedure for the detection and separation of the deadtime part must be applied. It is advisable to leave this task to the user in an interactive implementation.

OFFSET CANCELLATION

All step response identification methods require the determination of the steady state values at least of the output and the cancellation of the initial offset. Therefore a good offset cancellation must be a prime concern also for step response identification. For an interactive identification procedure the user may be asked to specify the offset manually. For an automated identification procedure the mean of the output signal may be calculated before applying the input step change and after the settling of the output.

CONCLUSION

Table based step response identification methods are based on the proper detection of few characteristic values and can easily produce simplified models with limited model complexity and accuracy. They are especially suited for the manual evaluation of step responses. However, the implementation of these table based methods requires a reliable detection procedure for the characteristic values which may cause problems in the presence of noise. Numerical methods can evaluate the complete step response data set and can therefore produce also more complex models, however, the accuracy of the identified models depends on the quality of the step response data sets which in the presence of noise should be improved by proper application of prefiltering techniques too.

ACKNOWLEDGEMENTS

The research project is carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

REFERENCES

[1] H. Rake, Step response and frequency response methods, Automatica, 16, 1980, 519-526

[2] L. Ljung and T. Glad, Modeling of dynamic systems, Prentice Hall, Englewool Cliff, NJ, ISBN-0-13-597097-0, 1994

[3] V. Strejc, Nährungsverfahren für aperiodische Übergangscharakteristiken, *Regelungstechnik*, 7, 1959, 124-128

[4] R. Isermann, Identifikation dynamischer Systeme 1, *Springer-Verlag, 2nd, no. 1*, ISBN-3-540-54924,1992

[5] G. Schwarze, Algorithmische Bestimmung der Ordnung und Zeitkonstanten bei P-, I- und D-Gliedern mit zwei unterschiedlichen Zeitkonstanten und Verzögerung bis 6. Ordnung, *Messen Steuern Regeln*,1964, 7, 10-19

[6] M. Radtke, Zur Approximation linearer aperiodischer Übergangsfunktionen, *Messen Steuern Regeln*, 9(6), 1966, 192-196

[7] M. Hankel and A. Reiner, Approximation der Sprungantwort von linearen aperiodischen Strecken, *at – Automatisierungtechnik*, R. Oldenbourg Verlag, 40(7), 1992, 263-266

[8] V. Strejc, Auswertung der dynamischen Eigenschaften von Regelstrecken bei gemessenen Ein- und Ausgangssignalen allgemeiner Art, *Messen Steuern Regeln*,1960, 3, 7-11

[9] S. Niu and D. G. Fisher, A multiple model least-squares estimation method, *American Control Conference*, Baltimore, 1994, 2231-2235

[10] S. Körner and R. Schumann, Robustified estimation of multiple order models, *IASTED Conference on Modelling, Identification and Control, Grindelwald*, Switzerland, 1998, to appear

Proceedings of the 17th IASTED International Conference
Modelling, Identification and Control
February 18-20, 1998, Grindelwald, Switzerland

ROBUSTIFIED ESTIMATION OF MULTIPLE ORDER MODELS

STEFFEN KÖRNER and REIMAR SCHUMANN

Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover
e-mail: koerner@repam.fh-hannover.de,
internet: www.repam.fh-hannover.de/automa

ABSTRACT

One important aspect in developing new identification techniques for industrial users must be the simple and robust applicability. The paper proposes the combination of the MMLS method with correlation techniques to the CorMMLS method thus allowing unbiased estimates also in the presence of arbitrary stationary disturbances with zero mean. The CorMMLS method can be used as base for solving practical problems that make identification difficult for inexperienced users. Solutions are outlined for the determination of the model structure, for the detection of deadtimes and for the automatic compensation of offsets.

Keywords: identification, least squares method, multiple order models, correlation

INTRODUCTION

Nowadays several approaches to process modelling are available, but since the development of conventional theoretical models proved to be time consuming and expensive the advantages of experimental identification methods for already existing processes are obvious. Although the identified models are valid only in the analysed operating range and do not necessarily comprise physically relevant parameters they are in most cases appropriate for control design. The least-squares (LS) method originally developed by Gauss in 1809 is widely used as a basic identification tool for parameter estimation, especially for discrete time models (e.g. [1], [2]).

Nevertheless the direct implementation of this method suffers from various numerical problems e.g. if the input signal excitation is small, the computer accuracy is low or the model is overparameterised. Numerical robustness can be improved by the application of square root filtering techniques (see e.g. [3]). Particularly the QR decomposition method represents a good compromise between numerical performance and algorithmic complexity. Furthermore the LS estimator has been improved introducing order recursive structures, which allow the parallel estimation of models from order 1 to n (where n is the maximum order regarded) with minimum computational load. A combination of square root filtering using the QR decomposition with an order recursive estimation scheme has been proposed in [4] and called the multiple model least square (MMLS) estimation method.

To gain better properties for LS estimation in the presence of noise many modifications have been proposed like the generalised LS (GLS), the extended LS (ELS) or the instrumental variable (IV) method, all of which can be easily incorporated into the MMLS estimator in order to minimise the estimation bias (see e.g. [2]). Another way to reduce the bias caused by noise is the use of correlation techniques in a two-step procedure as proposed by Isermann [3]. In the first step a nonparametric model in form of correlation functions is identified without the need for initial guesses concerning model structure and deadtime. In the second step the parametric model is identified from the correlation functions.

In this paper the combination of correlation techniques with the MMLS method is proposed. The resulting two-step method is called CorMMLS (correlation multiple model least square) that yields unbiased estimates also in the presence of coloured noise.

Modifications and extensions of the CorMMLS method are outlined to solve practical application problems with respect to identification in closed loop, model order and deadtime detection and offset cancellation.

THE MMLS ESTIMATION METHOD

This section briefly reviews the basic properties of the MMLS estimation method [4]. Assume that the process behaviour under investigation can be described by the following linear discrete time difference equation:

$$\begin{aligned} &y(t) + a_1 y(t-1) + \ldots + a_n y(t-n) \\ &= b_1 u(t-d-1) + \ldots + b_n u(t-d-n) + n(t) \end{aligned} \qquad (1)$$

Here y(t) and u(t) represent the process output and input signals respectively, n(t) is a coloured noise signal with zero mean, a_i, b_i (for i = 1,...,n) are the model parameters and d is the discrete deadtime. For a simplified description of the MMLS method the deadtime is assumed to be zero (d=0). Then the augmented data vector can be constructed from the measurement data with the inputs and outputs paired together as follows:

$$\psi(t) = [-y(t-n), u(t-n), \ldots, -y(t-1), u(t-1), -y(t)]^T \qquad (2)$$

Obviously this vector differs from the conventional LS data vector with its grouped elements {y(·),u(·)} and the inclusion of the current process output y(t). This special structure is the basis of the MMLS approach allowing its order recursive nature, which means that the calculation of models starts with the lowest order model and continues by calculating the higher order models successively. The augmented data vectors are grouped to a matrix to which the QR decomposition is applied as follows

$$\begin{bmatrix} \psi^T(1) \\ \psi^T(2) \\ \vdots \\ \psi^T(t) \end{bmatrix} = \mathbf{QR} = \mathbf{QD}_c\mathbf{U}_c \qquad (3)$$

with Q as an orthogonal matrix and **R** as an upper triangular matrix. The diagonal matrix $\mathbf{D}_c$ and the unit upper triangular matrix $\mathbf{U}_c$ are obtained by further decomposition of **R** utilising the Housholder transformation as outlined in [5]. Applying finally

$$\begin{cases} \mathbf{U}(t) = \mathbf{U}_c^{-1} \\ \mathbf{D}(t) = \mathbf{D}_c^2 \end{cases} \qquad (4)$$

the upper triangular parameter matrix U(t) contains all parameters of the models from order 1 to n shown in (equ. 5). The matrix **D**(t) contains the corresponding loss functions for the estimated models.

$$\begin{gathered} y(t) + \hat{a}_1 y(t-1) = \hat{b}_1 u(t-1) \\ y(t) + \hat{a}_1 y(t-1) + \hat{a}_2 y(t-2) = \hat{b}_1 u(t-1) + \hat{b}_2 u(t-2) \\ \vdots \\ y(t) + \hat{a}_1 y(t-1) + \ldots + \hat{a}_n y(t-n) = \hat{b}_1 u(t-1) + \ldots + \hat{b}_r u(t-n) \end{gathered} \qquad (5)$$

The computational effort needed for the calculation of all models including the corresponding loss functions is approximately the same as needed for the calculation of a single model of the highest (i.e. the n^{th}) order in the traditional way .

THE CORRELATION MMLS METHOD

The estimates of LS methods are generally biased if the noise signal n is not white noise. In order to get an unbiased estimate for general zero mean disturbance signals the MMLS method can be combined with correlation techniques. The following considerations are restricted to open loop measurement data with zero mean coloured noise n which is statistically independent from the input test signal u.

With a stationary (pseudo-)random input test signal the following correlation functions can be defined

$$\phi_{uy}(\tau) = \lim_{N\to\infty} \frac{1}{N+1} \sum_{t=0}^{N} u(t-\tau)\, y(t) \qquad (6)$$

$$\phi_{uu}(\tau) = \lim_{N\to\infty} \frac{1}{N+1} \sum_{t=0}^{N} u(t-\tau)\, u(t) \qquad (7)$$

with τ being discrete.

For a finite number of samples the correlation functions can be estimated by:

$$\hat{\phi}_{uy}(\tau) = \frac{1}{N+1} \sum_{t=0}^{N} u(t-\tau)\, y(t) \qquad (8)$$

$$\hat{\phi}_{uu}(\tau) = \frac{1}{N+1} \sum_{t=0}^{N} u(t-\tau)\, u(t) \qquad (9)$$

If the difference equation of the process model (equ. 1) is multiplied by u(t-τ) and summed up over time the following equation for the correlation function estimates arises:

$$\begin{aligned} &\hat{\phi}_{uy}(\tau) + a_1\hat{\phi}_{uy}(\tau-1) + \ldots + a_n\hat{\phi}_{uy}(\tau-n) \\ &= b_1\hat{\phi}_{uu}(\tau-d-1) + \ldots + b_n\hat{\phi}_{uu}(\tau-d-n) + \hat{\phi}_{un}(\tau) \end{aligned} \qquad (10)$$

where the last term represents the estimated correlation of input signal and noise. According to the assumption that the input signal u is independent from the noise signal n the following holds:

$$\lim_{N\to\infty} \hat{\phi}_{un}(\tau) = 0 \qquad (11)$$

This means that the influence of the corresponding estimated crosscorrelation function in (equ. 10) disappears with increasing N. This effect is shown in the simulation example, see table 1.

It is now straigthforward to determine the model parameters from (equ. 10) using the MMLS method. For this purpose a modified augmented data vector can be constructed as follows:

$$\begin{aligned} \varphi(\tau) = [&-\hat{\phi}_{uy}(\tau-n), \hat{\phi}_{uu}(\tau-n), \\ &\ldots, -\hat{\phi}_{uy}(\tau-1), \hat{\phi}_{uu}(\tau-1), -\hat{\phi}_{uy}(\tau)]^T \end{aligned} \qquad (12)$$

This vector corresponds to the augmented data vector as outlined above in (equ. 2) such that the MMLS method can be applied directly utilising correlation functions instead of measurement data. In this context correlation can be interpreted as a special prefiltering technique that removes the influence of zero mean disturbances.

The information necessary for the estimation procedure is contained in those crosscorrelation values that are significantly different from zero. The selection of relevant correlation function values reduces the number of data for the estimation. Due to the distinct shape of the crosscorrelation functions as shown in Fig. 1 two limits P and M (with M ≥ τ ≥ -P) may be determined that are valid for the crosscorrelation and autocorrelation function determining the relevant data for the estimation.

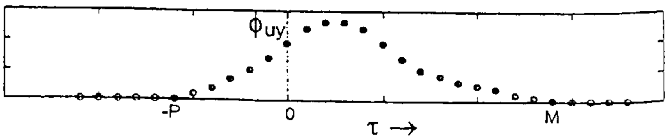


Fig. 1 Limits P and M for crosscorrelation function

MODIFICATIONS AND EXTENSIONS

The previous chapter has introduced the CorMMLS method for the case of open loop identification with a zero mean noise signal that is statistically independent from the input signal. Furthermore for simplicity the discrete deadtime d has been assumed to be zero. In practical applications, however, processes with deadtimes have to be identified in closed loop frequently. Furthermore the measured input and output signals may contain offsets which have to be cancelled before the application of the estimation algorithm. In the following some solutions for the described problems are outlined which can ease the practical application of the CorMMLS method considerably.

CLOSED LOOP IDENTIFICATION

It can be shown that the CorMMLS method can be applied with similar results also for closed loop identification if an additional (pseudo) random test signal u_T (which is statistically independent from n) is fed to the process input in addition to the controller output. If the correlation functions are built in the closed loop case using u_T instead of u the same properties hold with respect to noise elimination as discussed for the open loop case.

MODEL ORDER AND DEADTIME ESTIMATION

In the original version the MMLS method allows the parallel estimation of models of different orders. The scheme, however, can in principle be extended also to calculate in parallel models with different orders and different deadtimes including the corresponding loss functions. These can be used as base not only for an automated detection of the appropriate model order but also for the selection of a proper deadtime by selecting a good compromise between model complexity and loss either manually or by an automated procedure.

OFFSET CANCELLATION

In practice it is wise to care for the offsets and steady state values of the measured absolute input and output signals implicitly. These are defined as

$$Y(t) = y(t) + Y_{00} \quad and \quad U(t) = u(t) + U_{00} \qquad (13)$$

A practical way to get rid of the steady state values is to calculate the first difference of the measured signals

$$\begin{aligned} \Delta Y(t) &= Y(t) - Y(t-1) \\ &= [y(t) + Y_{00}] - [y(t-1) + Y_{00}] \qquad (14) \\ &= \Delta y(t) \end{aligned}$$

and to use the first differences of input and output signal for the identification procedure. This is possible as the model (equ. 1) holds also for the first differences and for the corresponding correlation functions.

SIMULATION EXAMPLE

In order to illustrate the behaviour of the CorMMLS method in the presence of coloured noise simulation studies were done comparing the performance with the original MMLS in open loop. The oscillating process of 2nd order is defined as

$$y(t) - 1.5y(t-1) + 0.7y(t-2) = u(t-1) + 0.5u(t-2)$$

with a sampletime of T_0 = 2s and different experiment durations. The disturbance is a coloured noise with a noise to signal ratio of 0.2. Selected simulation results are shown in table 1. The order was determined automatically for both methods resulting in the true model order.

method	*N*	$\hat{a}_1$	$\hat{a}_2$	$\hat{b}_1$	$\hat{b}_2$
MMLS	500	-1.442	0.652	1.064	0.557
CorMMLS	500	-1.503	0.701	1.088	0.398
MMLS	1500	-1.448	0.657	0.995	0.600
CorMMLS	1500	-1.506	0.707	1.003	0.496
model parameters		-1.5	0.7	1.0	0.5

Table 1. Comparison of MMLS and CorMMLS in the presence of noise

It can be seen that the CorMMLS has no problems to cope with coloured noise if the data sequence is sufficiently large.

CONCLUSION

The combination of correlation technique with the MMLS method results in a two-step method with good numerical properties for the estimation of parametric discrete time models. The concurrent estimation of multiple models makes this algorithm especially appealing if order and deadtime of the model are unknown. It has been shown that the MMLS method is numerically more reliable, more efficient and never worse than the LS estimates [4]. The presented CorMMLS method leads to unbiased estimates also in the presence of a zero mean noise signal as long as the input test signal applied in open or closed loop is statistically independent of the noise. The CorMMLS method has been successfully integrated into a software-tool for industrial computer aided identification that integrates the identification task into a blockoriented simulation environment in an userfriendly way [6].

ACKNOWLEDGEMENTS

The research project is carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

REFERENCES

[1] L. Ljung, System identification; Theory for the user, Prentice Hall, Englewool Cliff, NJ, 1987

[2] T. Söderström and P. Stoica, System identification, Prentice Hall International (UK) Ltd, 1989, ISBN-0-13-881236-5

[3] R. Isermann, Identifikation dynamischer Systeme 2, *Springer-Verlag, 2nd*, ISBN-3-540-54924,1992

[4] S. Niu and D. G. Fisher, A multiple model least-squares estimation method, *American Control Conference*, Baltimore, 1994, 2231-2235

[5] S. Niu, L. Ljung and Å. Björck, Decomposition methods for solving least-squares parameter estimation, *IEEE Transactions on Signal Processing*, 1996, 44(11), 2847-2852

[6] S. Körner and R. Schumann, A New Framework for Computer Aided Identifcation in Process Industry, *IASTED Conference on Modelling, Identification and Control*, 1998, to appear

Proceedings of the 17th IASTED International Conference
Modelling, Identification and Control
February 18-20, 1998, Grindelwald, Switzerland

A NEW FRAMEWORK FOR COMPUTER AIDED IDENTIFCATION IN THE PROCESS INDUSTRY

STEFFEN KÖRNER and REIMAR SCHUMANN

Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover
e-mail: koerner@repam.fh-hannover.de,
internet: www.repam.fh-hannover.de/automa

ABSTRACT

Good process models are an essential prerequisite for the use of modern control strategies in the process industry. In this paper a new framework for system identification aimed at industrial users is presented. A prototype realisation has been developed as a SIMULINK™ blockset integrating identification into the blockoriented simulation environment. It utilises advanced identification methods taking into account a-priori knowledge in a sensible way. Thus a quick and useable identification procedure based on process data is made possible even for those users who are inexperienced in identification.

Keywords: system identification, CACSD, process automation, user orientation

COMPUTER AIDED IDENTIFICATION

Worldwide competition in process industry requires processes which are made more profitable by improving quality, increasing throughput, decreasing maintenance effort and other operating costs while maximising profits. Moreover environmental problems demand the best possible use of resources and the minimisation of waste.

Conventional standard controllers being widely used in process industry cannot meet these demands in all cases because of increasingly complex process designs that mostly exhibit nonlinear behaviour and couplings. It has been shown in many practical applications that modern process control strategies are capable to tackle even very complex and difficult control tasks (see e.g. [1]). Nevertheless the crucial prerequisite for such advanced control strategies is a suitable process model. A practically oriented way to yield a process model from an operating process is the transformation of measurement data into a mathematical process model through identification.

Available tools for process identification have been frequently developed as part of CACSD (computer aided control system design) systems. Their application is normally aimed at experts and relies on the experience and cleverness of the user. Many tools have been programmed in MATLAB™, a numerical computation software that has established itself as a standard base for CACSD. MATLAB™ is used particularly at universities and in R&D departments, but it's use is rather restricted in most areas of process industry, where control is mostly done by commissioning engineers not being able to handle complex tools [2,3]. Consequently the need for the development of a MATLAB™ toolbox in form of a SIMULINK™ blockset was detected that is especially aimed at industrial users offering an intelligent user-friendly graphical user interface (GUI) and providing sensible access to identification.

THE ICAI TOOLBOX

A new toolbox called ICAI (Industrial Computer Aided Identification) has been developed as a SIMULINK™ blockset (see Fig. 1), thus integrating the identification task into MATLAB™'s blockoriented simulation environment. This allows the user to solve modelling, identification and control design including simulation in one single graphical environment without the arduous need to convert models and data according to the needs of separate programs or to use MATLAB™'s command line. The formerly presented ICAI toolbox [4] was completely revised to include GUI standards, different user profiles and improved data storage and retrieval. The ICAI blockset contains ICAI ID (identification) blocks which are simple in use providing a guided tour to identification and making use of a unified GUI that controls the standardised identification procedure [3] for each block. The *ICAI project* described below sets the frame for this task.

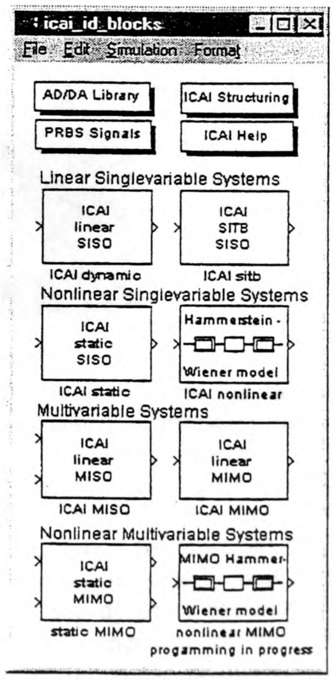


Fig. 1. ICAI ID blocks

ICAI PROJECT

The *ICAI project* settings (Fig. 2) must be specified for each block diagram comprising ICAI ID block(s). The project settings require information about the user profile, process type and the aim of the project, information that is used by ICAI or by the complementary control system design blockset currently under development. The password protected *user profile* controls the complexity of the graphical user interfaces of the ICAI ID blocks for the user levels *process personnel*, *area engineer* and *control expert*. This way a control expert will get more degrees of freedom and more sophisticated presentations of the identification results than process personnel, in order not to confuse the inexperienced user. At any time it is possible to change the project settings with the appropriate password. This enables for example the control expert to access and make use of the data and results that have been gained by process personnel. The *process specification by domain and type* allows the use of process model libraries if available. A sensible library approach including a concept of granularity that can be invoked by ICAI has been described in [5].

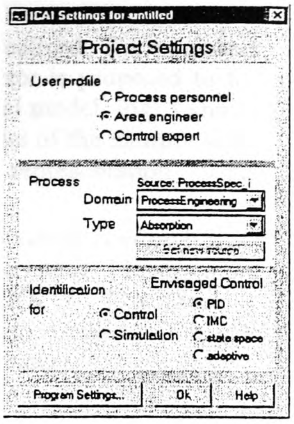


Fig. 2 ICAI Project

COMMON PROPERTIES OF ICAI ID BLOCKS

The necessary steps for an identification with ICAI are shown in *Fig. 3*. It starts with a double-click on an ICAI ID block. If the *ICAI project* has been already specified the user can directly start with the identification task otherwise the user has to specify the *ICAI project* first. Then the measurement data have to be loaded, which must be gained from suitable experiments following a special recipe as outlined in [4]. The ICAI GUI then offers context sensitive and user specific functionality for every step of the identification procedure. Relevant a-priori knowledge can be incorporated where available and used to structure and parameterise the identification procedure designed to meet the industrial users needs. Only few reliable identification methods are offered with extensive pre-parameterisation, such that the user has to deal with only few transparent design decisions. The validation of the results is done by graphical comparison of the simulated data from the identified model against the measurement data. If the user accepts the identification result the ICAI ID block changes its appearance (colour and label) as an indication that this block is finished and ready for simulation. This final model block is called ICAI model block, which can be copied into the ICAI library [8] that allows a quick retrieval of identified models.

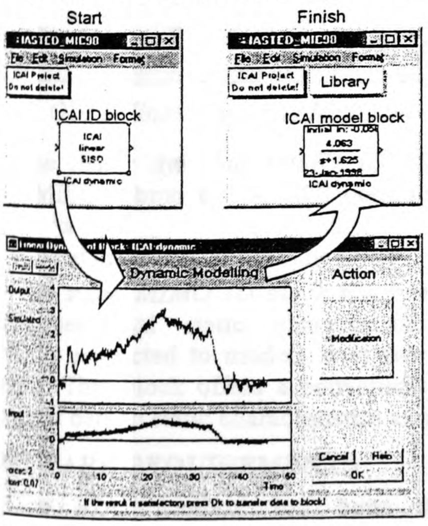


Fig. 3 ICAI sequence

THE IDENTIFICATION BLOCKSET

ICAI ID blocks represent just another type of SIMULINK™ blocks. The ICAI ID blockset shown in Fig. 1 comprises different blocks representing black boxes that support the modelling of unknown parts of the process model from measurement data.

ICAI SISO ID BLOCKS

The ICAI SISO ID blocks have been designed for the identification of linear and nonlinear SISO (single-input single-output) process models.

The *linear dynamic SISO ID block* supports the modelling of linear dynamic processes. For the identification procedure a two step identification method has been applied that combines correlation with multiple model least squares (MMLS) estimation as described in [6]. This block is able to handle signal offsets and drifts, to detect a deadtime and to check the identification results against a-priori knowledge.

The *static SISO ID block* generates a static nonlinear function from steady state data by interpolation or approximation. The steady state data can be directly loaded from a file or by graphical input. It is also possible to extract the steady state data from dynamic measurements automatically or manually provided that steady states are included in the measurements. A modified weighted mean method is used as default to fit the data.

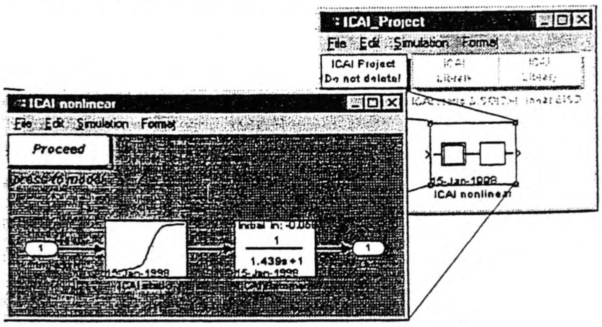


Fig. 4. Hammerstein submodel

The *nonlinear SISO ID block* supports the identification of simplified Wiener and Hammerstein models being suitable to represent numerous processes. The models are combined from a static and a linear ICAI ID block. The standard identification procedure is to model the static characteristic first utilising the functions of the static SISO ID block. Then the linear dynamic part of the model is identified based on the dynamic measurement data preprocessed in order to compensate for the static nonlinearity. If an inversion of the static characteristic is impossible only a Hammerstein model is identified.

Otherwise also a Wiener model is realised. The nonlinear model with the minimum output error is proposed to the user who has to decide about the final model. Fig. 4 shows an example for the resulting submodel of the identification procedure and the change in the representation of the successfully identified block.

The *ICAI SITB (System Identification Toolbox) SISO ID block* allows to access the System Identification Toolbox [7] from within SIMULINK™. This block is only accessible for control experts because it needs expertise to handle the advanced features of the system identification toolbox properly.

ICAI MIMO ID BLOCKS

The ICAI MIMO ID blocks have been designed for the identification of linear and nonlinear MIMO (multiple-input multiple-output) process models. The functionality corresponds to that of the ICAI SISO ID blocks. Standardised design paths are worked through such that also inexperienced users can succeed even in nonlinear MIMO identification.

The *linear dynamic MISO (multiple-input single-output) ID block* follows a similar procedure as the linear dynamic SISO ID block but it can handle in addition an arbitrary number of input signals. Fig. 5 illustrates an example for the identification of a three-input one-output model. The final identification result is a MISO model composed from SISO transfer functions.

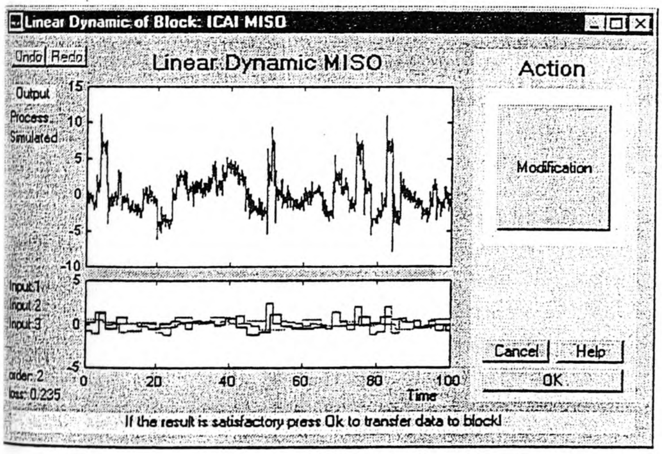


Fig. 5. GUI of linear dynamic MISO block

The *linear dynamic MIMO ID block* is composed from MISO submodels. A GUI controls as many MISO ID blocks as outputs are selected. These MISO ID blocks are worked through one after the other.

The *static MIMO ID block* has been designed to fit multidimensional static characteristics. Its GUI is currently restricted to models with two inputs and two outputs. This block offers a convenient way to fit measurement data to static characteristics as shown in Fig. 6.

SOFTWARE REQUIREMENTS

The ICAI toolbox has been developed under MATLAB 5.1 / SIMULINK 2.1. To run ICAI the signal processing toolbox and the control system toolbox are required, furthermore the system identification toolbox is needed for the ICAI SITB block.

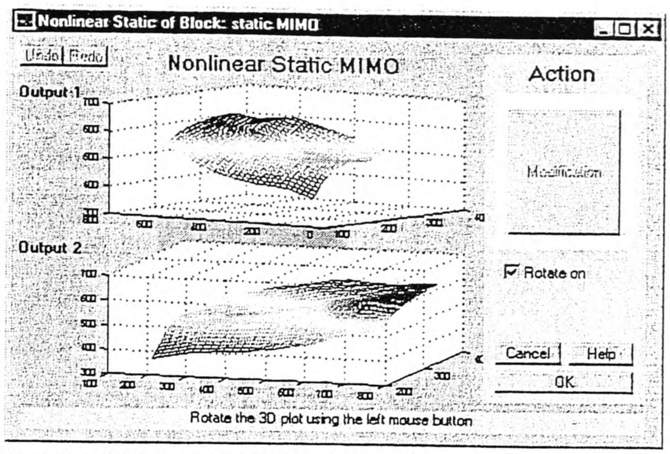


Fig. 6. GUI of static MIMO ID block

CONCLUSION

The ICAI blockset has been proposed as a new framework for identification in the process industry. It is integrated into SIMULINK™ and can be used even by inexperienced users who are capable to use this blockoriented simulation environment. Providing different access levels in the user profile standardised paths to the identification of single- and multivariable processes are offered utilising GUIs that are adapted to the users' knowledge and capabilities. A more detailed description of the ICAI toolbox with many figures and different presentations can be found in the internet [8].

ACKNOWLEDGEMENTS

The research project is carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

REFERENCES

[1] P. J.Gawthrop and J. W. Ponton, Improved control using dynamic process models, *TransIChemE*, 71(A1),1996, 63-70.
[2] V. Hahn.and G. Nöth, Anforderungen an regelungstechnisches CAE in der chemischen Verfahrenstechnik, *VDI/VDE-GMA Jahrbuch*, 1997, 260-280.
[3] S. Körner and R. Schumann, Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE, *VDI/VDE- GMA Kongreß - Meß- und Automatisierungstechnik*, Baden-Baden, 1996, 93-103
[4] S. Körner and R. Schumann, ICAI – A MATLAB™ toolbox for Computer Aided Identification, *IFAC Symposium on System Identification*, Kitakyushu, Japan, 1997, 1611-1616
[5] O. Ravn and M. Szymkat, Modelling and calibration with mechatronic blockset for SIMULINK, *11th IFAC Symposium on System Identification*, Kitakyushu, Japan, 1997, 1643-1648
[6] S. Körner and R. Schumann, Robustified estimation of multiple order models, *IASTED Conference on Modelling, Identification and Control, Grindelwald*, Switzerland, 1998, to appear
[7] L. Ljung, System Identification Toolbox User's Guide, (The MathWorks Inc., 1995), user's guide
[8] S. Körner: ICAI – Industrial Computer Aided Identification – A new MATLAB toolbox for industrial users, Online, *Internet*, 1997, http://www.repam.fh-hannover.de/automa/ICAIHelp/Intro.html

Industrietaugliche Identifikation mit ICAI

S. Körner, Hannover

Zusammenfassung: In diesem Beitrag wird das Konzept eines industrietauglichen rechnergestützten Identifikationswerkzeugs erläutert, das die Erzeugung mathematischer Prozeßmodelle aus experimentell gewonnenen Prozeßdaten mit vertretbarem Aufwand ermöglicht. Die Industrial Computer Aided Identification (ICAI) Toolbox wurde exemplarisch als eine neue SIMULINK™ Bibliothek zur Unterstützung der Identifkationsaufgabe unter MATLAB 5.1/ SIMULINK 2.1™ entwickelt. Die wesentlichen Elemente sollten aber prinzipiell in jede blockorientierte Simulationsumgebung integriert werden können. Ein wesentlicher Ansatz der ICAI Toolbox ist die Bereitstellung von drei Benutzerebenen, die je nach Wissensstand der Anwender vergeben werden können. Somit können sowohl Experten als auch unerfahrene Anwender bei der Identifikation von (Teil-)Modellen innerhalb einer blockorientierten Simulationsumgebung zielgerichtet unterstützt werden.

Schlüsselworte: RT-CAE, CACE, Identifikation, Industrieorientierung

EINLEITUNG

In der chemischen Verfahrenstechnik vertraut man immer noch darauf, daß PID-Regler bei gleichermaßen einfacher wie auch durchsichtiger Funktionalität für mindestens 95% aller Anwendungsfälle ausreichend seien. Nach Hahn und Nöth [1] führt diese Praxis in der chemischen Verfahrenstechnik jedoch mitunter zu so schlecht eingestellten Reglern, daß Anlagenfahrer bei Störungen gezwungen werden, die Regelung in Handbetrieb zu nehmen - ein Zustand, der zu Lasten der Produktqualität geht, wertvolle Arbeitskraft kosten und zu unsicheren Betriebszuständen führen kann. Zudem werden verfahrenstechnische Prozesse immer weiter vermascht, um mit einem Minimum an Ressourcen arbeiten zu können. Da aber aufgrund der Komplexität vermaschter Prozesse eine Handregelung nur schlecht möglich ist, empfiehlt sich der Einsatz regelungstechnischer CAE (Computer Aided Engineering) Systeme, die die Auslegung komplexer Regelsysteme unterstützen (Bild 1).

Notwendige Voraussetzung zur Anwendung moderner regelungstechnischer CAE-Systeme ist ein Prozeßmodell, das das dynamische Verhalten des Prozesses in dem betriebsrelevanten Arbeitsbereich beschreibt. Dabei wird das Modell um so komplexer sein, je größer der Arbeitsbereich gewählt wird, weil dann verstärkt nichtlineare und Kopplungseffekte auftreten können, die bei der Reglersynthese berücksichtigt werden müssen. Da in diesen Fällen lineare PID-Einzelregler nicht mehr ausreichen und eine Auslegung nichtlinearer vermaschter Regelsysteme auf heuristischem Wege wenig praktikabel ist, kann die Reglersynthese systematisch nur mit regelungstechnischer CAE-Unterstützung erfolgen. Bild 1 veranschaulicht die iterative regelungstechnische Entwurfsprozedur von der Prozeßmodellierung über die Reglersynthese bis hin zur Reglererprobung. Erst nach erfolgreichem Test im Modellregelkreis wird die Reglererprobung an der Industrieanlage durchgeführt. Falls die Performance der geregelten Anlage nicht zufriedenstellend sein sollte, muß entweder die Phase der Reglersynthese oder sogar die der Prozeßmodellierung wiederholt werden. Es ist offensichtlich, daß eine vernünftige regelungstechnische CAE-Unterstützung nur auf Grundlage einer geeigneten Prozeßmodellierung erfolgen kann.

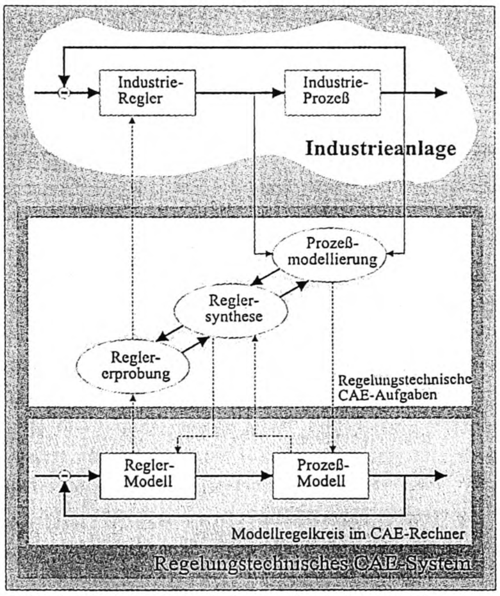


Bild 1: Regelungstechnische Optimierung einer Industrieanlage

PROZEßMODELLIERUNG

Bei der Prozeßmodellierung innerhalb einer blockorientierten Simulationsumgebung (wie z.B. SIMULINK™) wird erst ein Blockdiagramm erstellt, in dem verschiedene lineare und nichtlineare Funktionsblöcke miteinander verknüpft werden. Die Funktionsblöcke können aus verschiedenen Blockbibliotheken ausgewählt werden, die mitunter individuell erstellt werden und mehr oder weniger komplexe Elemente für einen bestimmten Funktions- oder Anwendungsbereich enthalten können. Je umfangreicher die Blockdiagramme werden, um so sinnvoller ist die Bildung von Teil- bzw. Makromodellen, was in verschiedenen Modellierungsebenen erfolgen kann und die Übersichtlichkeit das Blockdiagramms fördert.

Oftmals sind aber nicht alle Teile eines Prozeßmodells über Bibliotheken verfügbar oder aber die Parametrierung bereitet Schwierigkeiten. Dann bietet sich ein hybrider Modellierungsansatz an, der zusätzlich die Identifikation von (Teil-)Modellen ermöglicht, also die experimentelle Modellbildung auf Grundlage von Prozeßdaten erlaubt, die den betriebsrelevanten Arbeitsbereich des (Teil-)Prozesses genügend beschreiben. Für diesen Aufgabenbereich wurde die ICAI Toolbox entwickelt, die die Systemidentifikation von (Teil-) Modellen innerhalb einer blockorientierten Simulationsumgebung ermöglicht.

ICAI TOOLBOX

Ein wesentlicher Ausgangspunkt zur Entwicklung der ICAI Toolbox waren Interviews mit potentiellen Anwendern in der Verfahrenstechnik. Dabei stellte sich heraus, daß die bereits existierenden Identifikationswerkzeuge in der betrieblichen Praxis eher selten genutzt werden. Vielfach wurde die sinnvolle Unterstützung regelungstechnisch unerfahrener Benutzer gefordert - ein entscheidender Aspekt, dem kaum ein derzeit erhältliches Identifikationswerkzeug gerecht wird. Die drei wesentlichen Anforderungen in diesem Zusammenhang sind [2]:

- Integration der Identifikationsaufgabe in eine blockorientierte Modellierungsumgebung, so daß der Benutzer nicht zwischen verschiedenen Programmen für Identifikation, Reglersynthese und Simulation wechseln und die gewonnenen Daten mit jedem Wechsel konvertieren muß.
- Bereitstellung weniger, dafür aber leicht parametrierbarer und robuster Identifikationsverfahren.
- Unterstützung von Mehrgrößen-Identifikationsaufgaben durch standardisierte Vorgehensweisen (siehe auch [3]).

Verständlicherweise kann mit einer standardisierten und damit nicht allen Eventualitäten gerecht werdenden Vorgehensweise zur Systemidentifikation nicht immer ein optimales Modellierungsergebnis erzielt werden, wie es mit mehr Freiheitsgraden im Identifikationsprozeß möglicherweise erreichbar wäre. Dennoch ist insbesondere für einen industriellen Anwender, der kein Experte auf diesem Gebiet ist, ein aus einer standardisierten Vorgehensweise entwickeltes, nutzbares Identifikationsergebnis eine praktikable Grundlage für einen systematischen Reglerentwurf. Als Entwicklungsumgebung für die ICAI Toolbox wurde MATLAB™ gewählt, das sich zum Quasistandard für regelungstechnische Software entwickelt hat und neben einem umfangreichen Paket an mathematischen Routinen mit SIMULINK™ auch eine blockorientierte Simulationsumgebung bereitstellt.

DIE ICAI ID BLOCKBIBLIOTHEK

Die ICAI **ID** (**Id**entification) Blockbibliothek besteht aus Blöcken zur Identifikation von Eingrößen- und Mehrgrößenprozessen, die im SIMULINK™ Blockdiagramm jeder für sich einen unbekannten (Teil-)Prozeß darstellen.

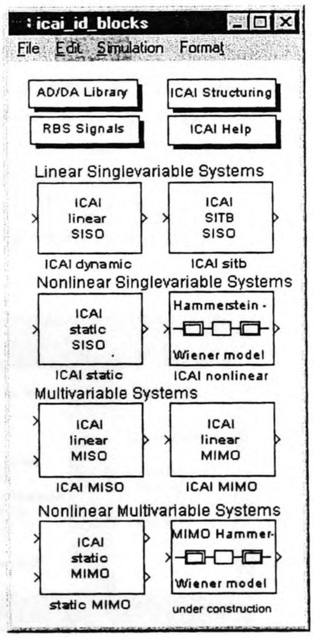


Bild 2: ICAI ID Blockbibliothek

Folgende ICAI ID Blöcke stehen zur Verfügung (Bild 2).

- *linear SISO* (single input single output): Eingrößenblock zur Identifikation von linearen Dynamiken.
- *SITB SISO*: Eingrößenblock zur Identifikation von linearen Dynamiken durch Integration der System Identifikation Toolbox [3].
- *static SISO*: Eingrößenblock zur Modellierung von statischen Kennlinien.
- *nonlinear SISO*: Eingrößenblock zur Identifikation von nichtlinearer Eingrößensystemen zusammengesetzt aus statischer Kennlinie und linearer Dynamik im Sinne von Hammerstein - oder Wiener - Modellen.
- *linear MISO* (multiple input single output): Block zur Identifikation von linearen Dynamiken mit mehreren Eingängen und einem Ausgang.
- *linear MIMO* (multiple input multiple output): Block zur Identifikation von linearen Dynamiken mit mehreren Eingängen und mehreren Ausgängen.
- *static MIMO*: Block zur Modellierung von statischen Kennlinienfeldern mit zwei Eingängen und zwei Ausgängen.

HANDHABUNG VON ICAI ID BLÖCKEN

Bei der Entwicklung der *ICAI ID Blöcke* wurde besonderen Wert auf einfache und durchsichtige Handhabung gelegt. Bild 3 zeigt die zur Nutzung von ID Blöcken notwendigen Schritte. Zuerst wird der *ID Block* aus der ICAI ID Blockbibliothek an die gewünschte Stelle im Blockdiagramm kopiert. Nach dem Start eines ID Blocks müssen die allgemeinen ICAI Projektbedingungen (*ICAI Project*) angegeben werden (siehe Bild 4), was allerdings nur einmal pro Blockdiagramm geschehen muß. Dann folgt die Identifikation auf Grundlage von Experimentdaten. Nach der Identifikation des (Teil-)modells kann das Ergebnis in die ICAI Ergebnisbibliothek (*ICAI Library*) geschrieben werden, bevor es als *ICAI Model Block* zur Simulation oder zum Reglerentwurf genutzt werden kann.

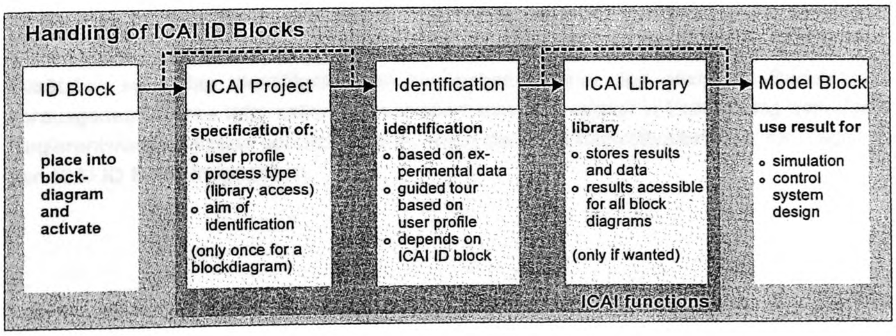


Bild 3. Handhabung von ICAI ID Blöcken

DAS ICAI PROJEKT

Das *ICAI Projekt*, das den Rahmen für den Identifikationsablauf bestimmt, muß einmal pro Blockdiagramm spezifiziert werden. Erst danach können ICAI Funktionen ausgeführt werden. Im ICAI Projekt wird zuerst ein paßwortgeschütztes Benutzerprofil abgefragt, das Auswirkungen auf die im Identifikationsverlauf angebotene Funktionalität hat. Derzeit sind drei Benutzerebenen eingerichtet. Die Benutzerebene Anlagenpersonal (Process personnel) beschränkt die Freiheitsgrade für regelungstechnisch eher unerfahrene Benutzer, indem kompliziertere Methoden ausgeblendet und die Darstellungen auf den Zeitbereich beschränkt werden. Auf der Betriebsingenieurebene (Area engineer) werden zusätzlich zu den Funktionen der Benutzerebene verschiedene Möglichkeiten zur Datenaufbereitung und Identifikation bereitgestellt. Die Expertenebene (Control expert) ermöglicht den Zugriff auf alle von der ICAI Toolbox zur Verfügung gestellten Funktionen.

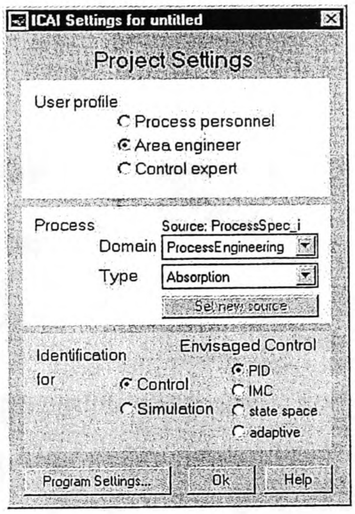


Bild 4: Optionen im ICAI Projekt

Sämtliche Projektdaten werden automatisch verwaltet und mit dem SIMULINK™ Blockdiagramm gespeichert. Ein Doppelklick auf den automatisch für jedes Blockdiagramm generierten *ICAI Project Block* ermöglicht die Änderung des ICAI Projektprofils. Dadurch werden beispielsweise Experten jederzeit in die Lage versetzt, ein vom Prozeßpersonal gewonnenes Ergebnis zu überarbeiten unter Zugang zu sämtlichen Funktionen.

Die Prozeßspezifikation nach Fachgebiet (Process Domain) und Prozeßtyp (Process Type) erlaubt den Zugriff auf SIMULINK™ Bibliotheken, die als Prozeßmodellbibliotheken zur Hinterlegung einmal gewonnen Wissens genutzt werden können und nach der Projektspezifikation automatisch aufgerufen werden. So ist zum Beispiel die Mechatronische Blockbibliothek von Ravn integriert worden [5], die u.a. eine Steuerung der Modellkomplexität zuläßt. Zudem dient die Prozeßspezifikation der Vorwahl von sinnvollen internen Projektparametern, die auch das Erscheinungsbild der graphischen Benutzeroberfläche beeinflussen.

Zusätzlich kann das Identifikationsziel angegeben werden, das geeignete Benutzerprofil vorausgesetzt. Diese Information soll vor allem von dem derzeit in Entwicklung befindlichen Reglerentwurfswerkzeug genutzt werden. Nach der Projektspezifikation wird der Zugriff auf den ICAI ID Block freigegeben.

IDENTIFIKATION MIT ICAI ID BLÖCKEN

Die Handhabung von ICAI ID Blöcken entspricht der Handhabung anderer SIMULINK™ Standardblöcke. Allerdings sind noch nicht fertig modellierte ID Blöcke solange farblich gekennzeichnet, bis die Identifikation erfolgreich abgeschlossen ist. Sobald der ID Block parametriert ist, wird er zum *ICAI Model Block,* was durch einen Farbumschlag kenntlich gemacht wird. Auf diese Weise können Blöcke, die noch nicht fertig modelliert sind, selbst in großen Projekten schnell erkannt werden. Zudem erscheint eine Warnmeldung, falls die Simulation mit noch nicht identifizierten ICAI ID Blöcken durchgeführt werden sollte.

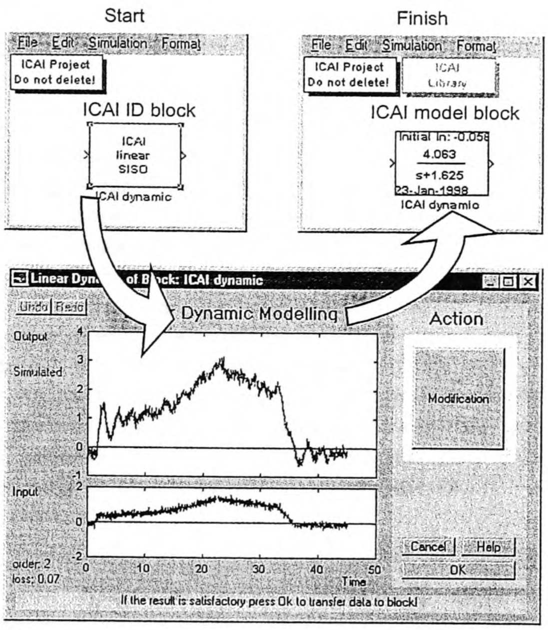


Bild 5. Identifikation mit einem ICAI ID Block

Sobald das ICAI Projekt (s.o.) spezifiziert ist, kann mit einem Doppelklick auf den ICAI ID Block das standardisierte graphische Benutzerinterface gestartet werden, das den Anwender durch die Identifikationsaufgabe führt und so gestaltet wurde, daß es ein Minimum an Einarbeitung erfordert. Dabei ist die Softwareoberfläche der eingestellten Benutzerebene angepaßt.

Die Eingabe der Versuchsdaten an den ICAI ID Block erfolgt vorzugsweise benutzergeführt aus MATLAB Dateien, in einigen Fällen aber auch von Hand unter Nutzung von ICAI Draw, einem Programmteil, der die grafische Eingabe von Daten ermöglicht. Relevantes Anwenderwissen wird abgefragt und genutzt, um den Identifikationsablauf zu strukturieren und zu parametrieren. Für die Identifikation sind nur wenige Standardlösungswege bereitgestellt, die in der Nutzung möglichst unkompliziert sind, so daß der Anwender nur wenige Design-Entscheidungen zu treffen hat. Schließlich hat der Anwender durch graphischen Vergleich des Prozeßsignalverlaufs mit dem simulierten modellbasierten Verlauf zu entscheiden, ob das Ergebnis ausreicht oder ob weitere Modifikationen nötig sind. Da die ICAI Model Blöcke weiterhin die Aktivierung der ICAI-Funktionalität erlauben, ist eine Überarbeitung von bereits erzielten Ergebnissen jederzeit möglich.

BESCHREIBUNG DER ICAI ID BLÖCKE

Im folgenden wird exemplarisch die Funktionalität der ICAI ID Blöcke beschrieben und einige Besonderheiten aufgezeigt.

Linear Dynamic SISO ID Block

Der Linear Dynamic SISO ID Block unterstützt die Identifikation eines linearen dynamischen Modells von Prozeßdaten. Es wird angenommen, daß die genutzten Daten von einem Experiment im offenen Regelkreis stammen, das den Prozeß im gewünschten Arbeitsbereich genügend angeregt hat, und daß der Prozeß entweder linear ist oder der Arbeitsbereich so klein, daß eine Linearisierung möglich ist. Als Standardmethode wurde eine Zwei-Schritt Identifikationsmethode implementiert, in der Korrelationstechniken und ein modifizierter MMLS (multiple model least squares) – Algorithmus kombiniert wurden, wie in [6] beschrieben. Mit dieser Methode wird ein Satz von Prozeßmodellen erzeugt, aus dem dasjenige mit dem geringsten Ausgangsfehler dem Anwender als Identifikationsergebnis vorschlagen wird. Fällt der graphische Vergleich des Prozeßsignalverlaufs mit dem simulierten modellbasierten Verlauf zufriedenstellend aus, wird das ausgewählte Identifikationsergebnis als Übertragungsfunktion in den Linear Dynamic ID Block geschrieben.

Static SISO ID Block

Der Static SISO ID Block kann statische Kennlinien beschreiben. Dazu müssen Prozeßdaten verfügbar sein, die eine ausreichende Anzahl an statischen Prozeßzuständen enthalten, um den nichtlinearen statischen Zusammenhang zu beschreiben. Als erstes Ergebnis wird eine Ausgleichskurve vorgeschlagen, die über eine standardisierte gewichtete Mittelwertbildung erzielt wird. Genügt dem Anwender die Güte dieser ersten Approximation nicht, können weitere Methoden genutzt werden, sofern dazu die Zugangsberechtigung besteht. Zuletzt wird das Ergebnis als Tabelle in den Static ID Block übertragen, der die Simulation des Blocks durch lineare Interpolation der Tabellenwerte ermöglicht.

Nonlinear SISO ID Block

Der Nonlinear SISO ID Block ist ein Makroblock, der die sukzessive Modellierung der statischen Kennlinie und der linearen Dynamik mit Hilfe des Static SISO und des Linear Dynamic SISO ICAI ID Blocks steuert (siehe Bild 6). Die Prozeßdaten müssen die statischen Prozeßzustände zur Modellierung der nichtlinearen statischen Kenn-

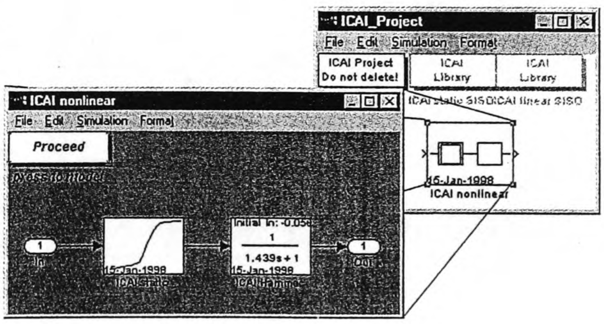


Bild 6. Nonlinear SISO ID Blocks

linie enthalten und zudem den Prozeß genügend angeregt haben, um die lineare Dynamik identifizieren zu können. Die Standardvorgehensweise sieht zuerst die Identifikation des statischen Modells in Form einer statischen Kennlinie vor. Danach wird unter Berücksichtigung der Effekte der statischen Kennlinie die lineare Dynamik über den Linear Dynamic SISO ID Block identifiziert. Das erfolgt erst im Sinne eines Hammerstein-Modells (statische Kennlinie vor linearer Dynamik) und dann als Wiener-Modell (lineare Dynamik vor statischer Kennlinie); letzteres nur, sofern die Kennlinie invertierbar ist. Man erhält schließlich zwei Modelle, von denen dem Anwender das mit dem kleineren Ausgangsfehler als Identifikationsergebnis vorgeschlagen wird.

Linear Dynamic SITB SISO ID Block

Der Linear Dynamic SITB SISO ID Block nimmt eine Sonderstellung unter den ICAI ID Blöcken ein, da er in benutzerfreundlicher Weise auf die Methoden der System Identification Toolbox (SITB, siehe [4]) zugreift. Dieser ICAI ID Block ist nur Experten zugänglich, da die SITB einerseits nur die Identifikation von diskreten Modellen unterstützt und andererseits zumindest Grundkenntnisse der angebotenen Verfahren und Darstellungsformen erfordert. Falls ein SITB SISO ID Block aufgerufen wird, kann entweder eine SITB-Datei (mit Zugriff auf alle genutzten Daten und Prozeßmodelle) importiert werden oder die System Identification Toolbox kann direkt aufgerufen werden. Wird eine SITB-Datei ausgewählt, so bietet der SITB SISO ID Block die Möglichkeit, aus allen bisher identifizierten parametrischen Modellen das bestgeeignete auszuwählen und direkt in den SITB SISO ID Block zu übertragen. Falls sich herausstellen sollte, daß die Güte eines SITB ICAI-Model Blocks nicht ausreicht, genügt ein Blockaufruf, um direkt in die System Identification Toolbox zu gelangen, mit Zugriff auf alle zuvor eingegebenen Daten und Modelle.

Linear Dynamic MISO ID Block

Die Funktionalität des Linear Dynamic MISO ID Blocks entspricht der des Linear Dynamic SISO ID Blocks mit dem Unterschied, daß dieser Block mehrere Eingänge besitzt. Zur Systemidentifikation wurden die im Linear Dynamic SISO ID Block genutzten Methoden auf Mehrgrößensysteme erweitert. Bild 7 zeigt die Signaldaten eines Prozesses mit drei Eingängen und einem Ausgang, aufgrund derer ein MISO Block generiert wird.

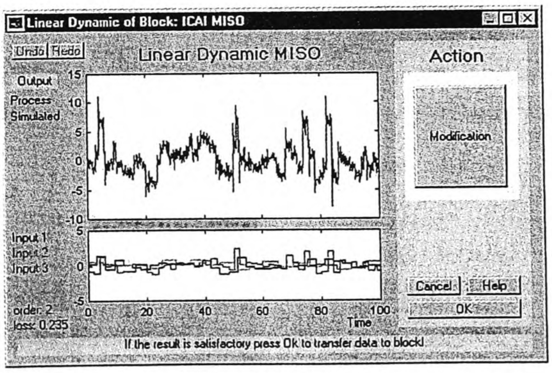


Bild 7. Oberfläche des Linear Dynamic MISO ID Blocks

Linear Dynamic MIMO ID Block

Der Linear Dynamic MIMO ID Block wird über einen Steuerblock sukzessive aus Linear Dynamic MISO ID Blöcken zusammengesetzt, wobei die Anzahl der Linear Dynamic MISO ID Blöcke der Anzahl der Ausgänge entspricht. In der Anwendung werden die MISO ID Blöcke nacheinander durchgearbeitet.

Static MIMO ID Block

Der Static MIMO ID Block wurde zur Modellierung von n-dimensionalen statischen Kennfeldern entworfen. Ebenso wie beim Static SISO ID Block müssen die Prozeßdaten eine genügende Anzahl an statischen Prozeßzuständen enthalten, damit das Kennlinienfeld approximiert werden kann. Die graphische Benutzeroberfläche (Bild 8) ist derzeit auf zwei Ein- und Ausgänge beschränkt.

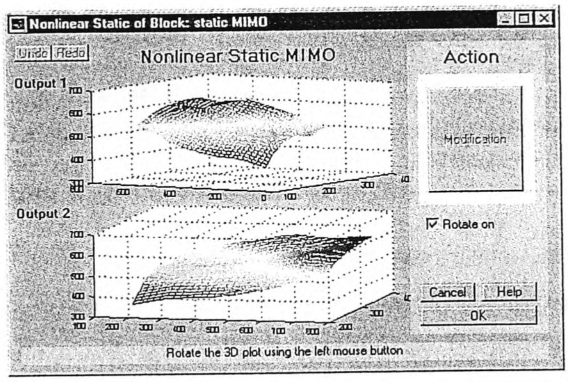


Bild 8. Kennfeld im Static MIMO ID Block

ICAI HILFEFUNKTION

Die Hilfe zu ICAI ist komplett im HTML-Format geschrieben und so umfassend, daß jederzeit schnelle Unterstützung über Standardbrowser wie zum Beispiel Netscape™ oder den Internet Explorer™ verfügbar ist. Zudem gibt es einen kleinen Lehrpfad durch die Hilfeseiten, der jedoch nicht notwendigerweise beschritten werden muß, um mit den ICAI ID Blöcken effektiv arbeiten zu können.

SOFTWAREANFORDERUNGEN

Die ICAI Toolbox wird unter MATLAB 5.1/SIMULINK 2.1™ entwickelt. Daneben sind folgende Toolboxen zur Anwendung der ICAI Toolbox notwendig:

- Signal Processing Toolbox
- Control System Design Toolbox
- System Identification Toolbox 4.0

ZUSAMMENFASSUNG

Zur Nutzung moderner regelungstechnischer Verfahren sind geeignete Prozeßmodelle eine unabdingbare Voraussetzung. Da die theoretische Prozeßmodellentwicklung sehr aufwendig, mühsam und teuer sein kann, ist die Entwicklung einfach bedienbarer, effizienter Identifikationswerkzeuge wünschenswert, die eine schnelle und brauchbare Prozeßmodellentwicklung auf Grundlage von Prozeßdaten auch für Mehrgrößenprozesse mit Nichtlinearitäten erlauben. Dabei muß die Softwareoberfläche dem vorhandenen Anwenderwissen angepaßt sein. Für unerfahrene Anwender müssen Standardlösungswege bereitgestellt werden, die so unkompliziert und robust wie möglich sind. Zudem ist es erforderlich, die Identifikationsaufgabe direkt in ein CAE-System zu integrieren und nicht ein losgelöstes Werkzeug zu schaffen, das zusätzlichen Schulungsaufwand erfordert.

Mit der ICAI Toolbox sind robuste Verfahren zur Systemidentifikation exemplarisch in MATLAB™'s blockorientierte Simulationsumgebung SIMULINK™ eingebettet worden. Das mit jeder Identifikationsaufgabe automatisch aktivierte *ICAI Projekt* sorgt für eine benutzerabhängige Programmsteuerung, die so gestaltet wurde, daß die Anwendung der bereitgestellten Verfahren besonders einfach und intuitiv ist. Weitere ausführliche Informationen zu der ICAI Toolbox sind über Internet erhältlich [7].

DANKSAGUNG

Dieses Forschungsprojekt wird zu großen Teilen von der Volkswagen Stiftung getragen. Projekt: REPAM 210-70631/9-38-1/92.

REFERENZEN

[1] Hahn, V. und G. Nöth: Anforderungen an regelungstechnisches CAE in der chemischen Verfahrenstechnik, VDI/VDE-GMA Jahrbuch, VDI-Verlag, Düsseldorf, 1997

[2] Körner, S. und R. Schumann: ICAI – A MATLAB™ toolbox for Computer Aided Identification, 11th IFAC Symposium on System Identification, Kitakyushu, Japan, 1997

[3] Körner, S. und R. Schumann: Inbetriebnahmeunterstützung bei PLS durch regelungstechnisches CAE, VDI/VDE- GMA Kongreß 1996 , Meß- und Automatisierungstechnik, Baden-Baden, VDI-Bericht Nr. 1282, 1996

[4] Ljung, L.: System Identification Toolbox User's Guide, The MathWorks Inc., user manual, 1995

[5] Ravn O. und M. Szymkat: Modelling and calibration with mechatronic blockset for SIMULINK, 11th IFAC Symposium on System Identification, Kitakyushu, Japan, 1997

[6] Körner S. und R. Schumann: Robustified estimation of multiple order models, IASTED Conference on Modelling, Identification and Control, Grindelwald, Switzerland, 1998

[7] Körner S.: ICAI – Industrial Computer Aided Identification – A new MATLAB toolbox for industrial users, Online, Internet, 1997
http://www.repam.fh-hannover.de/automa/ICAIHelp/Intro.html

Paper No. L-5c-01-5
Proceedings of the 14th IFAC, ISBN 0 08 043248 4

USER FRIENDLY CACSD FOR COMPLEX INDUSTRIAL PROCESSES

Birga Syska; Steffen Körner; Reimar Schumann

Forschungsschwerpunkt REPAM, Fachhocischule Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Germany,e-mail: syska@repam.fh-hannover.de, FAX +49-511-9296-111

EXTENDED ABSTRACT

In this paper the concept of an industrial CACSD (computer aided control system design) system is outlined which is geared at making available efficient theoretical control system design methods to the process industry. After discussion of general requirements the focus is set on the ICAC (Industrial Computer Aided Control) toolbox developed for MAT-LABISIMULINK™. This toolbox is aimed at industrial users like process engineers or corn-missioners who are not necessarily specialists in control system design. The ICAC toolbox provides specific methods for the design of linear and nonlinear single and multivariable control systems composed of industrially available flinction blocks such as PID controllers and typical nonlinear characteristics. Furthermore it allows simplified optimisation of arbi-trary industrial control schemes. The ICAC user interface supports 3 user levels with different degrees of flinctionality adapted to the user's knowledge.

KEYWORDS

• **CAC SD** • **Industrial Engineering** • **Process Industry**

Copyright © 1999 IFAC
14th Triennial World Congress, Beijing, P.R. China 409 L-5c-01-5

USER FRIENDLY CACSD FOR COMPLEX INDUSTRIAL PROCESSES

Birga Syska, Steffen Körner and Reimar Schumann

Forschungsschwerpunkt REPAM, Fachhochschule Hannover, Ricklinger Stadtweg 120, 30459 Hannover, Germany, e-mail: syska@repam.fh-hannover.de, FAX: +49-511-9296-111

Abstract: In this paper the concept of an industrial CACSD (computer aided control system design) system is outlined which is geared at making available efficient theoretical control system design methods to the process industry. After discussion of general requirements the focus is set on the ICAC (Industrial Computer Aided Control) toolbox developed for MATLAB/SIMULINK™. This toolbox is aimed at industrial users like process engineers or commissioners who are not necessarily specialists in control system design. The ICAC toolbox provides specific methods for the design of linear and nonlinear single and multivariable control systems composed of industrially available function blocks such as PID controllers and typical nonlinear characteristics. Furthermore it allows simplified optimisation of arbitrary industrial control schemes. The ICAC user interface supports 3 user levels with different degrees of functionality adapted to the user's knowledge. *Copyright © 1999 IFAC*

Key Words: CACSD, Industrial Engineering, Process Industry

1. INTRODUCTION

The design of complex control systems for industrial processes is in general still based on experience and trial and error rather than on systematic process analysis and control system design. Typically control systems in the process industry comprise either single PID control loops with clear association of the process input and controlled signal (e.g. heating and temperature) or standard control schemes developed by intuition through the years (e.g. cascade control using an underlying flow control loop). Although many of these control schemes seem to work rather satisfactorily, in most major control systems several poorly tuned or switched off controllers are encountered leading to unsatisfactory process behaviour or manual operation of the process (Hahn and Nöth, 1997). With increasing demands on process efficiency, product quality and environmental compatibility the need for better control and process optimisation leads to the question of how the potential of systematic process analysis and controller design methods developed by control theory over the last decades can be made available to the industrial control engineer to better the situation described. One of the possible answers is the use of industrial computer aided control system design tools tailored to the control design tasks and knowledge level of industrial engineers. The aim is to hide the complexity of theoretical methods under an industrial user interface and to adapt the design results to the realisation means in industrial process control systems.

2. INDUSTRIAL CACSD ENVIRONMENT

Most available CACSD systems have been developed in and for an academic environment. These systems serve mostly as testbeds for newly developed control methods providing a variety of methods and high degrees of freedom for the tuning of the methods. An industrial user, however,

ISBN: 0 08 043248 4

needs comprehensive CACSD tools which allow him

- to analyse practical problems encountered in the application of industrial control systems, e.g. by using process identification and simulation tools.
- to design or redesign industrial control schemes with function blocks available in industrial controllers, e.g. with (automatic) control scheme generators.
- to tune industrial control systems according to the requirements of the process operation, e.g. by using numerical or analytical optimisation tools.

For all these tasks the industrial user should not be bothered with the selection of various methods or unnecessary specification of design parameters as required usually in academic CACSD systems. An industrial CACSD system is needed which should provide for every step of the systematic control system design procedure, see Fig. 1, just one default preparametrized design method yielding usable results reliably (rather than optimal results after extensive selection and tuning of the optimal method).

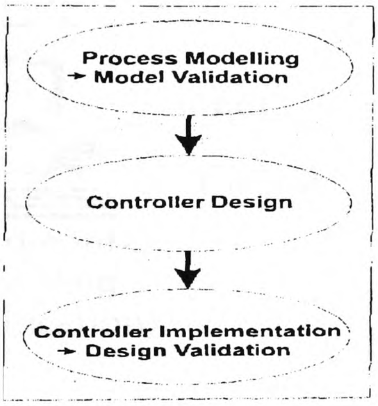


Fig. 1. Systematic Control System Design Procedure

3. THE ICACSD SYSTEM

The above considerations and extensive interviews with industrial users led to the design of the ICACSD (industrial CACSD) system (Schumann, *et al.*, 1996) which comprises in its present state four modules shown in Fig. 2.

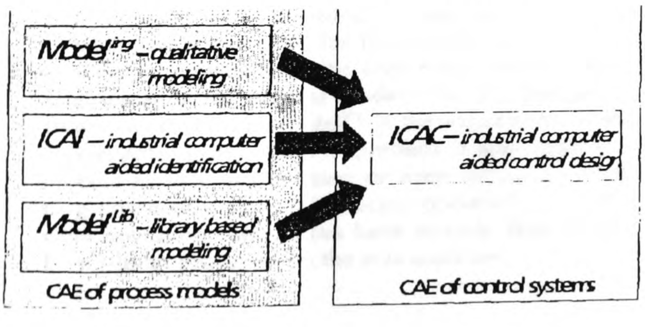


Fig. 2. Industrial CACSD modules

The ICACSD system represents the general frame for an industrial realization of the systematic control system design procedure (Fig. 1). The first user action is the specification of an ICACSD project, for which among other details a user profile and a process characterisation have to be defined, see Fig. 3.

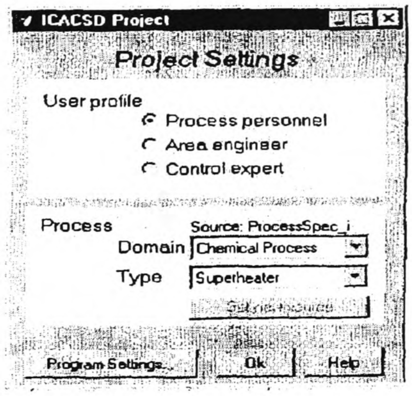


Fig. 3. ICACSD user profile and process specification

The user profile defines the degrees of freedom available for three different user levels: *process personnel* will get a standardized path through the CACSD procedure using preparametrized default methods with almost no degrees of freedom, the *area engineer* can at least access alternative design methods and the *control expert* will have access to all design methods and parameters available. By specifying the process domain and type, the user can add some general information about the process for which the control system is to be designed.

After completion of the project settings, the ICACSD scheme window opens, allowing the access to process modeling for which the user can select four model generation tools, see Fig. 4.

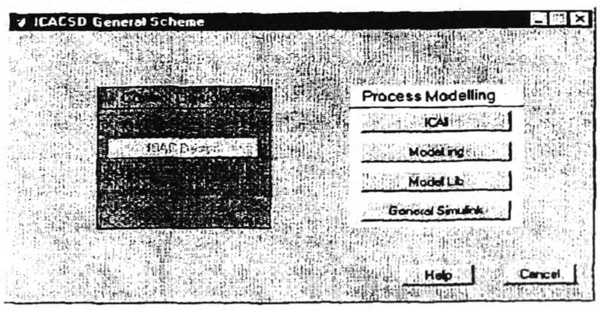


Fig. 4. ICACSD start menu

Besides the general possibility to use SIMULINK™ directly for the specification of a process model, ICACSD provides the choice between three toolboxes for the CAE of process models which are described in the next section.

ISBN: 0 08 043248 4

4. CAE OF PROCESS MODELS

The most important requirement for a systematic analysis and design of control systems is the availability of comprehensive process models. Therefore the CAE of process models is also of primary interest especially for the ICACSD system for which three toolboxes have been specified supporting the generation of process models in different ways and for different phases of process design and operation.

- the *ICAI* (Industrial Computer Aided Identification) toolbox for industrial computer aided identification (Körner and Schumann, 1997). *ICAI generates* structured linear and nonlinear single- and multivariable process models from measurement data of the process. ICAI is therefore especially useful during start-up and operation of the process where it can serve in addition to analyse the behaviour of the real process

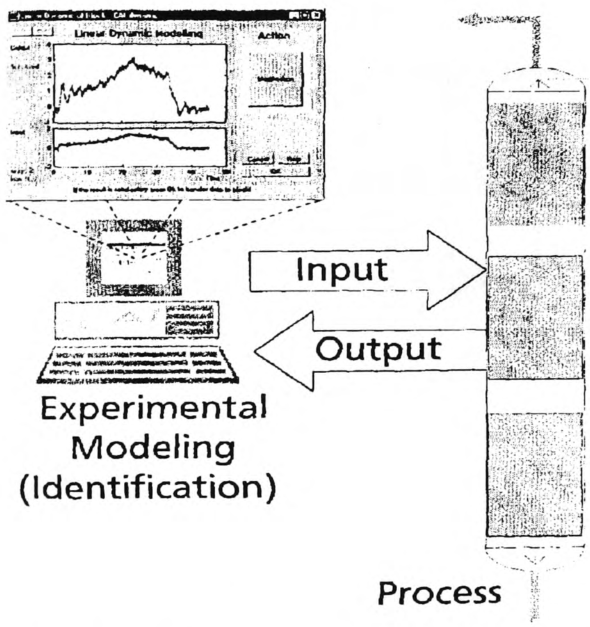


Fig. 5. Process modeling with ICAI

- the *Model^ing* toolbox for the qualitative design of process models using the process knowledge of the industrial user (Strickrodt, *et al.*, 1996). The knowledge *engineering approach* of Model^ing allows direct interaction with the area expert (i.e. the industrial user) without the need of a knowledge engineer who would normally be responsible for the translation of the unstructured knowledge of the area expert into a formal representation (or process model). As the primary information source of Model^ing is the experience of the area engineer, the Model^ing approach is applicable to a process already in operation or even during the planning phase of a standard process provided that sufficient operation experiences have already been gained from similar processes by the area engineer.

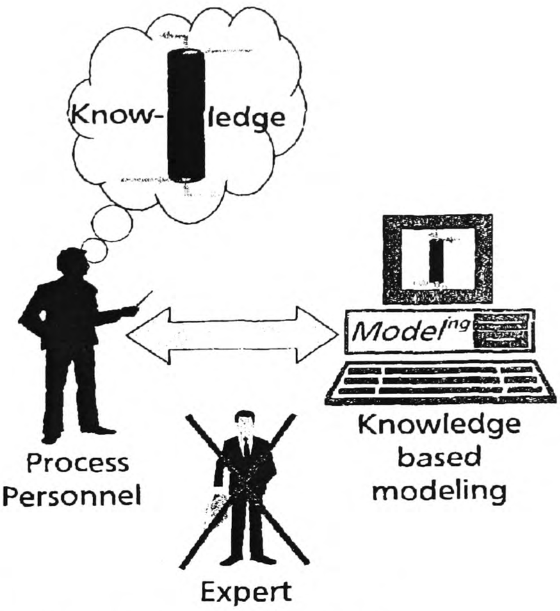


Fig. 6 Knowledge based process Model^ing

- the newly specified *Model^Lib* toolbox aimed at the aggregation of process models from partial (component) models.

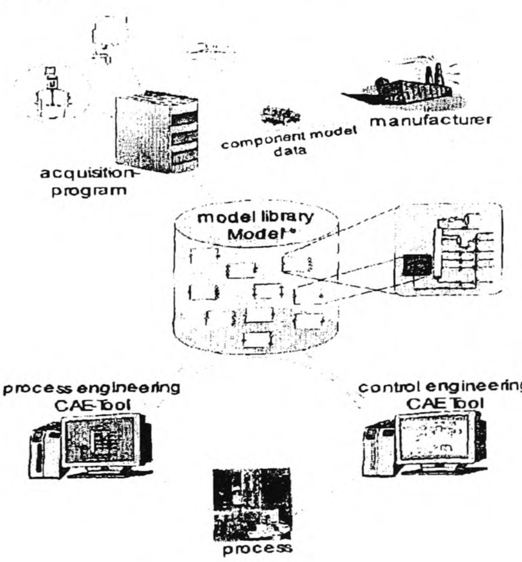


Fig. 7. Library based modeling

The basic idea of Model^Lib relies on the definition of an electronic catalogue of process components like valves, pumps, pipes, superheater etc., including especially dynamic models of the components' behaviour as required for the control system design. Such an electronic catalogue should be supported and filled by the component producers providing all technical information

ISBN: 0 08 043248 4

required for planning and maintenance of the process. During the process planning where the process is composed from such components, Model[Lib] should automatically generate dynamic models from the components submodels as required for the control system design for the regarded process area.

5. INDUSTRIAL COMPUTER AIDED CONTROL TOOLBOX – ICAC

After creation of the process model the ICACSD scheme changes its appearances, see Fig. 8. The process modeling menu is replaced by the process model (here: ICAI model), and in addition the control design block becomes accessible guiding the user through the ICAC toolbox. In a predefined sequence the following actions are organised:

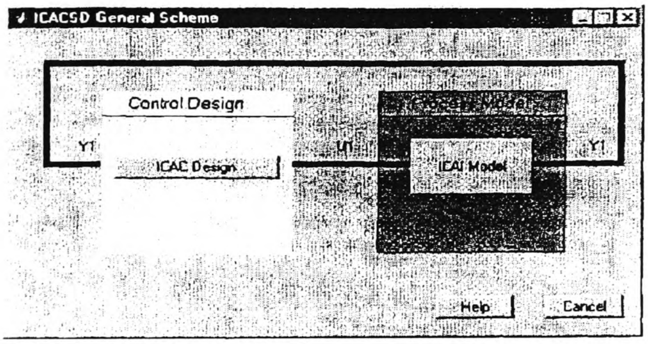


Fig. 8. Accessing ICAC in the ICACSD scheme

1) *Model Preprocessing.* Before designing the control system a general (SIMULINK™) process model may be preprocessed by linearisation or by applying ICAI to create a specific nonlinear single- or multivariable model structure. Original and preprocessed model may be compared by simulation in addition.
2) *Signal association.* In case that a MIMO control system has to be designed the user has to associate every process (model) output to be controlled with the primary process input from which it should be controlled preferably. Thus a symmetric control system structure is organised defining preference pairs of I/O signals.
3) *Control Design.* The control design (ICAC) window is shown in Fig. 9. The design procedure starts with the simplest control system structure (by default), the association of a linear PID controller to every pair of associated process I/O signals. The design method is predefined (numerical or analytical optimisation) and can be changed only by an area engineer or a control expert who, in addition, may also change the normally hidden design parameters. The simulated step responses of the closed loops allow an intuitive evaluation of the control performance which can be modified individually using the sliders for the control action. - If the control performance is not sufficient, alternative control system structures may be tried in a similar way by the user by simply clicking on the checkboxes for nonlinear SISO control and/or MIMO control. In the first case a nonlinear characteristic is added to every PID controller to compensate for process nonlinearities whereas in the second case process couplings are compensated systematically by a decoupling controller network.

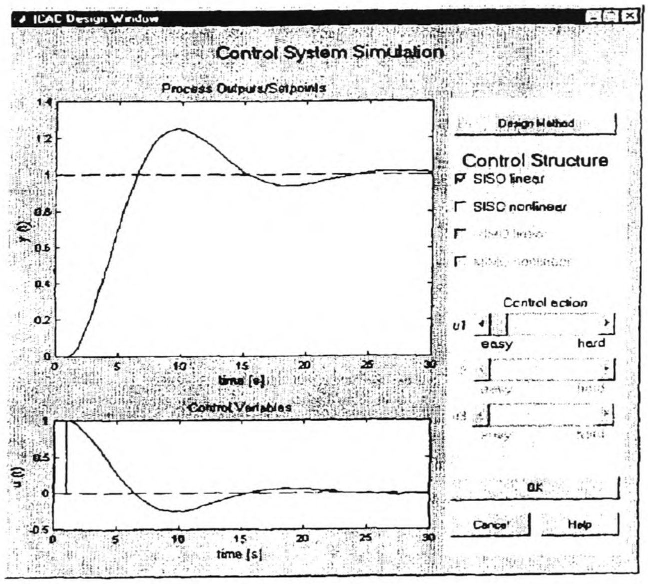


Fig. 9. ICAC design window

The resulting control system, however, consists in every case of linear PID-type control blocks possibly combined with nonlinear characteristic blocks, with which an industrial user is likely to be familiar and which furthermore can be realised with industrial process control systems. Having completed the controller design the user can choose between further evaluation of the design results using the original model (if the design was based on a preprocessed model) or a prototype implementation of the control system using an industrial process control system, or a repetition of the process modeling and/or controller design in case the user is not satisfied with the design results.

6. APPLICATION EXAMPLE

The application of the ICACSD toolboxes ICAI and ICAC will be demonstrated at a laboratory air conditioning plant, Fig. 10, for a which a humidity controller is designed. The plant consist of an air channel where the air flow can be changed by a fan. Air temperature ϑ and relative humidity φ in the mixing chamber at the air outlet are varied by a heater controlled by u_ϑ and by a humidifier controlled by u_φ.

ISBN: 0 08 043248 4

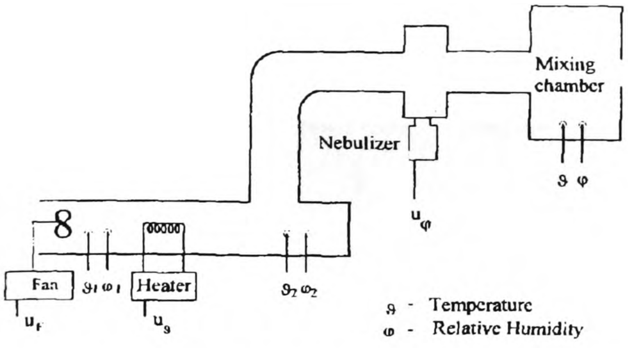


Fig. 10. Laboratory climate plant

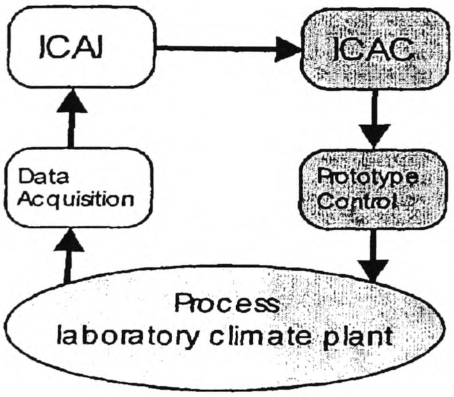


Fig. 11. General ICACSD procedure

The ICACSD procedure as shown in Fig. 11 is carried out in 4 steps

(1) *Data acquisition for identification.*
In the final version of ICACSD data acquisition will be accomplished by means of an interconnected process control system (PCS version) or in the standalone case by making direct use of an A/D-D/A process interface. For the demonstration experiment a separate data acquisition tool was used to collect the experimental data of the process inputs u_ϑ and u_φ and the process outputs ϑ and φ and to stored them in a MATLAB file ready for ICAI.

(2) *Process modeling with ICAI.*
Within ICAI the user on level "Process personnel" has just to select the MATLAB file with the measurement data and can then process these data directly and in a simple way to produce a two-input two-output model. The detailed handling within ICAI is described in (Körner and Schumann, 1997) indicating the simplicity of the user interface. Validation of the identified model is done by comparing the measured with the simulated process outputs, see Fig. 12. The submodel connecting u_φ with φ as required for the design of the humidity controller was identified by ICAI as a linear third order process.

$$G(s) = \frac{0.191 - 0.023s + 0.06s^2}{1 + 15.621s + 8.906s^2 + 2.49s^3}$$

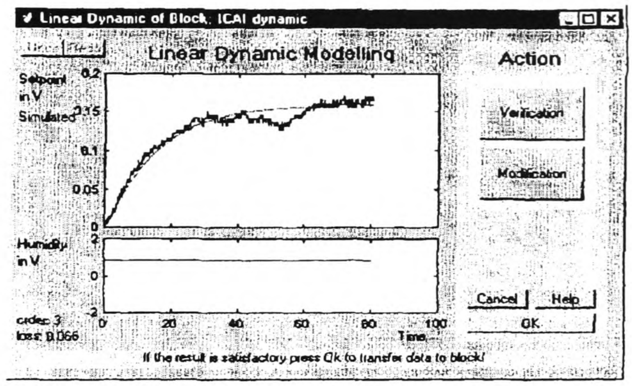


Fig. 12. Model validation with ICAI.

After the completion of the process identification the ICACSD menu changes to the form shown in Fig. 13 where the process model is presented with a canonical two-input two-output model structure.

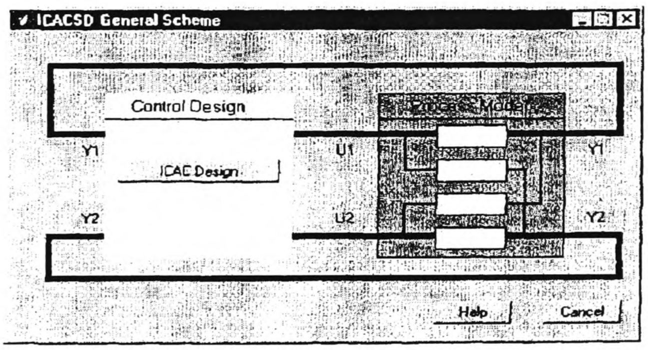


Fig. 13. ICACSD menu with structured process model.

(3) *Controller design with ICAC.*
The model produced by ICAI is selected by default when entering ICAC through the ICACSD menu, Fig. 13. In the simplest case, i.e. on the user level "Process personnel", controller design is simplified to the selection of the inputs and outputs (and thus of the respective submodel) for which the controller is to be designed and the variation of the control action in the ICAC design window, Fig. 9, using the respective sliders until the control performance is acceptable. In the demonstration example just u_φ with φ had to be selected for the design of the humidity controller.

(4) *Prototype control.*
For overall validation of the ICACSD design result the controller is directly applied to the original process. In the final state this will be accomplished either by implementing the controller in the interconnected process control system (PCS version) or in the standalone case using the A/D-D/A process interface. At the moment the prototype controller implementation is still accom-

ISBN: 0 08 043248 4

plished by making use of an external PC based realtime control system. Validation of the design result will be done by checking the simulated control system behaviour against the control behaviour at the real plant as indicated in Fig. 14.

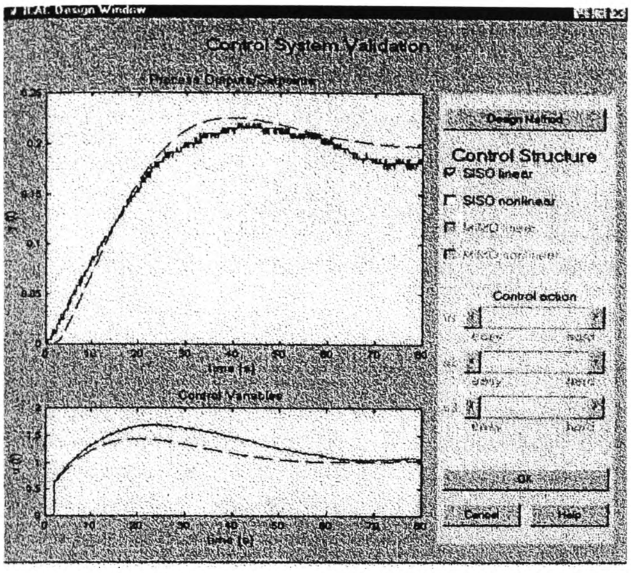


Fig. 14. Validation of the humidity controller

In case the controller performance is not acceptable ICAC and/or ICAI have to be used again to refine the design e.g. by adding nonlinear model and controller parts. As soon as the control performance is satisfactory the ICACSD procedure is finished.

7. CONCLUSION

The ICAC toolbox has been realised in parts as fast prototype. The implementation as SIMULINK/MATLAB™ toolbox is a continuing effort, the first part comprising the above described functionality will be completed in the next year.

The first tests will be run at an absorption process (miniplant) equipped with an industrial process control system serving as interface for data acquisition and as implementation tool for the designed control systems.

Future work will comprise in addition the design of more general control system structures including feedforward control as well as cascade control in arbitrary combinations.

8. ACKNOWLEDGEMENTS

The research was carried out with support of the Volkswagen foundation, REPAM, project 210-70631/9-38-1/92.

9. REFERENCES

Hahn, V., Nöth, G. (1997). Anforderungen an regelungstechnisches CAE in der chemischen Verfahrenstechnik, VDI/VDE-GMA Jahrbuch, VDI-Verlag, Düsseldorf.

Schumann, R., S. Körner, K.J. Baker and M. Strickrodt (1996). Shaping CACSD for practical use in process industry. IFAC World Congress, San Francisco.

Körner, S. and R. Schumann (1997). ICAI – A MATLAB™ toolbox for Computer Aided Identification. IFAC Symposium on System Identification, Fukuoka, Japan.

Strickrodt, M., R. Schumann and K. J. Baker (1996). An integrated knowledge engineering approach to process modeling for CACSD. IFAC World Congress, San Francisco.

ISBN: 0 08 043248 4

SHAPING CACSD FOR PRACTICAL USE IN THE PROCESS INDUSTRY

R. Schumann*, S. Körner*, K. Baker and M. Strickrodt****

** Fachhochschule Hannover, Fachgebiet Automatisierungstechnik, REPAM, Ricklinger Stadtweg 120, D-30459 Hannover, Germany*
*** University of Glamorgan, Dept. of Mech. and Man. Engineering, Pontypridd, Wales (UK)*

Abstract: During the last decade a variety of academic CACSD tools has been developed which allow the experimental use of computer aided process identification and controller design methods by academic experts. This paper proposes a new approach for the design of an industrial CACSD tool which is tailored to the requirements and abilities of industrial users in the process industry. The approach is based on a standardized CACSD procedure and a process model evolution scheme which simplify the use of CACSD methods under industrial conditions and relieve the industrial nonexpert user from the unnecessary theoretical load of academic CACSD programs.

Key Words: Industrial control; CACSD; identification; controller design

1. INTRODUCTION

Today a large number of CACSD (Computer Aided Control System Design) programs is available on the software market supporting more or less all CACSD phases like process identification and modeling, controller design, system simulation and analysis (Schumann, 1989). However, most programs are of academic origin providing analysis and design methods and tools developed in and for an academic environment. Now, in the process industry controller design tasks have to be solved by the process and control engineer for complex multi-input multi-output (MIMO) processes. Using academic CACSD programs for the solution of these design tasks will lead in general to mathematically complex process models and to the use of powerful theoretical controller design methods which, however, can be understood and handled only by academic control experts - even if the CACSD program is equipped with a sophisticated user guidance system as described in (Meier zu Farwig and Unbehauen, 1991). Moreover, most of the user interfaces of academic CACSD tools were designed to enable extensive tests of various algorithms and methods but do not support efficiently the solution of standard industrial controller design tasks.

This paper presents an industrial CACSD scheme which is tailored to the needs of the control engineer in the process industry. The design is based on numerous discussions with technicians and engineers in the process industry (Bayer, PreussenElektra) and in companies providing process control engineering, equipment and/or systems for this industry (Hartmann&Braun, Siemens). The industrial CACSD scheme is streamlined to support the industrial user on his traditional controller design path. It enables a more efficient and reliable solution for industrial controller design tasks than by purely manual design. The scheme includes:

1. a model evolution scheme for the adaptation of the process model complexity to the practical requirements and
2. the definition of a standardized CACSD procedure.

The paper is organized as follows: In the next section the traditional approach to control system design is outlined as it is still practised in the process industry today. Then the paper focuses on both components of the proposed industrial CACSD scheme, i.e. the model evolution strategy and the standardized CACSD procedure. An extensive design example illustrating the proposed industrial CACSD scheme will conclude the paper.

2. INDUSTRIAL CONTROL SYSTEM DESIGN

Practical control system design in process industry is mostly based on a rather rough description of the typically MIMO process. An example for this is shown in Fig. 1 where a steam generator is represented by a simple flow chart in which 6 measurement points (process outputs) and 4 manipulation points (process inputs) and 7 PID (Proportional plus Integral plus Differential) control blocks can be detected.

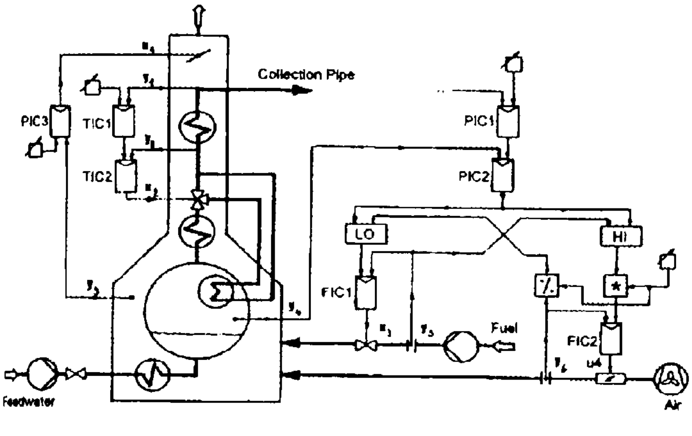


Fig. 1. Steam generator process

2.1. *Simple SISO Approach for MIMO Processes*

For many technical MIMO processes as for this steam generator standard control schemes (often complicated and nonlinear) are in use which have been developed by generations of process and control engineers in an intuitive way. However, without such a standard control scheme the starting point for practical controller design is in general the simplifying assumption that it is sufficient to split the MIMO process into independent SISO (Single-Input Single-Output) subprocesses by associating each process output to be controlled to the process input with the greatest influence on it. For each of these independent SISO main I/O paths a separate PID controller is implemented on the basis of rather rudimentary process informations like rough estimates for process gain and dominant time constant or possibly (and this is already looked at in industry as advanced time consuming and expensive procedure) based on step response experiments. The separate PID controllers are tuned by human expert knowledge - or better, the experienced industrial engineer just knows how to tune such a process by rules of thumb or by intuitive optimization. The restriction to PID controllers results from the fact that in industry these controllers are still standard. More advanced control algorithms like state controllers, discrete control algorithms or multivariable controllers are usually not available as standard function blocks in industrial controllers - and also not necessary to the understanding of the industrial control engineer because the functionality of these controllers is too complicated and difficult to tune.

2.2. *Intuitive Extension of the simple SISO Approach*

For 90% of industrial control design tasks the simple SISO approach with separate PID controllers is sufficient. Only if this approach fails due to unacceptable control performance a deeper process analysis is done by intuitive means in the sense that observed changes in process gains and time constants or coupling effects between the SISO subsystems are now taken into account for the controller design in addition. Then the beforehand strictly separated SISO control systems are supplemented with compensating elements to cope for the observed effects. So for the compensation of changing process gains a gain scheduling scheme is often used for the corresponding PID controller and crosscouplings between SISO subsystems are compensated by adding feedforward compensators etc., where all these measures are done more or less in the same intuitive way as for the design of the SISO PID controllers themselves. By time a complicated industrial control scheme may develop as indicated in Fig. 1 for the steam generator which may even contain such nonlinear elements like multiplication and division of signals, min/max-selection etc. and which is difficult to analyze theoretically.

In the next section, an industrial CACSD scheme for industrial controller design is described which is streamlined to the above described approach in the process industry and intended to make it more systematic and transparent.

3. INDUSTRIAL CACSD SCHEME

The proposed industrial CACSD scheme is based on two principles:

1. a model evolution scheme which includes four standard control system structures yielding the simplest solution with acceptable control performance and
2. a standardized CACSD procedure with reduced degrees of freedom with respect to process model and controller structure selection.

3.1. *Model Evolution Scheme*

The practical design path for industrial control systems as described above is starting with the simplest control system structure, i.e. separated SISO subprocesses controlled by individual PID controllers and extended to more complicated control schemes for compensation of process nonlinearities or coupling effects only in case the simple solution does not work sufficiently. The proposed industrial CACSD scheme follows this principle by defining a model evolution scheme comprising four different control system structures as shown in Fig. 2 and described in the following.

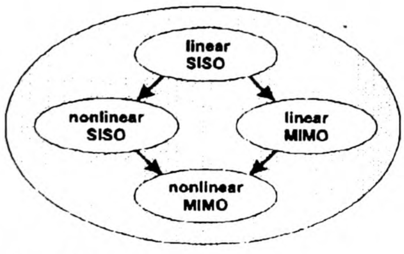


Fig. 2. Model evolution scheme

Linear SISO model. The first attempt for the controller design is based on the Linear SISO model which assumes that it is sufficient to represent the process by a model with separate linear SISO submodel blocks as shown in Fig. 3.

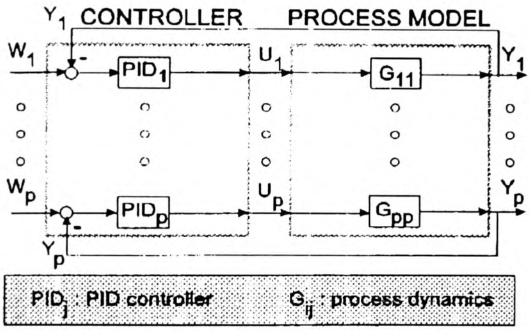


Fig. 3. Linear SISO process model and controller

The selection of the SISO main I/O paths is done by associating every process output to be controlled to the process input with the greatest influence on it. For each SISO submodel an independent linear PID controller is designed, see Fig. 3. Only in case that no acceptable control behaviour can be achieved using the standardized CACSD procedure as described in the next section the system structure is extended depending upon the observed unacceptable effects to one of the following alternatives.

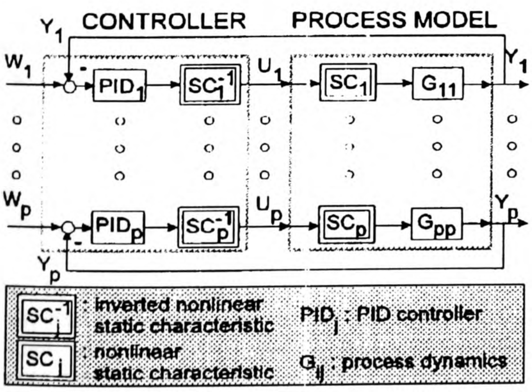


Fig. 4. Nonlinear SISO process model and controller

Nonlinear SISO model. In case that control problems are detected to be related with varying process gains, the Linear SISO model should be augmented to the Nonlinear SISO model. This model is combined from separate SISO submodels for the main I/O paths in form of simple Hammerstein (alternatively also Wiener) models each with a linear dynamic and a nonlinear static part, see Fig. 4. For each nonlinear SISO submodel a complementary nonlinear controller is designed with a nonlinear static block for compensating the submodel's nonlinearity and a linear PID controller tuned for the submodel's linear part, see Fig. 4.

Linear MIMO model. In case that primarily coupling effects deteriorate the control performance, the Linear SISO model should be extended to the Linear MIMO model reflecting also the crosscoupling effects between process inputs and outputs by additional linear coupling blocks, see Fig. 5.

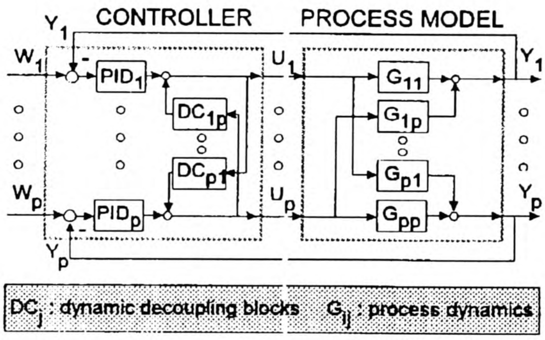


Fig. 5. Linear MIMO process model and controller

The corresponding standard controller structure contains singlevariable PID controllers for the main I/O paths as in the Linear SISO case, which are complemented with feedforward controllers designed to compensate for the effects of the linear dynamic coupling blocks of the process model.

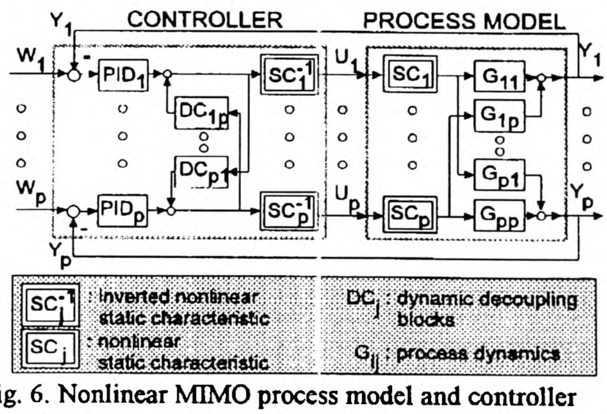


Fig. 6. Nonlinear MIMO process model and controller

Nonlinear MIMO model. Only in case that none of these alternative process model structures yields sufficient control behaviour the Nonlinear MIMO model may be tried as the most complicated model structure in the proposed model evolution scheme. This model structure can be gained either by supplementing the Nonlinear SISO model with nonlinear coupling I/O paths (each of which containing a linear dynamic and a nonlinear static block) or by extending the Linear MIMO block by nonlinear static blocks at each input (or output) of the model as shown in Fig. 6. In this case the corresponding controller structure is defined by extending the controller structure of the Linear MIMO model by nonlinear static compensation blocks at the process inputs, see Fig. 6.

3.2. *Standardized CACSD Procedure*

The standardized CACSD procedure is applicable to each of the above described models and described here for the case that the process model is generated by process identification from experimental data. The CACSD procedure can be split in three main CACSD phases, see Fig. 7.

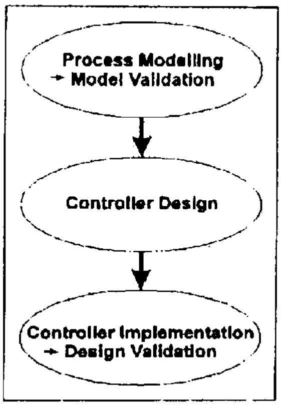


Fig. 7: Standardized CACSD Procedure

PHASE I: Process Modelling. As first step in the standardized CACSD procedure the process model is to be generated in the computer from experimental data. For the above specified 4 process model structures only two different CACSD tools are required:

1. an identification tool for generation of a linear dynamic SISO or MISO model from experimental data and
2. an identification tool for the determination of static characteristics in the SISO and MIMO case.

In case of the linear models the application of the first CACSD tool will lead directly to the required model. In case of the nonlinear models the static characteristics have to be determined first making use of the second CACSD tool; then by precompensating the model nonlinearities by their respective inverse the linear dynamic blocks are identifiable using the first CACSD tool. For validation of the process models graphical inspection is proposed allowing also an inexperienced user the detection of bad models by comparison of experimental and simulated data (in the future also other 'quality' measures will be used). In case that no good correspondence between experimental and simulated data can be achieved with the used process model structure Phase I of the CACSD procedure has to be repeated with the next more complex process model structure.

PHASE II: Controller Design: As shown in Fig. 3 to Fig. 6 the process model structure found in Phase I is directly reflected in the associated control system combined from:

1. linear single variable PID controller blocks tuned for the linear dynamic part of the associated main I/O path
2. nonlinear static blocks defined as inverse blocks of the corresponding process model nonlinearities and/or
3. linear feedforward compensating blocks tuned to reduce effectively the crosscoupling effects.

The tuning of the linear PID controllers and the feedforward compensating blocks can be done easily by numerical optimization in appropriately separated control subsystem. The predicted control system performance is checked by simulation of the complete control system with process model and controller, however, as the complete control system has been designed to cope only for the modeled effects only direct design errors can be detected which may accordingly be corrected by just repeating Phase II.

PHASE III: Controller Implementation: The implementation of the designed control system is a nontrivial task not only due to possibly unmodelled process model parts but also due to potential differences between the controller elements used in the simulation in Phase II and the ones really applied to the process with industrial control systems (modified PID algorithms, limiters, anti windup schemes etc.) which have to be taken into account. The crucial validation of the complete controller design is thus based on the comparison of the real control performance with the simulated one reached in Phase II. In case that the control performance at the real process is not sufficient and differs obviously from the simulated one the standardized CACSD procedure has to be repeated from Phase I with the next more complex process model structure.

4. PROTOTYPE REALIZATION OF THE INDUSTRIAL CACSD SCHEME

4.1. *Nonlinear MIMO Pilot Plant*

The scheme of the nonlinear two input two output pilot plant is shown in Fig. 8. Its main component is a semi-closed water tank filled with water by the waterpump (No.1 of Fig. 8) and with air by the airpump (No.3 of Fig. 8). Two valves, one for water level and one for air pressure (No.2 and 4) allow to adjust the operating point of this plant. Valves 5 and 6 provide the means to generate defined disturbances for the controlled variables water level Y_w and air pressure Y_a, which are manipulated by the control voltages U_w for the waterpump and U_a for the airpump. The plant is clearly crosscoupled in the sense that the waterpump does not only influence the water level, but also the air pressure and vice versa the air pump does also influence the water level.

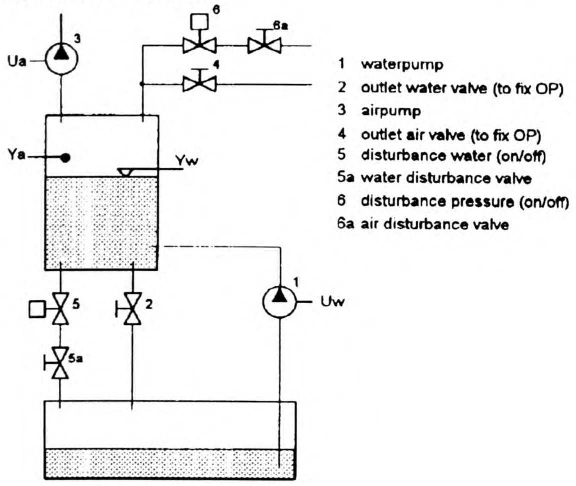


Fig. 8. Nonlinear MIMO pilot plant

4.2. *Utilized Tools and Methods*

As already pointed out numerous academic CACSD tools are available on the market (Schumann, 1989; Schmid, 1993; Frederick et al., 1992) but unfortunately there was no single tool available at the time of the prototype realization which was suited to support all tasks of the standardized CACSD procedure appropriately for an industrial environment. Thus a patchwork of tools had to be selected for the prototype realization, see Table 1.

Table 1 Utilized CACSD tools

Task within Standardized CACSD Procedure	**CACSD tool**
identification of linear dynamics	CADACS by University of Bochum
identification of static characteristics	*EASYSTAT* by FH Hannover
simulation and controller design	DORA 5.1 / DORA-Fuzzy by University of Dortmund
controller implementation validation	TCS with Loopdraw by EUROTHERM
simulation and presentation	SIMULINK / MATLAB by Mathworks

Identification of linear dynamics. CADACS with its realtime module for identification experiments provides various process identification methods. For the prototype realization the simple and reliable Moncman method was chosen which determines an n'th-order-lag-approximation model from a measured step response.

Identification of static characteristics. No commercial CACSD tool was found which supports this work effectively. So a proprietary CACSD tool, EASYSTAT, was created to deal with the time-consuming job of investigating steady state characteristics of SISO and MIMO processes. EASYSTAT allows automatic determination of static single and multidimensional characteristics in open and closed loop.

Simulation and controller optimisation. For this part DORA was chosen because especially the simulation part DORA-Fuzzy offers integrated, simple and efficient optimization facilities which allow PID parameter tuning in a block oriented environment. The PID controllers were optimized for setpoint changes using a quadratic controller design criterion, which balances control performance and actuator effort. For the decoupling feedforward controllers, standard lead/lag blocks were numerically optimized to reduce the coupling effects between the main I/O paths. The nonlinear characteristics of the control system were realized as inverse of the identified process nonlinearities using look-up tables.

Controller implementation and on-line validation. As typical industrial controller device the TCS (Turnbull Control Systems) 6370 controller was chosen as target system for the designed controller structure. The control system was implemented using the graphical blockoriented configuration software LOOPDRAW providing the means to realize nonlinear multivariable controller structures using linear dynamic blocks and static nonlinear blocks (as look-up tables).

4.3. *Experimental Results*

Now the complete industrial CACSD scheme will be illustrated by experimental results of the prototype implementation. Among the many results a ramped set point change (as normally applied in process industry) on the water level was chosen to demonstrate the overall performance of the designed control systems. For this purpose the process input variables water and air pump voltage U_w and U_a, as well as the process output variables water level Y_w and air pressure Y_a were recorded for the different cases

Linear SISO model. The application of the described standardized CACSD procedure produced the simulation and real time control results shown in Fig. 9. The control behaviour at the real plant shows oscillations and crosscouplings (which were expected) which the simulation does not show at all. The significant difference is obviously due to the fact that the simulated model does not cover any nonlinear or coupling effect. The observed differences and the poor control behaviour indicate that the model complexity is not sufficient.

Nonlinear SISO model. This was the first try to reduce the observed oscillation effects in the Linear SISO case. The simulated control system behaviour is identical to the linear SISO case (besides a change in the control signal scales due to the transfer of the process gains from the linear dynamic to the nonlinear blocks). However the real control performance becomes much better with respect to the oscillation effects observed in the first try, see Fig. 9. This is obviously due to the modeling and compensation of the process nonlinearities. Nevertheless the differences between simulation and real time results indicate still the existence of unmodelled parts in the process model and the observed real time control behaviour was not accepted.

Nonlinear MIMO model. To improve the results of the nonlinear SISO case the nonlinear MIMO model was tried (The linear MIMO approach was omitted due to the obvious existence of model nonlinearities). The comparison of the simulated control system with the realtime control experiments showed a much better coherence than in the first two cases. Furthermore, the crosscoupling effects are clearly reduced compared to the simpler process models, see Fig. 9. So the overall performance was accepted and the industrial CACSD scheme came to a successful end.

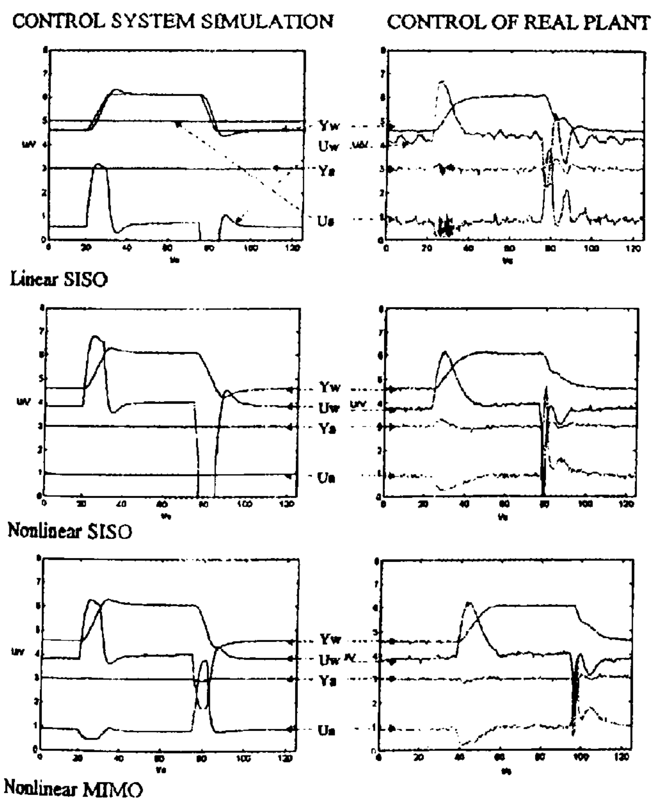


Fig. 9. Comparision of simulated and real process behaviour

5. SUMMARY (CACSD ASPECTS)

The proposed industrial CACSD scheme was designed for the solution of practical controller design tasks in the process industry. The combination of a standard model evolution scheme (from linear SISO to nonlinear MIMO) with a standardized CACSD scheme (including process identification, controller design and implementation) simplifies the CACSD procedure for the industrial user and allows a simple adaptation of the control system complexity to the practical requirements. The prototype application to a laboratory plant demonstrates the principal feasibility of this approach. Other applications, e.g. to climate plants, have shown similar results. However, it is clear that the proposed industrial CACSD scheme in its basic version has its limits with respect to the used model structures which have been selected to support rather the practized industrial design process than to fulfil theoretical conditions. So, future work will concentrate on the refinement of the scheme with respect to the use of alternative process model structures and an early detection of undermodeling. Also the use of alternative control structures oriented at more refined process model structures will possibly require other tuning procedures. Moreover, the prototype realization using a variety of different CACSD tools has to be replaced in the future by a new industrial CACSD tool realizing efficiently the outlined industrial CACSD scheme with a user interface designed for the industrial user.

6. ACKNOWLEDGEMENT

The research was carried out with support of the DAAD through the Anglo-German Academic Research Collaboration Programme ARC, Project 313-ARC-VII-93/86/scu, for the collaboration with the University of Glamorgan, Project 520. Further this research was partly supported by the Volkswagen foundation, project 210-70631/9-38-1/92.

7. REFERENCES.

Frederick, D K;Herget, C J; Kool, R; Rimvall, M, (1992), ELCS - The extended list of control software, *Dept. of Automatic Control*, Swiss Federal Institute of Technology (ETH), Zürich.

Meier zu Farwig, H and Unbehauen, H (1991). Knowledge-based system identification. *IFAC Symposium on Identification and System Parameter Estimation*, Budapest, Hungary.

Schmid, C (1993), Verfügbare Softwaretools Seminar "Rechnergestützte Analyse und Entwurf von Regelungssystemen", *VDI Seminarhandbuch, VDI-Bildungswerk, Düsseldorf.*

Schumann, R (1989). CAE von Regelsystemen mit IBM-kompatiblen Personal Computern. Automatisierungstechnische Praxis (atp), Vol. 31, No. 8, 349 -359.