Graphical User Interface Software for System Identification

Bart De Moor

Peter Van Overschee

Katholieke Universiteit Leuven
Department of Electrical Engineering (ESAT)
Research Group SISTA
Systems, Identification, System Theory and Automation

Kardinaal Mercierlaan 94
B-3001 Leuven, Belgium
tel: 32/16/220931 fax: 32/16/221855
email: bart.demoor@esat.kuleuven.ac.be
peter.vanoverschee@esat.kuleuven.ac.be

ESAT-SISTA/TR 1994-06

Contents

Chapter 1

Introduction and survey

"There is, however, a much larger sense in which changes in knowledge are causing or contributing to enormous power shifts. The most important economic development of our lifetime has been the rise of a new system for creating wealth based no longer on muscle but on mind. Labor in the advanced economy consists of people acting on information and information acting on people"
-excerpted from *Power Shift*, A. Toffler, 1990, pp 8-9.

In this Chapter, we present a general overview of the scientific contributions of this memorandum, which are concentrated around a new powerful generation of algorithms for system identification (called subspace algorithms) on the one hand, and their integration in a Graphical User Interface on the other hand.

More specifically, the research on system identification, model-based control and implementations into graphical user interfaces, when applied to real industrial processes and systems, will potentially result in products and production strategies that *are cleaner, safer and faster, are more user and cost efficient, allow for just-in-time and 'zero-defect' production, have narrower tolerances, are more flexible and responsive to user's specifications, have an improved performance, etc ....*
The crucial new ingredient here is that systems can be optimized when a model is available. Modeling allows to incorporate and exploit a priori information, ultimately leading to improved performance. In combination with a user friendly tool such as the Graphical User Interface we will discuss below, it allows to implement Rapid Prototyping Methodologies, either on a real life prototype plant or within a *virtual engineering* setting, in which the real plant is replaced by a(n) (approximate) model.

This Chapter is organized as follows: In Section 1.1. we first give a general chapter-by-chapter overview of the global text, while in Section 1.2. we situate the main results of this work in the wider context of the socio-economic reality, both in Europe and the World, which is further elaborated on in Section 1.3. In Section 1.4., we formulate the main conclusions. Acknowledgments can be found in Section 1.5.

1.1 A quick glance through this text

While in this Chapter we describe the main guidelines and the wider context of our work, in the Chapters to come we describe in some more detail the scientific and technical results and contributions. The different Chapters can be read quite independently from one another, in the sense that they basically treat the same problem but each time from a different point of view: Scientific (Chapter 2), Software-Technical (Chapter 3) or Applied-Industrial (Chapter 4).

- In Chapter 2, we concentrate on the major *scientific* contributions, which are all within the development of a new powerful breed of algorithms for system identification of multivariable systems. These algorithms, called **N4SID** [1], are based upon recently developed insights within the disciplines of system theory and numerical linear algebra and we will discuss (in terms that are not too mathematical) the main original contributions.

- In Chapter 3, we present a direct *software-technical spin-off* of the scientific insights of the previous Chapter. Here, we discuss our development of a software Toolbox for System Identification, named **ISID II**, which is based upon a user friendly Graphical User Interface. The net result is an easy-to-use package, which nevertheless is very powerful for producing mathematical models of complex industrial processes.

- The application of aforementioned insights for *rapid prototyping* within an industrial context is the subject of Chapter 4. Here we show how the combination of the powerful mathematical insights of Chapter 2, together with the user-friendly GUI of Chapter 3, allows a control-system design engineer to produce any model-based controller within a couple of hours. The system we analyze here is a glass-tube manufacturing process for which a model-based multivariable minimum-variance controller is designed.

[1] Read as a Californian license plate: *Enforce it.* Acronym stands for **N**umerical algorithms for **S**tate **S**pace **S**ubspace **S**ystem **Id**entification.

1.2 About this work and its context

After the microelectronics and integrated circuits era began in the late fifties, a phenomenal technology progress led to a spectacular and unprecedented evolution beyond all expectations. Microelectronics is now crucial to all information technology industries, such as the computer, consumer and communications industries. The steady decrease in integrated circuit cost per transistor has brought a stream of products to the market previously impossible to manufacture in a cost-effective way, if at all.
Information processing system designers now have the disposal of fast and inexpensive computers or signal processors, leading to an explosion of applications in, *e.g.*, digital signal processing applications in telecommunications, video and audio, radar and sonar, process control and automation.
Microelectronics is a challenging, but at the same time a very predictable technology. The available computing power is still steadily increasing, roughly doubling every two years. With the advent of multi-processor systems this increase is even more dramatic. The message is that in the future, systems designers will not have to worry so much about the available computing power. The question is, rather, whether it will be possible at all to take full profit from this evolution, as there is usually no point in solving the same problems at a speed which is doubled every two years.

The observation to be made in this proposal is that the most significant break-through can be achieved if we are prepared to drastically re-think our information processing problems, solutions and systems. Spurred by the significant advances in computer technology, information system research is to be directed towards developing *novel intelligent information processing systems.* Here 'intelligent' refers to the use of *advanced mathematical computations.*

Following similar tendencies as described above, the last two decades or so, research and education in electrical engineering is no longer confined to what could be considered the classical field of electricity, electronics or power engineering.
Throughout the world, electrical engineering departments have broadened their scope and have diversified their interests to start up research in image processing and enhancement, algorithms, telecommunications, signal processing, control theory, neural networks, artificial intelligence, fuzzy logic etc ... [2].

The present proposal is an excellent illustration of these new directions and evolutions. Two major developments are married into one powerful result, which is a Graphical User Interface Toolbox for System Identification. The two basic ingredients are illustrated in

[2] This is not only reflected in the research subjects of the Electrical Engineering Departments worldwide, but also in the courses that are taught by the staff of these departments. As an example, the basic courses in *Linear Algebra* in many places are given by people from EE Departments, despite the fact that this is a highly mathematical subject.

Figure 1.1:

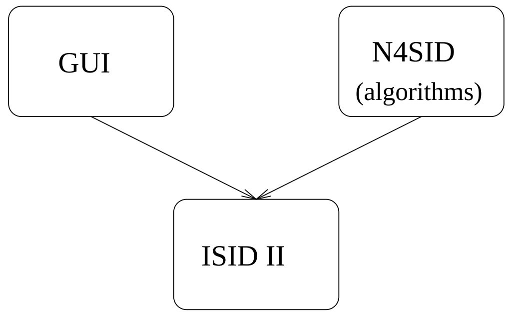

Figure 1.1 The main results of our work consist of 3 blocks: Scientific research for powerful identification algorithms (**N4SID**), development of a Graphical User Interface (GUI), and the combination of the two to allow for Rapid Prototyping Design of model-based control of industrial processes. The final result is the GUI based software toolbox **ISID II**. **N4SID** is described in Chapter 2, general ideas about GUIs and the Identification Toolbox **ISID II** are the subject of Chapter 3, while in Chapter 4 we describe a case study of rapid prototyping.

N4SID algorithms: [3] It turns out that for model-based control of many industrial processes, *linear dynamic models* provide a useful approach to model the system from experimental input-output measurements. The **N4SID** algorithms (Numerical Algorithms for Subspace State Space System Identification) which we have been developing in a series of publications [15] [16] [17] [18] [19] [20] [21] [22] [55] [59] [60] [61] [74] [75] [76] [77] [78] [79] [80] [81] [82] [83] [84] turn out to be powerful alternatives for the 'classical' identification algorithms that were the subject of intensive research in the 70s and 80s.

Among many others, an important conceptual idea behind the **N4SID** algorithms is to re-emphasize the concept of state within the system identification community. Trivial as this may seem, in system identification the relevance of the state of a system has been largely ignored, despite the fact that in control theory, the insight that state feedback is crucial has been around since the beginning of the sixties, for both linear and nonlinear systems.

N4SID algorithms first estimate/calculate the state (sequence), while next the (state space) model is determined. This is in sharp contrast with 'classical' algorithms where first a model is computed and only after that, when it is needed, the state (e.g. via a Kalman filter...).

Several other properties and advantages of **N4SID** that are highly relevant for practical applications will be discussed in this proposal.

[3] Pronounce as a Californian License Plate: *Enforce it*.

The Graphical User Interface (GUI) ISID II: The GUI we have been designing and implementing (see references [3] [81]) contains, besides the **N4SID** framework that we have developed ourselves, a whole breed of other functionalities, including preprocessing techniques, other identification algorithms and validation and plotting tools. One of our main features is a so-called Chain Window, in which the user gets an excellent overview of what she [4] is or has been is doing. Depending on the problem, there is also the opportunity to zoom in in less or more detail on one of the sub-functionalities. As we will describe in Chapter 3, GUI driven toolboxes belong to a third software generation, which has several benefits:

- First of all, the user can tackle much more serious problems because she is releaved from programming subtleties (almost all of the interaction with the software happens via the mouse buttons).
- This implies that bookkeeping of tasks, models, data sets and interconnections, becomes relatively straightforward.
- Next, there is quite some user guidance by the GUI in terms of options and defaults that can be chosen.
- Finally, GUIs are user friendly and one can learn to work with them in a couple of minutes, without the necessity of going through thick manuals (help functionalities are provided everywhere).

As a matter of fact, the GUI **ISID II** that we have been developing within Xmath[5], is one of the first GUIs to be implemented within the system identification community. We will come back to this development in Chapter 3.

The *synergetic* combination of these two elements has resulted in a powerful software toolbox, which is not only *user friendly* because of the extreme care by which the GUI was conceived and implemented but also because of the extreme elegance and power of the recently developed **N4SID** algorithms.

In this work, we show how complicated but extremely useful mathematical developments can be implemented in such a way that the potential user can successfully benefit from them without being bothered about the sometimes difficult fine mathematical detail. In this way, the powerful yet often unknown capabilities of (engineering) mathematics come within reach of the industrial control engineer, who plans to develop model-based control strategies for her plant at hand, but for several reasons cannot spend days,

[4] To quote Roger Penrose in *The emperor's new mind* (Vintage, London, 1990, pp.8): *"There is an inevitable problem in deciding whether to use the pronoun 'he' or 'she' where, of course, no implication with respect to gender is intended....* Accordingly, when referring to some abstract person, we shall henceforth use 'she' simply to *mean* the phrase 'she or he' (...and to make a point, especially within this technical environment... :-).

[5] Xmath is a trademark of Integrated Systems Inc., Santa Clara, California, USA.

weeks or months learning about and struggling to understand detailed (and sometimes difficult) mathematics.

It goes without saying that the approach in this proposal, to marry engineering mathematics with graphical user interfaces, can be applied *mutatis mutandis* to many other mathematical environments (such as *Computer Aided Control System Design* (CACSD), *Operations Research and Production Planning, Computer Integrated Manufacturing,* etc...)

1.3 Why should we care about intelligent software?

Before we dive into the technicalities of this work, we would like to situate it into a wider context. Indeed, the growing diversification of the world of electronics is not only reflected in research and education (see above), but, maybe more importantly, it is a general trend in Europe and the World. We only present two figures to illustrate this observation [6]. From these figures, it is clear that the future of Europe is no longer concentrated in the classical production sectors of the Third Industrial Revolution. Information technology has by now become one of the biggest sectors of the European economy and it is expected to become the largest by the end of the decade.
If we want to survive as one of the leading economical associations in the world, we should invest time, energy and money in our intelligence and creativity. Only in this way, we can maintain an economic growth which is badly needed from a socio-economic perspective and which in many ways is the best strategy to maintain a competitive position relative to the South-East Asian Tigers, Japan and the US [7].

The present work is a modest contribution to this larger program.

1.4 Conclusions

The essential contributions of our work, split up in scientific, technical and applied ones, are summarized in Figure 1.4.

[6] These figures are borrowed and translated from *Investeren in de toekomst: Elektronika: Centraal in een vernieuwd industrie- en technologiebeleid.* (Fabrimetal, Vlaamse Gewest, IMEC, 1993)

[7] A recent resolution of the European Parliament (January 1994, proposed by the commission on Economical and Monetarian Affairs and Industrial Policy) confirms the threat of an increasing dependency of the European electronics business from that of the US and Japan. The resolution proposes to strengthen the technological and financial basis of the European industry.

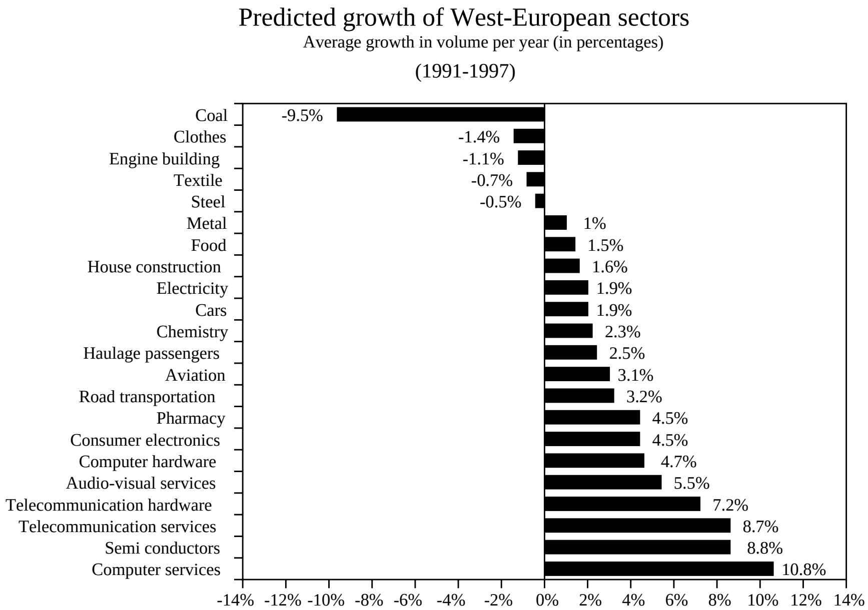

Figure 1.2 Prediction of the growth of some Western European Economical Sectors. Clearly, the classical heavy industrial activities are on the decline, in favor of the field of *Information Technology*. At present, Europe is undergoing a major shift from classical industrial activities towards 'postmodern' computer and telecommunications applications.

1.5 Acknowledgments

We're standing on the shoulders of giants...

This work would have been impossible without the wise advise of and many stimulating interactions with people from all over the world. We would like to thank Professors Gene Golub, Thomas Kailath, Robert Kosut and Stephen Boyd, all from Stanford University (California) for the many occasions at which we were their guests. That a real company is something completely different from our protected universitarian environment, was taught to us convincingly by Henk Aling and Alexandra Schmidt from Integrated Systems Incorporated, Santa Clara, California. Professor Lennart Ljung from Linköping University and professors Anders Lindquist, Björn Ottersten and Bo Wahlberg (Royal Institute of Technology, Stockholm) have shared with us lots of ideas and suggestions, many of which still need to be worked out. Professors Jan Maciejowski (Cambridge University, UK) and Michel Gevers (UCL, Belgium) have contributed a lot in creating a stimulating European Research Network on System Identification.
Last but not least, we would like to thank all our colleagues and friends in the Electrical

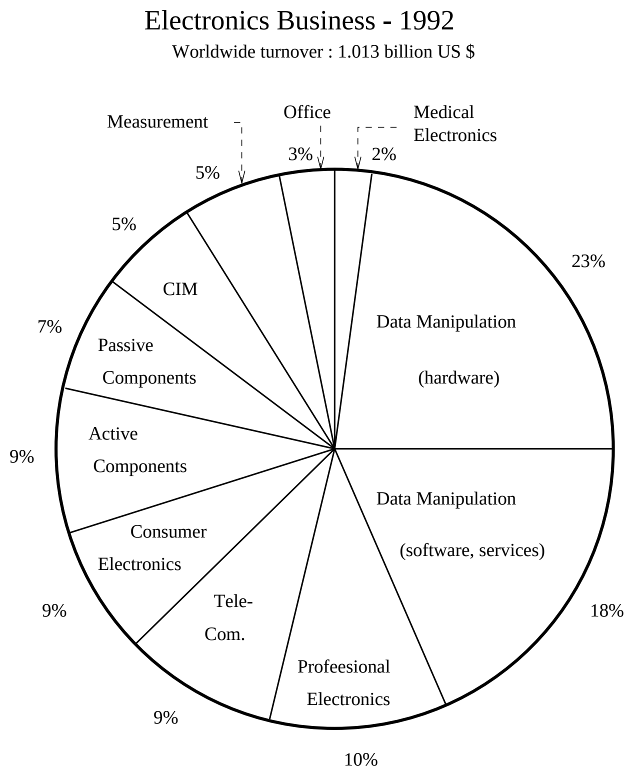

Figure 1.3 The Electronics Business Worldwide. It is obvious that what is commonly called *Information Technology* occupies a most prominent place. The work that is proposed here is mainly related to Computer Integrated Manufacturing and Data Manipulation (software).

Engineering Department of the Katholieke Universiteit Leuven.

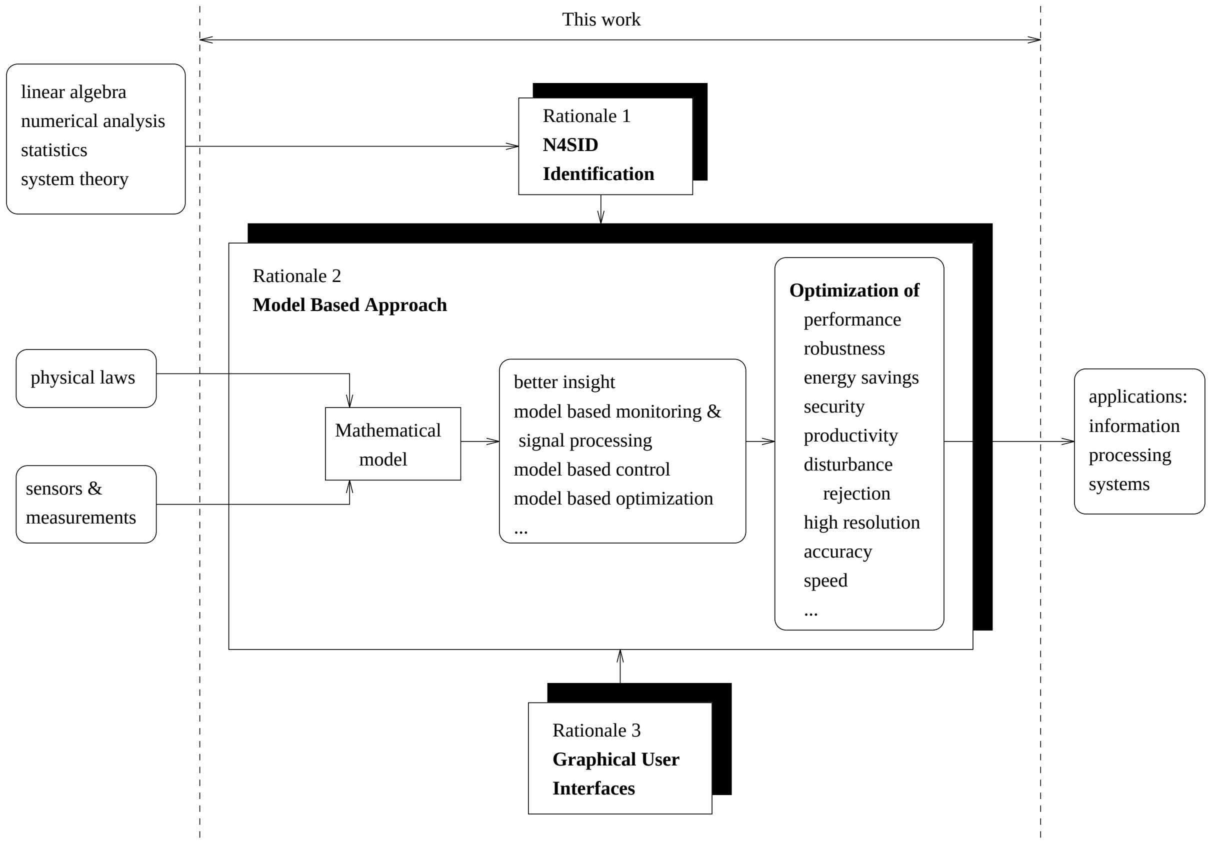

Figure 1.4 An overview of the different rationales of the work presented here. The main contributions are inside the dotted vertical lines. What is outside these lines, is important but will not be treated here in detail. We do not talk about the mathematical prerequisites and ongoing research in mathematical engineering, nor about the way experimental data are to be acquired to build our mathematical models. We also don't speak about the overall global integration to physically build information processing systems. The three main contributions of our work are:
1. Scientific: (Chapter 2) The discovery and development of new algorithms for multivariable system identification (**N4SID**). This ideal is driven by the fact that mathematical models are essential in designing better information processing systems, of which model-based controllers are but one special case.
2. Software-technical: (Chapter 3) The implementation of the Toolbox **ISID II** which is based on a Graphical User Interface.
3. Applied-Industrial: (Chapter 4) The validation of both the scientific insights and the software aspects within an industrial environment, clearly demonstrating that *rapid prototyping* now comes within reach of the control-system design engineer.

Chapter 2

Mathematical models and system identification

The development of Subspace Methods
is the most exciting thing
that has happened to system identification
the last 5 years or so....
Professor Lennart Ljung from Linköping, Sweden
at the SCIENCE-ERNSI workshop
Louvain-la-Neuve, October 2, 1993.

In this Chapter, we treat the main *scientific* contributions of our work. In Section 2.1., we first give a short but for our purposes sufficient motivation of the rationales behind *model-based control system design*, which is our main motivation to deal with the multivariable system identification problem. In Section 2.2., we discuss in some more detail the main contributions which make that our **N4SID** algorithms are excellent tools to work with in an industrial environment. We also provide some historical background and compare our achievements to previously existing approaches to find black box mathematical models of systems. Concluding remarks can be found in Section 2.3.

2.1 Models of systems and system identification

A dynamic system can conceptually be described as in Figure 2.1, which covers almost all physical, economical, biological, industrial, etc ...systems. One could distinguish between mental, intuitive or verbal models, or graphically oriented approaches such as graphs and tables, but we will mainly be interested in mathematical models. Such models are described as differential (continuous time) or difference (discrete time) equations.

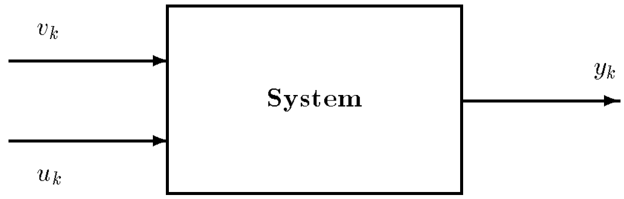

Figure 2.1 A dynamic system with deterministic inputs u_k, outputs y_k and disturbances v_k (see below). All arrows represent vector signals and k is the discrete time index. The user can control u_k but not v_k. In some applications, either u_k, v_k or w_k can be missing. The measured output signals provide useful information about the unknown system.

They describe the dynamic behavior of a system as a function of time. Mathematical models exist in all scientific disciplines, and, as a matter of fact, form the heart of scientific research itself. They are used for simulation, operator training, analysis, monitoring, fault detection, prediction, optimization, control system designs, quality control, *etc.* Most typically, models are highly useful in those situations in which experimenting with the real system is too expensive, too dangerous, too difficult or merely impossible. Last but not least, mathematical models are used for *control* and *feedback*, which, by the way, is one of the major engineering inventions.
The underlying motivation of our work is the claim that model-based solutions are superior in performance and robustness compared to plain heuristic approaches. This fundamental insight is not new, and has been at the heart of modern control theory ever since the path breaking work of Kalman at the beginning of the sixties.
We will however not go further into detail on how a controller needs to be designed, once there is a mathematical model. For this, there is a huge amount of scientific literature and even today, the topic is an active field of research. What is of interest to us here, is how such a mathematical model can be obtained! Surprisingly enough, this question has been neglected for a long time, even if all the work on control system design explicitly *assumes* that there is a model available. It is only the last decade or so that modeling of industrial processes has become a serious mathematical engineering discipline, in which we hope to have made some important contributions.

Basically, there are two main roads to construct a mathematical model of a dynamic system. Physicists will be interested in models (physical laws) that carefully explain the underlying essential mechanisms of observed phenomena and that are not *falsified* by the available experiments. The necessary mathematical equipment is that of *partial differential equations*. This is the analytic approach, which rigorously develops the

model from *first principles*.
For engineers however, this framework is often much too involved to be really *useful*. The reason is that engineers are not really interested in the exact model *as such*, but more in the potential engineering applications of models. In this perspective, a mathematical model is only one step in the global design of a system. The quality of a model is dictated by the ultimate goal it serves. *Model uncertainty is allowed as long as the robustness of the overall system is ensured.* Engineers -in contrast with mathematical physicists- are prepared to *trade-off model complexity versus accuracy*. A complex model will lead to a complex design, while a simplistic model will deteriorate overall performance and robustness of the final implementation. As an example, the best model for simulation (for instance a set of partial differential equations which accurately models the system's behavior) is not the best one for control, because, as a generic property of control system design, the complexity of the controller and the degree of difficulty associated with its implementation, are proportional to the model complexity. Therefore engineers will typically use *system identification* techniques to build their models. This is the field of modeling dynamical systems from experimental data: Experiments are performed on a system, a certain parameterized model class is predefined by the user and suitable numerical values are assigned to the parameters so as to fit as as closely as possible the recorded data. In this sense, system identification is the dynamic extension of curve fitting. Finally there is a validation step, in which the model is tried out on experimental data which were not used in the system identification experiment.

In Chapter 4, we describe an industrial process which perfectly illustrates the fundamentally different point of view between the two modeling approaches. The glass-tube manufacturing process described there could in principle be characterized completely using the laws of physics (in this case the laws that govern the behavior of solidifying glass). But, not only would this be a formidable task -if practically possible at all-, but even if there was such a model, it would be impossible to derive an appropriate control action to regulate the system, because of the complexity of the model. However, in Chapter 4, it will be shown how a relatively simple state space model, obtained by the mathematical methods described in this Chapter and the Graphical User Interface of the next Chapter, allows for the design of a high quality minimum variance controller. The message is that system identification however provides a meaningful engineering alternative to physical modeling. Compared to models obtained from physics, system identification models have a limited validity and working range and in some cases have no direct physical meaning. But, they are relatively easy to obtain and use and even more importantly, these models are simple enough to make model-based control system design mathematically (and also practically) tractable. Of course, there are still problems such as the choice of an appropriate model structure, the fact that many systems are time-varying and often largely underestimated measurement problems (appropriate sensors, sampling times, filters, outlier detection, etc...).
Let us conclude this section by saying that system identification, being a typical engi-

neering discipline, borrows many of its concepts and techniques from other mathematical and engineering fields (as was already suggested in Figure 1.4), such as optimization, numerical analysis, linear algebra, complex function theory, statistics, sensors and physical devices, experimental design, software engineering, etc ... and therefore is in many respects an interdisciplinary activity.

2.2 N4SID: A new generation of system identification algorithms

This section contains the central results of this Chapter. First of all, in Subsection 2.2.1., we describe the central importance of *state space models*, which is the type of models that's being delivered by **N4SID**. The acronym **N4SID**[1] stands for **N***umerical algorithms for* **S***ubspace* **S***tate* **S***pace* **S***ystem* **Id***entification* and we'll explain this mouthful of words in Subsection 2.2.2. In Subsection 2.2.3, we highlight the main innovations of our new algorithms with respect to existing 'classical' approaches. Subsection 2.2.4 illustrates the understatement that *we are standing on the shoulders of giants...*, in the sense that some of the concepts we use in our algorithms are more than 100 years old (besides more modern ones of course!). In Subsection 2.2.5, we summarize the main ongoing research activities.

2.2.1 State space models are good engineering models

It goes without saying that there is an infinite collection of mathematical models. In our work, we have restricted ourselves to discrete time, linear, time-invariant, state space models. From the number of epitheta used, this might seem like a highly restricted class of models (especially the fact they're linear), but, surprisingly enough, many industrial processes can be described very accurately by this type of models.
Moreover, by now, the number of control system design tools that are available to build a controller based on this type of models, is almost without bound. Especially for this reason, this model class is a very interesting one.
Mathematically, these models are described by the following set of difference equations[2]:

$$x_{k+1} = Ax_k + Bu_k + w_k \tag{2.1}$$

$$y_k = Cx_k + Du_k + v_k \tag{2.2}$$

with

$$\mathbf{E}[\begin{pmatrix} w_p \\ v_p \end{pmatrix} \begin{pmatrix} w_q^t & v_q^t \end{pmatrix}] = \begin{pmatrix} Q_s & S_s \\ S_s^t & R_s \end{pmatrix} \delta_{pq} \quad \geq 0 \tag{2.3}$$

In this model, we have

[1] Pronounce as a Californian license plate: *Enforce it*.

[2] $\mathbf{E}$ denotes the expected value operator and δ_{pq} the Kronecker delta.

- **vectors:** The vectors $u_k \in \mathbb{R}^{m\times 1}$ and $y_k \in \mathbb{R}^{l\times 1}$ are the measurements at time instant k of respectively the m inputs and l outputs of the process. The vector x_k is the state vector of the process at discrete time instant k and contains the numerical values of n states. These states do not necessarily have a direct physical interpretation but they have a conceptual relevance. Of course, if the system states would have some physical meaning, one could always find a similarity transformation of the state space model to convert the states to physically meaningful ones. $v_k \in \mathbb{R}^{l\times 1}$ and $w_k \in \mathbb{R}^{m\times 1}$ are unmeasurable vector signals. It is assumed that they are normally distributed, zero mean, stationary white noise vector sequences.

- **matrices:** $A \in \mathbb{R}^{n\times n}$ is called the system matrix. It describes the dynamics of the system (as completely characterized by its eigenvalues). $B \in \mathbb{R}^{n\times m}$ is the input matrix which represents the linear transformation by which the deterministic inputs influence the next state. $C \in \mathbb{R}^{l\times n}$ is the output matrix which describes how the internal state is transferred to the outside world in the measurements y_k. The term with the matrix D is called the direct feedthrough term. In continuous time systems this term is most often 0, which is not the case in discrete time systems due to the sampling [3].
  The matrices $Q_s \in \mathbb{R}^{n\times n}$, $S_s \in \mathbb{R}^{n\times l}$ and $R_s \in \mathbb{R}^{l\times l}$ are the covariance matrices of the noise sequence w_k and v_k.

A graphical representation of the system can be found in Figure 2.2.

Let us comment in some detail why it is often a good idea to try to fit experimental (industrial) process data to the model just described.

- First of all, for multiple-input, multiple output systems, the state space representation is the only model that is convenient to work with in *computer aided control system design* (CACSD). Most optimal controllers can be effectively computed in terms of the state space model, while for other system representations (such as e.g. matrix fractional forms) the calculations are not so elegant.

- Observe that we have collected *all* dynamics in one matrix A, that is to say that the eigenvalues of the matrix A will describe all the dynamical modes that have been measured, whether they come from the real system, from stochastic dynamic disturbances, from measurement sensors or the dynamics of the input actuators. This is quite unusual as compared to approaches that are described in the literature, in which one always carefully distinguishes between e.g. deterministic models (such as models for the 'real' system and sensor and actuator dynamics) and noise models for stochastic disturbances (as is for instance the case in the Box-Jenkins

[3] There are additional technical assumptions on the matrices A, B, C, Q_s, S_s, R_s in terms of controllability and observability, but we won't go into too much technical detail in this presentation.

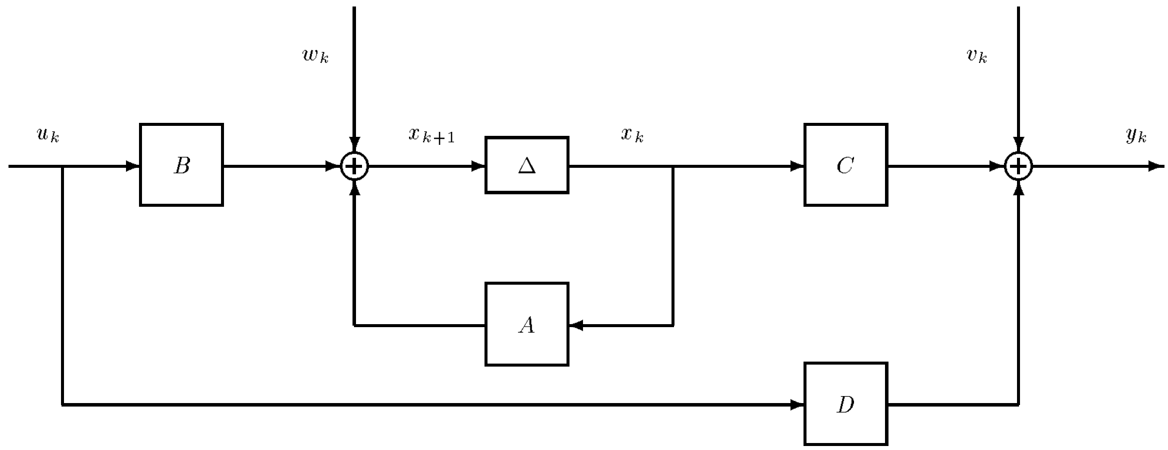

Figure 2.2 This picture is the same as the one in Figure 2.1. But here, we have restricted ourselves to finite dimensional linear time invariant systems to be identified. The vector signals u_k and y_k are available (measured) while v_k, w_k are unknown disturbances. The symbol Δ represents a delay. Note the inherent feedback via the matrix A (which represents the dynamics). Sensor or actuator dynamics are completely contained in A too. It is assumed that u_k is available without measurement noise.

approach [11]). The point here is that more often than not, we don't care about the precise origin of the dynamic modes, since, if they are important, they will certainly influence the controller action, independent of their origin. There is a modern trend in CACSD to define what is called a *standard plant* (see e.g. [12]), which contains the model of all disturbances, all sensors and the system model in one general state space description, which exactly corresponds to the model we will use.

- The assumption that the noise sequences v_k and w_k are Gaussian is quite natural for many applications, due to the central limit theorem (which here acts as an important engineering simplification). Of course, it can not always be made uncriticaly. Nevertheless, the approximation is often very satisfactory.
  Measurement noise (which is ubiquitous in industrial environments) is included in the stochastic white noise sequence v_k, while it is assumed that the input is applied to the system without distortion (in other words, we do not assume that the sequence u_k is corrupted by noise).

- A crucial question is of course why linearity would apply to everyday processes, since we all know that most phenomena are intrinsically non-linear. One reason is the experience that many industrial processes are really well approximated by linear finite dimensional systems and that sometimes, complex behavior can be captured by choosing the order n high enough. In order to cope with non-linearities, two measures are possible: Either the non-linearity is dealt with by identifying a time-varying system using a recursive updating of the model. This corresponds to a local linearization of the nonlinear system. A second possibility is provided by the observation that (mild) nonlinearities do not matter as they can be incorporated in the control design (robustness for dynamic uncertainties). Moreover, it is well known that a controller effectively linearizes the behavior of a system around a working point. Finally, we recall that the design of a controller is relatively easy for linear finite dimensional systems. As a matter of fact, this is the only class of systems for which CACSD is actually tractable in full generality and for which there is a complete rigorous theory available.

We are now ready to state the main mathematical problem of this memorandum:
Given input and output measurements $u_1, \ldots, u_N$ and $y_1, \ldots, y_N$, $(N \rightarrow \infty)$.
Find an appropriate order n and the matrices $A, B, C, D, Q_s, R_s, S_s$.

2.2.2 How do N4SID algorithms work ?

The goal of this subsection is to provide a verbal description of the main principles on which **N4SID** algorithms are based. For the fine mathematical details and elaborate proofs of the claims that are made here, the reader is referred to our scientific publications (see Subsection 2.2.4).

N4SID algorithms are based on concepts from system theory, (numerical) linear algebra and statistics, which is reflected in the following table that summarized the main elements.

System	Geometry	Algorithm
Determination of the order and the state	Projection (orthogonal or oblique)	QR-decomposition
	Determine finite dimensional subspace	(Generalized) singular value decomposition
System matrices	Linear relations	Least squares

The main conceptual novelties in **N4SID** algorithms are

1. The *state of a dynamical system* is emphasized in the context of system identification, whereas 'classical' approaches are based on an input-output framework. The difference is illustrated pictorially in Figure 2.3. This relatively recent introduction of the state into the identification area may come as a surprise since in control theory and the analysis of dynamical systems, the importance of the

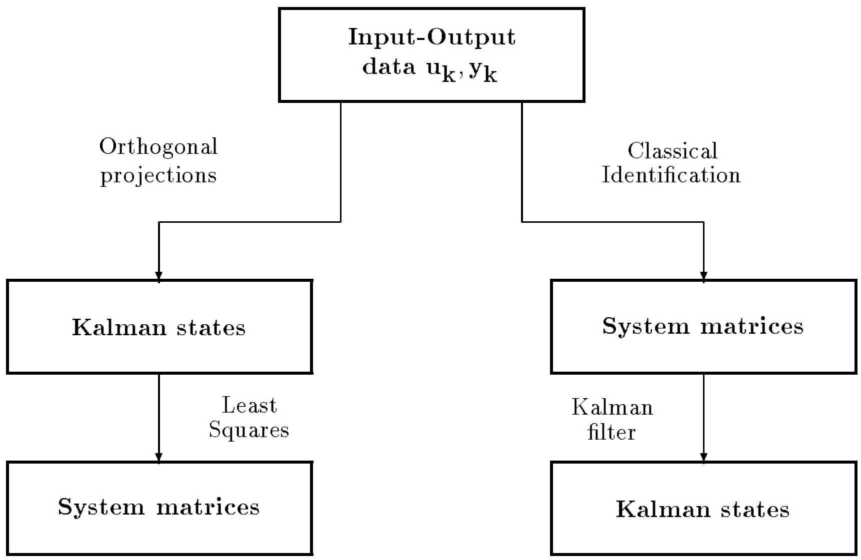

Figure 2.3 System identification aims at constructing state space models from input-output data. The left hand side shows the **N4SID** approach : first the (Kalman filter) states are estimated directly from input-output data, then the system matrices can be obtained. The right hand side is the classical approach : first obtain the system matrices, then estimate the states.

concept of state has been appreciated for quite some time now.
So an important achievement of the research in **N4SID** is to demonstrate how the Kalman filter states can be obtained from input-output data using linear algebra tools (QR and Singular Value Decomposition). An important consequence is that, once these states are known, the identification problem becomes a linear least squares problem in the unknown system matrices. This implies that one possible interpretation of **N4SID** algorithms is that they *conditionally linearize* the problem, which, when written in the 'classical' form of Prediction Error Methods, is a highly nonlinear optimization problem. Yet another point of view is that **N4SID** algorithms do not identify *input-output* models, but they identify *input-state-output* models.

2. Our system identification approach makes full use of the by now well developed body of *concepts and algorithms from numerical linear algebra*. While classical methods are basically inspired by least squares, our methods use 'modern' algorithms such as the QR-decomposition, the singular value decomposition and its generalizations, and angles between subspaces.
This is for instance illustrated by the fact that in **N4SID** the implementation of the algorithms is the same for SISO (single-input single-output systems) as for MIMO (multiple-input multiple-output).

3. Our approach provides a *geometric framework*, in which seemingly different models are treated in a unified manner. We think that the conceptual and algorithmic simplicity of our algorithms should be confronted with and compared to the sometimes extremely complicated and cumbersome arguments and approaches that are often found in present day system identification literature.

4. The conceptual straightforwardness of our algorithms translates into *user-friendly software implementations*. To give only one example: Since there is no explicit need for parametrizations in our geometric framework, the user is not confronted with highly technical and theoretical issues such as canonical parametrizations, and hence, at the level of possible choices to be offered by the software, we get *efficient* implementations.

2.2.3 What's new in N4SID ?

The mathematical engineering field of system identification has begun the blossom some 15 years ago with the work of Aström [7] [8], Box/Jenkins [11], Eykhoff [29], Ljung [52] (and many others, see e.g. [64] [69]). So it is a relatively young branch of research, the industrial spin-offs of which become only gradually visible now.
In this Subsection, we confront the innovations in **N4SID** with the properties of these 'classical' approaches.

Parametrizations: When viewed as a data fitting problem, it becomes clear that system identification algorithms require a certain user-specified parametrization. In **N4SID** we use full state space models and the only 'parameter' is the order of the system. For classical algorithmic approaches however, there has been an extensive amount of research to determine so-called *canonical* models, i.e. models with a minimum number of parameters (see e.g. [33] [38] [39] [41] [46] [54] [73]). There are however many problems with these minimal parametrizations.

- They can lead to numerically ill-conditioned problems, meaning that the results are extremely sensitive to small perturbations.
- There is a need for *overlapping* parametrizations, since none of the existing parametrizations can cover all possible systems. This implies that the user is

confronted with extremely difficult decision problems when trying to identify a linear system.

- Only minimal state space models, with initial state equal to zero are really feasible in practice. If there are for instance uncontrollable but observable (deterministic) modes, this requires special parametrizations.

The **N4SID** approach does not suffer from any of these inconveniences. The only parameter to be user-specified is the order of the model, which can be determined by inspection of certain singular values.

Convergence: When implemented *correctly*, **N4SID** algorithms are **fast**, despite the fact that they are using QR and Singular Value Decompositions. As a matter of fact, they are faster than the 'classical' identification methods, such as Prediction Error Methods, because they are not iterative. Hence there are also no *convergence* problems. Moreover, numerical robustness is guaranteed precisely because of these well-understood algorithms from numerical linear algebra. As a consequence, the user will never be confronted with hard-to-deal-with-problems such as lack of convergence, slow convergence of numerical instability.

Model reduction: Since our main interest lies in using these models in a Computer Aided Control System Design environment and because, when using linear theories, the complexity of the controller is proportional to the order of the system, one is always inclined to obtain models with as low an order as possible. Here, there is a fundamental difference between classical identification algorithms and **N4SID**. Classical algorithms will first obtain an high order system, and then apply model reduction algorithms (such as balanced realization [48] [62] or Hankel norm approximation [34]). In **N4SID**, the reduced model can be obtained *directly*, without having to compute first the high order model, and this directly from input-output data. This is illustrated in Figure 2.4. The interpretation is straightforward within Enns's [28] weighted balanced reduction framework.

2.2.4 Some historical elements and our publications

In this Subsection, we first give an historical survey of the several concepts that are present in **N4SID** and that make it to be one of the most powerful and sophisticated identification frameworks that is presently available. Then we give a short description about the origin of the work and related research.

The following table summarizes in a schematic way the different hallmark contributions and mathematical elements that have lead to and/or are incorporated in some way or another in **N4SID**. The idea is twofold: First of all, this table teaches us that certain concepts, such as e.g. angles between subspaces (Jordan, 1875) or the Singular

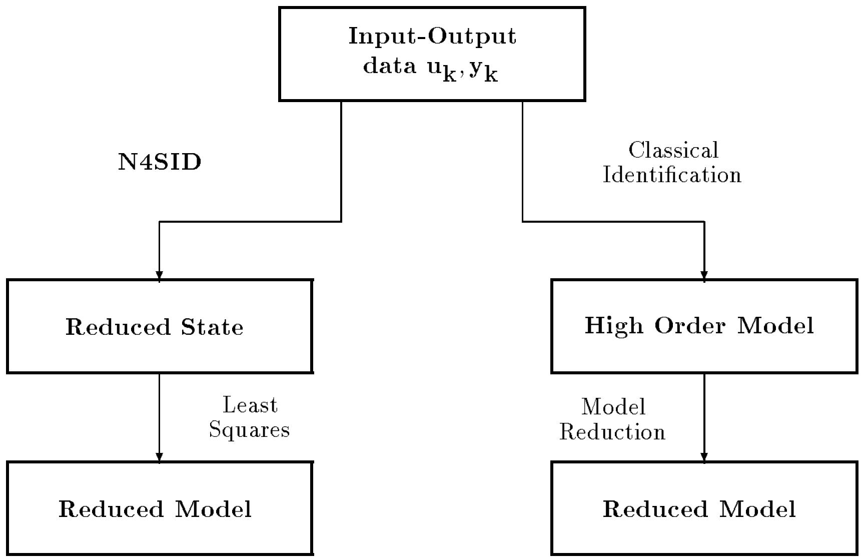

Figure 2.4 System identification aims at constructing state space models from input-output data. When a reduced order model is required, in the classical approach (to the right), one first identifies an high order model and then applies a model reduction technique to obtain a low order model. The left hand side shows the **N4SID** approach: Here, we first obtains a 'reduced' state sequence, after which one can identify directly a low order model.

Value Decomposition (Beltrami, Jordan, Sylvester, 1880's) need a long *incubation* period before they are applied in mathematical engineering. Secondly, it shows how clever combinations of seemingly unrelated concepts and techniques may lead to powerful algorithms, such as **N4SID**.

Of course, space does not permit us here to discuss these contributions in detail, but many of the papers we refer to are publically accessible and very interesting to read.

Let us now summarize the main direct sources of inspiration for our work in **N4SID**. First of all, **N4SID** algorithms are the input-state-output generalizations of the classical realization theory and algorithms of the seventies, which identify a state space model from impulse responses (Markov parameters), such as [25] [26] [42] [48] [62] [63] [70] [94]. The insights obtained in these works have really enhanced the understanding of the *structure* of linear systems and their identification. The first papers on obtaining models from input-output data which have influenced our work are [13] [36] [51] but more recently, also the work by Willems [93] was influential for the deterministic parts. Meanwhile, there were other insights obtained in a more statistically oriented context, such as the work by Akaike [1] [2], which introduced canonical correlations in the stochastic realization framework. Other influential work was done in [5] [6] [14] [23] [24] [30]. Related ideas on the combined stochastic-deterministic problem can be found in [50] [49] [87] [88].

Our work evolved from the development of subspace algorithms in a purely deterministic context [15] [16] [17] [18] [19] [20] [55] [59] [60] [61], over the purely stochastic problem [74] [75] [76], until we managed to combine the two approaches in one unifying framework [21] [77] [78] [80] [82] [83], finally emerging into software implementations [3] [81], which have been applied to real industrial processes [79] [84]. This work is however not yet completed, as is shown in the following Subsection.

Year	Name	Contribution	Discipline	Refs.
1809	Gauss	Least Squares	Statistics	[32]
1873	Beltrami	SVD	Algebra	[10]
1874	Jordan	SVD	Algebra	[44]
1875	Jordan	Angles between subspaces	Algebra	[45]
1883	Gram	QR	Algebra	[37]
1885	Sylvester	SVD	Algebra	[72]
1907	Schmidt	QR	Algebra	[68]
1913	Autonne	SVD	Algebra	[9]
1936	Eckart	SVD	Physics (!)	[27]
1936	Hotelling	Canonical correlations	Statistics	[43]
1960	Kalman	Kalman Filter	System Theory	[47]
1965	Golub/Kahan	SVD-algorithms	Numerical lin.alg.	[35]
1966	Ho/Kalman	Realization	System Theory	[42]
1974	Zeiger/McEwen	SVD & Realization	System Theory	[94]
1974	Akaike	Stochastic Realization	Statistics	[1] [2]
1976	Box-Jenkins	Box-Jenkins models	Statistics	[11]
1976	Faure	Stochastic linear systems	System Theory	[30]
1978	Kung	Realization theory	System theory	[48]
1987	Ljung	Prediction Error	System Theory	[52]
1987	Willems	Behavioral framework	System Theory	[93]

2.2.5 Further future developments

In this Subsection we try to assess the impact of the work on possible future research activities. At the same time this list shows that there is still much work to be done to further enhance **N4SID** methods.

- Let us point out that some of the ideas in this work have been extended to *descriptor* (singular) systems (see [61]), direct identification of continuous time systems (see

[59]), identification problems with known noise structure ([56]).

- Despite the fact that **N4SID** algorithms are quite fast, their speed can even be enhanced by exploiting the (block)Hankel structure of the input-output matrices (these are matrices of so-called low displacement rank). First results have been obtained in [91] [92].

- Another idea concerns the parallel implementation on fast arrays of processors, such as for instance systolic arrays. It turns out that there exist very elegant systolic array implementations for QR and SVD updating (see e.g. [57]).

- Much work remains to be done on the statistical analysis of **N4SID** algorithms. It turns out that the results are quite robust when compared to the maximum likelihood solution, but as of know, there is not a real good explanation for this behavior. Preliminary results have been obtained within the framework of array signal processing [65] [66] [67] [71] [89] [90]. A detailed statistical analysis could also lead to statistical order determination (in the sense of AIC (Akaike's Information Criterion) or Rissanen's MDL (Minimum Description Length).

- Some work needs to be done on the precise relationships of the model reduction interpretation of **N4SID** and the work by Enns [28] and Glover [34]. It would be nice if hard bounds could be obtained for the models delivered by **N4SID**.

- A precise analysis is to be made of the use of **N4SID** within the context of time-varying linear systems. Such systems occur for instance when nonlinear systems are linearized around a certain working point, which itself changes in time.

2.3 Conclusions

In this Chapter, we have highlighted the *scientific* contributions of our work, which can be summarized as follows: We have tackled the problem of multivariable system identification for multiple input multiple output, linear, combined deterministic-stochastic systems. Such models often provide good engineering models for real industrial plants (as will be illustrated in Chapter 4), especially for design of model-based controllers.
By combining insights, concepts and algorithms from system theory, (numerical) linear algebra and statistics, we have developed a new breed of system identification techniques, called **N4SID**, that do not suffer from the disadvantages of 'classical' identification approaches.

Let us conclude with two quotations from some survey papers. In the recently held Workshop on Future Directions in Circuits and Systems [31], it is emphasized that *.. these matrix-based signal processing algorithms are becoming increasingly important,*

(....) and need to be blended with traditional algorithms in a compatible and complementary way.
Prof. Lennart Ljung, one of the international experts in system identification, in his 1991 survey on Issues in System Identification [53], points out that it remains to be *established what these signal subspace methods have to offer and how they compare to conventional approaches....*

We hope that with this work we have bridged a little bit of this gap, a hope which is largely confirmed by the 1993 quote of the same Lennart Ljung at the beginning of this Chapter.

Chapter 3

A Graphical User Interface for System Identification

Less is more !
Prof. Stephen Boyd, Stanford University, August 1993
(while referring to GUIs).

In this Chapter, we first explain in Section 3.1. the trend in present days software to build work with Graphical User Interfaces (GUI). In Section 3.2., we describe our GUI, which was developed to do interactive system identification in an ultimately user friendly manner. It is called **ISID II**, which stands for *Interactive System Identification II* [1]. We describe the main features of **ISID II** while in Section 3.3. we give an overview of the algorithms that are implemented in **ISID II** (preprocessing, identification, validation and display). In Section 3.4. we summarize some statistical information on **ISID II** while concluding remarks can be found in Section 3.5.
Since it is impossible to summarize all the features of **ISID II**, let alone that we could visualize all its graphical functionalities, we would like to refer the interested reader to the **ISID II** manual [85] or just to have a feel-and-touch experience by having a free try-out session on the software itself. Moreover, it might be interesting to have a look at Chapter 4 too, since there we show the GUI in action in identifying a glass tube manufacturing process.

[1] **ISID II** was developed in Xmath, a trademark from Integrated Systems Inc., Santa Clara, California, USA, and is the successor of ISID I (which is a command-line user interface package for interactive system identification, see [4] for more details about **ISID I**).

3.1 Why A Graphical User Interface ?

To motivate the use of a Graphical User Interface (GUI) for system identification, a short history of user-interfaces is presented. This overview is confined to the history of user interfaces for identification and control of processes, which can be split up in three stages: Program User Interface, Command-line User Interface and Graphical User Interface. An overview of the discussion is given in Table 3.1.

Program User Interface: This is the era of lower level programming languages (Fortran, Pascal and C). Typically, the user-input consisted of programs with low level commands. To solve a problem, libraries of applicable functions had to be written. These functions were then combined to a program. The programming had to be done on a very low level, and it was only at the end, when all the bits and pieces were put together, that the results were obtained. If the results were not satisfactory, parts of the programs had to be rewritten. Due to the inflexibility of the programs, most of the time was spent programming. Investigation of the influence of different parameters (and different methods) on the result was hard and time consuming. Especially the intermediate bookkeeping tasks were very tedious.

Command-line User Interface: The second type of user interfaces is the command-line interface. This interface allows the user to enter commands at the command-line. Contrary to the previous generation, the effect of the commands could immediately be inspected. The commands also became more powerful (higher level). This made writing programs and experimenting with different methods less time consuming. Another feature was the bundling of programs into so called Toolboxes. These bundles contained all the necessary "tools" to solve a whole class of similar problems. There was for instance a Toolbox for control problems and one for identification problems. However, due to the complexity of the commands it was not easy for a novice user to start using these Toolboxes. First, she had to become familiar with all methods implemented in the Toolbox. Then, she also had to understand and study the sometimes complicated syntax of the commands. Finally, she had to understand how to *interconnect* the commands into a program that would solve his problem.

Typically the user had to read thick manuals with extended syntax conventions before she could start. On top of that, after she had read the manuals, it was not always clear how to solve her problem. This is because there was hardly any user guidance available (apart from the examples in the manuals). A thorough understanding of the implemented methods was thus still needed to use the Toolbox. Figure 3.1 shows a typical "Help" window of the Command-line User Interface software generation.

Even though a lot easier to use and more flexible than the Program User Interface programs, this new generation still had significant drawbacks i.e. the extended syntax, the complicated interconnections of commands and the lack of user-guidance.

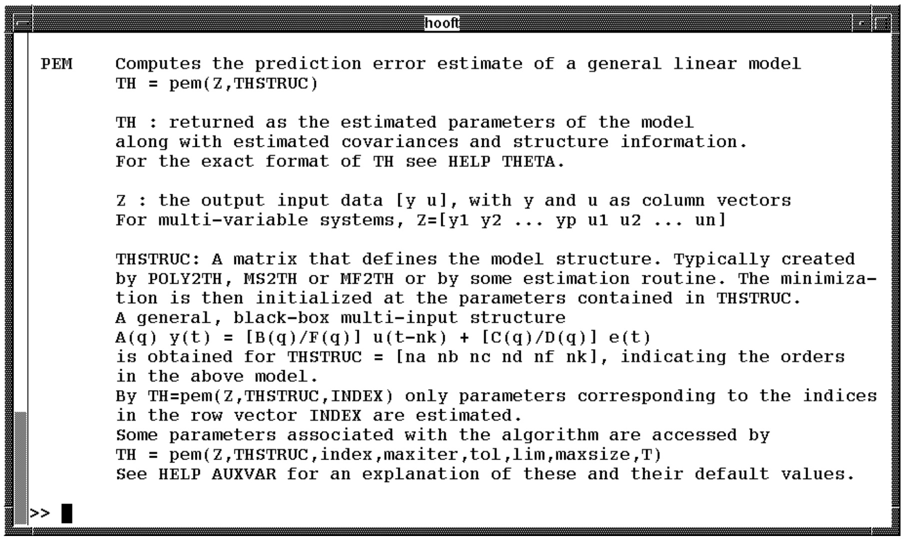

Figure 3.1 A typical help window for the command-line interfaces. While trying to solve the problem, the user often had to refer back to the cryptic on line help provided by the programs. Clearly, this "help" is hard to understand without extensive a-priori knowledge. The net result was that the threshold to start using this software intensively was quite large.

Graphical User Interface: The most recent interface is the Graphical User Interface (GUI). A GUI is a user interface made up of graphical objects such as menus, buttons and plots. Using it is straightforward since it only requires manipulation of the three mouse buttons and at rare occasions, typing in the name of an object or data file.

A first feature of a GUI is that it makes thick manuals virtually obsolete. Most graphical objects are clearly labeled so that their function is immediately clear. There is no need to study complicated syntax. An overview of the functionality can be found by browsing through the menus of the interface.

Another GUI feature is that the effect of changes in parameters (or methods) is depicted graphically. In the previous generation, the results were obtained as variables. These variables had to be transformed to figures to be interpreted. This made it necessary to add extra visualization commands. A GUI presents all results graphically, which excludes this last step. and which turns it into an elegant tool to perform varying parameter experiments.

On top of that, a GUI for identification and control system design provides the

	Program User Interface	Command-line User Interface	Graphical User Interface
Input	Programs	Command-line	Graphical
Output	Text	Variables	Graphical
Elementary Block	low level command	high level command	window
Start Threshold	very high	high	low
Flexibility	low	high	high
User guidance	none	limited	high
Syntax	complex	complex	simple
Expertise required	high	high	limited
Computer required	simple	simple	fast
Date	±1960-1982	1982-1990	1990-?

Table 3.1 Comparison of the distinct stages in the history of identification and control software.

user with guidelines to solve her problem. By equipping the GUI with a certain intelligence, (highlighting certain menus, buttons and plot handles), the user is guided through the interconnection of complicated functions. This interconnection is also graphically depicted, which enables the user to retain a clear overview (see for instance Figure 3.2).
A GUI for system identification enables a novice user to get acquainted with the software without the need for thick manuals or extensive (identification) expertise. In the next sections these advantages will become even more apparent.

3.2 ISID II: Where system identification and GUI meet

The combination of powerful numerical algorithms for system identification and a Graphical User Interface leads to intelligent and user-friendly identification software. The **I**nteractive **S**ystem **ID**entifiction software **ISID II**, which we have developed, contains 3 major concepts: the **ISID II** chain, the **ISID II** workspace and the **ISID II** algorithm windows. These concepts are now briefly reviewed.

Typically, an identification process consists of different actions that are executed one after the other. The classical steps are for instance: processing of the data, identification, validation. Each of these steps can itself be split up in a series and/or parallel connection of algorithms. Each algorithm is represented by a block. Putting these algorithm building blocks in a connected graph, leads to a “chain” of algorithms. This **ISID II** chain is the first major concept in **ISID II**.

The **ISID II chain** consists of an interconnection of algorithm building blocks and represents a user-defined identification process.

Figure 3.2 displays a typical chain. **ISID II** allows for *graphical programming* of the chain, which means that only the manipulation of mouse buttons is required to build a chain and tune its parameters. The algorithms (the building blocks) can be easily selected from the menus. They can be connected via a simple graphical interaction. The user is guided in making these connections since she can only connect compatible blocks (the output has to be compatible with the input).
Each of the algorithm building blocks contains a sophisticated algorithm, the parameters of which are automatically set to a meaningful default value. The user doesn't have to know the details of the algorithm, of which only the most important parameters are readily available for direct manipulation.
ISID II contains several *hard baked* chains, which represent programs that perform the most common identification procedures. For instance, the identification from filtered input output data using a least squares algorithm, followed by a cross validation of the obtained model. In this way, the novice user is given a guideline of how to use the identification algorithms, without the requirement of having any identification expertise and without having to read through the manual.

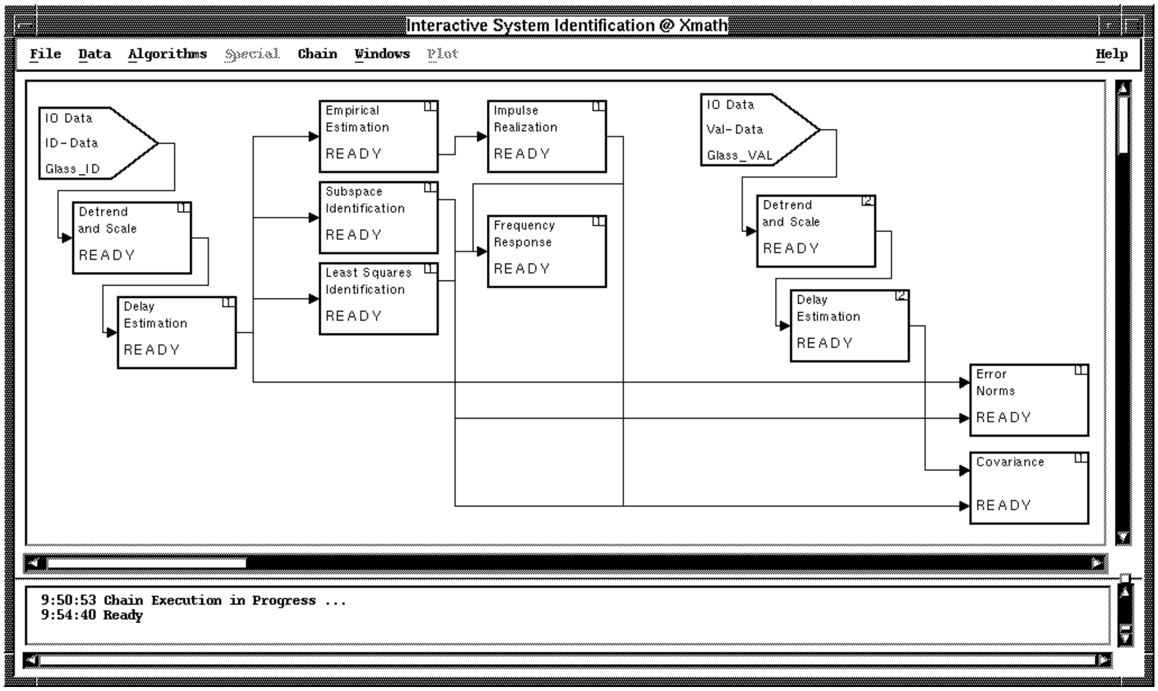

Figure 3.2 A typical **ISID II** chain representing a full identification process. Each block represents an algorithm (see Section 3.3). The blocks to the left are the input-output data and preprocessing blocks, followed by the identification blocks. To the left, we also see some validation blocks. Behind each block there is an algorithm window that visualizes the algorithm specific data and allows for adjustment of the parameters (see e.g. Figure 3.4.). This chain shown here in this Figure was used for the identification of a glass tube manufacturing process (see Chapter 4).

The **ISID II workspace** contains user defined data objects.

System identification typically requires different data objects. The most common ones are input-output data records, frequency and impulse response records and models. Each algorithm building block takes one type of object as its input and returns one object type as its output. Moreover, user defined data is read into **ISID II** as one of these objects, after which it is stored in the **ISID II** workspace. Outputs of algorithm-blocks can also be stored in this workspace. The workspace (Figure 3.3) is the second major concept in **ISID II**.
The start of the chain consists of special boxes that can load objects. These are the data boxes. For every type of data object there is a type of data box. These data boxes represent the "inputs" of the chain. Data objects from the workspace can be assigned

to these data boxes (data box type and object type should be compatible, which is verified automatically). In this way, user defined data (or previously saved outputs from algorithm building blocks) can be presented to the data boxes and thus to the inputs of the chain.

Figure 3.3 The **ISID II** workspace consists of a list of data objects. These data objects are loaded from user defined data. Alternatively they are outputs of algorithm blocks that were saved to the workspace. Through the data boxes, the objects in this list can be used as input to the chain. Recall that all of this can be done in an utmost elegant manner by just dragging and clicking with the three available mouse buttons.

The **ISID II algorithm windows** visualize the algorithm specific information. Algorithm parameters can be graphically adjusted.

System identification is intrinsically visualization intensive, given the large amount of information that needs to be processed (think about long measured data records, frequency responses, etc). Behind each algorithm building block there is an **ISID II** algorithm window (Figure 3.4). These algorithm windows represent the third major concept in **ISID II** as they visualize algorithm specific data. Moreover in the algorithm windows, parameters can be adjusted graphically by pushing buttons, dragging weight functions, clicking on singular value and error norm plots (order selection), adjusting peak shave levels and run time delays. The **ISID II** algorithm windows also facilitate visualization of the data by easy zoom in (magnifying glasses) and zoom out facilities on all plots.

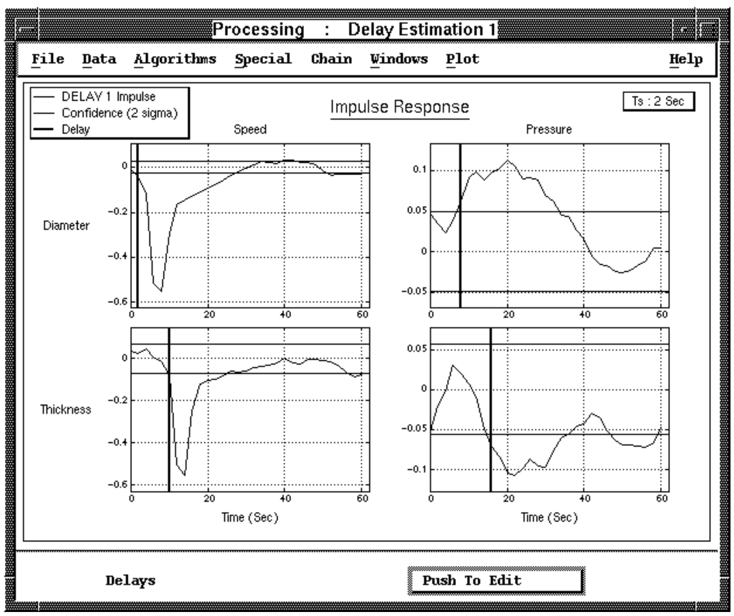

Figure 3.4 Behind each algorithm building block there is an algorithm window. These **ISID II** algorithm windows display algorithm specific data. These algorithm windows allow the user to change parameters graphically. Shown here is the window that goes with the delay estimation algorithm. The delays are indicated by the vertical lines. They can be changed by clicking on the vertical lines and dragging them to the desired value i.e. the intersection of the impulse response and the confidence bounds (horizontal lines). This contrasts with the Command-line User Interface where the user had to read the intersecting point from the scales.

The **ISID II software** allows for easy identification of Models.

Identifying a model with the **ISID II** software consist of the following steps:

- Build an **ISID II** chain or use one of the *hard baked* **ISID II** chains.
- Load the user data as an object in the **ISID II** workspace.
- Use the user defined object as an input to the chain.

Every algorithm in the chain will then successively execute, and automatically open up its corresponding algorithm window and calculate its output. The influence of algorithm parameters can now be inspected. The parameters can be changed through graphical interaction. The effect of these changes ripples through the chain, and all algorithm windows that depend on the changed data re-execute and update their plots.
There is also a *History window* which keeps track of all the experiments that have been performed and therefore acts as a bookkeeper.

3.3 An overview of ISID II algorithms

Each of the **ISID II** algorithm building blocks takes a data object as input and returns another data object at its output. The four data object classes (and their abbreviations) are: Input-Output Data (IO), Frequency Response (Freq), Impulse Response (Imp) and Models (Model).
There are 4 different algorithm building blocks, namely pre-processing, identification, validation and display algorithms. Since obviously we can not describe all of these functionalities in full detail (for which we would like to refer to interested reader to the manual [85] or just invite you to have a free try out of the software itself), we will restrict ourselves here to a mere enumeration of all the possibilities in the algorithm building blocks. In each of the following tables, the first column contains the name of the functionality, the second column is the data object that acts as an input, the third column the data object that is delivered as an output while the fourth column is a one-line description of the functionality.

3.3.1 (Pre-)Processing

Industrially measured data sets often need to be *pre-conditioned*. This processing makes the measured signals suitable for identification. The processing algorithms are:

Name	Input	Output	Functionality
Detrend and Scale	IO	IO	Scaling and trend removal (drift ...)
Peak Shaving	IO	IO	Removal of outliers (sensor failure ...)
Delay Estimation	IO	IO	Estimation and extraction of delays.
Filtering	IO	IO	Low, high or band pass filtering.
Post Sampling	IO	IO	Subsampling of a data set.
View and Split	IO	IO	Split up into Identification and Validation set.

3.3.2 Identification

ISID II contains a whole range of identification algorithms. Apart from the "classical" identification algorithms there are also two subspace identification algorithms (**N4SID**) implemented, which are presently the most powerful identification algorithms for multivariable linear systems and have been described extensively in Chapter 3. The collection of identification algorithms is the following:

Name	Input	Output	Functionality
Least Squares	IO	Model	Least Squares in time domain
Subspace Identification	IO	Model	**N4SID**
Instrumental Variables	IO	Model	Uses inputs as instruments.
Prediction Error Method	IO	Model	Classical cost function minimization.
Time Series	IO	Model	**N4SID** modeling of time series.
Empirical Estimation	IO	Freq & Imp	Non-parametric identification.
Frequency Least Squares	Freq	Model	Least squares in frequency domain.
Impulse Realization	Imp	Model	Realization of impulse responses.

3.3.3 Validation

The validation algorithms of **ISID II** allow to asses the "quality" of the identified models. This is done by inspection of different properties of the prediction errors. The validation algorithms are:

Name	Input	Output	Functionality
Error Norms	IO & Model	-	Displays prediction error norms.
Prediction	IO & Model	IO	Predicted signals.
Prediction Errors	IO & Model	IO	Prediction errors.
Covariance	IO & Model	Imp	Covariance of prediction errors.
Spectral Density	IO & Model	Freq	Spectral density of prediction errors.
Cross Correlation	IO & Model	Imp	Correlation inputs ↔ prediction errors.

3.3.4 Display

These algorithms are intended to display properties of the identified model. The model properties can then be interpreted by the engineer. The display algorithms are:

Name	Input	Output	Functionality
Frequency Response	Model	Freq	Displays frequency responses.
Impulse Response	Model	Imp	Displays impulse responses.
Spectral Density	Model	Freq	Displays spectral densities.
Covariance	Model	Imp	Displays covariances.
Poles & Zeros	Model	-	Displays poles & zeros.

3.4 Some statistical information about ISID II

ISID II contains more than 80 000 lines of code. The code was written in C (for the intensive computational tasks) and in MathScript.
There are 25 different algorithm windows. Each algorithm can have up to 10 different instances. There is no real danger of having the user confronted with too much information because, when not required, most of the information can be masked from the user when she wishes to do so.
ISID II is a toolbox within Xmath, which is in essence a matrix manipulation language in which also other GUI-based toolboxes are available (such as System Build (for graphically constructing and simulating linear and nonlinear systems), or ICDM for advanced control system design.

3.5 Concluding remarks

GUI driven toolboxes belong to a third generation of user-friendly software packages. Users can tackle more serious problems than previously possible because of the fact they

don't need to bother about programming subtleties. In addition, bookkeeping of tasks to do, of interconnections of models and data sets becomes straightforward. Finally, there is lots of user guidance and as a matter of fact, most of the time default settings are provided (and they represent the most commonly performed actions anyway).
We have been developing a GUI, called **ISID II**, which provides all of these functionalities for a *system identification* environment. Obtaining mathematical models from experimental data from industrial processes now comes within reach of *every* control system design engineer. Therefore, model-based computer aided control system design will become an important trend in the process industry within the next few years.

Chapter 4

Rapid Prototyping: An example

To illustrate the possibilities of **ISID II**, we consider an industrial case study. We will build a state space model (which was described in Chapter 2), from input-output records of a glass tube manufacturing process [1]. At the same time, we will show in some more detail the different GUI functionalities that we have been describing in Chapter 3.

In Section 4.1., we give a short description of the process. In Section 4.2., we build the Chain Window that allows to identify the process and we discuss some intermediate options and results. Although it is not a formal part of this memorandum, we show the results of a control design based on the derived model in Section 4.3., just to illustrate our point how relatively straightforward a controller can be designed, once a mathematical model is available. We end up with some concluding remarks in Section 4.4.

4.1 Problem Description

Figure 4.1 shows a schematic description of the glass tube production process. Quartz sand is fed to the machine at the top. The sand is melted to glass inside the furnace. The glass tubes are drawn at the bottom of the machine.

Inputs: The inputs are drawing speed and mandrill pressure. The drawing speed is the speed at which the tubes are pulled out at the bottom of the machine. The mandrill pressure is the pressure applied to the mandrill at the top of the machine.

[1] The data we use here are **real** industrial measurements on a real industrial commercial production plant. However, due to reasons of <u>confidentiality</u>, we are not allowed to reveal in detail exact physical production parameters.

Through the mandrill, this pressure is then applied to the inside of the tubes when they are pulled out of the machine.

Outputs: The outputs to be controlled are the geometrical parameters of the tube which are its mean diameter and its thickness.

Two input-output data sequences were measured (sampling frequency 2 Hz). The first one (1355 points, see Figure 4.2) is used for the identification of the process. The second one (893 points) is used for the validation of the results. For both input signals a pseudo random binary noise sequence was used as input.

4.2 Chain description and results

One of the main features of **ISID II** described in Chapter 3, was the Chain Window, which graphically represents the chain of algorithms to be executed to find a mathematical model of the process under study. Figure 3.2 shows the chain that was used to identify the glass tube manufacturing process. Figure 4.3 shows the full computer screen during the **ISID II** session, with all the algorithm windows open.

Processing: Both input-output data records are first detrended to remove the mean and the linear trends in the data. Since the measurements of the outputs can only be done when the tubes are sufficiently cooled down, there are significant time delays. The two delay estimation blocks estimate these delays and compensate for them by shifting the input and output signals in the appropriate direction. Figure 3.4 shows the algorithm window behind the delay estimation algorithm block.

Identification: Three different identification algorithms are applied to the data. By comparing the resulting models, we can enhance our confidence in them.

- A first model is obtained by realizing the impulse response obtained from the empirical transfer function.
- A second model is obtained from a subspace (**N4SID**) identification. The algorithm window corresponding to this identification block is shown in Figure 4.4.
- A last model is obtained by a least squares identification.

Validation: The models are validated by comparing the measured and simulated outputs (validation and identification data). Figure 4.5 shows the simulation error norms. The **N4SID** model (middle bar) has the smallest error and is thus used for control design. Through computation of the covariance of the prediction errors, the whiteness of these errors can be checked.

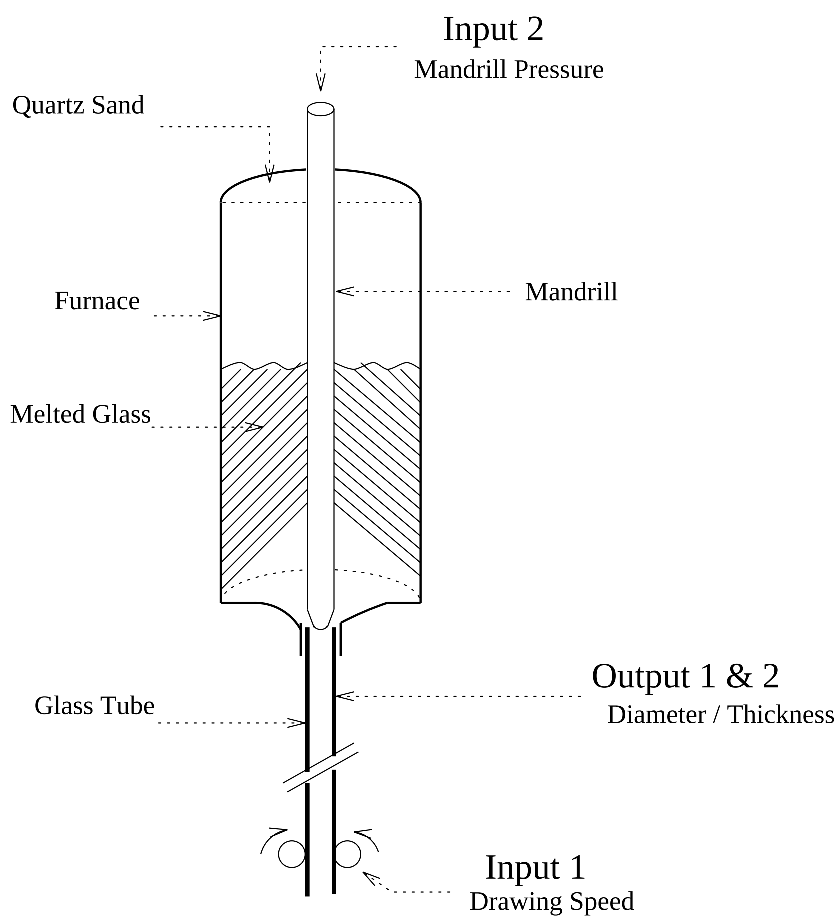

Figure 4.1 The glass tube manufacturing process. Inputs are drawing speed and mandrill pressure. Outputs are tube diameter and thickness. The input signals were pseudo-binary noise sequences. These inputs are sufficiently 'wild' to excite all the dynamic modes of the system (which is necessary since we want to control these modes afterwards). The tubes that are produced from these inputs are worthless since they are too irregular due to the wild inputs. This situation is typical for industrial system identification: There is a basic trade-off between the production loss that goes together with experimenting and the production quality enhancement as a result of the identification/model-based control design.

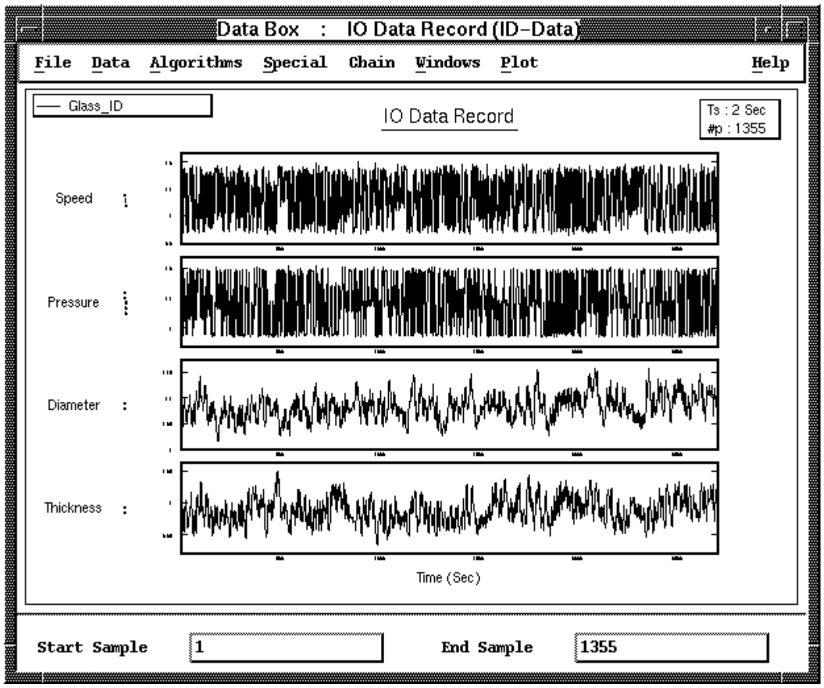

Figure 4.2 Data set used for the identification of the glass tube production process. The inputs (top two signals) are drawing speed and mandrill pressure. They are excited using pseudo-binary noise sequences. The outputs (bottom two signals) are tube diameter and thickness.

Display: The transfer functions of the obtained models are displayed on the same plot to compare the models in the frequency domain. The algorithm window is shown in Figure 4.6.

4.3 PIID control of the process

Even though this is not a part of the GUI software, we describe the result of as controller design to illustrate the use of the model identified using the **N4SID** algorithm. The scheme of the PIID controller is represented in Figure 4.7. The technique we have used was also developed by one of us. It is based on Multi-objective optimization of the free parameters of the controller and is extensively described in [86]. Figure 4.8 shows the closed loop results. Figure 4.9 illustrates the noise reduction and thus the quality enhancement that can be obtained with this controller (in simulation).

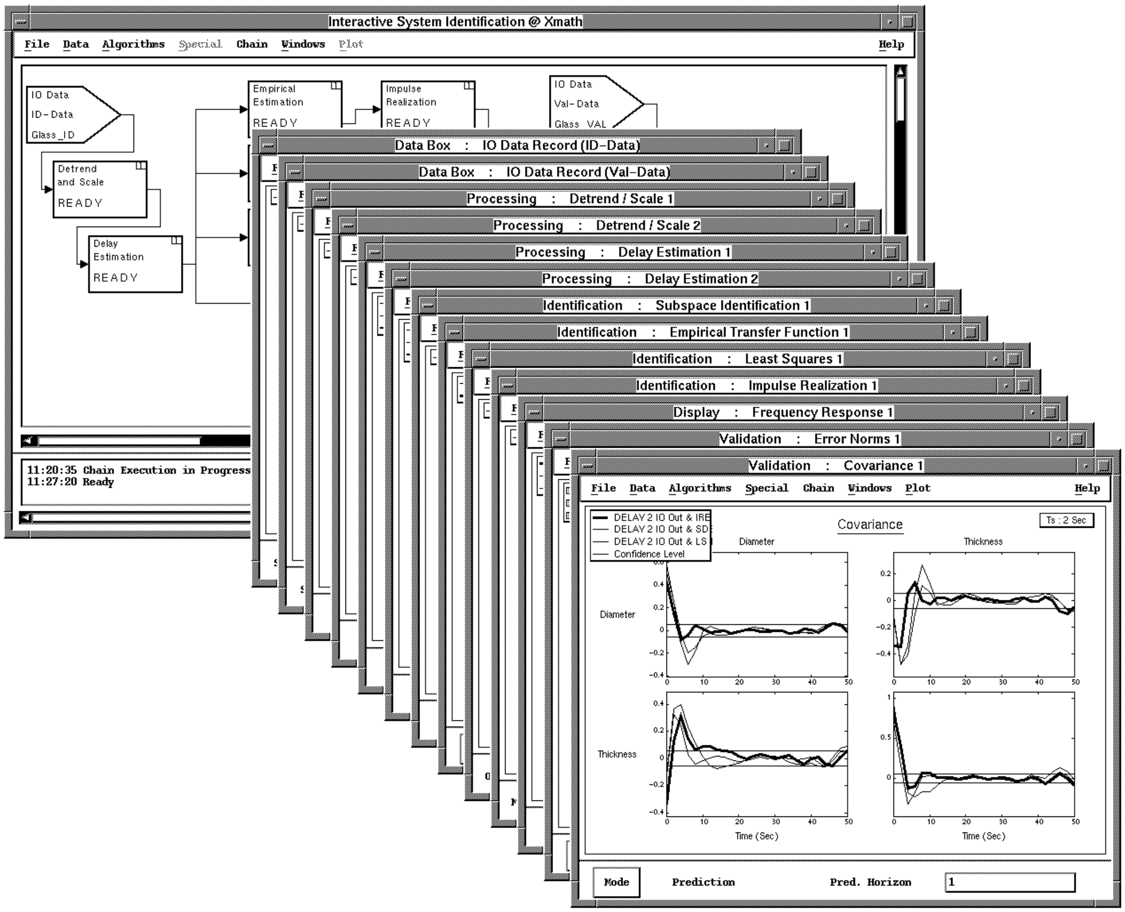

Figure 4.3 An overview of the **ISID II** session for the identification of the glass tube manufacturing process. The Chain Window is displayed in the top left corner (partially covered). All algorithm windows are organized in a "card system". **ISID II** contains several features toy manipulate and reorder this "card system" of windows.

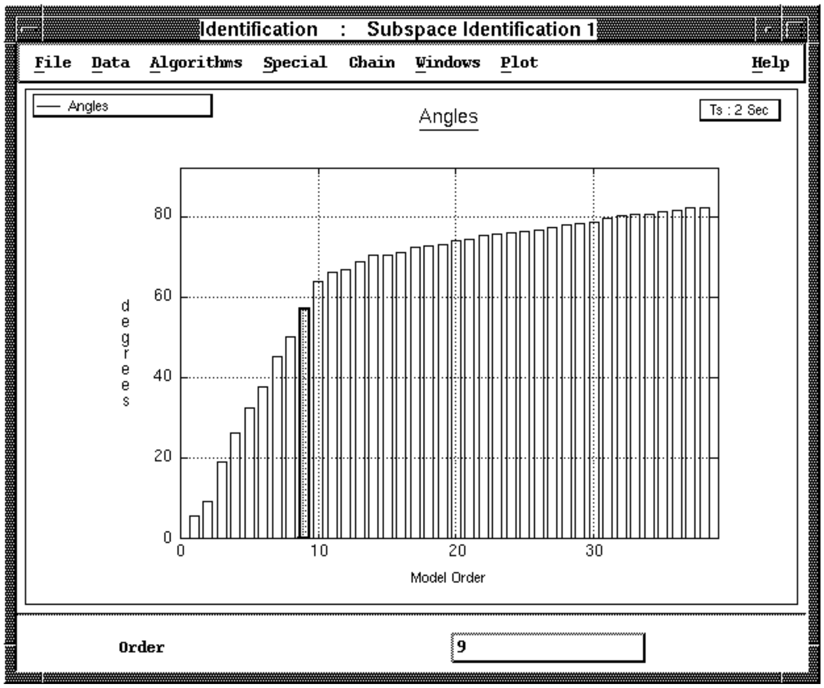

Figure 4.4 The algorithm window behind the subspace identification algorithm (**N4SID**). The plot displays the principal angles. These allow the user to make a decision on the order of the system. The order (9 in this case) can be selected by clicking and dragging with the left mouse over the desired orders.

4.4 Conclusions

In this final Chapter, we have been demonstrating in some detail the efficiency of using the GUI-based software tool **ISID II**. The overall design of a multivariable controller only takes a couple of hours, which creates a lot of time to try to find an optimal tuning and to pay attention to many additional performance and robustness specifications. It is well known in the model-based control system design literature that mathematical modeling of a multivariable system is the most time consuming part of a model-based control system design project. Therefore, using **ISID II**, rapid prototyping of multivariable controllers now comes within reach. Once state space models are available, one can concentrate one's attention to the controller design and, within a spirit of *virtual engineering*, develop and analyze control strategies, starting from the model or from several models.

We have illustrated these points on a *glass tube manufacturing* process. But in the

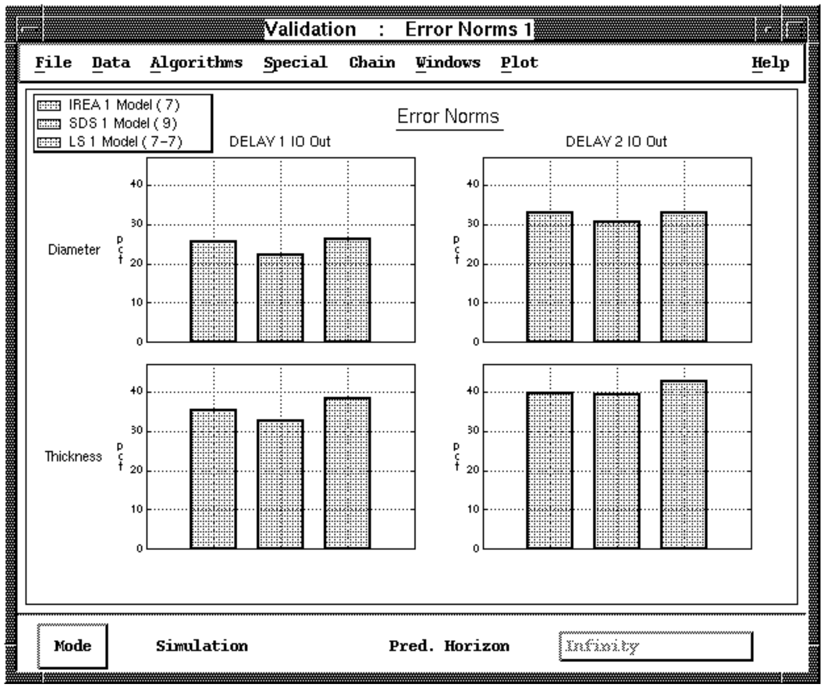

Figure 4.5 Algorithm window corresponding to the error norms algorithm block. The two left hand side plots show the error norms of the models applied to the identification data set (two outputs). The two right hand side plots show the same for the models applied to the validation data set. The three bars correspond to the three models (from left to right: impulse realization, **N4SID**, least squares). The error for the **N4SID** model is the smallest and this model is thus used for control system design.

mean time, we have build up additional equally successful experiences with other industrial production process, including HVAC (heating, ventilation and air-conditioning plants), flexible robot arms, fluttering airplane wings and several others. Others have used **N4SID** algorithms to design multivariable controllers for *Rapid Thermal Wafer Processing* devices [40] [91] [92].

We sincerely hope that in the near future toolboxes like **ISID II** will become indispensable components of intelligent control system design strategies.

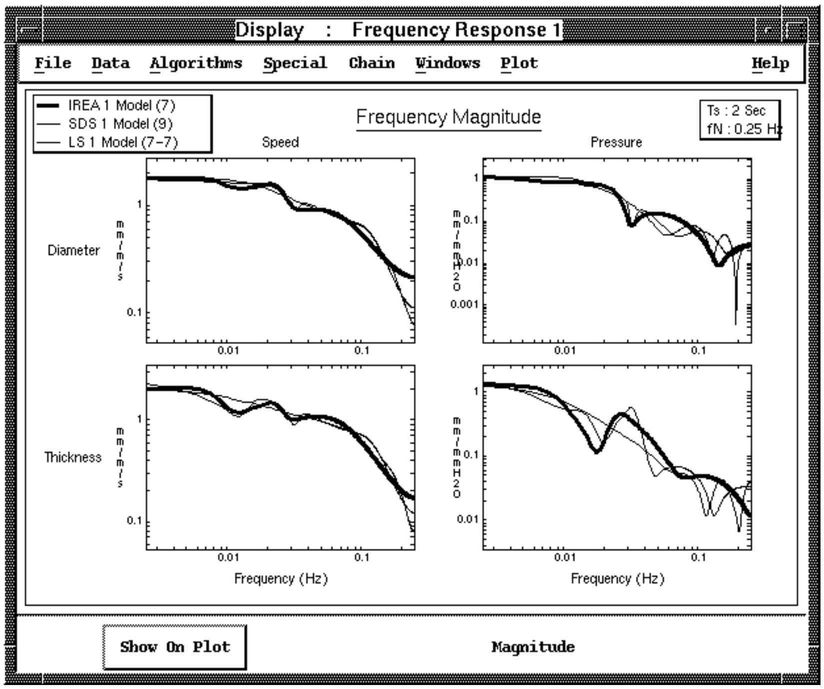

Figure 4.6 Frequency response of the three identified models, superimposed on each other. One can highlight one specific plot, corresponding to one specific model, by clicking with the left mouse button on the model's name. There are also several zoom in features available.

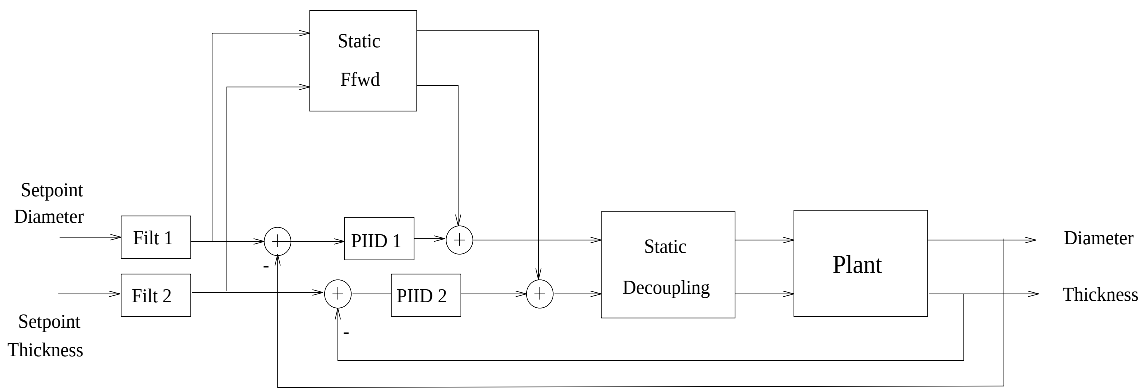

Figure 4.7 The control scheme used for the PIID control of the glass tube manufacturing plant. The controller consists of two feedforward filters, a static feedforward, a static decoupling and two PIID controllers to control the decoupled loops. The PIID parameters are tuned using a multi objective optimization algorithm.

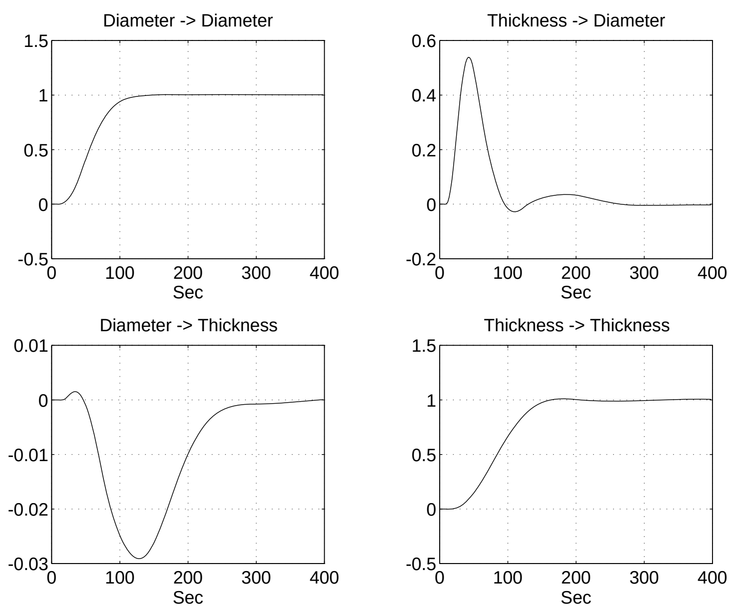

Figure 4.8 The closed loop step response of the PIID controlled glass tube manufacturing process. The PIID controller based on the **N4SID** model results in a smooth step response.

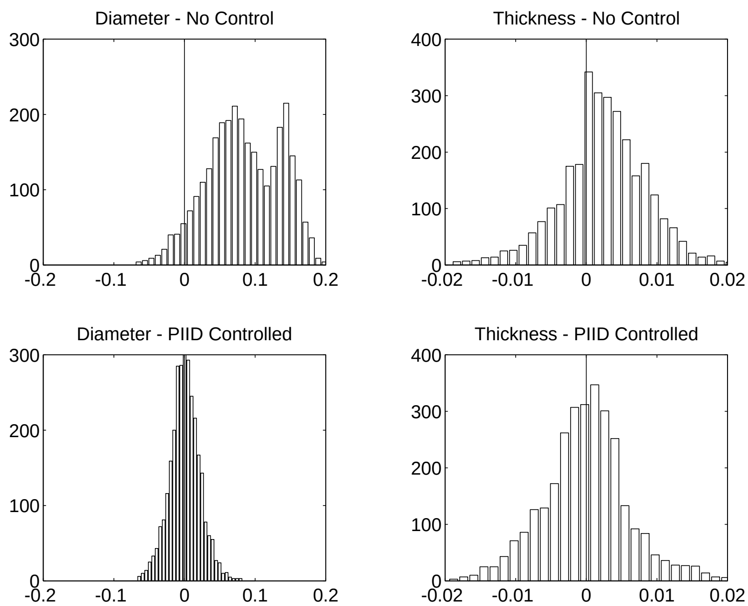

Figure 4.9 Illustration of the quality improvement. The top two figures show a histogram of the measured diameter and thickness. The reference setpoint for production is at zero (the vertical line). Clearly, both diameter and thickness are too large (on average). Especially the diameter does not satisfy the production specifications. The bottom two figures show the histograms of the controlled system. The variance on the diameter is a factor two smaller. The mean diameter is also exactly at its reference. The variance of the thickness is not reduced (not that important in the specifications). However the mean value is right at the specification now. This figure clearly illustrates the benefits of model-based control system design.

Bibliography

[1] Akaike H. *Stochastic theory of minimal realization.* IEEE Transactions on Automatic Control, 19, pp.667-674, 1974.

[2] Akaike H. *Markovian representation of stochastic processes by canonical variables.* Siam J. Control, Vol.13, no.1, pp.162-173, 1975.

[3] Aling H., Milletti U., Kosut R.L., Mesaros M.P., Van Overschee P., De Moor B. *An interactive system identification module for Xmath.* Proc. of the American Control Conference, June 2-4, 1993, San Francisco, USA, pp. 3071-3075

[4] Aling H., Kosut R., Van Overschee P., De Moor B., Boyd S. *Xmath Interactive System Identification Module.* Manual **ISID I** , Integrated Systems Inc., Santa Clara, CA, USA, 1993.

[5] Aoki M. *State space modeling of time series.* Springer Verlag, Berlin, 1987.

[6] Arun K.S., Kung S.Y. *Balanced approximation of stochastic systems.* SIAM J. Matrix Analysis and Applications, 11, pp.42-68, 1990.

[7] Aström K., Eykhoff P. *System identification - A survey.* Automatica, Vol.7, pp.123-167, 1971.

[8] Aström K., Wittenmark B. *Computer Controlled Systems: Theory and Design.* Prentice Hall, 1984.

[9] Autonne L. *Sur les matrices hypohermitiennes et les unitaires.* Comptes Rendus de l'Acadèmie des Sciences, Paris, Vol.156, 1913, pp.858-860.

[10] Beltrami E. *Sulle Funzioni Bilineari,* Giornale di Mathematiche, Battagline G., Fergola E. (editors), Vol. 11, pp. 98-106, 1873.

[11] Box G.E., Jenkins G.M. *Time series analysis, forecasting and control.* Revised edition, Holden-Day series in time series analysis and digital processing, Holden-Day, Oackland, 1976.

[12] Boyd S., Barrat C. *Linear controller design: Limits of performance.* Prentice Hall Information and System Sciences Series, Englewood Cliffs, NJ, 1991.

[13] Budin M. *Minimal realization of discrete linear systems from input-output observations.* IEEE Transactions on Automatic Control, Vol.AC-16, no.5, October 1971, pp.395-401.

[14] Caines P. *Linear Stochastic Systems.* Wiley Series in Probability and Mathematical Statistics, 1988.

[15] De Moor B., Vandewalle J. *A geometrical strategy for the identification of state space models of linear multivariable systems with singular value decomposition.* Proc. of the 3rd International Symposium on Applications of Multivariable System Techniques, pp.59-69, Plymouth, UK, April 13-15 1987.

[16] De Moor B. *Mathematical concepts and techniques for modelling of static and dynamic systems.* PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium, June 1988.

[17] De Moor B., Vandewalle J., Moonen M., Vandenberghe L., Van Mieghem P. *A geometrical approach for the identification of state space models with singular value decomposition.* 1988 Symposium on Identification and System Parameter Estimation, 27-31 August 1988, Beijing China, pp.700-704.

[18] De Moor B., Moonen M., Vandenberghe L. and Vandewalle J. *Identification of linear state space models with singular value decomposition using canonical correlation concepts.* in *'SVD and Signal Processing: Algorithms, Applications and Architectures'*, E. Deprettere (ed.), Elsevier Science Publishers B.V. (North-Holland), 1988, pp. 161-169

[19] De Moor B., Moonen M., Vandenberghe L. and Vandewalle J. *The application of the canonical correlation concept to the identification of linear state space models.* in A. Bensousan, J.L. Lions, (Eds) *Analysis and Optimization of Systems*, Springer Verlag, Heidelberg, 1988, pp. 1103-1114

[20] De Moor B., Moonen M., Vandenberghe L., Vandewalle J. *A geometrical approach for the identification of state space models with singular value decomposition. Proceedings of the International Conference on Acoustics, Speech and Signal Processing 1988*, New York, April 1988, pp. 2244-2247

[21] De Moor B. *Numerical algorithms for state space subspace system identification.* Academia Analecta, Klasse der Wetenschappen, Koninklijke Akademie voor Wetenschappen, België. To be published, 43 pp., 1994.

[22] De Moor B. *The singular value decomposition and long and short spaces of noisy matrices.* IEEE Transactions on Signal Processing, Vol.41, Issue 9, September 1993, pp.2826-2838.

[23] Desai U.B., Pal D. *A transformation approach to stochastic model reduction.* IEEE Transactions on Automatic Control, Vol.AC-29, no.12, December 1984.

[24] Desai U.B., Kirkpatrick R.D., Pal D. *A realization approach to stochastic model reduction.*, International Journal of Control, Vol.42, no.4, pp.821-838, 1985.

[25] Dickinson B., Morf M., Kailath T. *A minimal realization algorithm for matrix sequences.* IEEE Transactions on Automatic Control, Vol.AC-19, no.1., February 1974, pp.31-38.

[26] Dickinson B., Kailath T., Morf M. *Canonical Matrix fraction and state space descriptions for deterministic and stochastic linear systems.* IEEE Transactions on Automatic Control, Vol.AC-19,pp.656-667, 1974.

[27] Eckart C., Young G. *The approximation of one matrix by another of lower rank.* Psychometrika, 1, pp.211-218, 1936.

[28] Enns D. *Model reduction for control system design.* Ph.D.. dissertation, Dep. Aeronaut. Astronaut., Stanford University, Stanford CA, 1984.

[29] Eykhoff P. *System identification.* Wiley, London, 1974.

[30] Faure P. *Stochastic realization algorithms.* in 'System Identification: Advances and case studies'. Eds.: Mehra R., Lainiotis D., Academic Press, 1976.

[31] Final report of the CAS Workshop on Future Directions in Circuits, Systems and Signal Processing, IEEE Circuits and Systems Society Newsletter, Vol.1, No.3, September 1990, pp.14-17.

[32] Gauss C. F. *Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium.* F.Perthes and J.H. Besser, Hamburg, 1809 (reprinted in: Carl Friedrich Gauss Werke, Vol.VII, Königlichen Gesellschaft der Wissenschaften zu Göttingen, translation: Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections, Little, Brown and Co., Boston, 1857)(reprinted: Dover, New York, 1963).

[33] Glover K., Willems J. *Parametrizations of linear dynamical systems: canonical forms and identifiability.* IEEE Transactions on Automatic Control, Vol.AC-19, pp.640-645, 1974.

[34] Glover K. *All optimal Hankel norm approximations of linear multivariable systems and their* L^{∞} *error bounds.* International Journal of Control, 39, pp.1115-1193, 1984.

[35] Golub G., Van Loan C. *Matrix computations.*, Johns Hopkins University Press, Baltimore, Maryland, 1984.

[36] Gopinath B. *On the identification of linear time-invariant systems from input-output data.* The Bell System Technical Journal, Vol.48, May-June 1969, no.5, pp.1101-1113.

[37] Gram J.P. *Über die Entwickelung reeler Functionen in Reihen mittelst der Methode der kleinsten Quadrate.* Journal für die reine und angewandte Mathematik, 94, pp.41-73, 1883.

[38] Guidorzi R. *Canonical structures in the identification of multivariable systems.* Automatica, 11, pp.361-374, 1975.

[39] Guidorzi R. *Invariants and canonical forms for systems structural and parametric identification.* Automatica, 17, pp.177-133, 1981.

[40] Gyugyi Paul. *Model-Based Control Applied to Rapid Thermal Processing.* Phd Thesis Stanford University, CA, USA, June 1993.

[41] Hannan E.J., Deistler M. *The statistical theory of linear systems.* Wiley Series in Probability and Mathematical Statistics, John Wiley and Sons, New York, 1988.

[42] Ho B.L., Kalman R.E. *Efficient construction of linear state variable models from input/output functions.* Regelungstechnik, 14, pp.545-548,1966.

[43] Hotelling H. *Relation between two sets of variates.* Biometrika, Vol.28, pp.321-372, 1936.

[44] Jordan C. *Mémoire sur les formes bilinéaires.* J. Math. Pures Appl. II, Vol.19, pp.35-54, 1874.

[45] Jordan C. *Essai sur la géometrie à n dimensions.* Bull. Soc. Math. France, 3, 1875, pp.103-174.

[46] Kailath T. *Linear Systems.* Prentice Hall, Englewood Cliffs, New Jersey, 1980.

[47] Kalman R. *A new approach to linear filtering and prediction problems.* J. Basic Eng. 82, pp.35-50, March 1960.

[48] Kung S.Y. *A new identification method and model reduction algorithm via singular value decomposition.* 12th Asilomar Conf. on Circuits, Systems and Comp., pp.705-714, Asilomar, CA, Nov.1978.

[49] Larimore W. *System identification, reduced order filtering and modeling via canonical variate analysis.* Proc. of the American Control Conference, ACC June 1983, San Francisco.

[50] Larimore W. *Canonical variate analysis in identification, filtering and adaptive control.* Proc. 29th Conference on Decision and Control, Hawai, December 1990, pp.596-604.

[51] Liu R., Suen L.C. *Minimal dimension realization and identifiability of input-output sequences.* IEEE Transactions on Automatic Control, April 1977, Vol.AC-22, pp.227-232.

[52] Ljung L. *System identification - Theory for the User.* Prentice Hall, Englewood Cliffs, NJ, 1987.

[53] Ljung L. *Issues in System Identification.* IEEE Control Systems, Vol.11, No.1, January 1991, pp.25-29.

[54] Luenberger D.G. *Canonical froms for linear multivariable systems.* IEEE Transactions on Automatic Control, Vol.AC-12:290, 1967.

[55] Moonen M., De Moor B., Vandenberghe L., Vandewalle J. *On and off-line identification of linear state space models.* International Journal of Control, Vol.49, No.1, pp.219-232, 1989.

[56] Moonen M., Vandewalle J. *A QSVD approach to on- and off-line state space identification.* International Journal of Control, Vol.51, No.5, pp.1133-1146, 1990.

[57] Moonen M. *Jacobi-type updating algorithms for signal processing, systems identification and control.* PhD Thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium, November 1990.

[58] Moonen M, Ramos J. *A subspace algorithm for balanced state space system identification.* IEEE Transactions on Automatic Control, Vol. 38, pp.1727-1729, August 1993

[59] Moonen M., De Moor B., Vandewalle J. *SVD-based subspace methods for multivariable continuous time system identification.* "Identification of continuous-time systems". G.P. Rao, N.K. Sinha Eds, Kluwer Academic Publications, 1991, pp.473-488.

[60] Moonen M., De Moor B. *Comments on 'State-space model identification with data correlation'.* International Journal of Control, Vol.55, no.1, pp.257-259, 1992.

[61] Moonen M., De Moor B., Ramos J., Tan S. *A subspace identification algorithm for descriptor systems.* Systems & Control Letters, Volume 19, 1992, pp.47-52.

[62] Moore B.C. *Principal component analysis in linear systems: Controllability, Observability and Model Reduction.* IEEE Transactions on Automatic Control, Vol.AC-26, no.1, February 1981.

[63] Mullis C.T., Roberts R.A. *Synthesis of minimum round-off noise fixed point digital filters.* IEEE Transactions on Circuits and Systems, Vol.CAS-23, pp.555-562, 1976.

[64] Norton J.P. *An introduction to identification.* Academic Press, London, 1986.

[65] Ottersten B. *Parametric subspace fitting methods for array signal processing.* PhD Thesis, Information Systems Laboratory, Department of Electrical Engineering, Stanford University, CA, USA, December 1989.

[66] Ottersten B., M.Viberg, P.Stoica, A.Nehorai, *"Exact and large sample maximum likelihood techniques for parameter estimation and detection in array processing"*, in Radar array processing, S.Haykin, J.Litva, T.J.Sherpherd (eds.), Springer Verlag, 1993, pp. 99-151.

[67] Ottersten B., M.Viberg, T.Kailath, *"Analysis of subspace fitting and ML techniques for parameter estimation from sensor array data"*, IEEE Tr. on SP, Vol. 40, No.3, March 1992, pp. 590-600.

[68] Schmidt E. *Zur Theorie der linearen und nichtlinearen Integralgleichungen. I Tiel. Entwicklung willkürlichen Funktionen nach System vorgeschriebener.* Mathematische Annalen, 63, pp.433-476, 1907.

[69] Söderström T., Stoica P. *System Identification.* Prentice Hall International Series in Systems and Control Engineering, Prentice Hall, New York, 1989.

[70] Silverman L. *Realization of linear dynamical systems.* IEEE Transactions on Automatic Control, Vol.AC-16, pp.554-567, December 1971.

[71] Swindlehurst A., Roy R., Ottersten B., Kailath T. *System identification via weighted subspace fitting.* Proc. of the American Control Conference, pp.2158-2163, 1992.

[72] Sylvester J.J. *Sur la réduction biorthogonale d'une forme linéo-lin'eaire à sa forme canonique.* Comptes Rendus, CVIII., pp.651-653, 1889

[73] van Overbeek A.J.M., Ljung L. *On-line structure selection for multivariable state space models.* Automatica, 18(5), pp. 529-543,1982.

[74] Van Overschee P., De Moor B., Suykens J. *Subspace algorithms for system identification and stochastic realization.* Proc. Conf. on Mathematical Theory for Networks and Systems, MTNS, Kobe, Japan, pp.589-595, June 1991, Mita Press.

[75] Van Overschee P., De Moor B. *Subspace algorithms for the stochastic identification problem.* 30th IEEE Conference on Decision and Control, Brighton, UK, pp.1321-1326, 1991.

[76] Van Overschee P., De Moor B. *Subspace algorithms for the stochastic identification problem.* Automatica, Vol.29, no.3, 1993, pp.649-660.

[77] Van Overschee P., De Moor B. *Two subspace algorithms for the identification of combined deterministic stochastic systems.* Proc. of the 31st IEEE Conference on Decision and Control, December 16-18, 1992, Tucson, USA, Vol. 1, pp. 511-516

[78] Van Overschee P., De Moor B. *N4SID: Numerical Algorithms for State Space Subspace System Identification* Proc. of the World Congress of the International Federation of Automatic Control, IFAC, Vol.7, pp.361-364, Sydney, Australia, July 1993.

[79] Van Overschee P., De Moor B. *Subspace identification of a glass tube manufacturing process.* Proc. of the second European Control Conference, June 28-July 1, 1993, Groningen, The Netherlands, pp. 2338-2343

[80] Van Overschee P., De Moor B. *N4SID: Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems.* Automatica, Special Issue on Statistical Signal Processing and Control, Vol. 30, No. 1, 1994, pp. 75-93

[81] Van Overschee P., De Moor B., Boyd S., Aling H., Kosut R. *A fully interactive system identification module for Xmath (ISID)*, Acc. for publication in the Proc. of SYSID 94, 10th IFAC Symposium on System Identification, 4-6 July 1994, Copenhagen, Denmark

[82] Van Overschee P., De Moor B. *A Unifying Theorem for three Subspace System Identification Algorithms.* Accepted for publication in SYSID '94, 10th IFAC Symposium on System Identification, 4-6 July 1994, Copenhagen, Denmark.

[83] Van Overschee P., De Moor B. *A Unifying Theorem for three Subspace System Identification Algorithms.* Accepted for the American Control Conference, Baltimore, USA, June 1994.

[84] Zhu Y., Van Overschee P., De Moor B., Ljung L. *Comparison of three classes of identification methods.* Acc. for publication in SYSID '94, 10th IFAC Symposium of System Identification, 4-6 July 1994, Copenhagen, Denmark

[85] Van Overschee P., De Moor B., Aling H., Kosut R., Boyd S. *Xmath Interactive System Identification Module II,* Manual **ISID II**, 100 pp., Integrated Systems Inc., Santa Clara, CA, USA, 1993.

[86] Vanvuchelen P., De Moor B. *A Multi-Objective Optimization Approach for Parameter Setting in System and Control Design.* Proc. of the 16th IFIP Conference on system modelling and optimization, Compiegne, July 5-9, 1993, France pp. 63-66.

[87] Verhaegen M., Dewilde P. *Subspace model identification, Part I: The output-error state space model identification class of algorithms, Part II: Analysis of the elementary output-error state space identification algorithms.* Internal Report, Dept. of EE, Delft University of Technology, December 1990.

[88] Verhaegen M. *A novel non-iterative MIMO state space model identification technique.* Proc. 9th IFAC/IFORS Symp. on Identification and System Parameter Estimation, Budapest, Hungary, pp.1453-1458, 1991.

[89] Viberg M. *Subspace fitting concepts in sensor array processing.* PhD Thesis, Department of Electrical Engineering, Linköping University, Sweden, 1989.

[90] Viberg M., Ottersten B. *Sensor array processing based on subspace fitting.* IEEE Transactions on Acoustics, Speech and Signal Processing, ASSP-39, May 1991.

[91] Young Man Cho, Guanghan Xu, Thomas Kailath. *Fast Recursive Identification of State Space Models via Exploitation of Displacement Structure.* Information Systems Lab, Stanford University, 27 pp., July 1992 (submitted for publication).

[92] Young Man Cho. *Fast subspace based system identification: Theory and practice.* Information Systems Lab, Stanford University, CA, USA, PhD Thesis, August 1993.

[93] Willems J. *From time series to linear systems.* Automatica, Part I: Vol.22, no.5, pp.561-580, 1986, Part II: Vol.22, no.6, pp.675-694, 1986, Part III: vol.23, no.1, pp.87-115, 1987.

[94] Zeiger H., McEwen A. *Approximate linear realizations of given dimension via Ho's algorithm.* IEEE Transactions on Automatic Control, 19, pp.153, 1974.