Multivariable Control Systems: An Engineering Approach

P. Albertos
A. Sala

Springer

Advanced Textbooks in Control and Signal Processing

Springer
London
Berlin
Heidelberg
New York
Hong Kong
Milan
Paris
Tokyo

Series Editors

Professor Michael J. Grimble, Professor of Industrial Systems and Director
Professor Michael A. Johnson, Professor of Control Systems and Deputy Director
Industrial Control Centre, Department of Electronic and Electrical Engineering,
University of Strathclyde, Graham Hills Building, 50 George Street, Glasgow G1 1QE, U.K.

Other titles published in this series:

Genetic Algorithms: Concepts and Designs
K.F. Man, K.S. Tang and S. Kwong

Neural Networks for Modelling and Control of Dynamic Systems
M. Nørgaard, O. Ravn, N.K. Poulsen and L.K. Hansen

Modelling and Control of Robot Manipulators (2nd Edition)
L. Sciavicco and B. Siciliano

Fault Detection and Diagnosis in Industrial Systems
L.H. Chiang, E.L. Russell and R.D. Braatz

Soft Computing
L. Fortuna, G. Rizzotto, M. Lavorgna, G. Nunnari, M.G. Xibilia and
R. Caponetto

Statistical Signal Processing
T. Chonavel
Translated by Janet Ormrod

Discrete-time Stochastic Systems (2nd Edition)
T. Söderström

Parallel Computing for Real-time Signal Processing and Control
M.O. Tokhi, M.A. Hossain and M.H. Shaheed

Analysis and Control of Non-linear Process Systems
K. Hangos, J. Bokor and G. Szederkényi
Publication due January 2004

Model Predictive Control (2nd edition)
E. F. Camacho and C. Bordons
Publication due March 2004

P. Albertos and A. Sala

Multivariable Control Systems

An Engineering Approach

With 68 Figures

Springer

Prof. P. Albertos
Dr. A. Sala
Department of Systems Engineering and Control, Polytechnic University of Valencia,
C. Vera s/n, Valencia, Spain

British Library Cataloguing in Publication Data
Albertos Prerez, P.
Multivariable control systems : an engineering approach. –
(Advanced textbooks in control and signal processing)
1.Automatic control
I.Title II.Sala, Antonio, Doctor
629.8
ISBN 1852337389

Library of Congress Cataloging-in-Publication Data
A catalog record for this book is available from the Library of Congress.

ISSN 1439-2232
ISBN 1-85233-738-9 Springer-Verlag London Berlin Heidelberg
a member of BertelsmannSpringer Science+Business Media GmbH
http://www.springer.co.uk

Typesetting: Electronic text files prepared by authors
Printed and bound in the United States of America
69/3830-543210 Printed on acid-free paper SPIN 10922118

To Ester (P.A.), Amparo and Ofelia (A.S.)

This page intentionally left blank

Series Editors' Foreword

The topics of control engineering and signal processing continue to flourish and develop. In common with general scientific investigation, new ideas, concepts and interpretations emerge quite spontaneously and these are then discussed, used, discarded or subsumed into the prevailing subject paradigm. Sometimes these innovative concepts coalesce into a new sub-discipline within the broad subject tapestry of control and signal processing. This preliminary battle between old and new usually takes place at conferences, through the Internet and in the journals of the discipline. After a little more maturity has been acquired by the new concepts then archival publication as a scientific or engineering monograph may occur.

A new concept in control and signal processing is known to have arrived when sufficient material has evolved for the topic to be taught as a specialised tutorial workshop or as a course to undergraduate, graduate or industrial engineers. *Advanced Textbooks in Control and Signal Processing* are designed as a vehicle for the systematic presentation of course material for both popular and innovative topics in the discipline. It is hoped that prospective authors will welcome the opportunity to publish a structured and systematic presentation of some of the newer emerging control and signal processing technologies in the textbook series.

There are a considerable number of multivariable industrial processes which are controlled by systems designed using single-input, single-output control design methodologies. One reason for this is that multivariable systems textbooks often incorporate a significant amount of mathematics which tends to obscure the potential benefits that can be obtained from exploiting the multivariable structure and properties of multi-input, multi-output systems. In this new textbook, Pedro Albertos and Antonio Sala have made considerable efforts to discuss and illustrate the inherent meaning and interpretation of the principles within multivariable control system design. This is reflected in a book structure where after several chapters on models and linear system analysis Chapter 4 pauses to review the control roadmap ahead in the second part of the book. This roadmap has chapters devoted to centralised multivariable control methods, optimisation-based methods, robustness and implementation issues. In the presentation there is a clear indication

that the authors are very aware of current industrial control system design practice. This is seen through the choice of industrial and practical examples chosen to illustrate the control system principles presented. These include a paper machine head box control system, a (3×3) distillation column problem, a steam-boiler system and a really interesting ceramic kiln control system problem. The kiln problem is used to show that the industrial multivariable control system design problem has a wealth of associated problems which also have to be considered and solved. Indeed the ceramic kiln problem is similar to other processes like that of plate-glass manufacture, and the reheating of steel slabs in a walking beam furnace in the steel industry.

The discussion of the various issues in multivariable control system design is a particularly attractive feature of the book since this helps to put into context and perspective some difficult theoretical issues. The chapter on robustness (Chapter 8) is a good example of a discussion chapter from which the reader can decide whether to delve further into the supporting technical appendix and references. The book is suitable for final-year undergraduates, and graduate students who will find the valuable insights, and illustrative examples particularly useful to their studies of multivariable control system design and implementation. Lecturers and professionals in the control field will find the industrial context of the examples and discussions a refreshing change from the usual more straightforward academic multivariable systems control textbooks.

M.J. Grimble and M.A. Johnson
Industrial Control Centre
Glasgow, Scotland, U.K.
Summer 2003

Preface

"'Engineering approach' implies lots of shortcuts and simplifications. Simplification often means telling the truth but not the whole truth. If it were the whole truth, it would not be simple!" (Bob Atkins).

Control engineering is a multidisciplinary subject, useful in a variety of fields. Process control, servosystems, telecommunications, robotics, or social system dynamics, among others, require the concourse of automatic control concepts to better understand the behaviour of the respective processes and to be able to introduce changes in their dynamics or counteract the effect of disturbances.

Recent industrial trends in the implementation of control systems claim a wider perspective in the design, not just a collection of single-loop controllers, coping with a complex system with multiple interrelated variables to be controlled and having the option to manipulate multiple variables. The first step in this direction is to consider the control of multivariable systems.

The aim

Talking about control problems and moving to wonderful mathematical abstractions is very tempting. The complexity and elegance of many control problems have attracted the interest of theorists and mathematicians, developing more or less complex control theories that are not always well connected to the practical problems. However, in this book, the theory is used as a support to better understand the reasons and options of some control design techniques rather than to enter into the details of a given issue, even if this issue can be the matter of dozens of research papers.

The book presents the fundamental principles and challenges encountered in the control of multivariable systems, providing practical solutions but keeping an eye on the complexity of the problem to decide on the validity of the results. We are not interested in control design recipes, although guidelines

are welcome, and always try to analyse the proposed solution and suggest alternatives. The study of some of the control options for multivariable systems, their physical understanding and reasoning, and their tuning in real applications is the main aim of the book, devoting also some effort to the available tools used in their computer-aided design.

The content

The book is structured to cover the main steps in the design of multivariable control systems, providing a complete view of the multivariable control design methodology, with case studies, without detailing all aspects of the theory. An introductory chapter presents in some extent the general issues in designing control systems, guiding the reader through the subjects to be treated later on. As most control systems are conceived to be digitally implemented in a computer-based system, the use of process models is generalised and the control design approach is based on a model of the process. This is the subject of Chapter 2, where the representation of linear systems, in continuous and discrete time, is dealt with in some detail. Although there is an introduction to the modelling of non-linear processes, approximation techniques move the problem to the linear "arena", where the theory is simpler and well known and the concepts can be acquired more easily.

Chapter 3 deals with the tools for extracting properties from the models, including models of the process, the controller and the whole controlled system. This is a key chapter that provides the basis of analysing the behaviour of a system and its possibilities of being controlled. Emphasis is placed on the structural properties of multivariable systems and issues such as directionality and interaction, not relevant in single-input-single-output (SISO) systems, are discussed. Using the analysis results, model reduction techniques are introduced.

The general options in designing a control system are the subject of Chapter 4, where an analysis of the advantages and drawbacks of different control structures is presented. This is an introduction to the rest of chapters, where different design and implementation control techniques are developed. Decentralised and decoupled control is the subject of Chapter 5. Here, most of the ideas of SISO control system design are useful, being complemented with the analysis of the loops' interaction and the crucial issue of variables selection and pairing. Some guidelines for the setting of a multi-loop control system are discussed.

Full advantage of the internal representation is taken in Chapter 6 to introduce the centralised control structure. All the ideas presented in the previous chapters are used here to present a methodology for the integrated design of a control system with multiple controlled variables. The classical state feedback strategy, complemented with the use of state observers, provides a solution that is easily implemented in a digital computer and reduces the cost of instru-

mentation. Virtual sensors are a step further in the construction of observers and present an attractive solution from an engineering viewpoint.

One of the characteristics a control user demands is the interpretability of the tuning parameters of a control system. Cost or performance indices are a suitable way of expressing some requirements, even using limited options such as norms of variables, and optimal control provides an effective approach to controller design. In Chapter 7, the assumption of linear models simplifies the treatment and allows us to find a closed solution that can be implemented as a linear controller.

The treatment of uncertainty in the models (and even in the requirements) is fundamental from an engineering viewpoint, as the models are always partial representations of the process behaviour. An introduction to robust control design techniques, as a variation of optimal control, is presented in Chapter 8, complemented with additional readings and material presented in an appendix.

Implementation of the designed control is a key factor in the success of a control system. Many issues are application-dependent, but a number of general guidelines and warnings are the subject of Chapter 9. The integrated treatment of the control design and its implementation, in resource-constrained environments, is a matter of research interest and should be always kept in mind by a control design engineer.

The use of the internal representation provides a good framework for reviewing the control concepts in a general way, with validity not only for multivariable systems but also for SISO. The introduction of tools for analysing and designing robust control systems is also an added value of the book and a motivation to enter into this control design methodology.

As previously mentioned, all the relevant concepts are illustrated with examples, and programming code and simulation diagrams are provided to make the validation of the results easy as well as the grasping of the concepts. A number of case studies present a joint treatment of a number of issues. Unfortunately, there is no room to describe in full detail the practical design and implementation of a complete application. This is something that the reader could try to carry on in his/her own control problem.

A number of appendices include some reminders and new ideas to help in the reading of the main body of the book, giving a self-contained feature, always desirable in a book with a large audience. Of course, the analysis and representation tools are also developed, but they are always considered as a way of achieving the final control system design and evaluation. Examples with high-level simulation packages, mainly MATLAB®, are provided. The case studies and the chapter examples lead the reader into a practical perspective of the control solution. Usually, a full design completing additional issues of the case studies could be attempted as an exercise. Additionally, a large list of references will provide alternative reading to those interested in more rigorous treatment of the topic or more detailed specific applications.

The audience

This is not an ambitious book either from the theoretical perspective or from the end-user viewpoint, but it is trying to bridge the gap between these two extremes. The main goal is to present in an easy-to-read way the challenging control problems involving subsystems, interactions and multiple control objectives, introducing some tools for designing advanced control systems.

There is a wide audience of engineers and engineering students with a background of basic control ideas grasped in their previous studies or in their practical experience in designing systems. This is a heterogeneous audience coming from different fields, such as instrumentation in the process industry, the design of electronic devices, the study of vibrations and dynamics in mechanical systems or the monitoring of process units.

The book can be helpful in introducing the basic concepts in multivariable linear control systems to practical engineers. The control problems may be familiar to them and the presented tools will open their mind to find appropriated solutions.

For senior undergraduate students that previously had grounding in SISO control (PID, root locus, discretisation of regulators), the challenge is to convey the basic heuristic ideas regarding multivariable control design in a one-semester course, presenting the necessary mathematical tools as they are needed and putting emphasis on their use and intuition, leaving the details of the theory for more advanced texts.

Acknowledgements

We would like to thank our colleagues in the Department of Systems Engineering and Control at the Polytechnic University of Valencia for their helpful comments and suggestions, as well as to the many students who have shared with us the experience of discovering the interplay of teaching-learning the subject during the last years. Their further comments and remarks will be constantly appreciated. Feedback also from our readers, will help in improving the material in this book, both in further editions and in the instantaneous contact that the Internet provides nowadays.

Valencia,
August 2003

Pedro Albertos
Antonio Sala

Contents

1

Introduction to Multivariable Control

This introductory chapter is devoted to reviewing the fundamental ideas of control from a multivariable point of view. In some cases, the mathematics and operations on systems (modelling, pole placement, *etc.*), as previously treated in introductory courses and textbooks, convey to the readers an unrealistic image of systems engineering. The simplifying assumptions, simple examples and "perfect" model set-up usually used in these scenarios present the control problem as a pure mathematical problem, sometimes losing the physical meaning of the involved concepts and operations. We try to emphasise the engineering implication of some of these concepts and, before entering into a detailed treatment of the different topics, a general qualitative overview is provided in this chapter.

1.1 Introduction

The aim of a control system is to force a given set of process variables to behave in some desired and prescribed way by either fulfilling some requirements of the time or frequency domain or achieving the best performances as expressed by an optimisation index.

The process engineers design the process according to the best of their knowledge in the field and by assuming some operating conditions. Later on, the process will run under some other conditions that support external disturbances usually not well known or determined. Also, the characteristics of the process will change with time and/or the load. It is the role of the control system to cope with these changes, also providing a suitable behaviour.

The scope of the control tasks varies widely. The main goal may be to keep the process running around the nominal conditions. In other cases, the control purpose will be to transfer the plant from one operating point to another or to track a given reference signal. In some other cases, the interest lies in obtaining the best features of the plant achieving, for instance, the maximum

production, minimum energy consumption or pollution, or minimum time in performing a given task.

In a general way, the following control goals can be targeted [17]:

- regulation (disturbance rejection),
- reference tracking,
- generation of sequential procedures (for start-up or shut-down),
- adaptation (changing some tunable parameters),
- fault detection (to avoid process damage or provide reconfiguration),
- supervision (changing the operating conditions, structure or components),
- coordination (providing the set-points),
- learning (extracting some knowledge from the experience).

All these different activities result in very distinct control approaches and techniques. From logical and discrete-time controllers to sophisticated intelligent control systems where the artificial intelligence techniques provide the framework for emulating human behaviour, the multiple available tools for control systems design are complementary and, in the integral control of a plant, some of them are used in a cooperative way.

Our interest in this book will be mainly focused on the regulation and set-point tracking of a plant where a number of manipulated variables allow joint control of several process variables. In some sense, the coordination is considered, but with the specific viewpoint of joint control of a multivariable system. These problems are basic ones and appear recurrently in some others, such as batch processes, sequential control or fault tolerant control. But that means that no specific attention will be paid to the starting-up and shutting-down phases, although in some cases the tracking concepts could be applicable. Neither alarm treatment nor learning and adaptation will be considered, although, as previously mentioned, all these activities are usually required to automatically run a complex system.

It is interesting to note, at the very beginning, that the controller could be considered as a subsystem feeding the controlled system with the appropriate signals to achieve some goals. That is, the controller is selecting, among the options in some *manipulated variables*, the signals which are appropriate. With this perspective, the controller is guiding the process in the desired way, but the process itself should be capable of performing as required.

Traditionally, the role of the control system was to cope with the deficiencies of the controlled process and the undesirable effect of the external disturbances. Nowadays, there is a tendency to integrate the design of both the process and the controller to get the best performance. This may result in a simpler and more performing global system. For instance, taking into account the existence of the control system, a reactor or an aircraft can be designed in such a way that they are open-loop unstable and cannot run without control. But, on the other hand, the controlled system could be cheaper, faster, more reliable or more productive.

In order to illustrate some basic concepts and ideas to better outline the purpose of the book, let us introduce a process control example. Afterwards, we will outline basic ideas, further developed in the rest of the chapters.

Example 1.1. Let us consider a typical process unit for refining a chemical product. First, there is a mixing of two raw materials (reactives) to feed a distillation column where two final products are obtained, the head and bottom components. In order to run the unit, we must control the different flows of material, provide adequate temperature to the inlet flows and keep the desired operating conditions in the column by adjusting its temperature, pressure and composition. Some other complementary activities are required, such as agitating the content of the mix tank or keeping the appropriate levels in all vessels, including those of auxiliary or intermediate buffers. A simple layout of the unit is depicted in Figure 1.1.

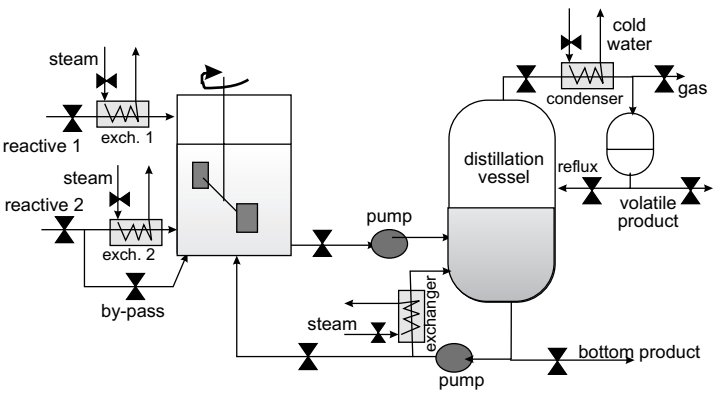

Figure 1.1. Distillation unit

The ultimate control goal is to obtain the best distilled products (maximum purity, less variance in concentration, ...) under the best conditions (maximum yield, minimum energy consumption, ...), also taking into consideration cost and pollution constraints. But before we begin to get the products, we must startup all the equipment devices, establish a regular flow of reactives, reach the nominal operating conditions and then keep the unit stable under production. Also, care should be taken about faults in any part of the unit: valves, agitator, existence of raw materials, heating systems, *etc.*

1.2 Process and Instrumentation

The process to be controlled is an entity of which the complexity can vary from something as simple as a DC motor or a water tank to a very complex system such as a mobile platform or an oil refinery.

Independently of its design, carried out taking into account control requirements or not, control design assumes that the equipment modules are

given and are already interconnected according to the guidelines of the process experts. Sometimes, analysis of expected performance with a particular control system may advise changes in the process or instrumentation (sensor and actuators); for instance, from the control viewpoint, it could be better to feed the mixed material or the reflux of the head product at a different column plate, but at this moment it is fixed.

In order to control the process, some *manipulated* variables should be available, allowing introduction of *control actions* in the process to force it towards evolving in the desired way. In the kind of processes we are going to deal with, more than one manipulated variable is always available, providing more richness and options in controlling the process. In an automatically controlled plant, these manipulated variables will act on the process through the corresponding actuators. To get information about the process, some internal variables should be measured, being considered as *output* variables. Again, more than one output variable will be considered. The control target could be these variables themselves or some other directly related to them: to keep them constant in a regulatory system, to track some references in servo systems, or to perform in some prescribed way with temporal, harmonic or stochastic properties.

In the distillation unit, Figure 1.1, there are many input variables. We can count as many as 14 valves, two pumps and an agitator. All of them can be used to drive the unit, being considered as manipulated variables, but most of them will be locally controlled or manually fixed and will not intervene in the control strategy. Many temperatures, flows, levels or concentrations at different points inside the unit can provide information about the behaviour of the plant, but not all of them will be measured. Even less will be controlled. The set of measurement devices, as well as the instrumentation required to condition the measurements, constitute the data acquisition system, which itself can be quite complex involving transducers, communication lines and converters. These devices will be also fundamental in achieving proper control [66].

The input variables or signals acting on the process but not being manipulated to achieve the control goals should be considered as *disturbances*. They are usually determined as a result of other processes or, in the simplest case, they are assumed to be constant. These disturbances can be *predictable* (deterministic) or not. For instance, in a rolling mill process, the arrival of a new block affecting the rolls' speed is an event that can be predicted but not avoided. Also, the disturbances can be *measurable* or not. Even some characteristics of the disturbance may be known in advance if it belongs to a class of signals. For example, in the distillation column we can get information about the raw materials' concentration, but it is fixed somewhere else and cannot be considered as a manipulated variable. Ambient temperature is also a partially predictable disturbance.

It is clear that unpredictable, unknown and unmeasurable disturbances are the worst ones to be counteracted by the control actions.

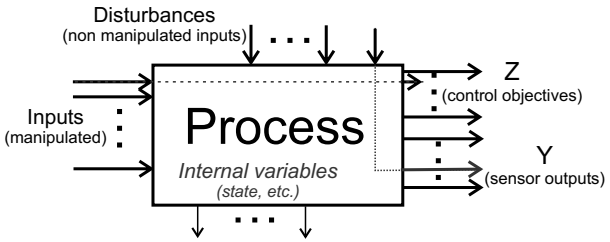

Figure 1.2. Process variables

1.3 Process Variables

To study the behaviour of a process is to analyse the involved variables and their relationship, what is represented by the *variable* and *process* models. As previously mentioned, with respect to the process shown in Figure 1.2, the variables can be:

- **external** or **inputs**, being determined by other processes or the environment, acting on the process and considered as:
  - **manipulated** or **control** variables, u, if they are used to influence the dynamics of the process. Actuators will amplify the control commands to suitable power levels to modify plant's behaviour,
  - **disturbances**, d, if they are uncontrollable outputs of other subsystems,
- **internal**, being dependent on the process inputs, system structure and parameters. We are interested in evaluating the behaviour of these process variables. They can be classified as:
  - **outputs** or measured variables, y, if they are sensed and provide information about the process evolution,
  - **controlled variables**, z, if the control goals are based on them. They can be outputs or not, depending on the sensors' availability and placement,
  - **state variables**, x, as later on properly defined, are a minimum set of internal variables allowing the computation of any other internal variable if the inputs are known.

¿From an information viewpoint, any process can be considered as an information processor, giving some outputs as a result of the processing of inputs and the effect of disturbances[1].

[1] As usual in control literature, it will be assumed that no significant amount of power will be extracted or introduced by the signal processing system (controller) into the process, *i.e.*, actuators are considered part of the system and sensors are ideal. Otherwise, energy balance equations in the process do change, and the model changes accordingly. For details on modelling and system interconnection relaxing those assumptions, see [102].

Dynamic physical systems evolve continuously over time. Thus, the involved variables are functions of time, as are real variables, and are represented by continuous-time (CT) signals, (for instance, $u(t), u \in \mathbb{R}^m, \quad t \in \mathbb{R}$ represents a set of m real CT variables). Nevertheless, for the sake of the analysis, only some characteristics of the signals could be of interest. For instance, if only the value of these variables at some given time instant is relevant, the variables' model is discrete and they are represented by discrete-time (DT) signals, time being an integer variable ($t \in \mathbb{Z}$). In digital control systems, signals are quantised ($u_i \in \mathbb{Z}$), due to the finite word-length of the internal number representation. Maybe only some levels of the variables are relevant. In the simplest case, only two options are considered and the corresponding variable is represented by a logical or binary signal ($u_i \in \{0, 1\}$).

Most of the distillation unit variables are modelled using CT signals, but, for the sake of control, could be treated digitally or logically. For instance, the agitator speed could be represented by two options: on and off. Some variables could be treated as CT signals if they lie inside a prescribed range of values, being considered as saturated or null if they are out of range. That is, the same physical variables can be represented by different signals depending on the purpose of their treatment.

As previously mentioned, in some cases, the value taken by a variable as a function of time is not relevant and we are interested in some periodic properties, such as the frequency components, in magnitude or phase. Harmonic analysis (Fourier transform) is thus appropriate. In some other cases, only the stochastic properties of the variables are of interest. Consider, for instance, the concentration of a distillation product. More than the punctual value of the concentration at a time instant, the interest of the user is in the average concentration in a reasonable interval, as well as the possible maximum deviations.

In the case of multivariable systems, it is also mandatory to scale the variables' magnitude in order to make them comparable, if different "errors" need to be used to compute the control actions. Usually, the significant variable ranges are normalised so some performance measures can be said to be fulfilled if a particular error is lower than 1. This is convenient for quick comparative analysis.

1.4 The Process Behaviour

Our main aim is the time response or frequency response of the controlled process, when subject to some given reference changes or expected disturbances (*forced response*). A related issue is the study of the behaviour of the autonomous process when it evolves from some non-equilibrium initial conditions (*free response*).

Systems could be considered as operators mapping a set of functions of time (inputs) onto another (outputs). We are interested in *dynamic* systems,

that is, those whose current internal variables depend on their past value and that of the inputs. Dynamic system analysis tools are applied for that purpose.

To simplify the study, some basic assumptions are made. It is clear that, once a time scale is selected, some dynamic subprocesses could be considered either infinitely fast (instantaneous or *static*) or infinitely slow. In this case, they are considering as generator of constant signals, without any significant evolution.

Linearity. This is the most important simplification [11]. A process (as any operator) is said to be linear if it accomplish the following linearity principles:

1. The response is proportional to the input. That is, if for a given input, $u(t)$, the response is $y(t)$, for an input $\alpha u(t)$ the response is $\alpha y(t)$, $\alpha \in \mathbb{R}$.
2. The effect of various inputs is additive. That is, if for a given input, $u_1(t)$, the response is $y_1(t)$, and for an input, $u_2(t)$, the response is $y_2(t)$, for an input $u(t) = \alpha_1 u_1(t) + \alpha_2 u_2(t)$ the response is $y(t) = \alpha_1 y_1(t) + \alpha_2 y_2(t)$.

Linearity allows us to "split" the study of the dynamic behaviour; the total evolution can be computed as the result of external inputs plus the effect of some initial conditions. It would also allow the study of the different manipulated variables, one by one, and the global behaviour would be determined by the way the different responses add up altogether. Linear operations with sets of variables are dealt with by means of vector and matrix algebra, and some results are direction-dependent, as we will see later on.

Linearisation is an approximation technique allowing the representation of the non-linear behaviour of a process by an approximate linear model. The linearisation approach is used to consider the relationship between incremental variables around an equilibrium state, but it requires continuity and differentiability in the non-linearities. Although this technique is not always applicable (consider, for instance a switching process), in many cases it provides good insight into the process behaviour and can be used in the design of a suitable controller.

Approximate linearisation should not be confused with exact linearisation. In some processes, with a suitable selection of variables (or a change of variables in a given model), or some changes in the system structure (for instance, by feed-backing some variables) the resulting model may have the linearity properties. Some ideas will be suggested in the final chapter of this book.

Time invariance. This is another practical assumption, implying that parameters and functions appearing in the process model do not change with time.

Usually, the process behaviour changes with time because some parameters, assumed to be constant, slowly vary with time. Another cause of "apparent" time-variation is non-linearity: changing the operating point changes the approximate linear behaviour. Slow, unmodelled, non-linear dynamics also may manifest that way, even without operating point changes.

Lumped parameters. In this case, time is assumed to be the unique independent physical variable and the devices' dynamics are modelled as punctual phenomena, without taking into consideration the distributed-in-space nature of most processes: communication lines, three dimensional plants, transportation systems and so on. In many cases, spatial discretisation (finite elements, finite differences) allows us to set up approximate models, allowing us to better accept this assumption.

Example 1.2. The distillation unit is a highly non-linear process, if it is consider in the full range of operating options. But if we consider its evolution around some stable production, an approximate linear model can be set up to relate the effects of input variations on the internal variables.

The distillation plant may also be considered as a typical distributed system. We can talk about the column temperature, but this temperature varies from point to point internally, as well as the temperatures of the metallic elements in the exchangers. There are transportation delays and, for instance, the instantaneous column inlet flow concentration and temperature is slightly different from those at the output of the mixer. Nevertheless, by either averaging values or making a spatial discretisation, a lumped parameter model will be useful.

1.5 Control Aims

Referring to a multivariable, multiple-input-multiple-output (MIMO) system where there are p-controlled variables, expressed as a vector $y(t)$, and m-manipulated variables, $u(t)$, and also considering a d-dimensional disturbance vector, $d(t)$, the control requirements for following a p-dimensional vector reference, $r(t)$, can be stated at different levels, as described below.

Ideal control. If the relationship between the process variables can be expressed by:

$$y = Gu + G_d d \tag{1.1}$$

where G and G_d are general operators, the ideal control action, leading to $y = r$ would be:

$$u = G^{-1}(r - G_d d) \tag{1.2}$$

This control could be perfect, but it is ideal and not achievable. There are many reasons for this, among them:

1. The operator G is usually not invertible.
2. Even if G were invertible, the resulting actions may be physically unfeasible.
3. The disturbance, d, may be unmeasurable.
4. The process and disturbance models (G and G_d) are not perfectly known.

Possible control. For the control problem to be feasible, some performance requirements from the ideal one must be relaxed so that they are compatible with the constraints in the available actuator powers and measurements. For instance, high-speed reference tracking and low control action are not simultaneously achievable. High attenuation of unmeasurable disturbances and high tolerance to modelling errors are also incompatible. So, the control design is a trade-off between conflicting requirements. Introducing additional sensors, actuators or control variables in a more complex *control structure* may improve the performance possibilities of the control system. Ideal control would be achieved under unlimited actuator power and *full information* sensing.

Optimal control. If the requirements are formulated as the minimisation of a cost index or the maximisation of a performance index, the resulting controller is called an "optimal" one. But optimality from a mathematical viewpoint does not mean the best from a user viewpoint, unless user requirements are properly translated into optimisation parameters (see below).

Practical control. The controllers above are theoretical controllers. They are based on models and ideal performances. Other than the issues above, practical controllers should consider that:

- the models represent an approximate behaviour of the actual process (sometimes *very* coarse) and this behaviour may change,
- formal control design specifications represent user requirements only approximately and partially,
- the process operation should be *robust* against moderate changes in the operating conditions, requirements and disturbances,
- the implementation of the controller is constrained to resources' availability.

Thus, practical controllers should prove to the end-users that they can consistently operate the process in an "automatic" way without the continuous surveillance of the operator. The complexity of the controller, the ease of parameter tuning, the interpretability of the different control actions, or its cost advantages are issues to be considered when selecting a control strategy.

1.6 Modes of Operation

As previously mentioned, the dynamic behaviour of non-linear processes may be quite different depending on the operating conditions regarding loads, disturbances and references. But any controlled process may operate in a variety of situations such as starting-up/shutting-down, transferring from some operating conditions to new ones, under constraints (alarms, emergency) or under the guidance of the operator. All this means different modes of operation requiring different control strategies. In some cases, the same controller but

with different parameters will be appropriate, but in some others new control structures or even no control at all may be needed.

In fact, any automatic control system should have the option of operating in at least two modes of operation:

- **manual control**, if the manipulated variables are determined by the operator,
- **automatic control**, where the manipulated variables are governed by the controller. This can be done in two basic settings:
  - **open-loop control**. There is no feedback from the process and the manipulated variables are determined by the control system based on the information provided by the operator or input measurements,
  - **closed-loop control**. The controller determines the manipulated variable based on the references and goals introduced by the operator and the measurements from the process.

  Of course, mixed strategies do exist.

One interesting practical problem is the transfer between modes of operation. In particular, "closing the loop" (as well as the transfer between operating conditions) is a difficult action requiring some expertise to avoid "bumping" and even instability in the process signals.

Example 1.3. The distillation unit will be usually warmed up in manual mode. The different flows and actions will be updated manually to reach the operating conditions. Information from the unit will guide the operator (acting as a controller) to drive the unit close to the desired conditions, in an open-loop operation. Finally, in a gradual way, the control loops will be closed, starting with those more independent or less influential on the overall behaviour of the unit.

1.7 The Need for Feedback

The ideas above can be expressed in a general and obvious way: at any moment, the appropriate control action depends on the situation of the process. To know the process situation implies getting some information from it: to *close the loop* [76].

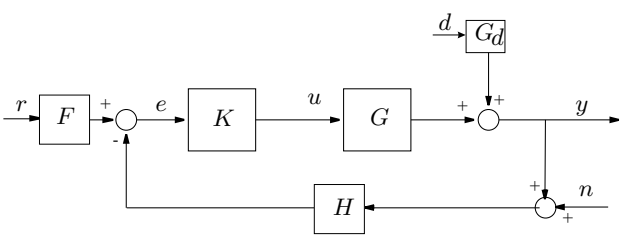

Figure 1.3. Basic control loop

Let us consider a linear SISO system, with a closed-loop control as depicted in Figure 1.3, where n is a measurement noise. Assuming a sensor system operator H, and a reference filtering F, the output can be expressed by[2]:

$$y = \frac{GK}{1+GKH}Fr + \frac{G_d}{1+GKH}d - \frac{GKH}{1+GKH}n \tag{1.3}$$

Reference tracking, $y = Tr$, can be achieved in open loop ($H = 0$) with $K = G^{-1}$, if r is such that physical realisability constraints are avoided (F may be a low-pass filter to achieve that).

However, *feedback* ($H \neq 0$) is necessary for disturbance rejection. Otherwise, the effect on output will always be G_d. Feedback is also needed to counteract *modelling errors*. In open loop, $T_{ol} = GK$ is achieved, and in closed loop, it is $T_{cl} = FGK/(1 + GKH)$. In many cases, a controller, K, can be designed so that T is any user-specified function. Let us depict the variations on the achieved behaviour with small variations of G:

$$\delta T_{ol} = K\delta G \;\Rightarrow\; \frac{\delta T_{ol}}{T_{ol}} = \frac{\delta G}{G} \tag{1.4}$$

$$\delta T_{cl} = \frac{FK}{(1+GKH)^2}\delta G \;\Rightarrow\; \frac{\delta T_{cl}}{T_{cl}} = \frac{1}{1+GKH}\frac{\delta G}{G} \tag{1.5}$$

so closed loop is advantageous if $(1 + GKH) > 1$, as it diminishes sensitivity to modelling error with respect to open-loop control.

Thus, feedback is necessary and appears as a solution for basic control problems such as disturbance rejection or reference tracking, also under modelling errors. In a first approach, if we consider a high-gain controller ($K >> 1$), the above equation can be simplified to:

$$y \approx \frac{F}{H}r + 0d - n \tag{1.6}$$

Thus, some interesting conclusions about feedback are:

- it works if the closed-loop system is stable (stable zeros of $1 + GKH$),
- the process and disturbance models (G and G_d) are irrelevant (insensitivity to modelling error),
- it injects sensor noise and its imprecision as additional deviation sources,
- for y to track the reference ($F = H$), a precise knowledge of the sensor system (H) is required,
- it (may) cancel the disturbances.

However, high-gain implies a greater chance of instability, and measurement noise must be filtered. A feedback control system is designed to achieve a compromise between disturbance rejection, noise filtering and tracking the

[2] Sometimes positive feedback is assumed, leading to denominator expressions in the form $1 - GK$ (see, for instance, Equation (8.16) on page 242).

reference, copying with some uncertainties in the models and guaranteeing some degree of stability.

Similar concepts are applicable to MIMO processes, although the complexity of the operators requires a more careful treatment.

Feedback control is based on the existence of an error or a discrepancy between the desired controlled variable and the corresponding measured output. By its conception, if there is no error no control action is produced. Thus, some kind of error should be always present.

In order to act before an error is detected in the system, if there are measurable disturbances or planned changes in the references, a *feedforward* (anticipatory) control may be useful. A control action is generated to drive the process in the required way, trying to reject the measurable disturbances and "filtering" the reference changes. Ideally, if the control given by (1.2) is applied, no error will appear. But this is not achievable in general. Thus, even in the case of feedforward control, a final feedback or reaction based on the knowledge of what is happening in the process is needed.

A more complete solution is to combine both structures to get a so-called two degree of freedom (2-DoF) control configuration. It is worth pointing out that the loop controller should take care of both the model uncertainties and the unmeasurable disturbances. The prefilter can implement a sort of open-loop control. The idea of reference prefiltering can be also extended to measurable disturbances, leading to additional feedforward control schemes.

1.8 Model-free *vs.* Model-based Control

Abstraction is a key feature of control engineering. To realise that the dynamic behaviour of an aircraft can be represented with the same tools, and even equations, as a distillation column provides a platform to conceive control systems in a generic way. But this common framework for representing the dynamics of different processes does not mean that the particular characteristics of any process are included in the generic model. First, remember that a model is always a partial representation of a process. Second, it is worth remembering that the requirements, constraints, operating conditions and many other circumstances may be very different from one system to another.

There are two basic approaches to getting the model of a process. On one hand, if the process is physically available and some experiments can be carried out, its dynamic behaviour can be captured and (partially) represented by a model, using "identification" and "parameter estimation" techniques [84]. On the other hand, if the operation of the process is fully understood and the governing principles are known, a first-principle model can be drawn, although some experiments should be carried out to determine some parameters.

A model-based control may be developed and, in this case, the main steps in the design will rest upon the mathematical treatment of the global *closed-loop model* [41].

Model-free control. In some cases, the control design is based on the emulation or repetition of some actions that have been proved to be appropriate. No model of the process is needed but it is required to deal with processes with similar behaviour. Also, by some experimental approach (or even by trial and error), some parameters of a given controller may be tuned to get the appropriate behaviour.

If many sensors and actuators are cleverly located (in a suitable control structure), successful on-line tuning of simple regulators may be carried out. However, a basic model is needed to understand the suitability of the different instrumentation alternatives.

1.9 The Importance of Considering Modelling Errors

It has been pointed out that a process model (system) is a partial representation of the process behaviour. The partial concept may refer to:

- some phenomena and/or variables have not been considered or, even if considered, they have been deemed as irrelevant. This results in a simpler model, a reduced model involving less variables or less complex equations (lower order in the differential equations),
- the model has been obtained under some operating conditions not representing the process behaviour in other situations where, for instance, the timescale, the frequency range or the magnitude of some variables is different or the presence of disturbances was not taken into account.

In this context, it is usual to consider local models (around given set-points or operating conditions), different timescale models (to analyse the transient or the steady-state behaviour) or low/high-frequency models.

Thus, a model should have some properties attached, such as its validity range, the parameters' accuracy or the uncertainty introduced by the missing dynamics, in order to be properly used.

It is worth mentioning that the quality of a model is strongly connected to the purpose of its use [52]. It is not the same to have a good model to fully design a distillation column, to understand its behaviour at a basic level or to design a control system for it. For these three different uses, three different models may be appropriate.

¿From a control viewpoint, some apparently very different process models may lead to the same controller and, *vice versa*, two slightly different models may behave in a totally different way if the same controller is applied to them. So, the importance of modelling error depends on the conditions of use of the model.

Example 1.4 ([16]). Let us consider two SISO systems represented by their transfer function (TF) :

$$G_1(s) = \frac{1}{s+1}; \qquad G_2(s) = \frac{1}{(0.95s+1)(0.025s+1)^2} \tag{1.7}$$

The reader can easily check that the open-loop step responses are almost identical. However, with a proportional regulator, $k = 85$, the first system exhibits a wonderful, desirable, closed-loop step transient but the second one is unstable.

If the models to be compared were:

$$G_1(s) = \frac{1}{s+1}; \qquad G_3(s) = \frac{10(s+1)}{s(10s+1)} \tag{1.8}$$

the second one is unstable in open-loop response. However, the behaviour with the above mentioned proportional regulator yields a very similar response in terms of settling time and final value.

Thus, in modelling a process, the control purpose should be kept in mind. *Identification for control* is the identification approach trying to get the best process model to design the control and, moreover, to combine the efforts of modelling and control design in the common endeavour of getting the "best" controlled system behaviour.

In this framework, the concept of *robustness* appears as a fundamental property of a practical control system. It does not matter how good a control system is if slight changes in the process/controller parameters or in the operating conditions result in a degraded control or even in unstable behaviour.

And it does not matter if the controlled plant behaves properly under many operating conditions if it fails to be under control or violates some constraints in some specific (possible) situations.

1.10 Multivariable Systems

If p-independent variables are selected to be controlled, then at least the same number of independent variables should be manipulated ($m \geq p$). Being independent means that they do not produce similar effects on the controlled variables, although the concept will be later formalised. If there are more manipulated than controlled variables, then there are more options to control the process and it is expected to achieve better performances. In general, the number of sensors need not be equal to the number of controlled variables p: as in the case of actuators, the more sensors, the better.

In dealing with multivariable systems, some extra concepts are relevant:

- **grouping (subdivision) and pairing.** In principle, one or more inputs may be "attached" to each controlled variable or group of them. The choice of groups influences:
  - **interaction.** The manipulated variables attached to one controlled variable may affect the others,
  - **dominance.** If the effect of the corresponding attached manipulated variable is greater than the others, the coupling presents dominance. The concepts of "greater" effect is, at this moment, qualitative. Later on it should be more precisely formulated,

- **conditioning**. Refers to the different "gains" a multivariable process may present according to the combination (direction) of inputs. If they are significatively different, the process is *ill conditioned* and it may be difficult to control.

The magnitude of all these coupling effects is strongly dependent on the measurement units. Thus, an appropriate scaling is always necessary so that all errors have comparable meaning.

In some cases, the selected manipulated variables and measurements will not appear to be convenient for control purposes and changes in the number and/or position of the actuators and sensors will be required.

In a multivariable control problem, the pairing of input/output variables, the effect of the interactions and the options for decoupling different control goals are issues to be considered at the earlier stages of the design [119, 46].

1.11 Implementation and Structural Issues

Although there are still many control systems based on pneumatic, hydraulic, electric or electronic components, most of them are implemented digitally. The operational amplifier has been the kernel of many control devices, but nowadays digital control technology is considered as the general implementation technology.

Thus, even if the controlled plant is a CT system, the controller is a DT one. If the sampling period is short enough, most of the CT techniques can be directly translated into DT implementations, but, if this is not the case, intersampling behaviour and performance degrading should be taken into account. Otherwise, pure digital control design techniques should be used.

Digital control simplifies the control implementation. At the end, the controller device is implemented as an algorithm, a small part of the application code. The same computer can be used, without additional cost, to implement many controllers. This involves not only specific hardware such as reliable computers and communication networking, but also a real time software to guarantee the actions delivering time. Also, in the case of MIMO systems, the dynamics of all the controlled variables is not necessarily the same, thus requiring different time scheduling. But due to the resources limitation there are a number of issues to be considered:

- word-length. Both the variables and the coefficients are discretised and represented by finite-length string of characters. The computation accuracy is bounded,
- time constraints. The same CPU must run a number of different tasks and the time available for each task is limited,
- reliability. The failure of a task may affect the execution of a critical one.

DT control implementation may be centralised or decentralised. In the first case, all the data are processed by a unique CPU producing the control signals

for all the manipulated variables. Reliability, networking and task scheduling are the main issues. In decentralised control, each control loop may be allocated to a digital unit, but still a lot of networking and real time scheduling is required to coordinate the behaviour of the whole system.

One additional feature of digital control is the ability to easily combine the control tasks at different levels: local regulation, coordination, supervision and operator interaction. In this setting, binary logic decision systems are coupled with the lower level controllers, leading to hybrid complex systems.

1.12 Summary of the Chapters

The previous sections have outlined the problems to be solved in designing a multivariable control system, from the conception, the modelling and simulation phase to the implementation. A deeper understanding of them and their solutions is the objective of the rest of the book. As described in the Preface, the book is organised in a set of main chapters covering the main topics and a set of appendices where revision of concepts or more involved details on some techniques are placed.

The next chapter discusses the different model types available and the transformations between them. Chapter 3 details analysis of the system properties that can be inferred from their models (gain, stability, structure). Chapter 4 describes in more detail the objectives of control and the alternatives in solving the associated problems briefly outlined in this introductory chapter (closed-loop properties, feedforward control, *etc.*). Chapter 5 presents the methodologies for controlling MIMO plants based on SISO ideas by setting multiple control loops, decoupling and creating hierarchies of cascade control. Chapter 6 describes some *centralised* control strategies, where all control signals and sensors are managed as a whole by means of matrix operations. Pole placement state feedback and observers are the main result there. Chapter 7 deals with controller synthesis by means of optimisation techniques. The linear quadratic Gaussian (LQG) framework and an introduction to linear fractional transformation (LFT) norm-optimisation (mixed-sensitivity) are covered. Chapter 8 discusses how to guarantee a certain tolerance to modelling errors in the resulting designs. It deals with the robustness issue from an intuitive framework and presents the basics of robust stability and robust performance analysis. Mixed sensitivity is introduced as a methodology for controller synthesis. Lastly, Chapter 9 deals with additional issues regarding implementation, non-linearity cancellation and supervision.

The appendices are devoted to review basic concepts on SISO, matrix analysis and signal and system norms, including also some technicalities about optimisation (derivation of the linear quadratic regulator equations), stochastic processes (derivation of the Kalman filter) and providing additional information on robust control methods.

2

Linear System Representation: Models and Equivalence

In this chapter, models used for system simulation and model-based control design are presented. The treatment is focused on linear systems and the linearised approximation of non-linear systems due to the necessary limitation in space. As disturbance rejection is a key objective in many control applications, disturbance models are also introduced.

2.1 Introduction: Objectives of Modelling

In the previous chapter, it was shown that the "ideal" control requires the inversion of the plant model. Thus, any control structure will take advantage of a good process model to compute the control action, even if the model is not perfect.

In this book, process models are tools for designing the control system, for simulating the behaviour of the controlled system, and for analysing its properties and evaluating the goals' achievement. Thus, their level of detail, range of validity and presentation will be determined by their use.

For each application, control goal or design methodology, a given model will be more or less suitable. Given a process, different models can be attached to it, some of them being equivalent, but, in any case, all them should be "coherent" [84].

For instance, for regulatory and tracking purposes, a CT/DT dynamic model would be required, but for production optimisation or management a simplified and aggregated model, or even a steady-state model, would be more appropriate. For alarm treatment, a discrete-event model will represent the evolution from one operating condition to the next, probably combined with some regulatory actions.

State-based models will be extensively used in this book, due to their relationship with first-principle models and their ease of computer implementation, as well as the availability of computer aided control system design packages for them.

Other than processes, signals should be also modelled. They can be considered as the output of processes (generators) with some particular properties. In particular, deterministic disturbances (such as steps, ramps or sinusoidal variations) can be modelled as the output of uncontrollable generators, and stochastic disturbances will be mainly modelled by their mean and variance properties.

2.2 Types of Models

As previously mentioned, based on the type of signals involved in the model we can find models of different natures: CT, DT, discrete-event, hybrid or stochastic models.

The usual CT and DT signals are functions of time, as defined in Appendix A.1. Multivariable signals are composed by stacking a set of individual signals in column vector form.

A *binary* or *logical* signal only takes two possible values, being synchronous if changes are only allowed at predefined time instants or asynchronous if the changes may happen at any moment.

Random variables and stochastic processes, being characterised by their statistical properties, will be considered in a later section and in Appendix E.

Although different kinds of models can be defined, unless otherwise stated, the hypothesis of Section 1.4, namely linearity, time-invariance and lumped parameters (finite-dimensional system), will be assumed to hold.

A non-linear system is a broader (and more common) representation of actual processes. The diversity of options and their specific and usually more difficult mathematical treatment puts the study of non-linear systems out of the scope of this book. Some simple cases will be outlined in Section 9.5. non-linearity, time-variation and spatial variation will be accommodated by control systems that are tolerant enough regarding modelling error.

Locality. The models usually only represent the relationship between increments of the variables around a given *operating point*. This is quite usual in modelling non-linear systems if we are interested in their approximate linearised behaviour around an equilibrium point. In general, non-linear models are better suited to modelling the global behaviour of a process, relating absolute values of the variables.

Variables. Based on the kind of variables involved, we can define: input/output or *external* models, and state variables or *internal* models. In the case of external models, we can also consider the so-called *black-box* models, where only the input and output variables are involved, or *white-box* models, where the internal structure of the process is somehow represented.

Methodology. The last distinction can be also related to the approach followed to obtain the model. If the basis of the process operation is known,

its dynamical behaviour can be expressed by balance and fundamental equations, leading to a *first-principle model*. If, on the other hand, this fundamental behaviour is unknown or the resulting equations would be too detailed and complicated, and it is possible to experiment with the process, its response to some excitation can be used to get an *experimental model* representing its input/output behaviour, without any reference to what happens internally.

2.3 First-principle Models: Components

Let us consider the following illustrative example.

Example 2.1 (First-principle modelling). Let us consider a continuous-flow stirred tank reactor (CSTR)[1], where a first-order exothermic reaction $A \rightarrow B$ happens, with a cooling jacket [89]. We have a rough model of the CSTR, knowing that due to the entrance of a flow rate input stream, F_o, with C_{ao} concentration of component A, at a temperature T_o, there is an internal level, h, temperature, T, and component A concentration, C_a, and there is an outlet flow rate, F, at temperature T and concentration C_a. This can be represented by the block shown in Figure 2.1, where the cooling jacket water flow, F_j, enters at temperature T_{jo}, leaving at temperature T_j, and the total jacket volume, V_j, is fixed.

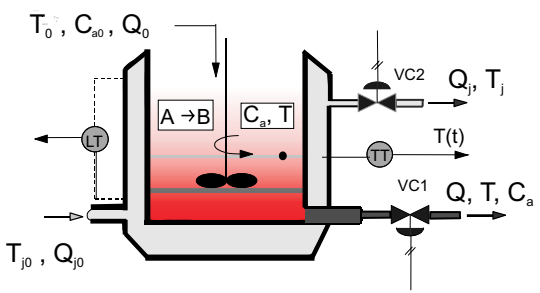

Figure 2.1. CSTR reactor

If all the processes inside the reactor are known, the following set of equations can be written:

1. Total mass balance:
$$\frac{dV}{dt} = F_o - F \tag{2.1}$$
2. Mass balance on component A:
$$\frac{d(VC_a)}{dt} = F_o C_{ao} - FC_a - \alpha V C_a e^{-\frac{E}{RT}} \tag{2.2}$$
where R is the perfect gas constant and α is the pre-exponential factor from the Arrhenius law, and E is an activation energy.

[1] The work about this process has been carried out in collaboration with M. Perez [101].

3. Energy balance in the reactor:

$$\frac{dT}{dt} = \frac{F_o T_o - FT}{V} + \frac{H\alpha}{\rho c_p} C_a e^{-\frac{E}{RT}} - \frac{UA}{\rho c_p V}(T - T_j) \tag{2.3}$$

where H is the reaction heat, ρ and c_p are the density and heat capacity respectively of the inlet and outlet streams, U is the overall heat transfer coefficient through the jacket, A is the transfer area.

4. Energy balance in the jacket:

$$\frac{dT_j}{dt} = \frac{F_j}{V_j}(T_{jo} - T_j) + \frac{UA}{\rho_j c_{pj} V_j}(T - T_j) \tag{2.4}$$

where ρ_j and c_{pj} are the density and heat capacity respectively of the cooling stream.

In this way, a set of non-linear differential equations represents the CSTR dynamics.

Simplifications. If we were only interested in the reactor components evolution, or the reaction were isothermal, the energy-balance equations would be missing:

$$\frac{dV}{dt} = F_o - F \qquad \frac{d(VC_a)}{dt} = (F_o C_{ao} - FC_a) - \alpha V C_a e^{-\frac{E}{RT}} \tag{2.5}$$

Temperature variations, if any, would amount to having time-variance in the parameters on the reduced model.

If interest were focused on long-term production, only the static relation among the variables would be relevant. Then, for given input constant values, if the operation is stable, an equilibrium point will be reached and a set of (algebraic) equations will model the steady-state behaviour. Steady-state equations are obtained by setting to zero all derivatives (as magnitudes are constant):

$$\begin{gathered} F = F_o; \qquad F_o(C_{ao} - Ca) - \alpha V C_a e^{-\frac{E}{RT}} = 0 \\ \frac{F_o}{V}(T_o - T) + \frac{H\alpha}{\rho c_p} C_a e^{-\frac{E}{RT}} - \frac{UA}{\rho c_p V}(T - T_j) = 0 \\ \frac{F_j}{V_j}(T_{jo} - T_j) + \frac{UA}{\rho_j c_{pj} V_j}(T - T_j) = 0 \end{gathered} \tag{2.6}$$

Similar to the reactor equations, basic equations describing the elements' phenomena can be used. We distinguish between static elements, leading to algebraic equations, or dynamic elements.

Static elements. Examples of static behaviour are, for instance:

- resistors: $V = IR$,
- wall heat transmission: $Q_{12} = k(T_1 - T_2)$,
- springs: $f = K(l - l_0)$,
- outlet flow: $F_{12} = k\sqrt{h}$,
- pipes: $F = k(P_1 - P_2)$,
- viscous friction: $f = kv$,
- balances, such as:
  - $\sum f_i = 0$, the total force applied to a body,

- $\sum F_i = 0$, the net flow in a pipe junction if there is no storage, or
- $\sum V_i = 0$ the total voltage drop in a loop, and so on.

The meaning of the different constants and parameters is clear for those introduced in the respective field. Some of these expressions are approximations of non-linear relationships (friction, spring, resistor) but some others, like the tank outlet flow, are explicitly non-linear. Under some circumstances, a linear approximation will be possible.

Dynamic elements. Dynamic elements are those where the involved variables are not related instantaneously but in different times or by time increments. For instance, accummulative or delay components, such as:

- capacitors ($\frac{dV}{dt} = \frac{1}{C}I$) and coils ($\frac{dI}{dt} = \frac{1}{L}V$),
- heat storage: $\frac{dT}{dt} = \frac{1}{MC_e}(Q_{in} - Q_{out})$,
- mass storage: $\frac{dV}{dt} = \frac{1}{M}(F_{in} - F_{out})$,
- motion equations: acceleration $\frac{dv}{dt} = \frac{1}{M}F_{tot}$, $\frac{d\omega}{dt} = \frac{1}{I}T_{res}$, or velocity $\frac{dx}{dt} = v$, $\frac{d\phi}{dt} = \omega$ for linear or angular motions,
- chemical reactions: $\frac{dx}{dt} = f(x_i, T)$, as previously used, where the product composition evolves with time,
- transportation belt: $m_{out}(t) = m_{in}(t - \tau)$,
- stack or queue systems: $n(k+1) = n(k) + \sum u_i(k)$.

We must notice that these relationships (and many others) are similar, leading to a component behaviour that is common to some of them. These analogies allow for a unified treatment of any dynamical system without much relevance of the supporting technology. It should be pointed out that the last dynamic equation is slightly different, as the time is discrete and the variables are assumed to be integers. We will see more of that later on.

Basic equations. The equation

$$\frac{dy(t)}{dt} = \alpha u(t) \tag{2.7}$$

represents the *storage* of u, α being a scaling constant to deal with the appropriate measurement units of y and u. In fact, the same relationship can be written as:

$$y(t) = \alpha \int_0^t u(\tau)\, d\tau + y(0)$$

The equivalent DT equations would be, respectively:

$$y(k+1) - y(k) = \alpha u(k) \qquad y(k) = \alpha \sum_{i=0}^{k-1} u_i + y(0) \tag{2.8}$$

In DT it is also very easy to represent some delays such as:

$$y(k) = u(k - d) \tag{2.9}$$

where d is the number of time delay intervals in this relationship.

These storage and delay equations are the basic dynamic equations and will be the kernel in defining the state space model in the next section.

2.4 Internal Representation: State Variables

A process is composed of a number of interconnected elements as described above. In order to completely define a process model, we must get the same number of independent equations as internal variables, allowing theoretically the computation of the internal variables given the external input, $u(t)$.

Some of the equations in the model will be dynamic and some of them will be algebraic. By substitution, some equations can be removed. A so-called *normalised representation* can be obtained by removing the algebraic equations and manipulating the rest of them to be expressed as storage equations (2.7) or (2.8).

State equation. If all the dynamic equations are first-order differential equations, they can be arranged in a normalised way. Denoting by *state variables* the storage variables previously defined, the process model could be summarised by the so-called state equation:

$$\frac{dx(t)}{dt} = f(x(t), u(t), t) \tag{2.10}$$

where $x \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}^m$ is the input vector and f is an n-dimensional vector of non-linear functions. The argument t explicitly indicates the possibility of being time-varying functions. This system representation is denoted as being an n-order system.

Output equation. If the output variables, $y \in \mathbb{R}^p$, are not the state variables, they will be related to them, and possibly also the inputs, by the (algebraic) output equation:

$$y(t) = g(x(t), u(t), t) \tag{2.11}$$

where g is a p-dimensional vector of non-linear functions. The set of state and output equations is denoted as a *system realisation* or normalised *state space representation*.

Example 2.2. Looking at the equations in Example 2.1, let us rewrite the model as state equations. Only Equation (2.2) should be transformed into:

$$\frac{dC_a}{dt} = \frac{F_o(C_{ao} - C_a)}{V} - \alpha C_a e^{-\frac{E}{RT}}$$

the rest of the equations, (2.1), (2.3) and (2.4), constituting a fourth-order system with states (V, C_a, T, T_j).

For DT processes, the state and output equations will be, respectively:

$$x(k+1) = f(x(k), u(k), k) \tag{2.12}$$

$$y(k) = g(x(k), u(k), k) \tag{2.13}$$

where the argument k stands for the integer instant of time.

The delay element, (2.9), in CT models requires an infinity of state variables to be represented (all the past input values), leading to infinite-dimensional systems. Some approximations are usually done, (2.42).

However, in a DT process, if the time delay is a multiple of the period, it will admit a simplified treatment introducing as many delayed variables as delay intervals. For instance, for $d = 2$, $y(k) = u(k-2)$ is equivalent to:

$$x_1(k+1) = x_2(k); \quad x_2(k+1) = u(k); \qquad y(k) = x_1(k) \tag{2.14}$$

The Concept of State

The previously defined state variables have a number of interesting properties:

- **memory**. They summarise the past history of the process,
- **state as internal variables**. They are not directly connected to the input, but their derivatives (in DT, future values) as well as any other process variable can be expressed as a function of these and the inputs,
- **minimality**. There is a minimum number of state variables so that the process internal model cannot be further reduced by removing any internal variable; otherwise, the dynamic equations will be of an order higher than one,
- **non-uniqueness**. Any set of n-independent internal variables, not directly connected to the input, represents the system. By independent we mean each one representing a different storage process.
  If the set vector $x(t)$ is a state vector, any variable vector $\bar{x}(t)$ such that

$$x(t) = T\bar{x}(t) \tag{2.15}$$

  T being an n-square regular matrix, is also a state vector.
  Indeed, knowing $x(t)$, the new state can be computed as $\bar{x}(t) = T^{-1}x(t)$ and *viceversa* (2.15). The T-matrix represents a linear state transformation or a *similarity transformation*. The transformation of Equation 2.10 yields:

$$\frac{d\bar{x}}{dt} = T^{-1} f(T\bar{x}(t), u(t), t)$$

  so it is also a normalised representation.

2.5 Linear Models and Linearisation

As previously mentioned, most of our study is referred to linear, time-invariant (LTI) systems. For this kind of systems, the state equation (2.10) is simplified: the time variable is no more an argument (time-invariance) and the functions f express *linear* combinations of the state and input variables. Thus:

$$\begin{aligned} \frac{dx_1}{dt} &= a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + b_{11}u_1 + \cdots + b_{1m}u_m \\ &\vdots \\ \frac{dx_n}{dt} &= a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n + b_{n1}u_1 + \cdots + b_{nm}u_m \end{aligned} \tag{2.16}$$

or, in matrix form

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.17}$$

Similarly, for the output equation (2.11), it would be:

$$\begin{aligned} y_1 &= c_{11}x_1 + c_{12}x_2 + \cdots + c_{1n}x_n + d_{11}u_1 + \cdots + d_{1m}u_m \\ &\cdots \\ y_p &= c_{p1}x_1 + c_{p2}x_2 + \cdots + c_{pn}x_n + d_{p1}u_1 + \cdots + d_{pm}u_m \end{aligned} \tag{2.18}$$

or, in matrix form:

$$y(t) = Cx(t) + Du(t) \tag{2.19}$$

The state space model involves the four matrices (A, B, C, D) having the following dimensions and meaning:

- **A**, the $n \times n$ system matrix, represents the internal interconnection among state variables,
- **B**, the $n \times m$ input matrix, represents the input-to-state direct connection,
- **C**, the $p \times n$ output matrix, represents the state-to-output direct connection, and
- **D**, the $p \times m$ coupling matrix, represents the input-to-output direct connection or input/output coupling.

So, in total, the number of parameters is $n^2 + n \times m + p \times n + p \times m$. A system will be denoted by the 4-tuple shorthand notation $\Sigma := (A, B, C, D)$.

It is easy to show that under a similarity transformation such as (2.15), the new equivalent state space model will be $(\bar{A}, \bar{B}, \bar{C}, \bar{D})$, being:

$$\bar{A} = T^{-1}AT; \bar{B} = T^{-1}B; \bar{C} = TC; \bar{D} = D \tag{2.20}$$

and the new state and output equations:

$$\begin{aligned} \dot{\bar{x}} &= \bar{A}\bar{x} + \bar{B}u \\ y &= \bar{C}\bar{x} + Du \end{aligned}$$

Some transformations are of special interest for deriving or emphasising some properties of the representation, denoted as *canonical representations*, usually

reducing the number of significant parameters. For instance, if the eigenvalues of A are distinct, it is possible to find a matrix T such that $\bar{A} = \Lambda$ is diagonal (or, in a general case, a Jordan canonical form). Appendix B details some essential definitions and properties regarding matrices.

If we want to make a distinction between manipulated and disturbance inputs, a generalised state space model can be expressed by:

$$\dot{x}(t) = Ax(t) + Bu(t) + B_d d(t) \tag{2.21}$$

$$y(t) = Cx(t) + Du(t) + D_d d(t) \tag{2.22}$$

where $d \in \mathbb{R}^q$ is a q-dimensional vector of disturbances.

Equilibrium points. Under some conditions, the non-linear model (2.10 and 2.11) can be approximated by a linear one, in particular, around an equilibrium point. For a constant value of the input vector, u^0, an *equilibrium point* is defined as the state vector (set of variables), x^0, solution of:

$$0 = f(x, u^0) \tag{2.23}$$

It is worth noting that for the same input u^0 there may be one, none or many solutions of (2.23), and thus, there will be the corresponding number of equilibrium points.

Any other process variable will reach a steady-state value. For instance, the output vector will be:

$$y^0 = g(x^0, u^0) \tag{2.24}$$

Note that the equilibrium point can be stable or unstable. Further detail will be given later.

Linearisation. If the vector functions in the non-linear model, f and g, are continuous and derivable at an equilibrium point (x^0, u^0), a *linearised* model can be attached to the process by means of a truncated Taylor series expansion of these functions.

For small input amplitudes, the second and higher-order terms are negligible, and an approximate linearised process model in the form (2.17)–(2.19) can be derived:

$$\dot{\bar{x}} = A\bar{x} + B\bar{u}; \qquad \bar{y} = C\bar{x} + D\bar{u}$$

where $\bar{x} = x - x^0$ and $\bar{u} = u - u^0$ represent the variable increments around the equilibrium point, and the elements of the corresponding matrices are the Jacobian components of the non-linear functions:

$$a_{ij} = \frac{\partial f_i}{\partial x_j}(x^o, u^o) \quad i, j = 1, \ldots, n \tag{2.25}$$

$$b_{ij} = \frac{\partial f_i}{\partial u_j}(x^o, u^o) \quad i = 1, \ldots, n \quad j = 1, \ldots, m \tag{2.26}$$

$$c_{ij} = \frac{\partial g_i}{\partial x_j}(x^o, u^o) \quad i = 1, \ldots, p \quad j = 1, \ldots, n \tag{2.27}$$

$$d_{ij} = \frac{\partial g_i}{\partial u_j}(x^o, u^o) \quad i = 1, \ldots, p \quad j = 1, \ldots, m \tag{2.28}$$

Remark 2.3. If the process evolves to another equilibrium point, the linearised model parameters change. If the variation in the variables is large enough to invalidate the higher-order terms' truncation, the linear model is not valid anymore.

Example 2.4. Let us consider the CSTR of Example 2.1. Assume a constant volume inside the reactor ($F_o = F$), as well as the nominal conditions and parameters in Table 2.1.

Table 2.1. Reactor notation and steady-state variables.

Variable	*Description*	*Value*
C_{a0}	Input concentration (kmol A/m^3)	8
V	Reactor volume (m^3)	1.36
T_o	Inlet flow temperature (K)	294.7
V_j	Jacket volume (m^3)	0.085
α	Arrhenius exponential factor (h^{-1})	7.08×10^{10}
E	Activation energy (kJ/kmol)	69815
U	Heat transmission coeff. (kJ/hm^{2}K)	3065
A	Heat transmission surface (m^2)	23.22
T_{jo}	Cooling water input temperature (K)	294.7
R	Perfect gas constant (kJ/kmolK)	8.314
H	Reaction heat (kJ/kmol)	69815
cp	Thermal capacity (kJ/kg·K)	3.13
cp_j	Water thermal capacity (kJ/kg·K)	4.18
r	Reactive and product density (kg/m^3)	800
r_j	Water density (kg/m^3)	1000

Substituting the data in the table into (2.2)–(2.4), the particular model is:

$$\frac{d(C_a)}{dt} = \frac{F_o}{1.36}(8 - C_a) - 7.08 \times 10^{10} C_a e^{-8397.3/T}$$

$$\frac{dT}{dt} = \frac{F_o}{1.36}(294.7 - T) + 197.4 \times 10^{10} Ca.e^{-8397.3/T} - 20.8987(T - T_j)$$

$$\frac{dT_j}{dt} = \frac{F_j}{0.085}(294.7 - T_j) + 200.3076(T - T_j)$$

Thus, the state equation with $x = [C_a \quad T \quad T_j]'$, and $u = [F \quad F_j]'$ would be:

$$\dot{x}_1 = f_1(x,u) = 0.7353u_1(8 - x_1) - 7.08 \times 10^{10} x_1 e^{-8397.3/x_2}$$

$$\dot{x}_2 = f_2(x,u) = 0.7353u_1(294.7 - x_2) + 197.4 \times 10^{10} e^{-8397.3/x_2} - 20.90(x_2 - x_3)$$

$$\dot{x}_3 = f_3(x,u) = 11.76u_2(294.7 - x_3) + 200.31(x_2 - x_3)$$

The operating point is defined assigning, for instance, the inlet flow $F_o = 1.13$ m^3/h and cooling flow in the jacket $F_j = 1.41$ m^3/h. Thus, the operating point, denoted by x_s, can be computed by solving the above system equation, making the

left-hand derivative terms null as in (2.6). Due to the non-linearity, in this case there are three equilibrium points. One of the options is:

$$x_s = [C_{as} \quad T_s \quad T_{js}]^T = [4.031 \quad 333.6 \quad 331.4]^T$$

The Jacobian elements, $a_{ij} = \frac{\partial f_i}{\partial x_j}(x_s, u_s)$, of the linearised model can be easily computed. For instance, the first linearised equation is:

$$\frac{d\bar{x}_1}{dt} = -(0.7353u_{1s} + 7.08 \times 10^{10} e^{-8397.3/x_{2s}})\bar{x}_1$$
$$+(-7.08 \times 10^{10} x_{1s} e^{-8397.3/x_{2s}} \frac{8397.3}{x_{2s}^2})\bar{x}_2 + 0.7353(8 - x_{1s})\bar{u}_1$$

Proceeding with all the elements, the following linear model is obtained:

$$A = \begin{pmatrix} -1.705 & -0.2519 & 0 \\ 23.088 & -28.71 & 20.9 \\ 0 & -200.3 & -216.89 \end{pmatrix}; \qquad B = \begin{pmatrix} 2.918 & 0 \\ -28.6 & 0 \\ 0 & -415.29 \end{pmatrix}$$

Using the MATLAB® command[2] **ss**, assuming $C = I$, that is, all the state components are measurable, and putting $D = 0$, the state space representation is created:

```
sys=ss(A,B,C,D)
```

The individual matrices can be retrieved as `sys.a`, `sys.b`, *etc.*, if needed.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `gradient,diff,ss`.

Remark 2.5. A linearised model can also be obtained around a nominal trajectory. If the system input is defined in a time interval, $u^0(t), \forall t \in [t_i, t_f]$, and the nominal state trajectory is given by $x^0(t)$, a linearised model can be attached to this nominal trajectory by relating the variations of the trajectory with respect to variations of the nominal input. Nevertheless, the linearised model would be, generally, time-varying, as the linear model will change with the state[3].

A good example of this situation is a web treatment station, quite common in the plastic, paper, metallurgical and many other industrial sectors. The web is transferred from a coil to another one through the treatment area. The basic goal is to control the rotational speed of the leading shaft as well as the position of the tensional arm to keep a constant web velocity and tension. Under nominal operating conditions, the diameter and inertia of the leading coil are increasing with time, thus the nominal values of the parameters change. The nominal shaft speed is time-varying to keep the web velocity constant, and a linearised model can be obtained around this reference.

[2] All the commands in this book refer to MATLAB® version 5.3.

[3] In fact, the Jacobian coefficients in (2.25) would be like: $a_{ij}(t) = \frac{\partial f_i}{\partial x_j}(x^o(t), u^o(t))$.

Advantages and drawbacks. The state space representation has the following advantages:

- it may be obtained by the process behaviour description, as realised with the reactor example, with an immediate physical interpretation, if so derived,
- it presents separately the input effect and the measurement system,
- it is valid for linear and most non-linear systems,
- it can also represent the action of external disturbances (they are just treated as non manipulated inputs),
- it provides the full (internal) description of the process, allowing the linking of the model with the internal structure,
- there is a bunch of control system design approaches based on this representation,
- it is valid for SISO and MIMO systems, with many common properties. One key point in the MIMO case is that a complex system of interactions is described with the minimal set of independent variables.

But also, some drawbacks should be taken into account:

- the model may be over-parameterised with respect to other representations,
- it is not unique and it may be unnecessarily complicated (*i.e.*, over-dimensioned), if irrelevant physical phenomena are modelled or a non-minimal representation is used,
- the *frequency response* is not easily connected with the parameters of the model,
- delays cannot be modelled in the normalised representation, only approximately, by (2.43).

Anyway, it will be shown that it is possible to get this representation from any other, and to transform it to any other as well, at least in the linear case.

Discrete Models

If instead of differential equations the process is defined by difference equations, a discrete state space model can be used. In this case, the state and output equations are, respectively:

$$x(k+1) = Ax(k) + Bu(k) \tag{2.29}$$

$$y(k) = Cx(k) + Du(k) \tag{2.30}$$

Linearisation. Equilibrium points in non-linear discrete systems are calculated by replacing all instances of a particular variable, x (with different delays), with an equilibrium value, x^0. For time-invariant DT systems, the equilibrium points are given by the solutions to:

$$x^0 = f(x^0, u^0); \qquad y^0 = g(x^0, u^0) \tag{2.31}$$

Once the (non-linear) equilibrium points are calculated, linearisation is carried out obtaining the Jacobian, as in the CT case.

2.6 Input/Output Representations

In the previous section, we have built up a model by connecting the components according to the system structure, but it is also possible to look at the process as an m-input/p-output information processor. If only these variables are used, the system representation will be composed of a set of p-differential equations of order 1 or greater, as an extension of the differential equation model of a SISO system. An example of this is:

$$\begin{aligned} \ddot{y}_1 + 2\dot{y}_1 - y_2 &= 0.3\dot{u}_1 + u_1 \\ \dot{y}_1 + \ddot{y}_2 &= 2.5u_2 \end{aligned} \tag{2.32}$$

This is a linear time-invariant CT two-input-two-output (TITO) system. Again, these differential equations could be non-linear and time-variant (even partial differential equations can be considered to represent distributed parameter systems or ∞-dimensional systems). For simplicity, we will assume the above equation system is formed by linear differential equations with constant coefficients.

2.6.1 Polynomial Representation

The Laplace transform (Appendix A) can be applied to the equation system, (2.32). If the terms involving the initial condition vanish or cancel, that is, if the initial variable values correspond to an equilibrium point of the system, an algebraic equation system on the Laplace variable, s, is obtained:

$$\begin{aligned} s^2y_1(s) + 2sy_1(s) - y_2 &= 0.3su_1(s) + u_1(s) \\ sy_1(s) + s^2y_2(s) &= 2.5u_2(s) \end{aligned}$$

If the terms related to the same variable are put together, the model can be expressed by the following matrix equation:

$$\begin{pmatrix} (s^2+2s) & -1 \\ s & s^2 \end{pmatrix} \begin{pmatrix} y_1(s) \\ y_2(s) \end{pmatrix} = \begin{pmatrix} (0.3s+1) & 0 \\ 0 & 2.5 \end{pmatrix} \begin{pmatrix} u_1(s) \\ u_2(s) \end{pmatrix} \tag{2.33}$$

In a generic MIMO system, the model would be:

$$D(s)y(s) = N(s)u(s) \tag{2.34}$$

where $D(s)$ and $N(s)$ are *polynomial matrices*, with dimensions $p \times p$ and $p \times m$ respectively, whose elements are polynomials on the s variable. This

new model, (2.34), is called the *differential operator* model, as a result of the use of the s-variable, or *polynomial representation*. Instead of the four real matrices (A, B, C, D) needed in the state space representation, only two matrices (D, N) represent the system, although their elements are polynomials (see Appendix B.6).

For a given system, the polynomial representation is not unique. In fact, if $P(s)$ is a square, $p \times p$, regular polynomial matrix, the process may be modelled by the new pair $(\bar{D}, \bar{N})$, being:

$$P(s)D(s)\,y(s) = P(s)N(s)\,u(s);\ \Rightarrow\ \bar{D}(s)\,y(s) = \bar{N}(s)\,u(s)$$

Example 2.6. A trivial example would be, in (2.33), to multiply by:

$$P(s) = \begin{pmatrix} 1 & 0 \\ -1 & (s+2) \end{pmatrix}$$

leading to an alternative representation of the same system with:

$$\bar{D}(s) = \begin{pmatrix} (s^2+2s) & -1 \\ 0 & s^2(s+2)+1 \end{pmatrix};\ \bar{N}(s) = \begin{pmatrix} (0.3s+1) & 0 \\ -(0.3s+1) & 2.5(s+2) \end{pmatrix}$$

The polynomial representation of DT models is similar, using the z-variable.

Advantages and drawbacks. The polynomial representation also presents some properties. Among the advantages:

- it relates the input and output variables,
- it may be used to describe subsystems,
- it may be obtained by linearisation of a set of non-linear differential equations,
- there are some *ad hoc* approaches to designing control systems based on polynomial operators,
- the number of parameters is lower than in the state space model.

However:

- it is not unique,
- it is not easy to handle (requiring symbolic computation and matrix inversions),
- it does not allow for easy autonomous system analysis (free response),
- it is only valid for linear systems.

2.6.2 Transfer Matrix

From (2.34), assuming $D(s)$ is invertible (*i.e.*, the p-differential equations are independent), the output can be expressed as a function of the input:

$$y(s) = D^{-1}(s)N(s)u(s) = G(s)u(s) \tag{2.35}$$

The rational matrix, $G(s) = D^{-1}(s)N(s)$, dimension $p \times m$, is denoted as the process *transfer matrix*, and its elements are quotients of polynomials (see Section B.6). It is a function of the Laplace variable, s, and can be considered as an operator analogous to transfer functions in SISO systems. This is one of the most attractive properties of this representation.

Example 2.7. In the example, (2.32), it is easy to obtain:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} (s^2+2s) & -1 \\ s & s^2 \end{pmatrix}^{-1} \begin{pmatrix} (0.3s+1) & 0 \\ 0 & 2.5 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

and, for instance, using the *Mathematica*® language:

```
GG=Inverse[{{s^2+2s,-1},{s,s^2}}].{{0.3s+1,0},{0,2.5}}
```

it yields:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} \frac{s(0.3s+1)}{s^3+2s^2+1} & \frac{2.5}{s^4+2s^3+s} \\ \frac{-(0.3s+1)}{s^3+2s^2+1} & \frac{2.5s+5}{s^3+2s^2+1} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{2.36}$$

In this case, the system is represented by one $p \times m$ matrix, $G(s)$, whose elements are rational functions. For instance:

$$G_{i,j}(s) = \left. \frac{y_i(s)}{u_j(s)} \right|_{u(s)_k=0, \forall k \neq j} \tag{2.37}$$

is the SISO transfer function between the input, u_j, and the output, y_i. That is, the Laplace transform of the i-output over that of the j-input, assuming that the rest of the inputs as well as the initial condition terms are null.

In general, $p \leq m$. That is, the number of output (controlled) variables is lower than the number of input (manipulated) variables. If $p = m$, $G(s)$ is a square matrix and some matrix operations will be allowed.

Again, the internal structure of the process is lost and this representation only provides information about the effect of the inputs in the outputs. If disturbances are considered, the model can be extended to become:

$$y(s) = G_p(s)u(s) + G_d(s)d(s) \tag{2.38}$$

where $d(s)$ is the Laplace transform of the vector of disturbances. Similarly, the element $G_{d_{i,j}}(s)$ is the transfer function between the j-disturbance and the i-output. Equation (2.38) is equivalent to (2.21) and (2.22).

Discrete case. The discrete transfer matrix representation of a general DT process, equivalent to (2.38), is expressed by:

$$y(z) = G_p(z)u(z) + G_d(z)d(z) \tag{2.39}$$

Advantages and drawbacks. The transfer matrix representation presents a number of advantages:

- it relates the input and output variables,
- it may be used as an operator to combine subsystems,
- it is unique (except for common factors in the rational elements, to be reduced),
- there are some *ad hoc* approaches to designing control systems based on this model structure,
- the number of parameters is minimised,
- its elements represent one-input to one-output connections, allowing experimental identification and partial model validation. Although it is out of the main scope of this book, its importance should be stressed. Some options for these tasks are outlined in Appendix A.4),
- in the same way, it shows up the *interactions* among different inputs and outputs,
- delays can be considered.

However:

- the elements are rational functions (including exponential terms if delays are considered),
- some global system properties, easily derived from the state matrix A, do not appear so clearly,
- it does not allow for the autonomous system analysis,
- it is only valid for linear systems.

Example 2.8. The internal representation of the CSTR, Example 2.4, is converted into a transfer matrix by the MATLAB® command `tf` (or `zpk` to get it in factorised form), as shown in the following example:

```
sys=ss(A,B,C,D); G=tf(sys); Gp=zpk(sys)
     TF from input 1 to output...          From input 2 to output...
      2.918 (s+57.43)(s+190.6)                    2186.3814
#1:  ----------------------------    #1:  ---------------------------
     (s+1.83) (s+54.36) (s+191.1)         (s+1.83)(s+54.36)(s+191.1)

      -28.6 (s+216.9)(s-0.6506)              -8679.561 (s+1.705)
#2:  ----------------------------    #2:  ---------------------------
     (s+1.83)(s+54.36)(s+191.1)          (s+1.83)(s+54.36)(s+191.1)

          5728.58 (s-0.6506)               -415.29(s+28.49)(s+1.922)
#3:  ----------------------------    #3:  ---------------------------
     (s+1.83)(s+54.36)(s+191.1)           (s+1.83)(s+54.36)(s+191.1)
```

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `ss,tf,zpk,ss2tf,tf2ss`.

Systems with Time Delay

Transportation phenomena, among others, introduce pure time delays in the relation between variables. If:

$$y(t) = u(t - \tau) \tag{2.40}$$

$y(t)$ is a delayed version of signal $u(t)$. In fact, if $y(t)$ is the output and $u(t)$ is the input of a system, this is a *distributed* process and the general mathematical treatment of this kind of system is rather complicated.

On the other hand, the Laplace transform is easy to apply because:

$$\mathcal{L}[y(t)] = e^{-s\tau}u(s)$$

In a delayed system, if the delays appear at either the input or output signals, it is quite common to model it by a transfer matrix.

Example 2.9 (MIMO experimental identification). A mixing process (see Section 5.8.2 on page 162 for description and analysis of steady-state behaviour) has an intermediate buffer tank, a solvent valve and a pure-product valve. Outputs of interest are concentration (C, %) and flow (F, l/s). The experimental response of both outputs to a 10% valve opening on each valve is depicted in Figure 2.2.

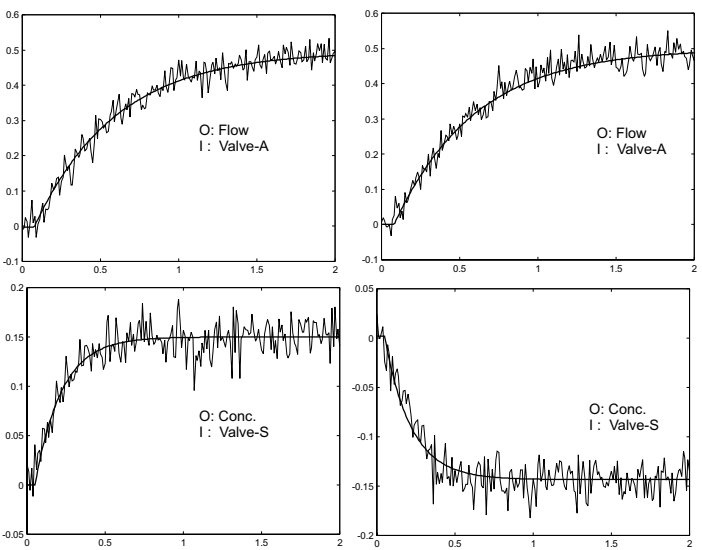

Figure 2.2. Mixing process: step response

Applying experimental identification (Section A.4), the following transfer functions can be approximately determined, targeting a widely-used first-order plus delay type of model:

$$\frac{F}{V_A} = \frac{5e^{-0.08s}}{0.52s+1}; \quad \frac{F}{V_S} = \frac{5e^{-0.08s}}{0.52s+1}; \quad \frac{C}{V_A} = \frac{1.5e^{-0.06s}}{0.17s+1}; \quad \frac{C}{V_S} = \frac{-1.45e^{-0.06s}}{0.17s+1}$$

also depicted in the above figure in thicker line. Hence, the experimental transfer function matrix is:

$$G(s) = \begin{pmatrix} \frac{5e^{-0.08s}}{0.52s+1} & \frac{5e^{-0.08s}}{0.52s+1} \\ \frac{1.5e^{-0.06s}}{0.17s+1} & \frac{-1.45e^{-0.06s}}{0.17s+1} \end{pmatrix} \tag{2.41}$$

It is difficult to know if the initial response, heavily masked by noise, includes a delay or it is caused by high-order dynamics. So, there are fundamental issues regarding quality of the model, signal-to-noise ratio, *etc.* that may determine the success of multivariable control strategies based on this crudely approximate model: Chapter 8 is devoted to them.

Unfortunately, the delays do not always appear in such a neat form as in the above example because there are internal and crossed delays and the elements of the transfer matrix can also be complicated[4].

A general approach to deal with time delays is to use the Padé approximation, converting the exponential into an approximated rational form. Based on the exponential approximation:

$$e^{-\tau s} = \frac{e^{-\frac{\tau}{2}s}}{e^{\frac{\tau}{2}s}} = \frac{1 - \frac{\tau}{2}s + \frac{\tau^2}{8}s^2 + \dots}{1 + \frac{\tau}{2}s + \frac{\tau^2}{8}s^2 + \dots} \tag{2.42}$$

the first-order approximation is:

$$e^{-\tau s} \approx \frac{1 - \frac{\tau}{2}s}{1 + \frac{\tau}{2}s} \tag{2.43}$$

Higher-order approximations are more accurate but introduce additional complexity into the model.

For DT representations, (2.9), if the time delay, τ, is a multiple of the time interval in the elements of the sequence, $\tau = dT$, it can be transformed into a shift, or, in $\mathcal{Z}$-transform, to multiply by the term z^{-d}, see Appendix A.3.

Transfer Matrix Poles and Zeros

The elements of the transfer matrix are rational functions of the Laplace variable, s. For a transfer function, $g(s) = \frac{n(s)}{d(s)}$, poles and zeros are the roots of $n(s)$ and $d(s)$ respectively (Appendix A.2.1).

The concept of pole and zero can also be defined for MIMO systems.

Given a transfer matrix, extract the common denominator

$$G(s) = \frac{N(s)}{d(s)}$$

[4] For instance, the closed-loop sensitivity function (4.5) of a SISO delayed system $G = e^{-s}(s+1)$ with proportional control $k = 4$ is $(1+kG)^{-1} = \frac{s+1}{(s+1)+4e^{-s}}$, so e^{-s} cannot be extracted as a factor.

where $N(s)$ is a polynomial matrix and $d(s)$ is a polynomial. Assume for simplicity that it has only single roots, $d(s) = \prod_{i=1}^{n}(s - p_i)$. These are the *poles* of the system. In order to determine the pole's multiplicity, decompose $G(s)$ into partial fractions by decomposing each element, $\frac{n_{ij}}{d(s)} = \sum_{l=1}^{n} \frac{\alpha_{ij,l}}{s-p_l}$. Thus, by suitably arranging α_{ij}, l in matrix form, we can express:

$$G(s) = \sum_{l=1}^{n} \frac{N_l}{s - p_l}; \qquad rank(N_l) = n_l$$

Then, the matrix rank n_l is the multiplicity of the pole p_l.

Assume a square system, $m = p$ for simplicity. In most cases, $G(s)$ is regular, *i.e.*, $rank(G(s)) = m$ for almost all s. The MIMO system transfer matrix *zeros* are the values z_i such that $rank(G(z_i)) < m$.

Let us consider the following trivial examples:

Example 2.10.

$$G(s) = \begin{pmatrix} \frac{-1}{s(s+1)} & \frac{1}{s} \\ \frac{-1}{s} & \frac{1}{s(s+1)} \end{pmatrix} = \begin{pmatrix} \frac{-1}{s} + \frac{1}{s+1} & \frac{1}{s} \\ \frac{-1}{s} & \frac{1}{s} + \frac{1}{s+1} \end{pmatrix} \tag{2.44}$$

$$G(s) = \frac{1}{s(s+1)} \begin{pmatrix} -1 & s+1 \\ -(s+1) & 1 \end{pmatrix} = \frac{1}{s} \begin{pmatrix} -1 & 1 \\ -1 & 1 \end{pmatrix} + \frac{1}{s+1} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.45}$$

Thus, $s = -1$ is a double pole and $s = 0$ is a single one. There are three poles at $p = 0, -1, -1$. For $s \to 0$, the transfer matrix

$$\lim_{s \to 0} G(s) = \lim_{s \to 0} \begin{pmatrix} \frac{-1}{s} & \frac{1}{s} \\ \frac{-1}{s} & \frac{1}{s} \end{pmatrix}$$

loses the rank in one order. Thus, there is a zero at $s = 0$.

Example 2.11.

$$G(s) = \begin{pmatrix} \frac{s+1}{s} & \frac{-2}{s} \\ \frac{-1}{s} & \frac{1}{s} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{\begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix}}{s}$$

Thus, $s = 0$ is a double pole; $z = -1$ is a local zero for g_{11}, but it is not a multivariable zero, because $rank(G(-1)) = 2$. On the other hand, it is easy to check that there is a multivariable zero at $z = 1$.

In the next chapter, a more complete treatment of poles and zeros in MIMO systems, for any representation, is presented.

2.7 Systems and Subsystems: Interconnection

Between a detailed internal representation and a pure black-box one, a system may be represented by the interconnection of a number of subsystems. If the model of each element is known, a global model can be obtained.

For example, in a control system, at least, two subsystems will always be considered: the plant to be controlled and the controller. The structure of each one may be also decomposed into a number of subsystems.

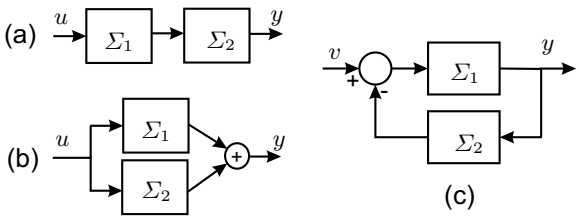

Figure 2.3. Series, parallel and feedback interconnection

2.7.1 Series, Parallel and Feedback Connection

Let us consider the basic subsystem interconnections. Let us define two subsystems, in state space and transfer matrix form:

$$\Sigma_i : \begin{array}{l} \dot{x}_i = A_i x_i + B_i u_i \\ y_i = C_i x_i + D_i u_i \end{array}; \qquad y_i(s) = G_i(s) u_i(s) \quad i = 1, 2 \tag{2.46}$$

Assuming appropriate dimensions in the input and output vectors, u_i, y_i, a global system may be arranged with joint state vector, $x = [x_1^T \; x_2^T]^T$, and its equation for each interconnection will be discussed.

Series connection. Also denoted by cascade connection. The input/output vectors to each subsystem in Figure 2.3(a) are:

$$y = y_2; \qquad u = u_1; \qquad u_2 = y_1$$

This results in the global state space system:

$$\dot{x} = \begin{pmatrix} A_1 & 0 \\ B_2 C_1 & A_2 \end{pmatrix} x + \begin{pmatrix} B_1 \\ B_2 D_1 \end{pmatrix} u \tag{2.47}$$

$$y = [D_2 C_1 \; C_2] x + D_2 D_1 u \tag{2.48}$$

and, in transfer matrix form:

$$y(s) = G_2(s) G_1(s) \, u(s) \tag{2.49}$$

Parallel connection. The interconnection in Figure 2.3(b) is represented by:

$$y = y_1 + y_2; \qquad u_1 = u_2 = u$$

This results in the global system:

$$\begin{array}{l} \dot{x} = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} + \begin{pmatrix} B_1 \\ B_2 \end{pmatrix} u \\ y = [C_1 \; C_2] x + (D_1 + D_2) u \end{array}; \qquad y(s) = (G_1(s) + G_2(s)) u(s) \tag{2.50}$$

Feedback connection. Figure 2.3(c). In this case, there is a loop and a summation point. Assuming an external input, v, null input/output coupling in the forward path ($D_1 = 0$) and negative feedback, the input/output vectors are:

$$y = y_1; \qquad u_1 = v - y_2; \qquad u_2 = y_1$$

This results in $y(s) = G_1(s)e(s) = G_1(s)(r - G_2(s)y)$, so the global system can be written as:

$$\dot{x} = \begin{pmatrix} A_1 - B_1 D_2 C_1 & -B_1 C_2 \\ B_2 C_1 & A_2 \end{pmatrix} + \begin{pmatrix} B_1 \\ 0 \end{pmatrix} v \tag{2.51}$$
$$y = (C_1 \ 0)x$$

$$y(s) = (I + G_1(s)G_2(s))^{-1} G_1(s) v(s) \tag{2.52}$$

In the last transfer matrix equation, the summation sign would be negative for positive feedback.

A system may be composed of a number of such interconnections, leading to a complex model, where also disturbances can be present. In these examples of interconnection, the ease of use of the transfer matrix as an operator is clearly illustrated.

Remark 2.12. In this and any other MIMO case, special care must be taken with block-diagram operations, as the matrix product is *not* commutative, contrary to the SISO case.

2.7.2 Generalised Interconnection

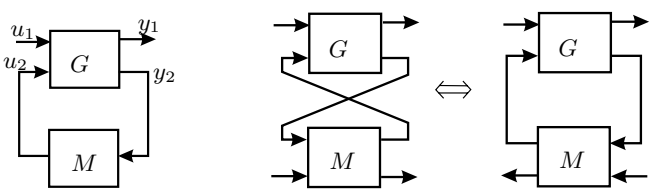

Figure 2.4. General interconnections

The interconnection between subsystems can be expressed in a general framework by just using two subsystems in a unique loop as in Figure 2.4 (left), where the subsystem in the feedback has a transfer matrix M, and the one in the upper subsystem is suitably partitioned into four blocks, G_{11}, G_{12}, G_{21} and G_{22}. In this framework, the equations are:

$$u_2 = My_2 = M(G_{21}u_1 + G_{22}u_2) \ \Rightarrow \ u_2 = (I - MG_{22})^{-1} MG_{21}u_1 \tag{2.53}$$
$$y_1 = G_{11}u_1 + G_{12}u_2 = \left(G_{11} + G_{12}(I - MG_{22})^{-1} MG_{21}\right) u_1 \tag{2.54}$$

Example 2.13. If $G = \begin{pmatrix} 0 & I \\ \Sigma_1 & 0 \end{pmatrix}$ the result is: $y_1 = M\Sigma_1 u_1$, *i.e.*, the series connection.

If $G = \begin{pmatrix} \Sigma_1 & I \\ I & 0 \end{pmatrix}$ the result is: $y_1 = (\Sigma_1 + M)u_1$, *i.e.*, the parallel connection.

If $G = \begin{pmatrix} I & -I \\ I & -I \end{pmatrix}$ Σ_1 the result is the feedback connection. Indeed, in this case, (2.54) has as TF matrix $\Sigma_1 - \Sigma_1(I + M\Sigma_1)^{-1}M\Sigma_1$. As:

$$(I + M\Sigma_1)^{-1}M\Sigma_1 + (I + M\Sigma_1)^{-1} = (I + M\Sigma_1)^{-1}(M\Sigma_1 + I) = I \qquad (2.55)$$

the output transfer matrix can be expressed as $\Sigma_1 - \Sigma_1(I - (I + M\Sigma_1)^{-1}) = \Sigma_1(I + M\Sigma_1)^{-1}$. Applying the push-through rule (B.5), Equation (2.52) is obtained.

In this way, by including in G not only the system equations but also the interconnection structure, many control problems of engineering significance can be cast as the block-diagram in Figure 2.4 (left). G is then called the *generalised plant*, one of its subsystems being the actual plant. Indeed, this formulation allows us to deal with open-loop, closed-loop and other set-ups in a unified way, including performance specifications, and also allowed a significant theoretical breakthrough in the 1980s (see Sections 4.5.3 and 7.4). This block-diagram and Equation (2.54), where the block-diagram has been simplified by ellimination of u_2 and y_2, are denoted as a *lower linear fractional transformation* (LFT), and the transfer matrix in (2.54) is usually represented in shorthand form as $\mathcal{F}_L(G, M)$.

The diagrams at the right of the referred figure present an even more general form of interconnection denoted as Redheffer star product, where each block has two input and two output vectors. It is easy to verify the correspondence to the previous models if appropriate elements of their transfer matrices are chosen. For instance, for

$$G = \begin{pmatrix} 0 & G_2 \\ I & 0 \end{pmatrix}; \quad M = \begin{pmatrix} G_1 & 0 \\ 0 & 0 \end{pmatrix}$$

the serial connection, (2.47), is also obtained. And similarly, the parallel and feedback connections can be represented. The LFT diagram is a particular case of the star product, if only $M_{11} \neq 0$, such as the M above.

The MATLAB® command `lft` allows us to compute transfer matrices of systems described in an LFT or star-product structure. In particular, the expression `sys=lft(P,K,nu,ny)` connects the first "nu" outputs of K to the last "nu" inputs of P, and the last "ny" outputs of P to the first "ny" inputs of K. The resulting system model, `sys`, maps the remaining inputs to the remaining outputs. If `nu` and `ny` are omitted, the system with lower input and output vector dimensions is assumed to be M in the LFT block-diagram.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `lft,feedback,connect,starp,series,parallel`.

2.8 Discretised Models

Discretised (sampled) signals. Although the physical variables involved in a process are mainly CT, in order to process them on a computer, to simulate the process behaviour or to design a digital controller, approximate DT models are required. In general, attached to a CT signal, a DT sequence can be defined by just taking the sampled values at some given time instants. For a regular sampling period, T, the elements of the sequence are $y_k = y(kT), \quad k = 0, 1, 2, \ldots$. If the unit time delay is expressed by the delay operator, z^{-1}, these *sampled-data* (SD) signals are represented by using the $\mathcal{Z}$-transform (Appendix A).

These sampled-data signals, under a fast enough sampling pattern, keep most of the information carried by the CT signal. Shannon sampling theorem [95] states that the sampling frequency must be twice the frequency up to which significant content is present in the CT signal.

DT process models. Similarly, a DT model of the CT process (or a *sampled-data* model) can be derived. It will approximate the process behaviour relating the involved variables. However, some warnings should be needed in this case:

- the attached DT model is an approximation of the CT process behaviour. Depending on the approximation criteria, the following situations may appear:
  - the CT model and the SD model have a similar time-response to a given input signal (impulse, step, ramp, ...),
  - the CT model and the SD model have a similar frequency-response in a range of frequencies,
  - the CT model and the SD model have a similar mathematical appearance (by substituting the time derivative by increments ratio, for instance).
- the intersampling behaviour may be rather different. Caution should be taken about the information lost from the input signal,
- the DT model parameters will change if the sampling period is changed.

The most usual and common discretisation is based on the use of a digital computer, leading to regular sampling characterised by a period T, together with a constant holding of the input during the same period. To get an SD-equivalent model of the process, the simplest discretisation approaches are based on the *approximation of the derivative operator*. There are many options to implement this approximation:

- **Euler.** For each derivative term, the following approximation is implemented:
$$\dot{x} \approx \frac{\Delta x}{\Delta t} \approx \frac{x_{k+1} - x_k}{T} \tag{2.56}$$
provided T is small enough. This approach can be also applied to non-linear systems. The equation (2.10) is transformed into:

$$\frac{dx(t)}{dt} = f(x(t), u(t), t) \rightarrow x_{k+1} = x_k + Tf(x_k, u_k, k) \tag{2.57}$$

The DT model will relate the DT signals. For this purpose, dealing with linear systems, the sequences are represented by their $\mathcal{Z}$-transform. To get the Euler-discretisation of a CT transfer matrix, the Laplace variable, s, is replaced by:

$$s = \frac{1 - z^{-1}}{T}$$

- **forward operator.** In a similar way, a CT transfer matrix can be approximated by replacing the Laplace variable, s, by:

$$s = \frac{z - 1}{T}$$

- **bilinear transformation.** Among the possible variations of the previous approximations, if the Laplace variable is replaced by:

$$s = \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}} \tag{2.58}$$

the transformed DT model will be stable if and only if the source CT model is so, which is indeed of interest. This is a useful discretisation approach, although it is only valid for linear systems[5].

In the next chapter, for LTI systems and based on the solution of the state equation, exact (step response) SD models will be also derived. Non-regular sampling may be considered, leading to more complex DT representations (Section 9.4).

2.9 Equivalence of Representations

All the representations of the same process should be coherent. If they relate to the same variables, they should be equivalent. Thus, given the state space model or the polynomial operator, the transfer matrix should be easily obtained. In fact, by applying the Laplace transform to Equations (2.17) and (2.19), assuming that the initial condition terms are null, the following equivalence is obtained:

$$sx(s) = Ax(s) + Bu(s)$$
$$y(s) = Cx(s) + Du(s)$$

[5] For computer simulation of non-linear systems, an approximate equivalent is the mid-point discretisation formula:

$$x_{k+1} = x_k + Tf(x_k + T/2f(x_k, u_k), \frac{u_k + u_{k+1}}{2})$$

$$y(s) = [C(sI - A)^{-1}B + D]u(s) = G(s)u(s)$$

Thus:

$$G(s) = [C(sI - A)^{-1}B + D] \tag{2.59}$$

Based on this equivalence, as

$$(sI - A)^{-1} = \frac{adj(sI - A)}{\det(sI - A)}$$

the poles of $G(s)$ are the eigenvalues of A (solution of the characteristic equation $\det(sI - A) = 0$).

An interesting representation for $(sI - A)^{-1}$ can be derived if this matrix is expanded in series $I + As + As^2 + \ldots$:

$$G(s) = D + CBs^{-1} + CABs^{-2} + CA^2Bs^{-3} + \ldots = \sum_{i=0}^{\infty} H_i s^{-i} \tag{2.60}$$

Also, for DT systems:

$$G(z) = D + C(zI - A)^{-1}B = D + CBz^{-1} + CABz^{-2} + \ldots = \sum_{i=0}^{\infty} H_i z^{-i} \tag{2.61}$$

H_i are denoted as *Haenkel parameters* or coefficients. In this DT representation, H_i is the impulse response at time i, because $y(z) = G(z)$ if $u(z) = 1$.

The reverse transformation is not so easy. First, the state representation is not unique, thus there will be many internal representations for the same input/output model. Second, some of these representations may be overdimensioned and so useless, spurious, internal variables will be included. One option would be to use the concept of memory or accumulation attached to the state variables and try to define them in this way. Another alternative is to attach a state variable to each process pole.

To properly deal with this issue, some knowledge about the process structure and properties should be available. This is the subject of the next chapter (Section 3.7.5), and thus the reverse transformation is postponed, with only a very simple example being presented now.

Example 2.14. Let us consider the 2×2 transfer matrix of Example 2.10. Based on the partial fraction decomposition, let us assign the state variable, x_1, to the pole at the origin and the variables x_2 and x_3 to those related to the pole at -1. A possible state space representation would be:

$$\dot{x}_1 = -u_1 + u_2; \quad \dot{x}_2 = -x_2 + u_1; \quad \dot{x}_3 = -x_3 - u_2$$

$$y_1 = x_1 + x_2; \quad y_2 = x_1 + x_3$$

That is:

$$A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}; \quad B = \begin{pmatrix} -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}; \quad C = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

Also, the transfer matrix may be extended to become:

$$G(s) = \begin{pmatrix} 0 & 1 \\ -1 & 2 \end{pmatrix} s^{-1} + \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} s^{-2} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} s^{-3} + \dots$$

From (2.35), the relationship between the transfer matrix and the polynomial representations is clear:

$$G(s) = D^{-1}(s)N(s)$$

Again, the reverse is not unique and many options are possible.

2.10 Disturbance Models

Disturbances are non-manipulated inputs to the process. They are generated elsewhere but affect the process behaviour and are signals coming from another processes.

2.10.1 Deterministic Signals

If the structure of these processes, denoted as *generators*, is known, the disturbances are said to be *deterministic*. For instance, polynomial disturbances can be considered as being generated by a chain of integrators. The initial condition of each integrator will determine one of the parameters of the polynomial signal. For instance:

- constant signals, $y = a_1$, are generated by:

$$\dot{x}_1 = 0; \quad x_1(0) = a_1; \quad y(t) = x_1(t)$$

- ramp signals, such as $y(t) = a_1 + a_2 t$, by:

$$\dot{x}_1 = x_2; \quad \dot{x}_2 = 0; \quad x_1(0) = a_1; \quad x_2(0) = a_2; \quad y(t) = x_1(t)$$

and, in general, *polynomial* signals, such as

$$y(t) = \sum_{i=1}^{i=n} \frac{a_i}{(i-1)!} t^{i-1}$$

are generated by:

$$\dot{x}_1 = x_2;\ \dot{x}_2 = x_3;\ \dots;\ \dot{x}_n = 0$$
$$y = x_1$$

where $a_i = x_i(0)$, and $n!$ denotes the factorial of n.

Discrete disturbances. If the generators are built up with accumulators instead of integrators, DT signals can be generated. For instance, with:

$$x_1(k+1) = x_1(k) + x_2(k); \quad x_2(k+1) = x_2(k) + x_3(k); \quad \dots$$
$$x_n(k+1) = x_n(k); \qquad y(k) = x_1(k)$$

the output is given by:

$$y(k) = x_1(0) + kx_2(0) + \frac{k(k-1)}{2}x_3(0) + \ldots = \sum_{i=1}^{n} \begin{pmatrix} k \\ i-1 \end{pmatrix} x_i(0)$$

Similar expressions can be obtained for exponential or sinusoidal signals.

The most common deterministic signals are summarised in Table 2.2. Disturbance randomness can be inserted if random impulse inputs, ψ, are assumed at time t.

Table 2.2. Generation of deterministic disturbances

	CT	DT
Steps	$d(t) \stackrel{\text{def}}{=} A$ $\dot{d} = \psi$	$d_k \stackrel{\text{def}}{=} A$ $d_{k+1} = d_k + \psi_k$
Ramps	$d \stackrel{\text{def}}{=} at + b$ $\dot{x} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \psi$ $d = [1 \quad 0]x$	$d_k \stackrel{\text{def}}{=} ak + b$ $x_{k+1} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x_k + \psi_k$ $d = [1 \quad 0]x$
Sinusoidal	$d \stackrel{\text{def}}{=} A\sin(\omega t + B)$ $\dot{x} = \begin{pmatrix} 0 & 1 \\ -\omega^2 & 0 \end{pmatrix} x + \psi$ $d = [1 \quad 0]x$	$d_k \stackrel{\text{def}}{=} A\sin(\omega Tk + B)$ $x_{k+1} = \begin{pmatrix} \cos\omega T & \sin\omega T \\ -\sin\omega T & \cos\omega T \end{pmatrix} x_k + \psi_k$ $d = [1 \quad 0]x$

Example 2.15. It is easy to show that, with the system defined by:

$$\dot{x} = a.x; \; x(0) = x_0; \quad y = x$$

the signal $y(t) = x_0 e^{at}$ is generated. Similarly, by:

$$\dot{x}_1 = x_2; \; \dot{x}_2 = w^2 x_1; \; x_1(0) = A\sin(\varphi); \; x_2(0) = A\cos(\varphi)$$
$$y = x_1$$

the signal $y(t) = A\sin(wt + \varphi)$ is generated.

2.10.2 Randomness in the Signals

It frequently happens that the model of the process generating the external signals is very complex or unknown. In this case, it is easier to characterise

the signals by their stochastic properties. They can also be considered as the output of stochastic generators or as the filtered response of an original stochastic signal. These disturbances may appear at the output (as measurement noise), at the input or at the state level. Let us briefly describe these types of variables:

- **measurement noise**. The output of the sensor system is affected by a number of random variables from which no information is available, such as device drifts or interference from other devices. The main goal would be to *filter*, reduce or cancel the noise, getting a *clean* measurement to feed the monitoring or control system,
- **stochastic inputs**. They affect the process behaviour as external variables and they are unavoidable. If we consider the positioning of an antenna, the force of the wind will present such a characteristic,
- **internal noise**. These are the most complicated disturbances and, in some cases, they can be modelled by assuming stochastic parameters in the process models. A way of dealing with this situation is by considering uncertainties in the model or in the variables. An introduction to the available tools is presented in Chapter 8.

In a general sense, the control goals under stochastic disturbances will be: to filter the measurement noise or output disturbances, to attenuate or cancel the input disturbances, probably by measuring them if it is possible, and to assure some performances under process noise, due to uncertainties in models or disturbances.

In order to characterise the stochastic signals, some basic statistic concepts and measurements about random variables should be remembered. Some of them are random variables, distribution and density functions, mean, variance, covariance, correlation, statistical independence and linear regression, among others. A short review of these concepts is included in Appendix E, where special attention is paid to the multivariable case.

2.10.3 Discrete Stochastic Processes

A signal, of which the value at any time instant is a random variable, is called a *stochastic process*. A discrete stochastic process is a *sequence* of random variables and can also be obtained as a result of sampling a CT stochastic process.

In the following, only discrete stochastic processes are considered. Their main properties are related to those of the random variables in the sequence, as well as their interaction. Let us review the most common (simplified) models of stochastic processes.

White noise. White noise (WN) is the simplest stochastic process, $\{\varepsilon_k\}$. The discrete random variables in the sequence are zero-mean ($E(\varepsilon_k) = 0$), and independent of each other. If their variance is $E(\varepsilon_k^2) = \sigma_k^2$, the *covariance* between samples j and k is:

$$\sigma_{jk} = \begin{cases} \sigma_k^2 & j = k \\ 0 & j \neq k \end{cases} \tag{2.62}$$

Due to the independence among variables, the knowledge of the current or past values in a sequence is useless for determining the future values. In this sense, WN cannot be compensated for or predicted.

If there is some kind of relationship among the variables, that is, if the random variables in the sequence are not independent, some additional information is available and the "noise" treatment may be easier. The rest of the stochastic processes are denoted as *coloured noise*. Some of them are easily modelled.

Random walk (drift). This can be considered as an accumulated white noise:

$$v_{k+1} = v_k + \varepsilon_k; \qquad \varepsilon_k \text{ is WN} \tag{2.63}$$

Thus, the difference between two consecutive variables is WN. Their covariance can be expressed by:

$$\sigma^2_{k(k+1)} = E(v_k v_{k+1}) = E(v_k(v_k + \varepsilon_k)) = E(v_k^2) + E(v_k \varepsilon_k) = \sigma_k^2$$

Coloured noise (general model). A general model of coloured noise can be represented as the output of a dynamic (DT) process:

$$\begin{aligned} x_{k+1} &= Ax_k + \varepsilon_k \\ v_k &= Cx_k \end{aligned} \tag{2.64}$$

where ε_k is a WN. Thus, based on the past and current values of the $\{v(k)\}$ sequence, some information about the future values can be estimated.

Similarly, if the noise is disturbing a DT process which has, in addition, manipulated inputs, the equations equivalent to (2.21) and (2.22) would be:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + B_d v_k \\ y_k &= Cx_k + Du_k + D_d w_k \end{aligned} \tag{2.65}$$

where v_k and w_k are white noises with covariance matrices V and W respectively. Usually, $D = 0$ and $D_d = I$, w_k being the measurement noise and v_k the process noise.

Input/output models

Other than the internal representation above, a monovariable stochastic process may be represented by a difference equation or, alternatively, using the general concept of the $\mathcal{Z}$-transform of a sequence, it can be also expressed by an input/output model such as:

$$v(z) = G_n(z)\varepsilon(z) \qquad \varepsilon_k \text{ is WN} \tag{2.66}$$

It is worth mentioning some typical representations that are rather common in literature [30, 84]:

- Moving-Average (MA). The signal is obtained as a weighted sum of current and past inputs:

$$y(k) = \sum_{i=0}^{n} b_i \varepsilon(k-i)$$

- Auto-Regressive (AR). The weighted sum of the signal is a random noise:

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \varepsilon(k)$$

- Auto-Regressive-Moving-Average (ARMA). This is a combination of the two above,

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \sum_{i=0}^{n} b_i \varepsilon(k-i)$$

- External-Auto-Regressive-Moving-Average (ARMAX). In this case, there is also a manipulated input, $u(k)$, leading to:

$$y(k) + \sum_{i=1}^{n} a_i y(k-i) = \sum_{i=0}^{n} b_i u(k-i) + \sum_{i=0}^{n} c_i \varepsilon(k-i) \tag{2.67}$$

However, the state space representation will be pursued in this book for disturbance rejection control designs, as all the above models can be expressed in a form similar to (2.64) or (2.65).

2.11 Key Issues in Modelling

As has been emphasised in this chapter, the model of a process is a partial representation of its behaviour. In this sense, the model we are interested in is a model suitable for designing the control system. Thus, the selection of the model and the modelling approach depends on the final goal:

- the details of the model are determined by the control goals. Static, low-frequency range, autonomous, DT or hybrid models are examples of options,
- the variables and relationships selected to build the model will depend on the effects we are interested in in our study. Some variables can be deleted or treated as disturbances if they are not so relevant, or they can become crucial if their particular effect has to be modelled,
- the control design methodology will recommend, or even determine, the type of representation,
- the knowledge and/or availability of the process to get actual data will allow for theoretical or experimental modelling techniques,
- the accuracy and complexity of the model will depend on the control requirements.

How do we obtain a model? As a summary, two main approaches can be followed:

- **first-principle (internal) model.** Leading to (detailed) non-linear models, with physical insight into the variables and equations, and a number of parameters to be determined. Techniques of model reduction may need to be applied,
- **experimental modelling** (Appendix A.4). By comparing the behaviour of the plant to that of some predefined models and structures, the models' parameters can be estimated, leading to (simple) approximate linearised models, being usually input/output representations. This approach is suitable for modelling disturbances if historical records of them are available.

According to the model purpose and the analysis of its properties, a change in the representation frame, its complexity, its range of validity or the number of involved variables may be suggested. Thus, the modelling phase should be revisited in the control design process, to enhance the available model to better fulfill the control requirements, the final goal in our study.

2.12 Case Study: The Paper Machine Headbox

The purpose of the headbox is to deliver a uniform and stable jet velocity profile in both cross and machine directions to form the paper sheet, Figure 2.5. The stock flows into the headbox chamber controlled by a valve. The top

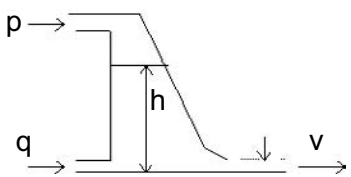

Figure 2.5. Headbox simple schema

of this chamber is filled with compressed air to dampen pressure pulsations. The airflow is also controlled by an input valve. The stock is homogenised in this chamber and it flows through the slice channel to the wire.

2.12.1 Simplified Models

First-principle. A very simple first-principle model can be derived by mass balances.

Stock balance in the headbox:

$$A(h)\frac{dh(t)}{dt} = q(t) - S(t)v(t)$$

where h is the stock level, in m, A is the headbox section, in m^2, q is the stock inflow, in kg/s, S is the slice lip section, in m^2, and v is the stock exit speed, in m/s.

The stock exit speed due to the total "head pressure" H, in m, is: $v(t) = \sqrt{2gH(t)}$; $\quad H(t) = h(t) + p(t)$

The air compression is assumed to be isothermal, thus $p(t)V_a(t) = const$, where p is the air pressure, in equivalent stock height (m) and V_a is the volume of air in m^3.

Air pressure variations are due to air net inlet flow and volume reduction, thus:

$$\frac{dp(t)}{dt} = \lambda(P_a(t) - p(t)) - \kappa\frac{dh}{dt}$$

where λ is the chamber dynamics inverse time constant, in s^{-1}, P_a is the source air pressure, in equivalent stock height, in m, and κ is a compression adimensional constant.

The slice lip exit area is manipulated by an external motor. In this simplified model, it is assumed constant or slowly-varying.

In this simple model, the head and pressure can be chosen as state variables, whereas the manipulated variables could be the stock inlet flow and the source pressure, leading to the state model:

$$\begin{aligned}\dot{h} &= -\frac{S}{A}\sqrt{2g(h+p)} + \frac{1}{A}q\\ \dot{p} &= \kappa\frac{S}{A}\sqrt{2g(h+p)} - \lambda p - \kappa\frac{1}{A}q + \lambda P_a\end{aligned}$$

Experimental. By applying steps at the inputs at an operating point, the elements of an approximate transfer matrix may be estimated. Additional transportation flow delays between the stock valve and the headbox chamber input may be realised. This can be done on the real plant or on a simulated model. This could result in a transfer matrix model obtained as in Example 2.9:

$$\begin{bmatrix} h \\ v \end{bmatrix} = \begin{bmatrix} \frac{e^{-s}}{1+s} & \frac{-1}{1+0.5s} \\ \frac{e^{-s}}{(1+s)(1+2s)} & \frac{0.2+s}{(1+0.5s)(1+2s)} \end{bmatrix} \begin{bmatrix} q \\ P_a \end{bmatrix}$$

Discrete model

From the first-principle non-linear state space representation, a simple DT model may be obtained by the Euler approximation of the two state variable derivatives.

Alternatively, from the experimental model above, the following discretisation approximations can be obtained:

1) Discrete transfer matrix. To get a discrete model, it could be easy to discretise each element of the transfer matrix, assuming a zero-order hold (ZOH) device in the input and a regular sampling period, T, at the output. If this period is assumed to be $T = 0.5$ s, the time delay of 1 s, will be equivalent to two delay units, z^{-1}. That is, $\mathcal{Z}[e^{-s}] = z^{-2}$.

Taking care of the delays, the MATLAB® command `Gd=c2d(G,.5,'zoh')` returns the DT transfer matrix:

$$G(z) = \begin{bmatrix} \dfrac{0.3935}{z^2(z-0.6065)} & \dfrac{-0.6321}{z-0.3679} \\ \dfrac{0.0489z+0.0381}{z^2(z-0.6065)(z-0.7788)} & \dfrac{0.2908z-0.2628}{(z-0.3679)(z-0.7788)} \end{bmatrix} \tag{2.68}$$

2) State representation based on physical variables. From this DT transfer matrix, grouping the common poles by rows or columns, the block-diagram in Figure 2.6 can be drawn.

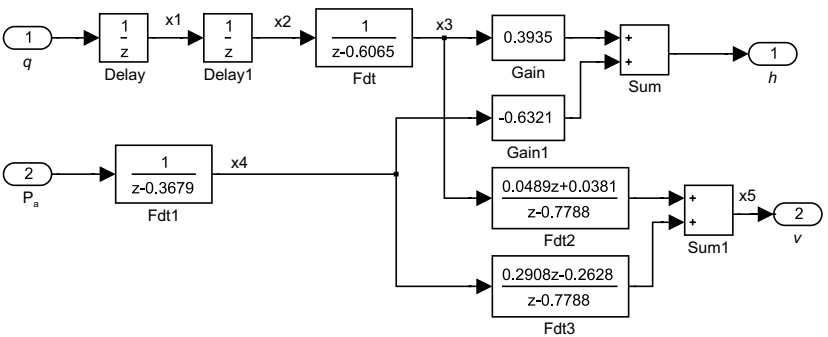

Figure 2.6. DT Block-diagram of the headbox (where $F \equiv q$)

Then, a state variable can be directly assigned to each first-order block (`Fdt2` and `Fdt3` share the same one), leading to the internal representation:

$$x_1(z) = \frac{1}{z}q(z) \quad \rightarrow \quad x_1(k+1) = q(k)$$

$$x_2(z) = \frac{1}{z}x_1(z) \quad \rightarrow \quad x_2(k+1) = x_1(k)$$

$$x_3(z) = \frac{1}{z-0.6065}x_2(z) \quad \rightarrow \quad x_3(k+1) = x_2(k) + 0.6065x_3(k)$$

$$x_4(z) = \frac{1}{z-0.3679}P_a(z) \quad \rightarrow \quad x_4(k+1) = 0.3679x_4(k) + P_a(k)$$

$$x_5(z) = \frac{0.0489z+0.0381}{z-0.7788}x_3(z) + \frac{0.2908z-0.2628}{z-0.7788}x_4(z)$$

$$(z-0.7788)\,x_5(z) = (0.0489z+0.0381)\,x_3(z) + (0.2908z-0.2628)\,x_4(z)$$

$$x_5(k+1) = 0.7788x_5(k) + 0.0489x_3(k+1) + 0.0381x_3(k)$$

$$+0.2908x_4(k+1) - 0.2628x_4(k)$$

and rearranging the last equation:

$$\begin{aligned} x_5(k+1) &= 0.7788x_5(k) + 0.0489\,[x_2(k) + 0.6065x_3(k)] + 0.0381x_3(k) \\ &\quad +0.2908\,[0.3679x_4(k) + P_a(k)] - 0.2628x_4(k) \\ x_5(k+1) &= 0.7788x_5(k) + 0.0489x_2(k) + 0.0678x_3(k) - 0.1559x_4(k) + 0.2908P_a(k) \end{aligned}$$

The output equation would be:

$$\begin{aligned} h(k) &= 0.3935x_3(k) - 0.6321x_4(k) \\ v(k) &= x_5(k) \end{aligned}$$

Altogether, the state representation is:

$$\begin{cases} x(k+1) = A_f x(k) + B_f u(k) \\ y(k) = C_f x(k) + D_f u(k) \end{cases}$$

$$A_f = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0.6065 & 0 & 0 \\ 0 & 0 & 0 & 0.3679 & 0 \\ 0 & 0.0489 & 0.0678 & -0.1559 & 0.7788 \end{bmatrix}; \quad B_f = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0.2908 \end{bmatrix}; \tag{2.69}$$

$$C_f = \begin{bmatrix} 0 & 0 & 0.3935 & -0.6321 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}; \quad D_f = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

2.12.2 Elaborated Models

An in-depth model has been developed at the Pulp and Paper Centre (University of British Columbia, Vancouver, Canada), reported in [127]. To express the dynamics of the upper chamber, the interconnected liquid and gas-flow systems are considered. A mass-balance equation for the stock and the stock flow out of the headbox can be obtained by applying:

$$\frac{dh}{dt} = q - C_s S\sqrt{2gh_j} - 2S_o\sqrt{2\frac{(p-p_0)}{\rho_w}}A \tag{2.70}$$

where h_j is the head at the slice lip, in m, C_s is the valve-sizing coefficient, S_o is the area of overflow valve opening, in m^2, p_0 is the atmospheric pressure, in m, and ρ_w is the density of stock, in Kg/m^{-3}.

A mass-balance equation for the air using the pressure density relationship gives the following equation in terms of pressure:

$$\frac{dp}{dt} = \frac{kp_0}{\rho_0}\left(\frac{p}{p_0}\right)^{\frac{(k-1)}{k}}\left[\frac{q_i - q_e}{V_a} + \frac{\rho_0 A(\frac{p}{p_0})^{\frac{1}{k}}\frac{dh}{dt}}{V_a}\right] \tag{2.71}$$

where ρ_0 is the air density, in kg/m^{-3}, κ is the specific heat ratio, q_i and q_e are the air inflow into the upper chamber and outflow through the bleed valve, in kg/s.

To write the equation which governs the rate of change of the head at the slice lip, h_j, the headbox inlet head, h_i, and the friction head loss, h_f, are considered. The following equation is obtained:

$$\frac{dh_j}{dt} = \frac{\sqrt{2gh_j}}{L}\left[H - h_j - h_f\right] \tag{2.72}$$

where L is the constant of the slice channel, in meters, and h_f is the friction head loss.

The stock level, h, in the headbox, the airpad pressure, p, and the head at the slice lip, h_j, are chosen as the state variables.

The stock inflow, q, the gas flow, q_e, defining the net air inflow and the slice lip area, S, are treated as inputs.

The headbox dynamics can thus be represented by a third-order non-linear dynamical system. The model is characterised by a number of construction-dependent parameters (A, V_a, S_o, h_f and C_{ds}) as well as some fundamental physical parameters (k, p_0, ρ_0, ρ_w and g). The values of q, $q_i - q_e$ or S_j depend upon the operating point. Knowing the design details of the headbox and the operating point, all these variables could be calculated quite easily.

The three equations are highly coupled. Equation (2.70) shows that if the input stock flowrate increases, the liquid level increases. This in turn increases the airpad pressure and the total head. This increases the outgoing stock flow. There is thus a self-regulating effect in the system.

Equation (2.71) shows that the airpad pressure will return to its original value when the equilibrium point is reached. Any deviation in the level will thus be contributing to the total head.

Changing the air output area changes the air outflow, affecting both the level and the pressure inside the airpad. An increase in the slice opening will produce a decrease in the level, airpad pressure and the total head.

The most important control problem for a headbox is to maintain constant jet velocity and to have a good dynamic behaviour when changing the grades. We must consider as disturbances the external pressure of air and flow inlet, as well as the consistency of the stock or physical characteristics of the flow resistance.

In spite of the model complexity due to non-linearities, the model order is reduced to 3, which is even lower than the experimental one obtained in the last chapter.

Linearisation. The previous non-linear model is useful for simulating and studying the dynamic behaviour of the headbox. To design control strategies, it is always convenient to have linearised models which approximate the non-linear dynamics for small disturbances around a steady-state point. In [127], the following linearised model is obtained for some given parameters and operating conditions, using the MATLAB® command `linmod`.

$$A = \begin{bmatrix} 0 & -0.00083 & -0.03 \\ 0 & -0.08849 & -1.041610 \\ 9.1371 & 9.1371 & -8.883063 \end{bmatrix}; \quad C = \begin{bmatrix} 1\ 0\ 0 \\ 0\ 1\ 0 \\ 0\ 0\ 1 \end{bmatrix} \tag{2.73}$$

$$B = \begin{bmatrix} 0.47597 & 0 & -0.048587 \\ 14.03999 & -0.00162 & -1.433194 \\ 0 & 0 & -57.9134 \end{bmatrix}; \quad D = \begin{bmatrix} 0\ 0\ 0 \\ 0\ 0\ 0 \\ 0\ 0\ 0 \end{bmatrix}$$

The eigenvalues of the system matrix A are $(-0.0005, -10.3254, -0.9950)$. Thus, the system has a very slow mode with a time constant of about 512 s and two fast modes with time constants of less than a second.

3

Linear Systems Analysis

This chapter covers the study of the dynamic behaviour of linear models. Differential equations and operations with the Laplace transform and its inverse are assumed to be the subject of introductory courses ([78, 49, 94] and basic outline in Appendix A). State space approach will be presented in some detail, with emphasis placed on obtaining reduced order models that are suitable for control purpose design. Important concepts in theory and practice (such as gain, poles, ...) will be presented as well, paying attention to the issues that are mainly related to multivariable systems, such as the system structure and the interactions.

3.1 Introduction

Most of the concepts and ideas developed in this book are based on process models. If the model is accurate enough, the analysis of its properties will provide an estimation of those from the actual process. If analysis is carried out previous to the controller design, it is denoted as *open-loop* analysis. The study of specific properties of a plant–controller combination is denoted as *closed-loop* analysis.

By this analysis a number of characteristics should be determined:

- **stability.** It is the basic feature of any controlled system. Related to this concept are issues such as degree of stability and, in a general way, the characterisation of the dynamic behaviour,
- **accuracy.** The system gains in steady-state, instantaneously, and as a function of the frequency. One major issue in MIMO systems is the concept of *directionality*: the gains depend on the input direction, that is, in the combination of inputs under study,
- **structure.** Interactions/independence among sets of variables, subsystem division,
- **robustness.** Availability for maintaining the above properties under changes or uncertainties in model parameters or disturbances.

All these properties are also present in the analysis of SISO systems, but the richness and complexity are by far larger in MIMO systems. In fact, provided that there is more than one manipulated variable, issues such as the selection of inputs and their purpose, or the study of interactions, cancellations or complementary effects, will condition the success of a control system.

Most of the concepts will be presented using the state space representation, introducing their interpretability and use with other representations.

3.2 Linear System Time-response

Continuous-time systems. Given a CT linear system, as defined in (2.17) (2.19):

$$\dot{x}(t) = Ax(t) + Bu(t); \qquad y(t) = Cx(t) + Du(t) \tag{3.1}$$

with initial conditions, $x(t_0) = x_0$, the state and the output may be computed at any time, $t \geq 0$, by solving this differential equation. The solution of a scalar first-order differential equation, $\dot{x}(t) = ax(t) + bu(t)$, is given by:

$$x(t) = e^{a(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{a(t-\tau)} b\, u(\tau) d\tau$$

where the first term is the solution of the homogeneous equation, $\dot{x}(t) = ax(t)$, and the second one corresponds to a particular solution for the input $u(t)$. In the same way, for the first-order differential state vector equation in (3.1), it is (see Appendix B.5):

$$x(t) = e^{A(t-t_0)}x(t_0) + \int_{t_0}^{t} e^{A(t-\tau)} Bu(\tau) d\tau; \tag{3.2}$$

$$y(t) = Ce^{A(t-t_0)}x(t_0) + C\int_{t_0}^{t} e^{A(t-\tau)} Bu(\tau) d\tau + Du(t) \tag{3.3}$$

Autonomous (free)-response. If the initial state is $x(0) = x_0$ and the input is null, $u(t) = 0$; $\forall t$, the state and output response is:

$$x(t) = e^{At}x_0 = \phi(t)x_0; \quad y(t) = C\phi(t)x_0 \tag{3.4}$$

where $\phi(t) = e^{At}$ is the *transition matrix*. Observe that by applying the Laplace transform to the state equation, with null input, it yields:

$$sx(s) - x_0 = Ax(s) \longrightarrow x(s) = (sI - A)^{-1}x_0$$

The Laplace transform of the transition matrix, $\phi(t)$, is denoted as the *resolvent matrix*, $\phi(s) = (sI - A)^{-1}$.

Forced response. If the initial condition term vanishes (the system is at an equilibrium point), the forced response due to a specific input is obtained:

$$x(t) = \int_{t_0}^{t} e^{A(t-\tau)} B u(\tau)\, d\tau; \tag{3.5}$$

$$y(t) = C \int_{t_0}^{t} e^{A(t-\tau)} B u(\tau)\, d\tau + D u(t) \tag{3.6}$$

Impulse response. Being at equilibrium, if input u_i is a unitary impulse, $\delta(t)$, the response is:

$$y(t) = C e^{At} b_i + D\delta(t)$$

where b_i denotes the ith column of B. If $D = 0$,

$$G(t) = C e^{At} B =$$

is denoted as the *impulse-response matrix*. The forced output may be expressed by a generalisation of the convolution formula (A.15):

$$y(t) = G(t) * u(t) = \int_{t_0}^{t} G(t-\tau) u(\tau) d\tau)$$

For details on the above issues, the reader is referred to [11].

Discrete-time systems. For DT systems

$$x_{k+1} = A x_k + B u_k \tag{3.7}$$

$$y_k = C x_k + D u_k \tag{3.8}$$

the solution of the state equation yields:

$$\begin{aligned} x_1 &= A x_0 + B u_0 \\ x_2 = A x_1 + B u_1 &= A^2 x_0 + AB u_0 + B u_1 \\ x_3 &= A^3 x_0 + A^2 B u_0 + AB u_1 + B u_2 \\ &\dots \end{aligned}$$

$$x_k = A^k x_0 + \sum_{j=0}^{k-1} A^{k-j-1} B u_j \tag{3.9}$$

$$y_k = C x_k + D u_k \tag{3.10}$$

From the above expression, the output can be also expressed by a discrete convolution:

$$y_k = \sum_{j=0}^{j=k} H(k-j) u_j \tag{3.11}$$

$$y_k = \sum_{j=0}^{j=k-1} C A^{k-j} B u_j + D u_k \tag{3.12}$$

where the Haenkel parameters are as defined in Equation (2.61):

$$H(j) = CA^jB \qquad j \geq 1 \tag{3.13}$$

$$H(0) = D \tag{3.14}$$

3.3 Stability Conditions

A general definition of stability should be applicable to any dynamic system and it would appear as: *a system is stable if it presents bounded outputs for bounded inputs.* Stability as such, in a general non-linear case, may be cumbersome to assess. In general, the stability of a (non-linear) process is studied around an equilibrium point or a nominal trajectory and, in this case, the process behaviour can be most of the time approximated by a linear model. This is called *local stability*, applying the term of *global stability* if this property is analysed for any input or initial condition of the process. In-depth analysis can be consulted in [74].

In the case of linear systems, stability is a more concrete property that can be unequivocally characterised.

Continuous-time. Matrix exponentials determine the time-response of CT systems. The reader is referred to Appendix B.5 for details. Particularly, if the state representation is in diagonalised canonical form ($A = \Lambda = diag\{\lambda_i(A)\}$), the transition matrix will also be diagonal:

$$e^{\Lambda t} = diag\{e^{\lambda_i(A)t}\} \tag{3.15}$$

and (3.2), as well as the system output, will be composed of a sum of these diagonal terms, denoted as *modes*. Thus, the system acts as a *generator* of signals represented by the modes.

As argued in (B.6), page 288, the exponential terms arising in the solution of (3.1) are the eigenvalues of A. For each exponential, the usual analysis in Section A.2.1 can be carried out regarding stability and approximate settling time.

As a conclusion, the *stability condition* for a CT linear system, such as (3.1), is:

$$Re\{\lambda_i(A)\} < 0, \qquad \forall i \tag{3.16}$$

where $Re\{\cdot\}$ stands for the real part.

> *A CT linear system is stable if there are not eigenvalues of the system matrix A (poles of the transfer matrix) in the right-half-plane (RHP) of the complex plant.*

Discrete-Time. With a similar reasoning, the stability condition for a DT linear system is:

$$|\lambda_i(A)| < 1, \qquad \forall i \tag{3.17}$$

A DT linear system is stable if there are not eigenvalues of the system matrix A (poles of the transfer matrix) outside the unit circle in the complex plane.

3.3.1 Relative Degree of Stability

The stability condition, (3.16), allows the comparison of the relative degree of stability among CT systems. If

$$Re\{\lambda_i(A)\} < -\alpha < 0, \qquad \forall i \tag{3.18}$$

then any state has a trajectory bounded by $|x_i(t)| < Me^{-\alpha t}$ for some constant M or, equivalently

$$\lim_{t\to\infty} e^{+\alpha t} e^{A(t-t_0)} x(t_0) = 0$$

This provides a measure for estimating the settling time. Nevertheless, for the output signal, matrix C may hide some modes (see Section 3.7).

A similar relative degree of stability can be defined for DT systems.

3.4 Discretisation

Although CT models better approximate the behaviour of actual systems, their control is usually implemented in a digital system, so the overall system is hybrid. To simplify its treatment, there are two common approaches:

1. Designing a CT controller, and trying to find an "equivalent" DT controller
2. Assuming a DT "equivalent" model of the process, and designing a DT controller.

This equivalence can be established using different criteria:

- matching the response to an equivalent input (step, impulse, ramp, ...),
- keeping a similar structure: substituting the derivative by a DT approximation (Euler, bilinear, ...), as seen in Section 2.8. This is the natural approach to discretise *polynomial representations*,
- preserving some properties, like stability (matching of poles and zeros).

In a control environment, the output is usually sampled periodically and, with the same frequency, the input is updated through a hold device (see Chapter 9 for implementation issues).

In a first approach, discretisation strategy will be based on the following assumptions[1]:

- assume synchronous and instantaneous sampling/updating processes, at a regular sampling period, T,
- replace any CT signal, $y(t)$, by its sequence, $y(kT)$. Usually, the $\mathcal{Z}$-transform of this DT signal, $y(z)$, is handled,
- assume the process inputs, $u(t)$, to be constant between sampling times (ZOH device). That is, $u(t) = u(kT); \forall\, t\ kT \leq t < (k+1)T$. Thus, the input is a sequence of steps.

Following this strategy for a SISO with transfer function, $g(s)$, the DT equivalent transfer function is obtained [21] by computing:

$$g(z) = (1 - z^{-1})\mathcal{Z}_T(\mathcal{L}^{-1}(\frac{g(s)}{s})) \tag{3.19}$$

Thus, the samples every T seconds of the step response of the CT system will match that of the DT system.

Discretisation of transfer matrices. Each element of the matrix can be discretised following the above approach. The result will be a DT transfer matrix, $G(z)$, matching the input step-response of the CT system. For each element $g_{ij}(s)$ of the matrix, (3.19) is applied, obtaining the corresponding $g_{ij}(z)$.

The MATLAB® command `c2d`, with arguments the CT transfer matrix, the sampling period and the discretisation method, gives as a result a DT transfer matrix.

Example 3.1. The DT model of the CSTR, Example 2.1, can be easily obtained, using:

```
GD=c2d(Gp,.01,'zoh');zpk(GD)
   TF from input 1 to output...        from input 2 to output...
    0.029249 (z-0.1489)(z-0.5626)        0.00020763 (z+2.16)(z+0.1357)
#1:------------------------------  #1: ------------------------------
   (z-0.1479)(z-0.5807)(z-0.9819)      (z-0.1479) (z-0.5807)(z-0.9819)

     -0.23504 (z-0.1044)(z-1.007)       -0.20653 (z+0.4445) (z-0.9831)
#2:------------------------------ #2: ------------------------------
   (z-0.1479)(z-0.5807)(z-0.9819)       (z-0.1479)(z-0.5807)(z-0.9819)

     0.13503 (z+0.4411) (z-1.007)        -1.5969 (z-0.745) (z-0.981)
#3: ------------------------------  #3: ------------------------------
   (z-0.1479)(z-0.5807)(z-0.9819)       (z-0.1479)(z-0.5807)(z-0.9819)
```

[1] The use of other sampling strategies, hold devices or the analysis of the inter-sampling behaviour of the CT signals is beyond the primary goal of this book. In [1, 2], a detailed exposition of these topics is presented. Section 9.4 outlines control under dual-rate sampling.

where a $T = 0.01\,s$ sampling period is selected and a step-response-equivalent DT model is defined by 'zoh'. Note the different pole position if a different sampling period (for instance $T = 0.1\,s$) is selected.

Discretisation of State Space Representations

Applying (3.2) to a sampling period, the hold device behaviour involves keeping constant the input in the inter-sampling interval:

$$x((k+1)T) = e^{AT}x(kT) + \int_{kT}^{(k+1)T} e^{A((k+1)T)-\tau)}d\tau Bu(kT) \tag{3.20}$$

that is:

$$x_{k+1} = \bar{A}x_k + \bar{B}u_k \tag{3.21}$$

where it is clear that these matrices can be obtained by:

$$\bar{A} = e^{AT}; \qquad \bar{B} = \int_0^T e^{A\tau}\, d\tau B \tag{3.22}$$

An alternative approach can be followed to compute these matrices, avoiding the integral involved in $\bar{B}$.

Consider an augmented "state" vector, including the equation $\dot{u} = 0$:

$$x_e = \begin{pmatrix} x \\ u \end{pmatrix}; \qquad \dot{x}_e = \begin{pmatrix} A & B \\ 0 & 0 \end{pmatrix}\begin{pmatrix} x \\ u \end{pmatrix} = A_e x_e \tag{3.23}$$

the intersample response can be interpreted as a "free response" of the augmented system. By (3.4), for $kT \le t < (k+1)T$:

$$x_e(t) = \begin{pmatrix} x(t) \\ u(t) \end{pmatrix} = e^{A_e(t-kT)}x_e(kT) \tag{3.24}$$

so, partitioning the matrix exponential, the matrices in (3.22) are:

$$\begin{pmatrix} \bar{A} & \bar{B} \\ 0 & I \end{pmatrix} = e^{A_e T} \tag{3.25}$$

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `expm, c2d, c2dm, d2c`.

Example 3.2. The MATLAB® code for discretising a system is (`Abar`=$\bar{A}$, `Bbar`=$\bar{B}$):

```
s = expm([[a b]*t; zeros(nb,n+nent)]);
Abar = s(1:n,1:n); Bbar = s(1:n,n+1:n+nent);
```

3.5 Gain

The concept of gain is very useful for characterising the behaviour of a system, usually assuming *stable* behaviour. It refers to the relation between the output and input "magnitude". But it can be defined as relating to the time behaviour (static, instantaneous), the frequency of the input and, in the case of MIMO systems, the direction of the input (the input vector components).

Scaling. It is clear that in order to compare magnitudes the measurement units are critical. Thus, to determine the gain or ratio between the magnitude of variables, even more so in a MIMO system, their measurement should be normalised so that they are comparable. The recommended scaling is:

- scale-manipulated variables so that the actuator range is mapped to $[-1, 1]$,
- scale disturbances so that its maximum estimated value is mapped to 1,
- scale-controlled variables so that one unit in the new scaling is equivalent to the maximum admissible errors.

In this way, individual units and limits for dozens of variables in a complex MIMO model can be forgotten and the gains that are discussed below can be compared with unity for controller validation. Scaling has no deep mathematical roots, but it is useful in practice. Also, it may avoid errors in determining how *well-conditioned* a problem is (see page 292). In some cases, measurements (may be different to *controlled variables*) should be scaled so that measurement noise is of a similar size (to help in variables' selection, Section 4.3).

3.5.1 Static Gain

Given a stable system, assume a constant input vector, $u(t) = \bar{u}$, the output reaching the value $y(t) = \bar{y}$. The static gain (also denoted as DC gain) is $\bar{G}$ such that:

$$\bar{y} = \bar{G}\bar{u}$$

From (2.17) and (2.19), letting $\dot{x} = 0$, the static gain is:

$$0 = A\bar{x} + B\bar{u}; \quad \bar{y} = [-CA^{-1}B + D]\bar{u} \tag{3.26}$$

which requires A to be regular.

Other representations. Applying the final value theorem of the Laplace transform:

$$\bar{y} = \lim_{t\to\infty} y(t) = \lim_{s\to 0} sy(s) = \lim_{s\to 0} sG(s)u(s) \tag{3.27}$$

Thus, as $u(s) = \frac{\bar{u}}{s}$:

$$\bar{G} = \lim_{s\to 0} G(s) = G(0) \tag{3.28}$$

Again, recalling (2.59), $\bar{G} = -CA^{-1}B + D$ is also obtained.
For DT systems, the static gain is:

$$\bar{G} = \lim_{z \to 1} G(z) = G(1) = C(I - A)^{-1}B + D \tag{3.29}$$

The static gain for polynomial representations is obtained by solving the system equation after deleting all the derivative terms.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `dcgain,ddcgain`.

3.5.2 Instantaneous Gain

In a similar way, the immediate response of a system to a step change in the input can be computed by applying:

$$y(0^+) = \lim_{t \leftarrow 0} y(t) = \lim_{s \to \infty} sy(s) = \lim_{s \to 0} sG(s)u(s) \tag{3.30}$$

and:

$$G_\infty = \lim_{s \to \infty} G(s) \tag{3.31}$$

Recalling (2.59),

$$G_\infty = D \tag{3.32}$$

That is, the input/output coupling matrix. For DT systems, the instantaneous gain is also given by D.

The coupling matrix, D, is usually null in physical systems: there is no "immediate" response to an external action, also in sampled-data systems where a minimum delay of one sampling period is included. On the other hand, it is normally non-zero in the controller subsystems: for example, the proportional gain is an instantaneous one.

3.5.3 Directional Gain

To evaluate how big a gain is, some measurements about the size (norm) of a matrix should be used. A summary of matrix norms is included in Appendix C. The spectral radius of a matrix, as defined by

$$\rho(G) = \max_i |\lambda_i| \tag{3.33}$$

is a useful measure of the matrix size. However, the eigenvalues are only defined for square matrices. Furthermore, they refer to a "gain" where the output vector and input vector have the same direction. Relaxing this assumption, other directions with higher gain may be found.

Given an input vector with constant magnitude, $|u(t)| = u_0$, the output magnitude depends on the input direction. It is interesting to know in which directions the extreme gains (maximum and minimum) are obtained. This could be relevant in determining how to act or what to measure in a controlled plant. For this purpose, we recall the singular value decomposition (SVD) of a matrix (Appendix B).

Given the matrix $G \in \mathbb{R}^{p\times m}$, $(y = Gu)$, its singular values are

$$\sigma_i = \sqrt{\lambda_i\{GG^T\}}; \qquad \sigma_i \geq 0 \tag{3.34}$$

In the appendix, it is shown that there is an SVD of G expressed by $G = U\Sigma V^T$, where Σ is a diagonal matrix. The number of non-zero entries in Σ is equal to the rank of G, and it is at most $\min(p, m)$. $U \in \mathbb{R}^{p\times p}$ and $V \in \mathbb{R}^{m\times m}$ are unitary matrices with orthonormal column vectors, u_i and v_i respectively, denoted by "input" (V) and "output" (U) coupling matrices. If complex matrices (such as those in the frequency-response) are considered, then $G = U\Sigma V^H$, and U and V are complex matrices; Σ, however, is still real diagonal.

The physical meaning is as follows. Assume a unitary input, $u = v_i$. The output is $y = \sigma_i u_i$, that is, a vector of magnitude σ_i and direction u_i. So, the action of G is a combination of scaling and rotations, and its directional gain is a sort of ellipsoid with extreme gains:

$$\text{max. gain: } \bar{\sigma} = \max_i \sigma_i = \sigma_1; \qquad \text{min. gain: } \underline{\sigma} = \min_i \sigma_i = \sigma_s \tag{3.35}$$

If there is a big difference between the extreme gains, the matrix is said to be *ill-conditioned*. This is characterised by the condition number defined by:

$$\gamma(G) = \frac{\bar{\sigma}(G)}{\underline{\sigma}(G)} \tag{3.36}$$

Ill-conditioned plants will be difficult to control because the response is very different depending on the direction of the input (B.14), with high sensitivity to actuator modelling errors.

Example 3.3. Let us assume the matrix

$$G = \begin{pmatrix} 45 & -72 & -5 \\ 0.3 & 1.5 & -0.2 \\ 27 & -45 & 9 \end{pmatrix}$$

The SVD gives (MATLAB®, `[U,S,V]=svd(G)`):

$$U = \begin{pmatrix} u_1 & u_2 & u_3 \end{pmatrix} = \begin{pmatrix} 0.8503 & -0.5262 & 0.0047 \\ -0.0112 & -0.0270 & -0.9996 \\ 0.5261 & 0.8499 & -0.0289 \end{pmatrix}; \qquad S = \begin{pmatrix} 99.819 & 0 & 0 \\ 0 & 10.32 & 0 \\ 0 & 0 & 1.025 \end{pmatrix}$$

$$V = \begin{pmatrix} v_1 & v_2 & v_3 \end{pmatrix} = \begin{pmatrix} 0.5256 & -0.0717 & -0.8477 \\ -0.8507 & -0.0386 & -0.5242 \\ 0.0049 & 0.9967 & -0.0813 \end{pmatrix}$$

For an input vector: $u \approx v_3$, the gain is $g_{min} \approx \underline{\sigma}_3$ and the output is in the direction of $y \approx u_3$:

$$y = G * [-0.85 \ -0.52 \ 0]^T = [-0.81 \ -1.03 \ 0.45]^T; \qquad |\, y \,| = 1.39$$

If the sign of the second input is changed, the output dramatically changes:

$$y = G * [-0.85 \ 0.52 \ 0]^T = [-75.69 \ 0.52 \ -46.35]^T; \qquad |\, y \,| = 88.7$$

In this case, the maximum gain $g_{max} \approx \underline{\sigma}_1$ would be for an input $u \approx v_1$, and the output is in the direction of $y \approx u_1$:

$$y = G * [0.5 \ -0.8 \ 0]^T = [80.1 \ -1.05 \ 49.5]^T; \qquad |\, y \,| \approx 94.2$$

Diagonal dominance. From the control viewpoint, a particularly interesting condition of a gain matrix is the so-called *diagonal dominance*. For a square-gain matrix $G = \{g_{i,j}\}$, if $\forall i$, $\exists j_i$ such that $j_i \neq j_k$ if $i \neq k$ and

$$g_{i,j_i} > \sum_{j \neq j_i} |g_{i,j}|; \qquad \varrho_i = \frac{\sum_{j \neq j_i} |g_{i,j}|}{g_{i,j_i}} < 1 \tag{3.37}$$

there is a diagonal dominance in the sense that by a suitable raw and column sorting, the matrix gain will appear almost as a diagonal matrix, and G will be non-singular. The ratio ϱ_i is denoted as the *Gershgorin radius*. The dominance can be analysed either by rows (as in the above definition) or by columns (see Appendix, page 288).

Example 3.4. Given a gain matrix, $y = Gu$,

$$G = \begin{pmatrix} 6 & 40 & 0 \\ 6 & 0.3 & 0 \\ 0.23 & 0.12 & 8.25 \end{pmatrix}$$

there is a clear dominant effect (or coupling) between $y_1 \leftrightarrow u_2$, $y_2 \leftrightarrow u_1$ and $y_3 \leftrightarrow u_3$. The Gersghorin radii are $\varrho_1 = 0.15$, $\varrho_2 = 0.05$, $\varrho_3 = 0.042$. Observe that the column Gersghorin radii do not show any dominance.

The diagonal dominance concept may be used to analyse the possibility of matching (pairing) input/output variables in decentralised control (Section 5.2.3).

3.6 Frequency response

Frequency-response techniques have been extensively applied in the study of SISO systems. We recall that the steady-state output of a linear system under

a sinusoidal input is also sinusoidal with the same frequency, the gain and the phase shift being determined by the frequency response (see Appendix A).

For MIMO systems, the frequency response, $G(j\omega)$, is a matrix and, as previously seen, its "gain" depends not only on the frequency but also on the input direction, that is, on the input combination. Thus, it will be interesting to determine the properties of the gain matrix for each frequency, $G(jw) = G(s)|_{s=jw}$:

- the rank of the gain matrix, to determine the existence of an imaginary zero if it decreases at a given frequency,
- the extreme gains, $\bar{\sigma}_G(jw)$ and $\underline{\sigma}_G(jw)$, and the condition number, in particular, around critical frequencies (bandwidth, cut-off, resonance, ...), as they are used in input/output controllability analysis and closed-loop performance and robustness analysis (see Section 8.5),
- the diagonal dominance, to evaluate the possible effects of interactions if a control is applied. The computation of the Gersghorin radii (3.37), in particular at the critical frequencies, allows us to determine a sufficient condition that guarantees the system's closed-loop stability (5.13).

This kind of analysis for a general frequency response of a MIMO system will point out the maximum and minimum reachable gains for a given input direction and at a given frequency. The MATLAB® command `sigma` computes these gains. Similarly, diagonal dominance can be also determined as a function of the frequency .

Example 3.5. The following MATLAB® code

```
A=[-2 1 2 .4;1 -2 1 .6;-.5 .25 .1 -.05;0.2 .1 .1 -.45];
B=[1.7 -.1;.11 1.4;0 0;0 0.01]; C=[1 -0.2 0 0;-0.3 1.5 0 0];
pt=ss(A,B,C,0);
```

simulates a TITO fourth-order system. Assume two separate output feedback proportional controllers of gain $k_p = 4$, controlling y_1 by u_1 and y_2 by u_2. By using (2.51) or the following MATLAB® code:

```
K=[4 0;0 4]; errclsedloop=feedback(eye(2),pt*K,-1);
eig(errclsedloop);
```

a stable controlled system is obtained with closed-loop poles located at {-12.27,-6.88,-0.26,-0.4}, the closed-loop error, $(r - y)$, being represented by the function `errclsedloop`. If we compute the extreme error gains as a function of the frequency, by means of

```
sigma(errclsedloop);
```

the result being plotted in Figure 3.1, it can be realised that at low frequencies, for a given combination of inputs, the error "gain" can reach the value of +14 dB. The error gain is reduced below −6 dB in the range of frequencies between 0.3 and 4 rad, and the controller will be safely efficient only in this range, in the sense that it will follow both references with less than 50% error. In fact, the closed-loop error static gain is:

$$G_0 = [I - CA^{-1}BK]^{-1} = \begin{pmatrix} 4.4808 & -0.6622 \\ 2.8926 & -0.2726 \end{pmatrix}$$

with $\bar{\sigma} = 5.38$ and $\underline{\sigma} = 0.13$, showing a possibly inappropriate steady-state behaviour unless the desired references always lie in the low-error direction.

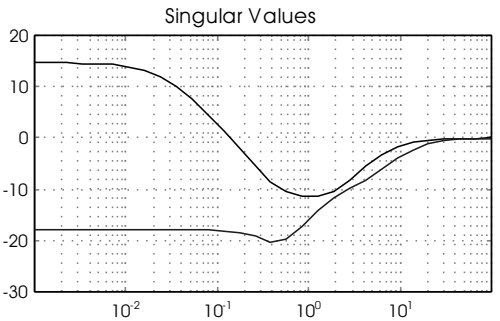

Figure 3.1. Max/min singular value of the error gain matrix

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `sigma,dsigma`.

System norm. Evaluating the maximum gain, $\bar{\sigma}(G(j\omega))$, and its maximum over all the frequency range, the worst-case amplification for any input signal is obtained. The value:

$$\|G\|_\infty = \sup_w \bar{\sigma}(G(j\omega)) \tag{3.38}$$

is denoted as the ∞-norm of the system. Further details are available in Appendix C.

Example 3.6. The above closed-loop error function has a worst-case maximum gain given by: `norm(errclsedloop,inf) = 5.38`, so there exists one reference signal for which error "power" is more than five times the set-point. If details on it are needed, then the `sigma` plot provides additional information.

3.7 System Internal Structure

After the study of the stability and gain issues, the next part of this chapter is devoted to the analysis of the structural properties of a system. That is, the connection between input, state, output and disturbance variables.

From a control point of view, the final goal is to determine the effect of external inputs on the controlled variables, but, in order to design the control and to better understand the global behaviour of the system, all the interconnections will be relevant. Thus, we will consider the connection of:

- input-to-state,
- state-to-output,
- input-to-output,
- disturbances as special inputs.

Although the main interest, from a control viewpoint, is the input/output connection, the analysis of the interactions with the state is relevant for selecting an adequate internal representation for:

- system implementation,
- model reduction,
- appropriate control design method.

3.7.1 Reachability (State Controllability)

State controllability is the system property of it being possible to find a control input signal to drive the state of the system from one arbitrary state to another. The physical interpretation is that directly or indirectly, the input vector independently reaches all the system states. To derive a *reachability test*, we will refer to a DT linear time-invariant system and its solution (3.9).

In particular, after n iterations:

$$x_n = A^n x_0 + A^{n-1}Bu_0 + A^{n-2}Bu_1 + \cdots + ABu_{n-2} + Bu_{n-1}$$

Thus, any targeted state $x^* = x_n$ will be reachable from x_0 if there is at least one solution of the equation system:

$$x^* - A^n x_0 = \begin{pmatrix} A^{n-1}B & A^{n-2}B & \dots & AB & B \end{pmatrix} \begin{pmatrix} u_0 \\ \vdots \\ u_{n-1} \end{pmatrix} \tag{3.39}$$

If the reachability[2] matrix is defined by:

$$\mathcal{C} = \begin{pmatrix} A^{n-1}B & A^{n-2}B & \dots & AB & B \end{pmatrix} \tag{3.40}$$

we can establish the following:

[2] In many references it is called the "controllability" matrix.

Reachability test. The system (3.7) is reachable if its reachability matrix, $\mathcal{C}$, is full rank. The control sequence could be obtained from:

$$\begin{pmatrix} u_0 \\ \vdots \\ u_{n-1} \end{pmatrix} = \mathcal{C}^{\dagger}(x^* - A^n x_0) \tag{3.41}$$

where $()^{\dagger}$ stands for the right pseudoinverse matrix.

In this case, $x^* - A^n x_0$ is a linear combination of the columns of $\mathcal{C}$. If the rank were $n_1 < n$, additional time (additional actions) will not improve the result[3]. In the following, unless otherwise stated, it will be assumed that $x_0 = 0$.

Relative degree of reachability

Looking at the reachability test, ($rank(\mathcal{C}) = n$ or n-linearly independent columns of $\mathcal{C}$), this is a mathematical test and a number of variations and approximations can be considered, leading to different concepts of partial reachability.

If $rank(\mathcal{C}) = n_c < n$, only a subspace of dimension n_c is reachable. That is, under a suitable state representation, the input will not be able to modify the value of $n - n_c$ state variables. To find a state transformation to make explicit which state variables are controlled and which are not, we know that the independent columns of $\mathcal{C}$ determine a base of the reachable subspace. The unreachable subspace will be defined by the $n - n_c$ vectors orthogonal to the columns of $\mathcal{C}$, that is, the solutions of:

$$\mathcal{C}^T x = 0 \tag{3.42}$$

This can be computed, for instance, by the MATLAB® command `null(C')`, or by singular value decomposition (Section B.4.1).

Reachable subsystem. To find the similarity transformation to make explicit the reachable state variables, take into account the complementary and orthogonal character of both subsystems. Thus, given (A, B), with reachability matrix, $\mathcal{C} = c$, compute:

```
T1=subnc=null[c']; T2=subc=null[subnc']; T=[T2 T1]
```

The T-state transformation will result in:

$$\bar{x}_{k+1} = \begin{pmatrix} \bar{A}_{11} & \bar{A}_{12} \\ 0 & \bar{A}_{22} \end{pmatrix} \bar{x}_k + \begin{pmatrix} \bar{B}_1 \\ 0 \end{pmatrix} u_k \tag{3.43}$$

showing that the first part of the state vector is reachable and the second one is not. If the unreachable subspace is stable, the system is said to be *stabilisable*.

[3] This matrix construction stops at A^{n-1} because, by the Cayley-Hamilton theorem, it is well known that A^n can be expressed as a linear combination of $A^i, i = 0, 1, ..., n-1$ (see Appendix B).

Single-input reachability. Instead of considering the full input vector, the reachability from a single-input variable can be determined. If the input matrix B is expressed by $B = (b_1 \quad b_2 \dots b_n)$:

$$rank(\mathcal{C}_i) = rank(\begin{pmatrix} A^{n-1}b_i & A^{n-2}b_i & \dots & Ab_i & b_i \end{pmatrix}) = n(i) \tag{3.44}$$

will show the dimension, $n(i)$, of the subspace reachable with input u_i. If $n(i) = n$, the whole system is reachable by this input and the pair (A, b_i) is said to be *cyclic*.

Example 3.7. Given the fourth-order two-input system defined by:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -2 & -1 & -1 & 0 \\ -2 & -1 & 0 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 0 \\ .1 & 0 \\ 0 & 1 \\ .1 & 1 \end{pmatrix}$$

the following properties can be computed:

- the reachable subspace dimension: `rank(ctrb(A,B)) = 3`,
- the explicit unreachable state variables:

```
nc=null((ctrb(A,B))'); c=null(nc');
T=[c nc]; A1=inv(T)*A*T, B1= inv(T)*B
A1 =                                        B1 =
  0.0000   0.5774  -0.5774  -0.5774            0         0
  2.0000  -0.5000  -0.8660  -1.3660      -0.0366    1.0000
  2.0000   0.8660  -0.5000  -0.3660       0.1366    1.0000
  0.0000   0.0000   0.0000   0.0000       0.0000    0.0000
```

  showing that the new fourth state variable

$$\bar{x} = T^{-1}x; \qquad \bar{x}_4 = [0 \qquad 0.2113 \qquad 0.7887 \qquad -0.5774]x$$

  is not reachable. If the uncontrollable mode were unstable, the system would be *non-stabilisable* (there is a "disconnected" unstable state). This would be the case if these matrices were from a CT system,
- the single-input reachability (for u_1): `b1=B(:,1); rank(ctrb(A,b1)) =3`.

Minimum reaching time. Looking at the $m \times n$ columns of $\mathcal{C}$, (3.40), let us find the first n-independent ones, and form a modified $n \times n$ reachability matrix as:

$$\bar{\mathcal{C}} = \begin{pmatrix} b_1 & Ab_1 & \dots & A^{\mu_1-1}b_1 & b_2 & Ab_2 & \dots & A^{\mu_2-1}b_2 & \dots & b_m & Ab_m & \dots & A^{\mu_m-1}b_m \end{pmatrix} \tag{3.45}$$

This arrangement shows that we can use a set of sequences of length μ_i of actions in the input u_i, $i = 1, \dots m$, to drive the system between two states[4]. Thus, if

$$\mu = \max_i \mu_i$$

a sequence of μ (the system reachability index) inputs will be required to reach an arbitrary state: the reachability index is the fastest time any arbitrary state can be reached.

[4] μ_i is called the reachability index of input i.

Reachability effort. Rank computation, in theory, involves the evaluation of determinants. If the rank of a matrix is n, that means the determinant of an $n \times n$ submatrix of this matrix is different from zero. However, that procedure is numerically unreliable, and rank determination based on the SVD decomposition is advised (see Appendix B.4.1).

The control sequence to drive the state is obtained by multiplying $\mathcal{C}^{\dagger}$ by the factor related to the desired state $(x^* - A^n x_0)$. Thus, a norm of this matrix will determine the control effort required to reach an arbitrary final state or, equivalently, the minimum gain of the reachability matrix (it will be the maximum gain of the pseudoinverse).

The reachability effort is defined as:

$$E_\rho = \bar{\sigma}(\mathcal{C}^{\dagger}) = \frac{1}{\underline{\sigma}(\mathcal{C})} \tag{3.46}$$

These measurements of the degree of reachability can be used to compare different control alternatives for the same or different processes. They may be also used to determine the suitability of choosing the actuators' placement.

Example 3.8. Let us consider the reachable part of the system considered in Example 3.7, and compute some properties:

```
A11=                          B1=
  0.0000   0.5774  -0.5774      0.0000        0
  2.0000  -0.5000  -0.8660     -0.0366   1.0000
  2.0000   0.8660  -0.5000      0.1366   1.0000
```

- for the two inputs: `rank([B11 A11*B11])=3` thus, the reachability time is two periods, $2T$,
- the reachability effort

  ```
  svd(ctrb(A11,B11))={1.9486,1.7402,0.5336}
  svd([B11 A11*B11])={1.7364,1.0124,0.0995}
  ```

  is much lower if three periods are used (0.0995/0.5336),
- the subsystem is reachable with any of the inputs:

  ```
  rank(ctrb(A11,B1(:,1))) =3
  svd(ctrb(A11,B1(:,1))) ={0.3031,0.1270,0.0450}
  rank(ctrb(A11,B1(:,2))) =3
  svd(ctrb(A11,B1(:,2))) ={1.9319,1.7321,0.5176}
  ```

  but the control effort will be much larger for some targets with the first input ($\underline{\sigma}_1 = 0.045$), compared with the second one ($\underline{\sigma}_2 = 0.5176$).
  On the other hand, there is not much benefit in using the first input, as two inputs have $\underline{\sigma} = 0.534$ and the subsystem could only be controlled by the second input ($\underline{\sigma} \approx \underline{\sigma}_2 = 0.518$).

3.7.2 Observability

State observability is a dual property of reachability, but relates state and output vectors. The information we get from the system state is through the output vector (the sensors). Thus, state observability is a system property related to it being possible to compute the state of the system from the output observations. The physical interpretation is that, directly or indirectly, all the state variables should influence the output in one way or another. To derive an *observability test*, we will refer again to a DT linear time-invariant system (3.7).

Assume an initial state, $x_0 \neq 0$, and, to simplify the notation, a free evolution ($u_k = 0, \forall k$). The sequence of measurements would be:

$$\begin{aligned} y_0 &= Cx_0 \\ y_1 &= Cx_1 = CAx_0 \\ y_2 &= CA^2x_0 \\ &\cdots \\ y_{n-1} &= CA^{n-1}x_0 \end{aligned}$$

or, in matrix form:

$$Y = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{pmatrix} x_0 = \mathcal{O}x_0; \qquad \mathcal{O} = \begin{pmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{pmatrix} \tag{3.47}$$

where $\mathcal{O}$ is denoted as the *observability matrix*[5]. Dually, we can establish the following:

Observability test. The system, (3.7), is observable if the observability matrix, $\mathcal{O}$, is full rank. The initial state can be computed as:

$$x_0 = {}^{\dagger}\mathcal{O}Y$$

where ${}^{\dagger}()$ stands for left pseudoinverse.

Relative degree of observability

Similar to the analysis of the relative reachability, we can define:

- **partial observability**. If $rank(\mathcal{O}) = n_o < n$, the whole state space is not observable, and there is a subspace of dimension $n - n_o$ without any effect on the output. This subspace is formed by the non-trivial solutions of:

[5] A similar concept is sometimes called the reconstructivity matrix.

$$0 = \mathcal{O}x^*$$

i.e., the null space of $\mathcal{O}$. The *observable* subspace is orthogonal to it, generated by the rows of $\mathcal{O}$. If the unobservable subspace has unstable dynamics, even if the output trajectories are acceptable, the system will sooner or later "blow up" due to the hidden instability. If there is no unobservable unstable dynamics, the system is said to be *detectable*.

- **single output observability**. If the observability matrix $\mathcal{O}_i$ constructed with only the i-th row of C (denoted by ${}_iC$) (MATLAB®, `obsv(A,C(i,:))` is full rank, the system is fully observable by y_i. Otherwise, $rank(\mathcal{O}_i) = n_o(i)$ shows the dimension of the observable subspace for this output.
- **observation time**. Observability indices (ν_i) can be attached to the outputs by sorting the first n-independent rows of the observability matrix:

$$\bar{\mathcal{O}} = \begin{pmatrix} {}_1C \\ {}_1CA \\ {}_1CA^{\nu_1-1} \\ \vdots \\ {}_pCA^{\nu_p-1} \end{pmatrix}$$

The observability time will be $\nu = \max_i\{\nu_i\}$.

- **detectability degree**. Based on (3.47), if the lowest gain of the observability matrix is small, changes in the state will not be easily detectable. Thus, the detectability degree will be expressed as $\underline{\sigma}(\mathcal{O})$.

3.7.3 Output Reachability

The reachability concept can be also applied to the output instead of the state. A system is *output-reachable* if there is a sequence of control actions driving the output from an initial value to a final one. It is easy to derive an output reachability test by just looking at the DT convolution (3.11), such as:

A system is output-reachable if:

$$rank([D\ CB\ CAB\ \ldots\ CA^{n-1}B]) = p \tag{3.48}$$

The output reachability time will be the lowest i such that:

$$rank([D\ CB\ \ldots\ CA^{i-1}B]) = p$$

(It will be instantaneous if $rank(D) = p$).

Observe that if a system fulfills this property it does not mean that a given output can be maintained at a set-point. It only means that this output value can be reached at a given time instant.

All these measurements of the degree of observability can be used to compare different measurement alternatives and may be also used to determine the suitability of choosing the sensors' placement.

3.7.4 Remarks on Reachability and Observability

It should be emphasised that these concepts are related to the system structure and connectivity. No regulation properties can be, in rigour, inferred.

That is, if a system is fully reachable (output-reachable), it does not mean that the system can be regulated around any reachable state (output). Furthermore, reachability implies bringing the system to an arbitrary state in a very short time, and the control effort, (3.46), is a worst-case measure. In regulation tasks, the target state is an equilibrium one, and the desired settling time is usually substantially larger than the system order. So, the guidelines on page 69 are only orientative and must be confronted with other methodologies of actuator selection. Similar remarks can be made about the degree of observability, $\underline{\sigma}(\mathcal{O})$. In this sense, the concept of input/output controllability, introduced in Section 3.9.1, gives a complementary understanding of the control possibilities of a process.

Partial input (output) vector reachability (observability) is also a measurement of a controlled system *integrity*, that is, it allows us to determine if a system remains reachable (observable) if there is a failure in some actuators (sensors). Fault handling is discussed in Section 9.6.

Although the development has been justified for DT systems, the following remarks are appropriate:

- **continuous-time systems**. For CT systems, the reachability and observability matrices are the same, although their derivation is a bit more complicated. Also applicable are the concepts of degree of fulfillment of these properties, although the reachability time has no sense if the control input is not bounded,
- **sampled-data systems**. If a reachable (observable) CT system is sampled with sampling period T, the resulting SD system keeps the same properties. The only exceptional situation of losing these properties happens if the sampling frequency ($w_s = 2\pi/T$) is a multiple of any internal resonant frequency.

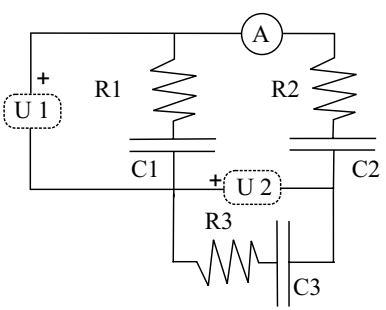

Figure 3.2. Example electric circuit

Example 3.9. For the electric circuit in Figure 3.2, where the state variables are the capacitor voltages, V_i, and the outputs are sensed by the current meter shown in the figure and a voltmeter in capacitor 3, it is easy to derive the following state and output equations:

$$\frac{d}{dt}\begin{pmatrix} V_1 \\ V_2 \\ V_3 \end{pmatrix} = \begin{pmatrix} \frac{-1}{R_1C_1} & 0 & 0 \\ 0 & \frac{-1}{R_2C_2} & 0 \\ 0 & 0 & \frac{-1}{R_3C_3} \end{pmatrix}\begin{pmatrix} V_1 \\ V_2 \\ V_3 \end{pmatrix} + \begin{pmatrix} \frac{1}{R_1C_1} & 0 \\ \frac{1}{R_2C_2} & \frac{1}{R_2C_2} \\ 0 & \frac{1}{R_3C_3} \end{pmatrix} u$$

$$y = \begin{pmatrix} 0 & \frac{-1}{R_2} & 0 \\ 0 & 0 & 1 \end{pmatrix} x + \begin{pmatrix} \frac{1}{R_2} & \frac{1}{R_2} \\ 0 & 0 \end{pmatrix} u$$

By simple inspection of the reachability and observability matrices (do that as an exercise), the following conclusions are easy to derive:

- the circuit is unobservable (x_1 is not observable). Only x_2 is observed by y_1, and x_3 by y_2,
- the system is fully reachable with two inputs. Input 1 controls two state directions, except if $R_1C_1 = R_2C_2$, when it only controls one (two identical circuits in parallel). Input 2 controls, in any case, 2 directions (V_2 and V_3)[6].

3.7.5 Canonical Forms

A reachable (observable) system can be transformed to canonical representations pointing out this property. These representations will be also useful in illustrating some control properties, as shown later on in the control design chapters.

Let us assume a SISO system with transfer function:

$$G(s) = \frac{b_1 + b_2 s + \ldots + b_n s^{n-1}}{a_1 + a_2 s + \ldots + a_n s^{n-1} + s^n}$$

It can be represented by the state model in the so-called *reachable canonical form*:

$$A_c = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ -a_1 & -a_2 & -a_3 & \ldots & -a_n \end{pmatrix}; \quad B_c = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}; \quad C_c^T = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix} \tag{3.49}$$

It is easy to verify that this model is reachable and the state variables are the derivatives of the first one:

[6] Note that the above conclusions have been obtained under the condition that eliminating one input means setting $u_i = 0$, *i.e.*, short-circuiting, and *not* opening the circuit.

$$x_{i+1} = \frac{dx_i}{dt} = \frac{d^i x_1}{dt^i}, i = 1, \ldots, n-1 \tag{3.50}$$

$$\frac{dx_n}{dt} = \frac{d^n x_1}{dt^n} = -\sum_{i=0}^{n-1} a_i x_i + u \tag{3.51}$$

$$\frac{d^n x_1}{dt^n} + a_{n-1}\frac{d^{n-1} x_1}{dt^{n-1}} + \cdots + a_1 \frac{dx_1}{dt} + a_0 x_1 = u \tag{3.52}$$

$$x_1(s) = \frac{1}{a_1 + a_2 s + \ldots + a_n s^{n-1} + s^n} \tag{3.53}$$

That is, the system can be represented by a chain of n-integrators, the output being a combination (b_i) of the states.

In the DT setting, the canonical form implies:

$$x_{i+1,k} = x_{i,k+1} = x_{1,k+i}, i = 1, \ldots, n-1$$

leading to a chain of pure delays instead of integrators.

For any other state representation, (A, B, C), it is easy to find the similarity transformation, T_c, to obtain the canonical reachable form. First, the coefficients a_i are those of the characteristic polynomial, $\det(sI - A)$. Taking into account the special form of A_c and B_c, by (2.20), the columns of $T_c = [t_1 \quad t_2 \quad \ldots \quad t_n]$ are recursively computed:

$$T_c B_c = B \Rightarrow t_n = B \tag{3.54}$$

$$T_c A_c = A T_c; \qquad t_{n-1} - a_n t_n = A t_n \Rightarrow t_{n-1} = (A - a_n I) t_n \tag{3.55}$$

$$\ldots \tag{3.56}$$

$$t_1 = A t_2 - a_2 t_n \tag{3.57}$$

with the final check that $-a_1 t_n = A t_1$.

In a dual way, an observable canonical form may be defined, taking the output as the first state variable if the transfer function is proper (see, for instance, [94] for details).

MIMO Systems

Transfer matrix to SS description. By repeating the above realisation for a MIMO plant, and stacking all elements together, a state space representation (non-minimal, see next section) can be obtained.

Example 3.10. For a TITO transfer matrix given by $G(s) = \begin{pmatrix} G_1 G_2 \\ G_3 G_4 \end{pmatrix}$, the above technique for each of the elements yields a realisation (A_i, B_i, C_i, D_i). The overall system can be expressed as:

$$\dot{x} = \begin{pmatrix} A_1 & 0 & 0 & 0 \\ 0 & A_2 & 0 & 0 \\ 0 & 0 & A_3 & 0 \\ 0 & 0 & 0 & A_4 \end{pmatrix} x + \begin{pmatrix} B_1 & 0 \\ 0 & B_2 \\ B_3 & 0 \\ 0 & B_4 \end{pmatrix} u \tag{3.58}$$

$$y = \begin{pmatrix} C_1 & C_2 & 0 & 0 \\ 0 & 0 & C_3 & C_4 \end{pmatrix} x + \begin{pmatrix} D_1 & D_2 \\ D_3 & D_4 \end{pmatrix} u \tag{3.59}$$

MIMO canonical forms. For MIMO systems there is the possibility of defining some canonical forms. The most well known is the Luenberger canonical form. It can be shown that any reachable system can be transformed into a state representation where there is a chain of μ_i-integrators attached to each input component. The procedure can be easily derived [67] and it is illustrated in the following example.

Example 3.11. Let us consider the DT headbox model[7] (2.69) in page 50. The reachability matrix is full rank and the reachability indices are $\mu_1 = 3$ and $\mu_2 = 2$. The first five columns of the reordered reachability matrix, (3.45), are:

$$\bar{\mathcal{C}} = \begin{pmatrix} b_{f,1} & A_f b_{f,1} & A_f^2 b_{f,1} & b_{f,2} & A_f b_{f,2} \end{pmatrix}$$

where $b_{f,i}$ stands for the ith column of matrix B_f.

1. Compute the inverse, $\bar{\mathcal{C}}^{-1} = \begin{pmatrix} \upsilon_1 \\ \upsilon_2 \\ \upsilon_3 \\ \upsilon_4 \\ \upsilon_5 \end{pmatrix}$.
2. Take the μ_1-th and $(\mu_1 + \mu_2)$-th rows (denoted as υ_3 and υ_5 respectively).
3. Use the similarity transformation, T_L, composed as: $T_L = \begin{pmatrix} \upsilon_3 \\ \upsilon_3 A_f \\ \upsilon_3 A_f^2 \\ \upsilon_5 \\ \upsilon_5 A_f \end{pmatrix}$.
4. Compute the canonical form:

$$A_L = T_L A_f T_L^{-1} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0.6065 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0.7711 & -2.0961 & 0 & -0.2865 & 1.1467 \end{bmatrix}$$

$$B_L = T_L B_f = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}; \quad C_L = C_f T_L^{-1} = \begin{bmatrix} -0.9315 & 0 & 0 & 0.4923 & -0.6321 \\ 0.6585 & 0 & 0 & -0.2628 & 0.2908 \end{bmatrix}$$

In this case, two chains of pure delays appear, ended by the state variables x_3 and x_5.

[7] Part of this example as well as the case study in this chapter and Chapter 6 have been worked out in collaboration with J.V. Roig [4].

3.8 Block System Structure (Kalman Form)

Now, we are in a position to analyse some structural properties of the system.

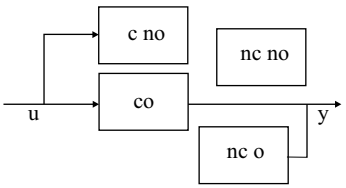

Figure 3.3. Block decomposition

Similar to the transformation of an internal representation to make explicit the reachable and unreachable parts, (3.43), in the general case, the state vector may be split into four subvectors involving the four possible combinations about reachability and observability properties. This canonical structure, known as the *Kalman form*, is composed of blocks, as depicted in Figure 3.3:

unreachable and unobservable (*ncno*),
reachable and unobservable (*cno*),
observable and unreachable (*nco*), and
reachable and observable (*co*).

Example 3.12. Let us consider a two-input-one-output system defined by:

```
A =
   -0.0450   -0.4592   -0.1223    0.3892
    0.4659   -2.2628    0.3591    0.7119
    1.1125    0.4726   -1.3667    0.0886
   -0.8736   -0.2277    1.1511   -0.4256
B =
    0.4513    0.1423
    0.3791    0.5195
    0.5774    0.0000
    0.5650    1.3076
C =
    0.4513    0.3791    0.5774    0.5650
```

The system is unreachable and unobservable as the "practical" controllability and observability ranks are 2:

```
c=ctrb(A,B); svd(ctrb(A,B))=2.1210    1.2338    0.0007    0.0001
o=obsv(A,C); svd(obsv(A,C))=4.8097    1.0047    0.0000    0.0000
```

Moreover, as `rank([c';o])= 3`, there is a subspace (dimension 1) which is *ncno*.

The system decomposition is obtained as follows[8]:

[8] Those commands were typed for a system whose state-space model had more precision in the coefficients. Copying the above matrices (A, B, C) yields rank

```
ncno=null([c';o]); ncno' =   -0.7604   -0.2387    0.5774    0.1776
nco=null([c';ncno']); nco' = -0.3502    0.8830    0.0000   -0.3127
cno=null([o;ncno']); cno' =  -0.3090    0.1404   -0.5774    0.7426
co=null([ncno';nco';cno']); co' =  0.4513  0.3791   0.5774  0.5650
```

By the similarity transformation T, `T=[co cno nco ncno];`:

```
 Ad=inv(T)*A*T =
   -0.1000    0.0000   -1.0000    0.0000
    0.0000   -1.0000    0.0000    2.0000
         0    0.0000   -2.0000    0.0000
    0.0000    0.0000    1.0000   -1.0000
Bd=inv(T)*B=
    1.0000    1.0000
    0.0000    1.0000
    0.0000    0.0000
    0.0000    0.0000
Cd=C*T=
    1.0000    0.0000    0.0000    0.0000
```

a block decomposition is achieved. By this decomposition, the reachability and observability properties are derived by inspection. In this case: x_2 and x_4 are unobservable and x_3 and x_4 are unreachable. Thus, only the state x_1 links the input and output of the system.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `ctrb, ctrbf, obsv, obsvf, null, rank,svd`.

3.8.1 Minimal Realisation

If we compute the transfer matrix of a state space representation, only the reachable *and* observable subsystems will appear. Thus, in a Kalman form representation, only the `co` subsystem is required to implement the transfer matrix. State space representations without unreachable or unobservable modes are denoted as *minimal realisations*.

Example 3.13. Let us consider again Example 3.12. The transfer matrix is:

---

4 for controllability and observability matrices (!), but there is no error in the procedure. This is one of the reasons that justifies the need for SVD in practical cases with imprecision (as in this case, "only" four significant digits). The numerically reliable way of obtaining `ncno` is typing `[u s v]=svd([c';o])`, realising how low the last diagonal element in `s` is, and, assuming it to be zero, taking the last column of `v` as a basis for `ncno`.

```
 g=tf(S);  gf=zpk(g)
Transfer function from input 1 to output:
         s^3+4s^2+5s+2              (s+2) (s+1.009) (s+0.991)
--------------------------- = ---------------------------------
s^4+4.1s^3+5.4s^2+2.5s+0.1999  (s+2)(s+1.009)(s+0.991)(s+0.09996)
Transfer function from input 2 to output:
         s^3+4s^2+5s+2              (s+2) (s+1.009) (s+0.991)
--------------------------- = ---------------------------------
s^4+4.1s^3+5.4s^2+2.5s+0.1999  (s+2)(s+1.009)(s+0.991)(s+0.09996)
```

So, both elements are identical and some terms cancel. Thus, the transfer matrix will be:

$$y(s) = \frac{1}{s+0.1}\begin{pmatrix}1 & 1\end{pmatrix} u(s)$$

This result could be extracted directly from the Kalman form, just considering the first-order `co`-subsystem. The number of states in the minimal realisation can be evaluated by the rank of the *product* of the observability and controllability matrices.

As previously seen, the concept of "minimal realisation" is connected to cancelling terms in the state representation, but it also has the same meaning dealing when with transfer matrices.

Let us consider this reduction, for the sake of clarity, in a SISO case.

Example 3.14. Given the transfer function, in factorised form:

$$g(s) = \frac{(s+3.02)(s+1.04)(0.00071s+1)}{(s+4)(s+2.994)(s+1.012)}$$

some pole-zero cancellations clearly appear. Let us look for a state representation of $g(s)$ and its properties,

```
n =[0.0007 1.0029 4.0622  3.1408];
d =[1.0000 8.0060 19.0539 12.1197];
g=tf(n,d);
[A,B,C,D]=tf2ss(n,d)
rank(ctrb(A,B))=     3
rank(obsv(A,C))=     3

svd(ctrb(A,B)) = {46.4732,0.4762,0.0452}
svd(obsv(A,C)) = {83.1521,0.0672,0.0015}
svd(obsv(A,C)*ctrb(A,B))={265.1,0.03,0.001}
```

pointing out that there is only one mode that matters for controllability and observability. Proceeding as before, but giving greater tolerance to the reachability and observability computation, only one state will remain. For example:

```
rank(obsv(A,C),0.1))=1
```

A more systematic approach to model reduction is presented later on in Section 3.10.

Example 3.15 (Headbox system). Let us now consider again the simple model of the headbox, and look for a mathematical internal representation without looking at the physical interpretation of the state variables.

Taking the DT transfer function in (2.68) (including the delays), and using the MATLAB® command:

$$[A_{11},\, B_{11},\, C_{11},\, D_{11}] = \mathsf{ssdata(G(1,1))}$$

a realisation of the first TF element is obtained. To get the full realisation we can proceed in different ways:

a) Repeating the sequence above for each element of the DT transfer matrix. This will lead to an augmented state representation, surely unreachable and unobservable.
b) Reducing to a common denominator the elements of $G(z)$ by columns and realising with the same state variables all these transfer functions. Attached to each input (column) a reachable subsystem will be realised. In this case, the global system could be unobservable.
c) Doing the same by rows, to get a set of p observable subsystems. The global system could be unreachable.

Let us illustrate the first option, obtaining (A_{ij}, B_{ij},C_{ij}):

$$A_{11} = \begin{bmatrix} 0.6065 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}; \quad B_{11} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}; \quad C_{11} = \begin{bmatrix} 0 & 0 & 0.3935 \end{bmatrix}; \quad D_{11} = 0$$

$$A_{12} = \begin{bmatrix} 0.3679 \end{bmatrix}; \quad B_{12} = \begin{bmatrix} 1 \end{bmatrix}; \quad C_{12} = \begin{bmatrix} -0.6321 \end{bmatrix}; \quad D_{12} = 0$$

$$A_{21} = \begin{bmatrix} 1.385 & -0.472 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}; B_{21} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}; C_{21} = \begin{bmatrix} 0 & 0 & 0.049 & 0.038 \end{bmatrix}; D_{21} = 0$$

$$A_{22} = \begin{bmatrix} 1.1467 & -0.2865 \\ 1 & 0 \end{bmatrix}; \quad B_{22} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; \quad C_{22} = \begin{bmatrix} 0.2908 & -0.2628 \end{bmatrix}; \quad D_{22} = 0$$

The global system would be:

$$A = \begin{bmatrix} A_{11} & 0 & 0 & 0 \\ 0 & A_{12} & 0 & 0 \\ 0 & 0 & A_{21} & 0 \\ 0 & 0 & 0 & A_{22} \end{bmatrix}; \quad B = \begin{bmatrix} B_{11} & 0 \\ 0 & B_{12} \\ B_{21} & 0 \\ 0 & B_{22} \end{bmatrix};$$

$$C = \begin{bmatrix} C_{11} & C_{12} & 0 & 0 \\ 0 & 0 & C_{21} & C_{22} \end{bmatrix}; \quad D = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix}$$

We know that this is a fifth-order system, but we obtained a 10-dimension state vector. This means that there are five extra states which are unreachable, unobservable or both. We can check that the system is unreachable and unobservable by computing the rank of the reachability and observability matrices. To reduce the model, the MATLAB® command

$$[A_m,\, B_m,\, C_m,\, D_m] = \mathsf{minreal}(A,\, B,\, C,\, D,\, tol)$$

will provide a minimal realisation, such as:

$$A_m = \begin{bmatrix} 0.9755 & -0.0237 & -0.0864 & -0.0658 & -0.0521 \\ 0.0352 & 0.5262 & -0.0956 & -0.4850 & 0.5282 \\ 1.0630 & -0.0770 & 0.0364 & -0.0374 & -0.1158 \\ 0 & -0.0145 & 0.9184 & -0.0161 & 0.0007 \\ 0 & -0.0349 & -0.3802 & 0.2921 & 0.2312 \end{bmatrix}$$

$$B_m = \begin{bmatrix} -1.4057 & 0.0593 \\ 0 & -0.5369 \\ 0 & 0.1122 \\ 0 & -0.3741 \\ 0 & -1.1259 \end{bmatrix}; \quad C_m = \begin{bmatrix} 0 & 0 & 0 & -0.0471 & 0.5771 \\ 0 & 0 & 0 & -0.1256 & -0.2165 \end{bmatrix}$$

The argument 'tol' allows for defining the tolerance in admitting a matrix as singular[9].

Of course, the MATLAB® command

```
Gd=tf(ss(Am,Bm,Cm,0))
```

will return the original transfer matrix. The MATLAB® command `ss`, applied to the full matrix G also yields a state space representation in one step.

3.8.2 Balanced Realisation

A balanced realisation is a minimal realisation where each mode has the same relative degree of reachability and observability. In particular, as a very special case, a realisation with the same reachability and observability matrices is balanced.

Example 3.16. Let us consider again Example 3.14, where the system has a third-order model (A, B, C, D). The transfer function is:

$$g(s) = \frac{0.00071s^3 + 1.003s^2 + 4.062s + 3.141}{s^3 + 8.006s^2 + 19.05s + 12.12}$$

or, pointing out the related factors (poles and zeros):

$$g(s) = \frac{(s + 3.02)(s + 1.04)(0.00071s + 1)}{(s + 4)(s + 2.994)(s + 1.012)}$$

$$g(s) = 0.2591 \frac{1 + 0.3311s}{1 + 0.334s} \frac{1 + 0.9615s}{1 + 0.9981s} \frac{1 + 0.00071s}{1 + 0.25s}$$

Compute a similarity transformation to get the Jordan canonical form:

```
[T,J]=jordan(A);
T=                                    J=
    0.1729    5.3225   -4.4954          -1.012     0       0
   -0.1709   -1.3306    1.5015             0      -4.      0
```

[9] In fact, the *minreal* command uses the argument 'tol' to delete the modes attached to the lowest singular value of $\mathcal{C}$ or $\mathcal{O}$, as outlined in Example 3.12.

Example 3.17. Consider the transfer matrix:

$$Y(s) = \begin{pmatrix} \frac{1}{s+1} & \frac{1}{s+2} \\ \frac{2s+0.5}{s+1} & \frac{2}{s+1} \end{pmatrix} U(s)$$

We realise that there are poles at $s = -1$ and $s = -2$. If a state representation of each element is carried out, the total system would be:

$$\dot{x} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -1.5 & 0 \\ 0 & 2 \end{pmatrix} u$$

$$y = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} x + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} u$$

It is easy to check that this fourth-order system is not a minimal realisation. Either by looking at the minimal realisation or proceeding like in Section 2.6.2, a double pole in $s = -1$ and a single one at $s = -2$ are obtained. The third pole in $s = -1$ does not belong to the co-subsystem.

$$y(s) = \left(\begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ -1.5 & 2 \end{pmatrix} \frac{1}{(s+1)} + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \frac{1}{(s+2)} \right)$$

where the first term of the right-hand side is the direct input/output coupling matrix (D), the instantaneous gain matrix, and the rank of the numerator matrix of pole in -1 is 2, and that of -2 is 1.

Regarding the determinant-based methodology, the first-order minors are the elements of the matrix, with poles in -1 and -2. If we compute the second-order minor, that is, the determinant of the transfer matrix, it would be:

```
In[17]:=  G={{1/(s+1),1/(s+2)},{(2s+.5)/(s+1),2/(s+1)}}
In[18]:= Simplify[Det[G]]
```

yields:

$$\frac{-2\ (-1.20377 + s)\ (1.45377 + s)}{(1+s)^2\ (2+s)}$$

so the least common multiple is indeed $(1 + s)^2(2 + s)$.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `eig, pole, mpole, damp, ss2zp, tf2zp`.

Zeros

Dually to the poles, the zeros can be considered as signal sinks (notch or blocking filters). If a SISO system presenting a zero at $s = z$ is excited by an input, $u(t) = e^{zt}$, the output only contains components from the system

modes (poles), and the components Ae^{zt} are "blocked". In this case, the zeros are the roots of the transfer function numerator polynomial.

In MIMO systems, it was pointed out in Section 2.6.2 that the zeros are the values of s reducing the rank of $G(s)$. This means that the previous blocking effect appears if the exciting signal is applied *in a given direction*, so the concept may be refined with directionality issues.

Zeros can be computed as the values of s such that, replacing below $y(s) = 0$, the matrix equation

$$\begin{pmatrix} sI - A & -B \\ C & D \end{pmatrix} \begin{pmatrix} x(s) \\ u(s) \end{pmatrix} = \begin{pmatrix} 0 \\ y(s) \end{pmatrix} \tag{3.60}$$

has a non-trivial solution for x, u, *i.e.*, the determinant of the left-hand side matrix vanishes. In this way, non-zero input would be "blocked" from appearing at the output.

Transfer matrix. They can be also computed as the roots of the numerator of the transfer matrix determinant if all the poles, with their respective multiplicity, appear at the denominator.

Example 3.18. In the previous example, the zeros may be computed using the $G(s)$-determinant. It was:

$$\frac{-2\ (-1.20377 + s)\ (1.45377 + s)}{(1 + s)^2\ (2 + s)}$$

all the poles appearing in the denominator.

Alternatively, using the internal representation (3.60) and *Mathematica*®:

```
In[36]:= PP={{s+1,0,0,-1,0},{0,s+2,0,0,-1},
        {0,0,s+1,0,-1},{1,1,0,0,0},{-1.5,0,2,2,0}}
Factor[Det[PP]]
Out[36]= -2(s-1.20377)(s+1.45377)
```

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `ss2zp,tf2zp,zero,tzero`.

If a state representation, a system realisation, is not a minimal one, that means that some modes or poles are unreachable or unobservable. Thus, the corresponding system matrix eigenvalues will not appear as transfer matrix poles. In this case, we will consider that there are cancellation or *decoupling* zeros.

For instance, if a system is expressed in the Kalman form:

$$A = \begin{bmatrix} A11 & 0 & A13 & 0 \\ A21 & A22 & A23 & A24 \\ 0 & 0 & A33 & 0 \\ 0 & 0 & A34 & A44 \end{bmatrix}; \quad B = \begin{bmatrix} B1 \\ B2 \\ 0 \\ 0 \end{bmatrix}; \quad C^T = \begin{bmatrix} C1 \\ 0 \\ C3 \\ 0 \end{bmatrix}$$

the eigenvalues of A_{22} are output decoupling zeros and those of A_{33} are input decoupling zeros. The eigenvalues of A_{44} could be considered in both classes. The input/output poles are the eigenvalues of A_{11}, and the "true" *transmission zeros* are those from (A_{11}, B_1, C_1, D).

As is well known, the right-half-plane (RHP) zeros, introduce additional difficulty in controlling a system. Systems with RHP zeroes are denoted as *non-minimum-phase* ones (NMP). They produce what it is called an *inverse response*. That is, for instance, if a step input is applied, the response goes in a sense (positive or negative) and the final value is of opposite sign. Dealing with MIMO systems, the multivariable zeros are not correlated with the zeros of the SISO transfer functions relating each pair input/output. Thus non-minimum-phase multivariable zeros may appear even in the case that no single transfer function has such a kind of zero. Further issues with RHP zeros are discussed in Sections 5.3 and 8.3.2.

Also it is worth realising that a MIMO system may have individual zeros in the same position as MIMO poles, without cancelling, as the example below shows.

Example 3.19. The system:

$$G(s) = \begin{pmatrix} \frac{1}{(s+1)} & 0 \\ 0 & \frac{(s+1)}{(s+2)} \end{pmatrix}$$

has poles in $\{-1, -2\}$ as well as one zero at $\{-1\}$, but they `do not` cancel: they belong to different internal variables. In general, they will act in different directions.

Another interesting property is that, for square plants, its poles are the zeros of the inverse and *vice versa*.

3.9 Input/Output Properties

Knowledge about the internal structure of a system provides criteria for: 1) selecting the placement of actuators and sensors to reach all the interesting parts of the system to be controlled and 2) pruning the irrelevant internal variables or the spurious variables appearing in the process of transforming the model representation. The partial degree of reachability and observability also helps in understanding its dynamic behaviour, as previously mentioned.

But once these properties have been verified, nothing is said about a number of issues that matter for control purposes:

- is it possible to keep the outputs (or the controlled variables) of the system at the set-points?
- is it possible to drive the outputs (or the controlled variables) tracking some references?
- how large control actions are required?
- is there any risk for hidden instability to appear?
- are the theoretically required controllers feasible?

- are there numerical problems in computing the system properties?
- is it possible to reject the disturbances?
- which uncertainty in the process model or in the disturbance knowledge is tolerable to keep some control properties?

All these properties are better connected to input/output process characteristics and should be analysed once the structural properties are fixed.

3.9.1 Input/Output Controllability

Most of these properties are summarised in the so-called input/output controllability. Unfortunately, there are not so clear criteria for evaluating the input/output controllability of a process, and even the definition of the concept is not unanimous in the literature. We will accept the following definition:

> *A system is input/output controllable if it is possible to implement a control system suitable for matching the control requirements in reference tracking and disturbance rejection, in spite of uncertainties in the process model.*

In rigour, to assess I/O controllability, based on the above definition, the only way to prove it is by designing a suitable controller. However, to assess the "possibility" of finding such a controller before investing effort in its design, some elementary tests can be carried out, based on the same ideas as those illustrated in Example 3.5.

Perfect controller. Given a plant model, $y = Gu + G_d d$, and a set-point, r, an "ideal" controller would apply $u = G^{-1}(r - G_d d)$. As commented in Section 3.5, input scaling is assumed to map $[-1, +1]$ onto the operational actuator range. Then:

- an obvious condition is that the number of control inputs should be equal to or larger than the number of controlled variables ($m \geq p$) if arbitrary set-points on them must be pursued,
- for disturbance rejection, scaling so that the maximum expected disturbance has unit magnitude, the ideal controller might be feasible if $\overline{\sigma}(G^{-1}G_d) \leq 1$, *i.e.*, the worst-case disturbance to ideal input gain does not imply actuator saturation,
- for reference tracking, scaling outputs so their desired range is mapped onto $[-1, 1]$, the condition is $\overline{\sigma}(G^{-1}) = \underline{\sigma}(G) > 1$,
- for tolerance to uncertainty and reasonable "invertibility", the condition number of G should be "small".

These conditions may be evaluated at different frequencies to assess at which frequencies ideal control is possible. However, if the zero-error target is relaxed, allowing for certain admissible deviations, the above conditions can be relaxed a bit, but get more complex. The reader is referred to [119] for details.

These conditions can be used as a guide for actuator placement, complementing reachability-based ones, or as a guide for determining the maximum level of performance achievable on a particular configuration.

Limitations on achievable performance

Feasible controller. Note that the above analysis refers to an ideal control and not to a "practical" control. The possibility of practical implementation implies some constraints in the control, in addition to the already (approximately) considered saturation bounds:

- the controller should be realisable (finite high-frequency gain[10]). The high-frequency gain limitation is also needed to avoid amplification of measuring noise and to tolerate modelling errors (see Chapter 8),
- unstable poles or zeros cannot be cancelled,
- delays cannot be inverted,
- enough and appropriate control actions should be available (related to the conditioning and minimum-gain issues commented on above).

Based on these constraints, it is clear that there may be lots of additional issues to discuss apart from those related to the controller: there are process, or model, limitations to achievable performances. If an assessment of these limitations is carried out in the starting phase of controller design, then, if they are unacceptable, a process redesign or model re-identification might be advised.

Some of these issues will be discussed in future chapters, as more notions on different solutions to the control problem are presented. However, they are summarised below, as it is important to point out that:

> *limitations must be considered as process-model properties, independent of the to-be-designed controller.*

First, ideal control (involving matrix inversion) is not achievable, as some of the above considerations regarding feasible controllers usually apply. However, there is a bunch of subtle limitations, only appearing when a detailed analysis of the frequency domain behaviour of the process is carried out. Just to mention the more apparent ones:

- the disturbances acting on a range of frequencies over the bandwidth of the actuators cannot be cancelled,
- the measurement noise frequency range may be within the desired bandwidth,
- the achievable bandwidth is limited if there are RHP zeros (for high gain the system becomes unstable),

[10] The process is invertible with finite high-frequency gain only if the direct input/output coupling matrix D is full rank. However, that situation is unusual.

- open-loop unstable plants have lower bounds in bandwidth (feedback is needed to stabilise the plant but bandwidth, in general, increases with loop gain),
- combining both previous reasons, open-loop unstable plants with RHP zeros are difficult to stabilise,
- uncertainty in the model limits achievable performance targets to ensure that simulated results are consistent with practical implementation.

As previously mentioned, details on these aspects will be discussed in Chapters 7 and 8.

3.10 Model Reduction

It is rather common to represent a dynamic system with a model whose order is higher than required to fulfill a control purpose. In the case of a process, a state model derived from an input/output representation may present unreachable and unobservable modes that must be pruned. Also, as seen in other examples, there are modes with small influence, due to either an approximate cancellation with poles or because their timescale is out of our range of interest.

There are control design techniques leading to high-order complex controllers. This is the case in cancellation controllers and in many of the model-based control design approaches.

In many cases, to achieve some controlled process performances, it suffices to use or to implement a simplified, reduced order, dynamic system. Reduction means approximation. Thus, in a reduced order model, some properties of the original system are captured but some other may be lost. Let us consider two typical approaches: approximating the dynamic behaviour keeping the same static gain, and approximating the behaviour at some critical range of frequencies, without taking care of the steady-state.

3.10.1 Time Scale Decomposition

Given a dynamic system (A, B, C, D), if there are two different time scales of behaviour, it can be decomposed into two subsystems such as:

$$\begin{pmatrix} \dot{x}_1 \\ \epsilon\dot{x}_2 \end{pmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u; \quad y = \begin{pmatrix} C_1 \\ C_2 \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

where ϵ is very small.

- **slow behaviour**. If, in the limit, it is considered that $\epsilon \to 0$, the derivative term of the second state subvector vanishes. Thus, the solution of

$$0 = A_{21}x_1 + A_{22}x_2 + B_2u$$

will determine the value of x_2:

$$x_2 = -A_{22}^{-1}(A_{21}x_1 + B_2u)$$

The reduced order system will be:

$$\dot{x}_1 = (A_{11} - A_{12}A_{22}^{-1}A_{21})x_1 + (B_1 - A_{12}A_{22}^{-1}B_2)u$$

and the fast behaviour, characterised by the state subvector x_2 is considered as instantaneous. This approximation is called *residualisation*,

- **truncation**. Assume a similarity transformation, T, in such a way that $A_{12} = A_{21} = 0$ and A_{11} and A_{22} are block-diagonal (Jordan canonical form), being $\lambda(A_{11}) \ll \lambda(A_{22})$. The dynamics of the state subvector x_2 components is much faster and they can be deleted.
  Keeping the model (A_{11}, B_1, C_1), the contribution of the fast modes has been removed. This simplest reduction does not keep the same static gain of the model,
- **fast behaviour**. Alternatively, to analyse the transient behaviour or the fast mode response, a timescale change such as $t = \epsilon\tau$ will lead to the new state equation:

$$\begin{pmatrix} \frac{1}{\epsilon}\frac{dx_1}{d\tau} \\ \frac{dx_2}{d\tau} \end{pmatrix} = Ax + Bu$$

  Proceeding as before, $\epsilon \to 0$, the derivative of the first state subvector vanishes, that is, $x_1(\tau) = \bar{x}_1$ remains constant, and the system dynamics is characterised by:

$$\frac{dx_2}{d\tau} = A_{22}x_2 + A_{21}\bar{x}_1 + B_2u$$

Example 3.20. Let us consider the manipulation of a tank level, h, by means of a valve, whose dynamics is characterised by a time constant of $\tau = 40$ ms. The partial transfer functions are:

$$g_t(s) = \frac{10}{(s+1)}; \qquad g_v = \frac{2}{(1+0.04s)}$$

so the global model is $G(s) = g_v(s)g_t(s)$. A state space model could be:

$$\begin{aligned} \dot{x}_v &= -25x_v + 50u \\ \dot{x}_t &= -x_t + 10x_v \end{aligned}$$

It can be written (with $\epsilon = 0.04$) as:

$$\begin{bmatrix} \epsilon\dot{x}_v \\ \dot{x}_t \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ -1 & 10 \end{bmatrix} \cdot \begin{bmatrix} x_v \\ x_t \end{bmatrix} + \begin{bmatrix} 2 \\ 0 \end{bmatrix} u$$

If $\epsilon \to 0$, $x_v \approx 2u$ and the approximate model becomes $\dot{x}_t = -x_t + 20u$. It will allow us to study the dynamics of filling the tank, assuming an instantaneous behaviour of the valve.

On the other hand, if the model is rewritten assuming a timescale of 10 ms, taking $\varepsilon = 0.01$, the state representation would be:

$$\begin{bmatrix} \dot{x}_v \\ \frac{1}{\varepsilon}\dot{x}_t \end{bmatrix} = \begin{bmatrix} 0 & -2.5 \\ -1 & 10 \end{bmatrix} \cdot \begin{bmatrix} x_v \\ x_t \end{bmatrix} + \begin{bmatrix} 5 \\ 0 \end{bmatrix} .u$$

For $\varepsilon \to 0$, $\dot{x}_t \to 0$, that is, $x_t = const.$ and the reduced model will be $\dot{x}_v = -2.5x_v + 5u$, allowing for the simulation of the valve dynamics.

3.10.2 Balanced Reduction

Reducing the model order, other than the timescale, the input/output relevance of each mode may be directly considered. In a balance realisation, the less reachable (and observable) modes are candidates to be deleted.

Example 3.21. Let us consider again Example 3.14, where the system was represented in a balance realisation by:

$$Ab = diag[-1.012 \quad -4 \quad -2.994]; \quad Bb^T = Cb = [\sqrt{0.0095} \quad \sqrt{0.9626} \quad \sqrt{0.0252}]$$

It appears clear that the second pole is the one contributing more to the input/output relationship. Again, to get a reduced (first) order model we can truncate or residualise the system (Ab, Bb, Cb).

Example 3.22. Let us consider the SISO system:

$$g(s) = \frac{1}{(s+0.1)(s+1)(s+10)} = \frac{0.1}{s+0.1}\frac{1}{s+1}\frac{10}{s+10} = \frac{0.1122}{s+0.1}\frac{-0.1235}{s+1}\frac{0.0112}{s+10}$$

The following reductions can be obtained:

- by deleting the factor $\frac{10}{s+10} \approx 1$ (residualisation):

$$g_1(s) = \frac{0.1}{(s+0.1)(s+1)}$$

- by deleting the term $\frac{0.0112}{s+10} \approx 0$ (truncation):

$$g_2(s) = \frac{-0.0113s + 0.0998}{(s+0.1)(s+1)}$$

- by reducing the term $\frac{0.0112}{s+10} \approx 0.00112$ (residualisation):

$$g_3(s) = 0.00112 + \frac{-0.0113s + 0.0998}{(s+0.1)(s+1)}$$

Alternatively, getting a balanced state space realisation (Ab, Bb, Cb, Db)

```
[A,B,C,D]=tf2ss(1,[1 11.1 11.1 1]);
s=ss(A,B,C,D);
sb=balreal(s);
[Ab,Bb,Cb,Db]=ssdata(sb)
```

and applying a balanced truncation (`modred(sb,3,'del')`) or, alternatively, residualisation (`modred(sb,3)`):

```
rsb=modred(sb,3);
gb1=tf(rsb)
     0.0007249 s^2 - 0.009611 s + 0.09962
gb1= ------------------------------------
          s^2 + 1.096 s + 0.09962
rsb2=modred(sb,3,'del');
gb2=tf(rsb2)
      -0.007277 s + 0.09648
gb2= ----------------------
     s^2 + 1.062 s + 0.09655
```

The behaviour of all the models is rather similar, but differences may be detected at given frequencies, in particular the static and instantaneous gains. For comparison, the norm (maximum difference in frequency-response) of the modelling error can be calculated:

```
norm(s-g1,inf)=0.0091, norm(s-g2,inf)=0.002
norm(s-g3,inf)=0.0011, norm(s-gb1,inf)=norm(s-gb2,inf)=7.25e-04
```

Example 3.23. Let us now consider a simple academic MIMO system, to illustrate the reduction of "common" poles hidden by the approximation implied in an experimental identification. In the example below, everything is clear by just looking at the transfer matrix, but the approach could be applied with similar success to models not so evident, as the balanced reduction detects that situation autonomously.

Assume an experimental model, $G(s)$, and its approximation:

$$G(s) = \begin{bmatrix} \frac{1}{s+1} & \frac{2}{s+2} \\ \frac{2.1}{s+1.1} & \frac{4}{s+2.1} \end{bmatrix}; \qquad G_a(s) = \begin{bmatrix} \frac{1}{s+1} & \frac{2}{s+2} \\ \frac{2.1}{s+1} & \frac{4}{s+2} \end{bmatrix} \approx G(s)$$

pointing out that the system could be, in fact, approximated by a second-order system, the two outputs being proportional.

If we type in MATLAB® an immediate realisation of G (A,B,C,D):

```
A=[-1 0 0 0;0 -2 0 0;0 0 -1.1 0;0 0 0 -2.1];
B=[1 0;0 2;2.1 0;0 4];C=[1 1 0 0;0 0 1 1];
```

and apply the following sequence of commands:

```
s=ss(A,B,C,0);
g=tf(s);
sb=balreal(s);
drsb=modred(sb,[3 4],'del'); drg=zpk(drsb)
```

the second-order reduced model is:

```
   TF from input 1 to output ...       from input 2 to output ...
       1.038 (s+2.075)                     2.0383 (s+1.077)
#1:  -------------------          #1:   ------------------
     (s+2.077)(s+1.078)                  (s+2.077)(s+1.078)
```

```
       2.0796 (s+2.075)                      3.979 (s+1.077)
#2:  -------------------              #2:  -------------------
     (s+2.077)(s+1.078)                    (s+2.077)(s+1.078)
```

This model keeps the high-frequency behaviour of the original plant (check it by plotting `sigma(g-drg)`), but not the DC gain. Note that, as an alternative, we may keep the DC gain assuming instantaneous behaviour of the to-be-deleted subsystem (*residualisation*), yielding in this case a nonzero D matrix in the state space representation. The approximated model would be then more complicated, with an additional zero at each element of the transfer matrix:

```
 rsb=modred(sb,[3 4]); rg=zpk(rsb)
   TF from input 1 to output...          from input 2 to output...
     -0.037152 (s+2.073)(s-29.01)          -0.018918 (s+1.076)(s-109.8)
#1:  ----------------------------    #1:  ----------------------------
           (s+2.076)(s+1.076)                    (s+2.076)(s+1.076)

     0.019995 (s+2.073)(s+102.9)           0.010401 (s+1.076)(s+380.4)
#2:  ----------------------------    #2:  ----------------------------
           (s+2.076)(s+1.076)                    (s+2.076)(s+1.076)
```

The norm of the modelling error is 0.047. The DC gain has maximum gain 3 and minimum gain 0.001, so if the modelling error were in the direction of the lower DC gain associated vectors, the simplification might not be appropriate (the plant is severely ill-conditioned at steady-state). Residualisation (keeping DC gain) in this case seems more appropriate, as the first truncated model has 10 times more gain in the lowest direction than that of the real plant G[11].

3.11 Key Issues in MIMO Systems Analysis

In order to characterise the behaviour of a controlled plant, as well as to initially validate a control system design, a bunch of analysis tools should be available. The goal of these tools should be not only to evaluate a given system but also to provide rules of thumb about how to improve some desired performances or to avoid some constraints' violation. A key concept introduced at the very beginning is that the control system cannot make "miracles": in the end, the control system is selecting the best action among the possible ones to reach some goals. If the process is not capable of performing in the desired way, no controller will do the job.

Thus, a crucial part of the analysis of a controlled system is the performance limitations. This could recommend us to think about process

[11] In most cases, keeping the DC gain is *not* necessary to get a good approximate model for closed-loop control, as low-frequency errors may be easily compensated for by integral action. Other frequency ranges are indeed more important, as discussed in Chapter 8.

changes before the control design is initiated, including the integrated process-controller design. This is nowadays a hot topic, being an active subject of control research and process design.

Structure evaluation. Structural properties are among the main factors in limiting the achievable performances. This has a lot to do with the selection of variables. The addition of a sensor or actuator or the appropriated pairing of variables, a matter to be discussed in the next chapter, may improve the structure of the process, allowing for better controlled performances.

We must consider the process structure and model from the control viewpoint. It is worth recalling that a good model for control does not mean a good model for other purposes, for instance, to predict the process output.

Gains. Directionality is also a fundamental issue in MIMO systems. It is sometimes surprising that systems behaving properly when a single loop is analysed present an unacceptable behaviour under simultaneous changes in the whole system. The advantages and drawback of the interaction should be taken into account to look for the appropriated control structure.

Performance evaluation. The worst-case condition should be considered to evaluate the performances of a MIMO system. Even without uncertainty in the model, unexpected behaviour may appear for some input combination.

3.12 Case Study: Simple Distillation Column

In the column shown in Figure 3.4, with lateral extractions, the concentration of the extracted flows (D, D_1 and D_2) is controlled by manipulating their flows, the concentration of the (constant) inlet flow, d_F, acting as a disturbance.

This process, with the particular selection of input and output variables, presents an interesting triangular structure. Note that if the bottom product extraction flow is also manipulated, strong interaction will appear.

Based on an experimental modelling run, the following transfer matrices have been approximated:

- output concentration increments *versus* extracting flows changes (transfer matrix $G(s)$):

$$\Delta Y(s) = \begin{bmatrix} \frac{0.7e^{-3s}}{18.5s+1} & 0 & 0 \\ \frac{5e^{-1.23s}}{14.6s+1} & \frac{0.58}{10.3s+1} & 0 \\ \frac{0.25}{9.4s+1} & \frac{3.65}{14.25s+1} & \frac{1.65}{9.8s+1} \end{bmatrix} (-\Delta U(s))$$

- concentration increments of outlet flows *vs.* that of inlet flow:

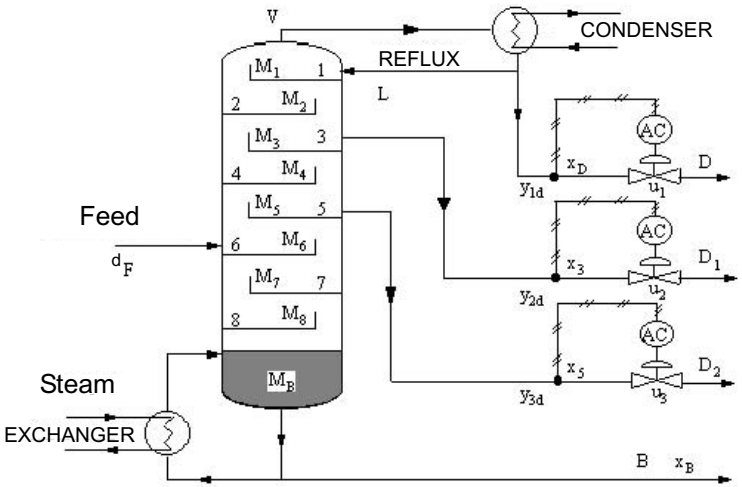

Figure 3.4. Distillation column

$$G_d(s) = \frac{\Delta Y(s)}{\Delta d_F(s)} = \begin{bmatrix} \frac{0.6e^{-1.6s}}{18.3s+1} \\ \frac{5.2}{15s+1} \\ \frac{20}{9.5s+1} \end{bmatrix}$$

a) Reduced minimal DT realisation. For control purposes, we would like to get a DT reduced-order model, but it should be validated comparing the time-responses. Thus, the following actions are taken:

1. Reduce the transfer matrix elements, using the following time constants for the approximate transfer functions: $\tau = 10$, 14.6, 18.4 s.
2. Use a sampling period $T = 1.5$ s, and get the ZOH equivalent DT transfer matrix, $G(z)$.
3. Obtain a minimal realisation (A, B, C, D) using as many physical variables as possible.
4. Compare the CT full model step response with that of (A, B, C, D).

1. Model reduction

This rough approximation, also considering delays that are multiple of the applied sampling period, will lead to:

$$\tilde{G}(s) = \begin{bmatrix} \frac{0.7e^{-3s}}{18.4s+1} & 0 & 0 \\ \frac{5e^{-1.5s}}{14.6s+1} & \frac{0.58}{10s+1} & 0 \\ \frac{0.25}{10s+1} & \frac{3.65}{14.6s+1} & \frac{1.65}{10s+1} \end{bmatrix}; \quad \tilde{G}_d(s) = \begin{bmatrix} \frac{0.6e^{-1.5s}}{18.4s+1} \\ \frac{5.2}{14.6s+1} \\ \frac{20}{10s+1} \end{bmatrix}$$

2. Discretisation

Element-by-element discretisation using, for instance, the MATLAB® command `c2dm` with the `zoh` method, yields the transfer matrices:

$$G(z) = \begin{bmatrix} \frac{0.0548}{z^2\,(z-0.9217)} & 0 & 0 \\ \frac{0.4882}{z\,(z-0.9217)} & \frac{0.0808}{z-0.8607} & 0 \\ \frac{0.0348}{z-0.8607} & \frac{0.3564}{z-0.9024} & \frac{0.2298}{z-0.8607} \end{bmatrix}; \quad G_d(z) = \begin{bmatrix} \frac{0.0470}{z\,(z-0.9217)} \\ \frac{0.5077}{z-0.9024} \\ \frac{2.7858}{z-0.8607} \end{bmatrix}$$

3. Minimal (physical) realisation

To get a realisation we can basically follow two different approaches:

i) To analyse the system's structure and detect row (column) common modes to be attached to the same output (input) variable.
ii) To get a partial realisation of each element, add altogether and reduced afterwards.

We will follow the first approach, to keep an eye on the physical meaning of the variables. In this way, the block-diagram depicted in Figure 3.5 is obtained. We have considered a common pole at $(z - 0.8607)$ attached to output y_3, as well as a delay on input u_1 dealing with y_1 and y_2.

From this diagram, Figure 3.5, a state variable may be assigned to each first-order block, leading to a seventh-order model. Using the notation $x_k = x(k)$, it yields :

$$x(k) = \left[\, x_1(k)\; x_2(k)\; x_3(k)\; x_4(k)\; x_5(k)\; x_6(k)\; x_7(k) \,\right]^T$$

The full model, including the disturbance, can be expressed by the following state equations:

$$\begin{cases} x_1(k) = u_1(k-1) \\ x_2(k) = x_1(k-1) + 0.8571 d_F(k-1) \\ x_3(k) = 0.0548 x_2(k-1) + 0.9217 x_3(k-1) \\ x_4(k) = 0.4882 x_1(k-1) + 0.9024 x_4(k-1) + 1.0400 d_F(k-1) \\ x_5(k) = 0.8607 x_5(k-1) + 0.0348 u_1(k-1) + 0.2298 u_3(k-1) + 2.7858 D_F(k-1) \\ x_6(k) = 0.8607 x_6(k-1) + 0.0808 u_2(k-1) \\ x_7(k) = 0.9024 x_7(k-1) + 0.3564 u_2(k-1) \end{cases}$$

the output equations being:

$$\begin{cases} y_1(k) = x_3(k) \\ y_2(k) = x_4(k) + x_6(k) \\ y_3(k) = x_5(k) + x_7(k) \end{cases}$$

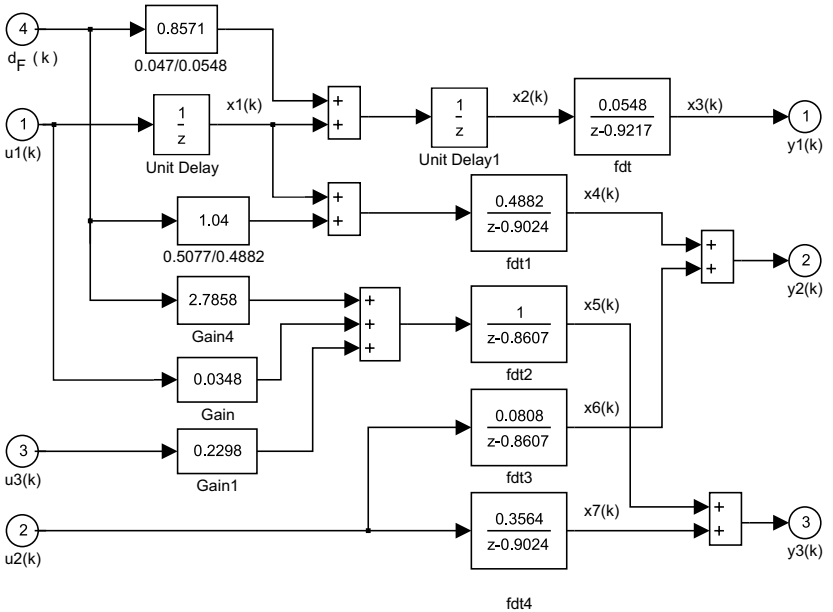

Figure 3.5. Approximate DT model

The disturbed model, Σ_d, is $x_{k+1} = Ax_k + Bu_k + Fd_F$, $y_k = Cx_k$, with:

$$; A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0548 & 0.9217 & 0 & 0 & 0 & 0 \\ 0.4882 & 0 & 0 & 0.9024 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.8607 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.8607 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.9024 \end{bmatrix}$$

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0.0348 & 0 & 0.2298 \\ 0 & 0.0808 & 0 \\ 0 & 0.3564 & 0 \end{bmatrix}; \quad F = \begin{bmatrix} 0 \\ 0.8571 \\ 0 \\ 1.0400 \\ 2.7858 \\ 0 \\ 0 \end{bmatrix}; \quad C = \begin{bmatrix} 0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,1\,0 \\ 0\,0\,0\,0\,1\,0\,1 \end{bmatrix}$$

4. Open-loop model validation

In order to check the model approximation, the step response of the "real" plant, that is, the model $G(s)$, and the DT approximate model are compared. In Figure 3.6, the output responses to a simultaneous unit step change in the three inputs are plotted for both systems, assuming no disturbances. Although

the ZOH discretisation preserves the step response, the small acceptable differences shown in the plots are due to the assumed approximations in the model parameters.

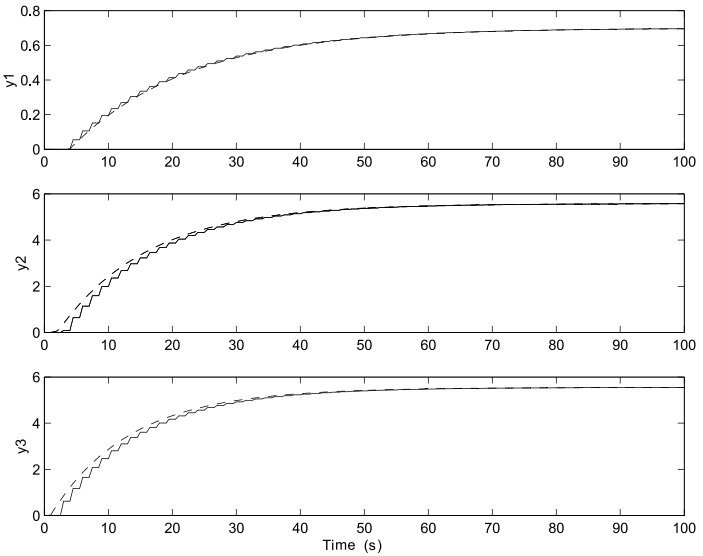

Figure 3.6. Step response of the CT and the approximate DT column model

b) Structural analysis. Mainly as a matter of training and analysis, we can study the reachability and observability conditions for the Σ_d model. As an example, let us consider the following options:

1. Determine the lower number of concentration sensors and their placement to detect the three controlled variables.
   If we compute the observability matrix rank for each of the output vector components, that is, the observability of the pairs $(A,_i C)$, we will realise that no output can observe the full state, and thus, we need more than one output to observe the three controlled variables. In fact, this can be easily deduced by just looking at the block-diagram in Figure 3.5, concluding that we need three sensors.
2. To counteract a disturbance, Δd_F (unitary step), compute the control sequence (extraction flows) to get back to the initial conditions:
   i) In the shortest time.
   ii) With the lower variation in the control flows, and a sequence of n (system order) actions.

   Let us discuss in some detail these two disturbance-rejection alternatives.
   i) For the first requirement, the reachability index should be computed:

Build up the reachability matrix of the pair $(A,\ B)$:

$$\begin{aligned} Co &= \left[\, B\ AB\ A^2B\ A^3B\ A^4B\ A^5B\ A^6B \,\right] \\ &= \left[\, b_1\ b_2\ b_3\ Ab_1\ Ab_2\ Ab_3\ A^2b_1 \cdots A^6b_1\ A^6b_2\ A^6b_3 \,\right] \end{aligned}$$

with $m \times n = 3 \times 7 = 21$ columns.
As its rank is 7, let us form a partial matrix taken the first 7-independent columns:

$$Co_1 = \left[\, b_1\ b_2\ b_3\ Ab_1\ Ab_2\ A^2b_1\ A^3b_1 \,\right]$$

which yields the following conclusions:

- u_1-reachability index: $\mu_1 = 3$,
- u_2-reachability index: $\mu_2 = 2$,
- u_3-reachability index: $\mu_3 = 1$,
- system reachability index: $\mu_{Co} = \max(\mu_1, \mu_2, \mu_3) = 3$.

Thus, the state reachability time is 4, and the control actions to be applied are:

$$U_{Tmin} = \left[\, u_1(3)\ u_2(3)\ u_3(3)\ u_1(2)\ u_2(2)\ u_1(1)\ u_1(0) \,\right]^T$$

Assuming an initial state x_0 at the time the step disturbance is applied, the control sequence to drive the state to the equilibrium (origin) will be computed by[12]:

$$U_{Tmin} = Co_1^{-1} \left(x(4) - A^4x(0) - \left[\, F\ AF\ A^2F\ A^3F \,\right] \begin{bmatrix} d_F(3) \\ d_F(2) \\ d_F(1) \\ d_F(0) \end{bmatrix} \right)$$

In particular, if $(d_F(k) = 1)$, and $(x(0) = x(4) = 0)$, the control will be:

$$U_{Tmin} = -Co_1^{-1} \left[\, F\ AF\ A^2F\ A^3F \,\right] \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -37.1775 \\ -\ 0.8571 \\ 0 \\ -\ 232.0935 \\ 249.2297 \end{bmatrix} = \begin{bmatrix} u_1(3) \\ u_2(3) \\ u_3(3) \\ u_1(2) \\ u_2(2) \\ u_1(1) \\ u_1(0) \end{bmatrix}$$

the control effort being:

$$E_{Tmin} = \sum_{k=0}^{3} \sum_{i=1}^{3} \left(u_i(k)\right)^2 = 1.17365 \times 10^5$$

[12] Equation (3.41) must be suitably modified by moving the effect of disturbances (non-manipulated inputs) to the right-hand side.

ii) Minimum control effort (in $n = 7$ intervals)
In this case, the system evolution is given by:

$$x(7) = A^7 x(0) + \begin{bmatrix} B & AB & \cdots & A^6 B \end{bmatrix} \begin{bmatrix} u(6) \\ \vdots \\ u(1) \\ u(0) \end{bmatrix} + \begin{bmatrix} F & AF & \cdots & A^6 F \end{bmatrix} \begin{bmatrix} d_F(6) \\ \vdots \\ d_F(1) \\ d_F(0) \end{bmatrix}$$

The pseudoinverse ($Co^\dagger$) provides the minimum norm solution for the control sequence. Thus, for $x(0) = 0$:

$$U_{Emin} = Co^\dagger \left(x(7) - A^7 x(0) - \begin{bmatrix} F & AF & \cdots & A^6 F \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \right)$$

$$U_{Emin} = \begin{bmatrix} u_1(0) & u_2(0) & u_3(0) \\ u_1(1) & u_2(1) & u_3(1) \\ u_1(2) & u_2(2) & u_3(2) \\ u_1(3) & u_2(3) & u_3(3) \\ u_1(4) & u_2(4) & u_3(4) \\ u_1(5) & u_2(5) & u_3(5) \\ u_1(6) & u_2(6) & u_3(6) \end{bmatrix} = \begin{bmatrix} 82.6985 & 0 & -6.4345 \\ 51.5983 & 0 & -7.4758 \\ 13.7274 & 0 & -8.6857 \\ -31.9367 & 0 & -10.0913 \\ -86.5520 & 0 & -11.7244 \\ -0.8571 & 0 & -13.6218 \\ 0 & 0 & -15.8263 \end{bmatrix}$$

the control effort (much lower than before) being:

$$E_{Emin} = \sum_{k=0}^{6} \sum_{i=1}^{3} \left(u_i(k) \right)^2 = 1.9049 \times 10^4$$

Note that the second input is not used at all.

Gains. It is clear that the instantaneous gain is null and the static gain is given by

$$\bar{G} = \begin{bmatrix} 0.7 & 0 & 0 \\ 5 & 0.58 & 0 \\ 0.25 & 3.65 & 1.65 \end{bmatrix}$$

4

Solutions to the Control Problem

This chapter deals with a description of the solutions to the control problem available to the designer. Most of these possibilities will be detailed in subsequent chapters, but the objective of the chapter is to outline the context and the practical possibilities.

4.1 The Control Design Problem

A basic and obvious idea, key conclusion from the previous chapter, should be clarified from the beginning, if a control system is designed to control a given process:

> *The process should be capable of behaving in the required way. The best control subsystem will just provide the most suitable inputs to the process to fulfill the goals.*

This has two main consequences:

1. If capabilities required to meet some goals are not built into the process, the control cannot achieve them.
2. The control system is "selecting" the best input, among the possible ones, to fulfill the goals.

Based on this evidence, there is a tendency nowadays to integrate the design of the process and its control.

For instance, in designing the control of a multiple-stage reactor process, the integral design of the control of the whole system, as well as forcing the interaction among stages, may reduce the control effort and achieve much better performances. A good example of this can be seen in [116].

Usually, a control system should be designed to work together with an already existing process. The control design problem can be stated at local, supervisory or even plant-wide level. Some ideas about the higher levels will

be discussed in the last chapter. If we consider the local level, the usual steps in designing the control are:

1. Define which are the components of the process to be controlled, which equipment parts are manipulable and which are fixed. Also define which are the variables of interest.
2. Define the user control goals.
3. Get a draft model of the process and the attached signals.
4. Select which will be the manipulated and measured variables.
5. Choose a suitable *control structure*.
6. Translate into a control language (also appropriate for the selected control structure) the user requirements.
7. Apply the controller design methodology based on the decisions taken in the previous steps (variables, models, goals).
8. Validate (by simulation or experimentally) the design, and tune the controller parameters.
9. Define the controller implementation. In the case of digital controllers, select the hardware and software to fulfill the control requirements.
10. Install the control in the process.
11. Evaluate the controlled system performances.

Most of these activities should be done iteratively. For instance, if the designed controller does not match the requirements, either the control structure or the variables selection should be revisited. If the implementation of the controller introduces additional constraints (time delays, computation time, resources availability, *etc.*), the design should be reconsidered, taking into account the new requirements. If the controlled system is driven out of the region of validity of the model, a new model should be obtained or the uncertainties should be reduced.

In the previous chapters, the modelling and analysis issues have been detailed. This can be done with a rough idea about the control goals. Nevertheless, once the user control goals have been stated, some decisions should be made to design the controller.

- which variables are selected as controlled, measured and manipulated and which ones are going to be treated as disturbances?
- based on this selection, which are the achievable performances?
- to meet the control requirements, which is the most suitable control design approach?

4.2 Control Goals

From a set of usually ambiguous "user" control goals, mainly based on qualitative and economical requirements as well as operational constraints, the desirable controlled plant performances for "control design" should be derived. They may concern different properties, such as:

- reference tracking, to follow changes in the set-points or references,
- control decoupling, to better understand and tune the different subprocesses or control variables,
- disturbance rejection, to cope with non-manipulated external variables,
- measurement noise rejection, to be able to use "imperfect" sensor and transmission systems,
- robustness against changes in the plant (model) or expected disturbances.

Some of these goals may be contradictory so this is a multi-criteria decision problem. A suitable trade-off is the most we can achieve.

Tracking performance. The classical performance measures in reference tracking, such as overshoot, settling time, bandwidth, cut-off frequency or resonance peaks, among many others (see the classical books of Franklin [49], Kuo [78], *etc.*), should be taken with caution in MIMO systems. The main concern is the gain *directionality*. An adequate response of a controlled variable if two inputs are acting *sequentially* can become unacceptable (for instance, exhibiting significant inverse response) if both are applied *simultaneously*. This is the effect of the process interactions.

Also, the usual conclusions given by the pole location, the basis of the pole assignment control design technique, should be treated carefully because, as we have seen, the system poles and zeros are not the poles of a given input/output relationship.

Control decoupling. The previous concerns ask for some kind of decoupling if generalisations of SISO criteria are to be applied. If the process is rearranged in such a way that each input (or block of inputs) only influences one output, the controlled process structure is simplified and it can be analysed as composed by m subprocesses. This is the main advantage, as the controllers can be designed and tuned independently. We will see in the next chapter some techniques to achieve this decoupling.

But the interactions are natural in many processes and they can work in the right way to control the process. In a general case, it seems wasteful to use some control effort in decoupling the process and then requiring an additional effort to control the subprocesses if it is in the opposite direction.

Diagonal dominance was a concept introduced for gain matrices, but it can also be extended to transfer matrices. In this case, the dominance can be established from different points of view. Other than the gain (in steady-state or at a critical frequency), it is interesting to consider the delays and/or the controllability of each output/input pair, as analysed in the next section.

Triangularity of transfer matrices is also a convenient property allowing the design of the controllers in sequence, considering the former ones as generators of disturbances for the later ones.

Disturbance rejection. We should consider two different types of disturbances:

Process disturbances are those modifying the internal process variables. These should be counteracted by modifying the manipulated control inputs. Thus, it would be convenient to know them in advance (in some cases, the reference can be considered as such a disturbance, being known by the user). Otherwise, we will be interested in measuring them or, at least, in estimating their current value. This will allow us to generate the extra control signals. *Input disturbances* are a particular case, acting at the process input. They are the easiest to be cancelled once their value has been measured or estimated.

Measurement noise is a disturbance which only has effect on the process if it is fed back by a controller. Thus, in this case, the best option is to filter it to avoid it acting as a disturbing input to the controlled process. *Inferential control* (Section 5.5.1) is a modern control technique generating the control action from estimated "noise-free" measurements. Of course, they cannot cope with the other kind of disturbances or model uncertainty.

Robustness. Model uncertainty has been the big issue in modern control design techniques. A good model-based control system could become useless if the plant behaviour strongly differs from that of the model. Thus, model uncertainty should be assessed and the designed loop should behave appropriately, taking into account the uncertainty when designing the controller.

Again, in the MIMO systems, interactions should be taken into account. A control loop may be robust to uncertainties in the elements of different subprocesses, if they appear separately, but drastically losing its properties if the uncertainties appear *simultaneously* in different parts of the process. *Robust control* issues will be the subject of a later chapter.

4.3 Variables Selection

In developing a model of a process, we may consider a very large number of external and internal variables, all of them related to the partial phenomena we are interested in: hydraulic, thermal, concentration, movement, energy and so on. But, certainly, we do not need to control all of these and we do not need either to measure or to manipulate every one. That means, a selection of variables should be done.

Three selections must be carried out:

- selection of controlled variables,
- selection of measurements,
- selection of manipulated variables.

System analysis knowledge may help in some cases to take decisions based on a full-size model. However, its usefulness is limited, and these selections should be also guided by experience and common sense.

Controlled variables. Ideally, the controlled variables must verify the following requirements:

1. They must relate to the primary user goals.
2. Primary goals should not be excessively dependent on small deviations of them (otherwise, small controller errors will unacceptably degrade overall objectives).
3. Its optimal set-point must not depend on the value of exogenous disturbances (to avoid designing a "good" controller following a "wrong" set-point).
4. Disturbances must cause small deviations on them (so low control "intensity" is needed).
5. Process state and input/output controllability are sufficient to actually design satisfactory controllers for them.

Some of the ideas in Section 5.5.1 are based on the above issues. If a set of variables fulfilling these requirements cannot be found, the process is "difficult to control" and redesign may be the wisest choice.

Example 4.1. [119] In an oven, setting the heating power as controlled variable (manipulating the supply voltage) is not a wise choice, as its optimal set-point is heavily dependent on factors such as heat loss, oven size, *etc.*, whereas the oven temperature is a much better choice.

State Representation

Let us assume a state space description of the plant to be controlled. As we have seen in the last chapter, some internal variables may be deleted by reducing the model order. The order may have been reduced to achieve a minimal realisation and to eliminate the state variables with reduced relevance into the (input)-to-(controlled variables) dynamics. In the following, it is assumed that fundamental control objectives may be approximately cast as objectives on the state variables.

Measurements. Note that controlled variables and measured variables are not the same thing. The issue now is to consider which of the available sensors (combinations of state variables) should be used.

This requires an analysis of the output equation, (2.11) or (2.19). Of course, the pair (A, C) should be observable. The best would be to measure all of the state ($C = I$), but this can be unfeasible (there is no sensing device for some variables or they are not reachable) or too costly. From this viewpoint, the best sensing option would be the set of outputs so that the degree of observability is maximum. This means minimum observation time and maximum detectability. This will come out from an analysis of the observability matrix. The measurement conditions and, as a result, the measurement noise, could be an issue to include or delete a given sensor. For the above results to be significant, matrix C should be scaled so that all sensors have comparable noise amplitude.

Actuators. For simplicity and economical reasons, the number of manipulated variables should be as low as possible. This requires us to analyse the state equation and, particularly, the reachability of the pair (A, B). Again, the maximum degree of reachability would be required, with the minimum number of control inputs. But, in this case, we must keep in mind that these control inputs should be also able to counteract the rest of external variables, that is, the disturbances. Thus, the generalised state model (2.21) should be taken into account.

Input/output properties

As a further consideration, not all the states could be equally relevant in our control goal. Thus, other than the input-state-output relationship, the direct input/output relation should be considered. Based on the analysis of the transfer matrix, we can deduce the more or less close connection of the outputs to the inputs. And we must consider two issues: time and gain.

Desirable input/output relationships must have simple dynamics. A control action (or a measurement) will be less interesting as far as the delay it introduces increases. For that purpose, let us define the *output reachability-time matrix*

$$\mathbf{R} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ r_{p1} & r_{p2} & \cdots & r_{pm} \end{pmatrix}; \quad {}_iC * A^r * B_j \begin{matrix} \neq 0 & r = r_{ij} + 1 \\ = 0 & r \leq r_{ij} \end{matrix} \tag{4.1}$$

This matrix represents the input/output distance, in time. It is also denoted as the *relative degree matrix*, because its elements are the relative degree of the transfer function elements of the transfer matrix.

In order to consider the magnitude, other than the static and instantaneous gains, the *characteristic matrix* defined by:

$$\mathbf{J} = \begin{pmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1m} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_{p1} & \gamma_{p2} & \cdots & \gamma_{pm} \end{pmatrix}; \quad \gamma_{ij} =_i C * A^{r_{ij}+1} * B_j \tag{4.2}$$

represents the first action for each input/output pair. These ideas are helpful mainly in decentralised control and decoupling (Section 5.3).

Feedforward disturbance rejection. As previously mentioned, disturbance variables are also candidates to be measured, if possible, to allow the fast generation of the counteracting control actions.

Transfer Matrix: Extreme Gains and Variables Relevance

The I/O indicators just presented (characteristic and relative order matrices) only provide partial information, related to high-frequency behaviour. To decide about the selection of variables, the frequency-dependent directional gains give interesting and complementary information. Again, these gains are sensitive to scaling. The variables should be normalised, according to scaling remarks in Section 3.5, to get comparable and relevant information.

Recall that the SVD of a matrix, Section 3.5.3, provides the maximum, ($\bar{\sigma}$), and minimum, ($\underline{\sigma}$), gains, and its ratio (the condition number, (3.36), $\gamma = \bar{\sigma}/\underline{\sigma}$). From the frequency response matrix of a process, these gains may be computed as a function of the frequency.

Let us assume a process with over-dimensioned input and output vectors so that a selection of a subset of them must be carried out.

Actuators. First, let us consider the transfer matrix in $z = G_z u$, where z are p_z controlled variables and u is a candidate actuator set. Carrying out the SVD, $G_z = U\Sigma V^H$, the first p_z columns of matrix V are the combinations of manipulations with highest effect on the control objectives, so if a particular actuator u_k has no significant components v_{ik} on them, it may be discarded. Note that as V may be frequency-dependent, there exists the possibility of needing one set of actuators for control in one frequency range and a different set for a different frequency range. That possibility is, indeed, not unusual in practice, as cascade control examples in Section 5.4.2 point out.

Measurements. In sensor selection, the key issue is how measurement noise and disturbances corrupt a, perhaps indirect, measurement of the controlled variables. A detailed treatment of some of the issues involved appears in Section 5.5.1. Assuming all sensors have been scaled so that measurement noise is similar, as $y = Gu = GG_z^{-1}z$, the SVD decomposition of $GG_z^{-1} = U\Sigma V^H$, will point out which sensors (G over-dimensioned) better "measure" the controlled variables: those will be the first columns of U. Note that this is a coarse criterion as the effect of process disturbances is not accounted for (for further details, see Section 5.5.1). The frequency-dependent nature of U may advise different sensors for different frequency ranges. Again, this is usual in practice, and it is behind some cascade-control strategies as well (Section 5.4.1).

Of course, the more sensors and actuators the better, but the more expensive the instrumentation is. A comparison with the magnitude of the unavoidable noise in the process will also indicate the uselessness of variables attached to lower singular values (low-gain actuators or sensors).

In many cases, it is wished that controlled variables and measured ones are the same. So, the issue is selecting a "good" subset of p inputs and outputs on the "big" G so that the "underlying" process is under control. This has close resemblance with the state space analysis previously discussed.

Conditioning. Apart from the issues of "gain" just discussed, sensitivity to uncertainty is also an issue: conditioning should be reasonable if high performance needs to be achieved. So, if the SVD is carried out for subsets of inputs and outputs, the selection of variables can be based on getting the minimum condition number (closer to 1), indicating a similar behaviour, and hence, difficulty to control, in any direction. And this can be done for static gains as well as for gains at some critical frequencies.

Based on the SVD techniques, the following guidelines can help in determining a good selection of variables:

- given a process with m_0 possible manipulated variables and p_0 measurable internal variables, carry out SVD of the gain matrix (it could be the static gain or that at a critical frequency) after a suitable scaling,
- determine the condition number, γ_0, and the lowest singular value, $\underline{\sigma}_0$. A high value for γ_0 indicates poor conditioning, and a low value of $\underline{\sigma}_0$ points out poor action in some directions,
- analyse the columns of U (V) and assign each one of them to the most weighted output (input) vector component. Any key output (input) component should be in this group,
- to compare different sets of input/output choices, γ (the lower the better) and $\underline{\sigma}$ (the higher the better) can be assessed.

This decomposition can be also used for system decoupling, as detailed in the next chapter. However, in a general system, there may be many (hundreds) of I/O combinations to check. Another useful variable selection tool, the relative gain array (RGA), is based on the interactions among variables and will be detailed in the next chapter. It can be used for a global preliminary screening, avoiding most of the unsuitable combinations [119].

Apart from the above considerations, the presence of RHP zeros, delays, *etc.* should be taken into account. Of course, the cost of the sensor/actuator alternative is an important factor. A wrong variable selection will limit the achievable performance and robustness of the final control scheme.

4.4 Control Structures

The selection of the variables to be used as control variables as well as the information used to generate the control actions will determine the control system structure. We must consider, at least, the following structures:

Open loop *vs.* closed loop. In an open-loop control structure, the control actions are generated based on external information: set-points or objectives, initial conditions, disturbances, operator data, and so on. A good model of the process is required and there is no option to cope with unexpected changes in the plant (either disturbances or plant changes). On the other hand, closed-loop control uses the information from the plant to generate the control. There

are many options for dealing with disturbances, reference tracking and uncertainties, but the big issue is the controlled plant stability (and performances). One obvious drawback of closed-loop control is the requirement of the existence of errors to act. Thus, a combination of both structures may allow better results.

Single/multiple loop. In feedback controlled MIMO systems, the vector of input actions may be computed altogether from the full set of measurements and available data or, alternatively, the information is split into blocks to determine each of the control actions. In this case, for each input, the remaining blocks of information can be considered as disturbances. This structure could be also denoted as centralised/decentralised control.

Two degree of freedom. The control action may be computed in two phases. First, the control error is evaluated and the control is based on the error. This is a feedback control action. Afterwards, an additional control action is computed based on the external inputs. There are two degrees of freedom to design the controller, and the design can be split to achieve tracking (references) and regulation (output feedback) performances.

Multi-level control. Groups of input (output) variables can be treated jointly to control a process variable. They will act locally, receiving commands (set-points) from higher decision levels and sending information back to these upper coordination levels.

4.5 Feedback Control

As previously mentioned, the main problem in closed-loop control is to design the elements of the controller to achieve the required closed-loop performances and, in any case, to assure the stability of the controlled plant.

In order to analyse the fulfillment of these properties, the relationship between different variables should be taken into account. For this purpose, with reference to Figure 1.3 on page 10 (assuming, for simplicity, $F = I$), the output of the controlled plant, under negative feedback, can be expressed by:

$$\begin{aligned} y &= G(s)\,(u + d_u) + G_d(s)\,d \\ u &= K(s)\,e \\ e &= r - H(s)\,(y + n) \end{aligned}$$

where u_d is an input disturbance vector, K is the controller transfer matrix, n is a measurement noise and H is the measurement device dynamics, the measurement noise filter or the feedback controller (in a 2-DoF control structure with $F \neq I$, Section 4.7). Carrying out suitable matrix operations, the output vector can be expressed by:

$$y = (I + GKH)^{-1}(GKr + Gd_u + G_d d - GKHn)$$

or, equivalently, by suitable definitions of M, T and S (see below):

$$y = Mr + SGd_u + SG_d d + Tn \tag{4.3}$$

The control action is:

$$u = (I + KHG)^{-1}(Kr - KHGd_u - KHG_d d - KHn)$$

so, applying the push-through rule (B.5):

$$u = (I + KHG)^{-1}Kr - KHS(Gd_u + G_d d + n) \tag{4.4}$$

Thus, the following matrices are relevant:

1. Loop transfer matrix: $L = GKH$, which determines the loop stability if there are no pole-zero cancellations in the three factors (see next section).
2. Sensitivity matrix

   $$S = (I + L)^{-1} = (I + GKH)^{-1} \tag{4.5}$$

   which defines the effect of the feedback on the general disturbances, d.
3. Complementary sensitivity matrix

   $$T = (I + GKH)^{-1}GKH \tag{4.6}$$

   defining the effect of the feedback in the noise rejection. It is denoted as complementary sensitivity because, from (2.55):

   $$T(s) + S(s) = I \tag{4.7}$$

4. Reference tracking matrix: $M = (I + GKH)^{-1}GK$, allowing to analyse the output/reference properties.
5. Input-gain matrix: KHS, expressing the effect of disturbances and measurement noise on the control action.

In the following, $H = 1$ is assumed, the matrices above being simplified to:

$$L = GK; \quad S = (I + GK)^{-1}; \quad T = M = (I + GK)^{-1}GK \quad S + T = I \tag{4.8}$$

4.5.1 Closed-loop Stability Analysis

Internal stability. From (4.3) and (4.4), the stability of the closed-loop system requires that the **four** transfer matrices: S, SG, KS and KSG have no RHP poles[1] (assuming, of course, no unstable uncontrollable/unobservable modes in K or G). In that way, y and u are bounded for bounded d, d_u, n, r. If this property holds, the loop is said to be *internally stable*.

[1] Note that G_d is stable (from the assumption of bounded disturbances) and, as $S + T = 1$, T is stable if and only if S is stable. Furthermore, regarding the reference term in (4.4), from (B.5), $(I + KG)^{-1}K = KS$.

State space representation. Assume a plant $G := (A, B, C)$ without input/output direct coupling, $D = 0$, in feedback connection with a controller $K := (A_c, B_c, C_c, D_c)$. The closed-loop system model is given by (2.51). So, its stability will be determined by the poles of the closed loop system matrix

$$\bar{A} = \begin{pmatrix} A - BD_cC & BC_c \\ B_cC & A_c \end{pmatrix}$$

Alternatively, the open-loop L is the series connection GK, with a state space representation $(A_L, B_L, C_L, D_L = 0)$ given by (2.47). Connecting by negative feedback, it is easy to show that:

$$\bar{A} = A_L - B_L C_L$$

To calculate the eigenvalues, using Schur's formula (B.4) twice, we can express:

$$\det(sI - A_L + B_L C_L) = \det \begin{pmatrix} I & -C_L \\ B_L & sI - A_L \end{pmatrix} = \det(sI - A_L) \det(I + C_L(sI - A_L)^{-1} B)$$

Thus, the following interesting property is derived:

$$\frac{\det(sI - \bar{A})}{\det(sI - A_L)} = \det(I + L(s)) \tag{4.9}$$

So, if there are no common poles in open and closed loop, the closed-loop poles are determined by the solution of

$$\det(I + L(s)) = 0$$

$I + L(s)$ is denoted as the *return difference* operator: it must have no RHP zeros for loop stability.

RHP cancellations. It can be shown, based on the above results, that the closed loop is internally stable if and only if there are no RHP pole-zero cancellations between G and K and $S = (I + L(s))^{-1}$ is stable. In this way, only *one* closed-loop transfer matrix must be checked, instead of four.

Frequency Domain

Nyquist criterion. In control, frequency domain techniques are mainly valuable because they allow us to determine closed-loop stability properties based on the analysis of the open-loop frequency response. The stability of a MIMO system can be analysed by the Nyquist plot of $\det(I + L(jw))$. In a simple way, we can say that if $L(s)$ is stable, the closed-loop system is stable if the Nyquist plot of $\det(I + L(s))$ does not encircle the origin. To use the same analysis as in SISO systems, we may plot the scalar function

$$l^*(jw) = \det(I + L(jw)) - 1$$

and check for encirclements of the point $(-1, 0)$.

The small-gain theorem. An intuitive result, quite useful in frequency domain stability analysis, is the small-gain theorem. If $L(s)$ is stable, the closed loop system is stable if

$$\|L(jw)\| < 1 \qquad \forall w$$

For linear systems, the result is easily obtained from the Nyquist criterion. It is very conservative but it can be used to ensure the stability of some special closed-loop configurations (see Chapter 8). A detailed discussion is available in Appendix C.4.

High-gain control. Note that for those frequencies with very high gain in the direct path, $|G(jw)K(jw)| \gg 1$, the sensitivity, (4.5), vanishes and the complementary sensitivity, (4.6), reaches unity, being $M(s) \approx H^{-1}(s)$. This implies, from (1.6):

- process model uncertainties are not so relevant with high-gain feedback control ($G(s)$ "disappears"),
- the reference tracking matrix is the inverse of the sensing matrix. That, is, it emphasises the relevance of the data acquisition system: *without quality sensors good control is not possible*,
- measurement noise fully affects output,
- the effect of the disturbances also vanishes.

So, it seems that high-gain controllers are a viable solution with good sensors available. However, apart from obvious saturation issues, the above argument is not totally right, and high gain is not a viable solution in most applications. In Section 8.2.2, the issue is discussed in more detail.

Remark 4.2. A rule of thumb dealing with SISO closed-loop systems is that the stability is degraded if the loop gain is increased or additional delay is introduced in the loop. This is true for almost any system, except for those called *conditionally stable* systems, which are only stable for a range of gain or phase lag[2]. Caution should be taken with these concepts in MIMO systems.

4.5.2 Interactions

Assume a SISO system with a well-designed closed-loop control, u_1-y_1, having other inputs and measured variables but not yet used for control. Now, another input is generated (by a second controller, u_2-y_2) to control one extra output, closing a second loop and forming a TITO system. If performance of the first loop remained unchanged, *i.e.*, if closing the second loop did not affect the input/output behaviour of the initial control pair, then there would be no interaction between these loops.

Unfortunately, this is *not* the usual case because the transfer matrix is not diagonal. If some variables of a process are independently controlled there is

[2] These are the so-called gain and phase conditionally stable systems, see [6].

an interaction among the different loops and the performances may be severely degraded.

Let us consider the simple case in a TITO feedback loop such as the one in Figure 1.3 on page 10 where the feedback controller is a diagonal one. If the diagonal elements, K_1 and K_2, are designed such that

$$T_1 = \frac{G_{11}K_1}{1 + G_{11}K_1}; \quad T_2 = \frac{G_{22}K_2}{1 + G_{22}K_2}$$

meet some requirements, the actual behaviour will be determined by:

$$T = [I + GK]^{-1}GK; \quad K = diag[K_1 \quad K_2]$$

In particular, the stability would be defined by the roots of:

$$\det(I + GK) = (1 + G_{11}K_1)(1 + G_{22}K_2) - G_{12}G_{21}K_1K_2$$

It is clear that, only in the case $G_{12} = 0$ or $G_{21} = 0$ (triangular plant), this condition is reduced to $\det(I+GK) = (1+G_{11}K_1)(1+G_{22}K_2)$ and the global stability is as designed for the single loops. In Section 5.2, this issue is further discussed.

4.5.3 Generalised Plant

In the 1980s, a general solution [44, 133] for a wide variety of control problems was found via the *linear fractional transformation* formalism, by formulating them in the generalised interconnection form in Section 2.7.2.

To achieve this, a controller can be connected to the plant via a set of sensed variables (either outputs or direct measuring of some of the disturbances) so that a transfer matrix K calculates a set of manipulated inputs to the process. Figure 4.1 depicts the idea in block-diagram form, where the actual plant and the control structure (information flow) and control objectives conform the generalised plant, P.

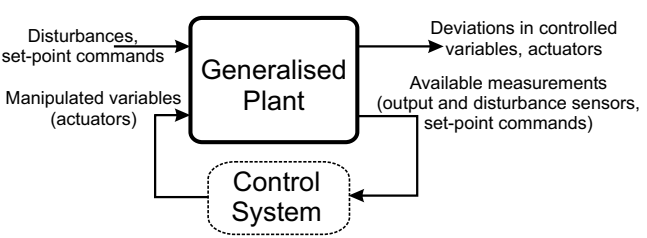

Figure 4.1. A general configuration of a control loop

If some of the regulator inputs are measurable disturbances or user-defined set-points, then the direct input/output coupling D_{12} of the generalised plant

is non-zero. This block-diagram is named in the literature as a *lower* linear fractional transformation. Note the similarity with Figure 1.2.

Some examples will help in grasping the ideas involved in the procedure of generating the generalised plant. Sections 7.4 and 8.6.1 detail some approaches to solving the control synthesis problem in this framework.

Example 4.3 (Open-loop control). The block-diagram in Figure 4.2 shows a MIMO open-loop reference tracking problem cast into the general configuration. From the figure, it is clear that the generalised plant has a transfer function matrix given by:

$$\begin{pmatrix} \text{errors} \\ \text{controller input} \end{pmatrix} = \begin{pmatrix} I & -G \\ I & 0 \end{pmatrix} \begin{pmatrix} r \\ u \end{pmatrix} \tag{4.10}$$

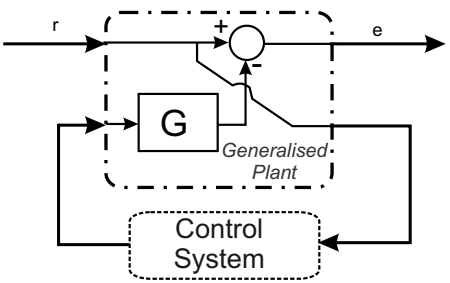

Figure 4.2. Open-loop control

Example 4.4 (Disturbance rejection: closed-loop control). The block-diagram in Figure 4.3 shows a MIMO disturbance-rejection problem with sensors measuring the disturbed output. The generalised plant has a transfer function matrix given by:

$$\begin{pmatrix} \text{errors} \\ \text{controller input} \end{pmatrix} = \begin{pmatrix} I & G \\ I & G \end{pmatrix} \begin{pmatrix} d \\ u \end{pmatrix} \tag{4.11}$$

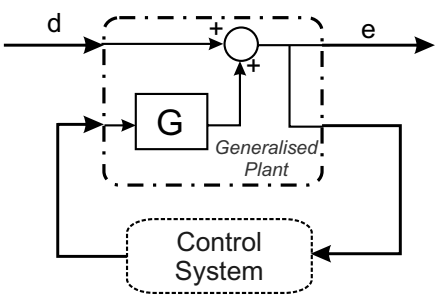

Figure 4.3. Closed-loop disturbance rejection.

4.5.4 Performance Analysis

There are several criteria for defining "good nominal performance" in terms of the closed-loop transfer matrices defined in this section, to be discussed below. In addition to that, tolerance of the solution to modelling errors should be assessed (see Chapter 8).

Settling time. It is a direct generalisation of the concept of SISO systems. In MIMO plants, the settling time for different input/output pairs may be different. It is related to the position of the system poles (see Appendix A) and its internal structure (controllability and observability).

Overshoot. This common SISO requirement is cumbersome when generalised to MIMO systems, as interactions cause deviations in all variables when subject to set-point changes or disturbances. Although it can be checked for particular input/output combinations, it is not usually used during the design phase, in favour of more tractable integral-squared-error criteria (Chapter 7).

Steady-state gain. Position errors for reference tracking are determined, in most loops, by the DC gain of the sensitivity function (4.5). For disturbance rejection, the steady-state gain of SG_d in (4.3) may be determined. To summarise all combinations, if the worst-case gain $\overline{\sigma}(S)$ (or $\overline{\sigma}(SG_d)$) is below a suitable bound, steady-state performance conditions are deemed satisfactory.

Bandwidth. Many control requirements can be cast in the frequency domain, in particular, ability to track references varying up to a maximum rate and ability to reject disturbances whose frequency components are mainly concentrated on a particular band. In particular, the *effective control band* is defined as the frequency band where the worst-case sensitivity $\overline{\sigma}(S)$ is *below* -6 dB, indicating a minimum attenuation of 50% of output disturbances or tracking with stationary error less than 50% (see Example 3.5). In many cases, good performance is required at zero frequency (DC position error), so reduced sensitivity is achieved from zero to a particular frequency ω_B: the *closed-loop bandwidth* is defined as the frequency where $\overline{\sigma}(S)$ crosses -6 dB.

For disturbance rejection, comparison of the worst-case open-loop response, $\overline{\sigma}(G_d)$, and the closed-loop one, $\overline{\sigma}(SG_d)$, will determine how effective control design has been[3]. Figure 7.4 on page 217 depicts such a situation in the context of a distillation case study.

Optimality. As previously commented, the different possibilities on allowed amplitude of actuator commands, allowable errors, amplitude of disturbances and set-points to be tracked, *etc.* pose a difficult problem if all of these requirements must be individually accounted for in the design process. However, they

[3] With a suitable scaling of the variables involved, approximate determination of satisfactory performance can be conveniently assessed by comparing a particular worst-case gain with 1. Details will be given in Section 8.6.1, after introducing many additional concepts needed.

may be approximately expressed as a cost index (sum of squares) so that optimisation can be carried out. Optimisation-based control is a powerful strategy with significant impact in practice. Chapter 7 is fully devoted to this issue.

4.6 Feedforward Control

Feedback control can compensate for disturbances and modelling errors if suitably designed. Furthermore, by interpreting the set-point changes as disturbances (see Section 6.1.2), with a suitable high-gain feedback, satisfactory reference tracking can be also achieved. In fact, the feedback-only solution is widely used in most control applications.

However, there might be circumstances where:

- the most significant disturbances are measurable, and
- non-measurable disturbances are small, and
- the plant is *stable* and a reasonably accurate *model* is available (perhaps even non-linear), and
- high performance is desired in either reference tracking or measurable disturbance rejection (or both),
- a particular key sensor (or group of them) is unavailable but a model to compute this variable is available.

In these circumstances, feedforward control can be a valid solution. In fact, the addition of feedforward strategies (see Section 4.7) in a closed-loop control is key to ensuring high-performance reference tracking as well as sufficient robustness to modelling errors (see Section 8.4.3).

Example 4.5. In process industry, pH control can have a strongly non-linear characteristic under common circumstances, due to the logarithmic nature of the measurement and valve non-linearities. Handling the problem by pure (linear) feedback control meets with many difficulties. In many cases in practice, the only way to achieve satisfactory disturbance rejection (with a reasonable buffer tank size, where the acid + base mixing is carried out) is to implement non-linear feedforward strategies based on measurements of the incoming flow and pH, perhaps as a part of a more complex structure incorporating feedback, such as gradual control (Section 5.5.4) and cascade control (Section 5.4).

4.6.1 Manual Control

The most elementary control problem is how to place the actuators to achieve a particular desired operating point in a set of output variables, y_d, for a stable plant[4].

[4] Errors in these calculations have to be compensated for (if sensors are available) by integral action in feedback controllers. Note that if set-point changes seldom occur, then the transient of the integral action (usually slow) will be irrelevant in practical terms. However, if set-point changes are frequent, then a slow correction of offsets might be a significant performance drain.

If a model (2.10) is available, reaching the desired output means solving for u the steady-state equations (derivatives equal to zero).

In the case of linear state space CT equations, the result is the DC gain, (2.59), $\bar{G} = -CA^{-1}B + D$, so $y_d = \bar{G}u^* = -(CA^{-1}B + D)u^*$ and hence:

$$u^* = \bar{G}^{-1}y_d = (-CA^{-1}B + D)^{-1}y_d \tag{4.12}$$

if the DC gain matrix is invertible[5].

If the plant is non-square and there are more sensors than actuators, then the desired point might be non-reachable (if y_d is not a combination of the columns of the DC gain matrix). Then, only a subset of the outputs must be selected, or the whole solution must be solved by least squares, using the *left pseudoinverse* in Appendix B.2, so $u^* = (\bar{G}^T\bar{G})^{-1}\bar{G}^T y_d$. If there are more actuators than outputs, the problem has infinite solutions so a subset of the actuators can be fixed or the configuration with minimum squared deviations can be also solved by least squares, using the *right pseudoinverse*: $u^* = \bar{G}^T(\bar{G}\bar{G}^T)^{-1}y_d$.

The most improvement in achievable performance appears when inversion of *non-linear* DC characteristics is implemented directly on the control software. In fact, its generalisation means inversion of *dynamic* non-linearities. In Section 9.5, simple illustrative examples of these techniques will be discussed.

Experimental determination of the DC gain. Even if a model is not available, performance gains can be achieved if, let us say, PID controllers are "helped" by suitably incrementing inputs when changing set-points, see (8.5). The needed increments can be obtained by using as DC gain the result of a step-response identification experiment. If the plant is non-linear, the output increments depend on the starting point. Eventually, its evaluation at different operating points gives a hint on the plant's degree of non-linearity.

Model-free manual control

Using manual control, by positioning the actuators in a constant setting a desired value appears at the process outputs of interest (of course, only in the absence of disturbances). Experimental determination of the DC gain matrix via step response will allow us, starting at a certain operating point, to determine the increments in actuator values to achieve the desired output increments. If the plant is non-linear, the output after the change in actuators will not be the desired one, but reasonably near, if the non-linearity is not very severe. In this case, the same DC gain matrix can be used to iterate a couple of steps to approximate the plant's output to its desired values.

Example 4.6. A mixing tank has two outputs (height y, concentration x), and two valves (V_A, V_S) acting as inputs. If, starting from an initial situation where both

[5] Note that, for instance, if A is singular, the DC gain does not even exist (the plant is unstable as singular A implies eigenvalues at $s = 0$).

valves are opened 30%, the outputs are (0.7 m, 50%), opening valve V_A to 40% gives, in steady-state, (1 m, 62%). Then, opening valve V_B to reach (40%, 45%) yields (1.35 m, 41%). In this case, a coarse approximation to the DC gain is:

$$\begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} \frac{0.3}{0.1} & \frac{0.35}{0.15} \\ \frac{0.12}{0.1} & \frac{-0.21}{0.15} \end{pmatrix} \begin{pmatrix} V_A \\ V_S \end{pmatrix} \tag{4.13}$$

So, a valve increment from the last situation to achieve a set-point (1.5 m,47%) can be calculated: $\Delta V = G^{-1}\,(0.15, 0.06)^T$. If, due to non-linearity, the set-point is not achieved, a second increment may be calculated with the same approximate gain.

4.6.2 Open-loop Inversion and Trajectory Tracking

The problem of obtaining a desired output can be formulated for a whole trajectory. In DT, if the desired output of a process with model $G(z)$ is $y_d(z)$, using the $\mathcal{Z}$-transform, the feedforward action, $u_{ff}(z)$, must verify:

$$y_d(z) = G(z)u_{ff}(z) \;\Rightarrow\; u_{ff}(z) = G(z)^{-1}y_d(z) \tag{4.14}$$

which is, for an arbitrary y_d, unfeasible as $G(z)$ contains delays. If $y_d(z)$ is the output of a reference model, $M(z)$, to an external input, *i.e.*, $y_d(z) = M(z)r(z)$, the open-loop regulator would be:

$$u(z) = G(z)^{-1}M(z)r(z) \tag{4.15}$$

where $M(z)$ must be selected to make $G^{-1}M$ feasible (proper, with reasonable maximum gain in all frequency ranges) and stable. Usually, G^{-1} should be obtained by means of symbolic computation software. If the original plant is non-minimum-phase, then G^{-1} will be unstable, unless M cancels the model's RHP zeros. An analogous derivation can be carried out for CT systems.

If the model is in state space form, its inverse realisation is given by:

$$\bar{x}_{k+1} = (A - BD^{-1}C)\bar{x}_k - BD^{-1}y_k; \quad u_k = D^{-1}C\bar{x}_k + D^{-1}y_k$$

Note that it cannot be inverted if D is singular.

Remark 4.7. The exact tracking of an arbitrary trajectory or reference model in discrete-time systems usually causes oscillations in the control variable. To avoid undesirable effects, in practical implementations, the control action should be filtered with a low-pass filter (including it in M), rejecting higher-frequency components that otherwise would excite high-frequency ranges of the plant (usually poorly modelled). It is also common to add a notch-filter at the Nyquist frequency $\omega_s/2$, such as the factor $0.5(1 + z^{-1})$.

4.6.3 Feedforward Rejection of Measurable Disturbances

In some cases, disturbances are measurable and a model of their effect is available, either from experimental data or from first-principle equations.

Example 4.8. Let us consider some common cases:

- in a tank level-control, feed pressures or flows of the incoming fluids can be measured and their effect determined from simple balance equations,
- in a neutralisation process (pH control), the incoming pH and flow can be measured,
- temperature control: measuring ambient temperature, solar radiation, detecting door/window openings, *etc.*,
- position control: knowing beforehand the mass of an object to be moved.

Availability of that information can contribute to improving the performance of a control system.

Let us have a model of the disturbance effect, for example by linearising first-principle equations, as (2.21):

$$\dot{x} = Ax + Bu + B_d d; \quad y = Cx \tag{4.16}$$

No direct coupling between the outputs and the inputs (d and u) is assumed, for simplicity, at this moment.

The simplest case is when the disturbances enter on the input channel (*i.e.*, there exists a matrix M such that $B_d = BM$). In this case, applying the control action:

$$u = -Md \tag{4.17}$$

the disturbances are cancelled, as $\dot{x} = Ax + B(-Md) + (BM)d = Ax$ is now independent of d.

To check if the input channel condition is fulfilled, the system of equations

$$Bu = -B_d d$$

should be solved for u. The usual situation is the order of the system, n, being greater than the number of inputs, m, so there are n equations with m unknowns. The least squares solution is:

$$u = -(B^T B)^{-1} B^T B_d d$$

Perfect disturbance rejection is possible if:

$$B_d = B(B^T B)^{-1} B^T B_d \tag{4.18}$$

and if this is the case, the above formula also yields M. In other case, if $(I - B(B^T B)^{-1} B^T) B_d$ is small compared to B_d, then approximate rejection can be carried out[6].

If the disturbance does not enter on the input channel or, also, if the disturbances directly affect the output to be controlled so $y = Cx + D_d d$, $D_d \neq 0$, then this proportional feedforward does not work well. Two alternatives appear:

[6] $B(B^T B)^{-1} B^T$ is a projection matrix, in the sense that it yields the orthogonal projection of any set of vectors (at its right side) onto the column space of B.

- achieve rejection at steady-state,
- achieve rejection in some frequency range (or at a particular frequency, not necessarily DC, for example, when a system is subject to measurable sinusoidal disturbances) by *dynamic* feedforward.

Steady-state rejection. The steady-state rejection implies choosing u_{ff} such that, in steady-state, with a CT model:

$$\begin{aligned} 0 &= Ax + Bu_{ff} + B_d d \quad \Rightarrow \quad x = -A^{-1}(Bu_{ff} + B_d d) \\ y &= Cx + D_d d = -CA^{-1}(Bu_{ff} + B_d d) + D_d d \end{aligned}$$

a solution to $y = 0$ can be found if the following system of equations can be solved for u_{ff}:

$$-(CA^{-1}B_d + D_d)d = CA^{-1}Bu_{ff} \tag{4.19}$$

Depending on the number of sensors, actuators, disturbances and the system properties, the problem can have an exact unique solution (if $CA^{-1}B$ is square and invertible) or it might need to be solved in a least-squares sense.

Dynamic rejection: transfer matrix models. The second alternative, with more complex implementation, involves calculating the transfer function matrix from disturbances to outputs[7]. Let us call it $G_d(s) = C(sI - A)^{-1}B_d + D_d$. Then, by applying the control action:

$$u_{ff} = -G^{-1}G_d d$$

it is straightforward to verify that the overall effect of d is zero. However, $G^{-1}G_d$ might be non-realisable so further low-pass filters, F, should be added to achieve a feasible implementation:

$$u_{ff} = -G^{-1}FG_d d \tag{4.20}$$

discarding the goal of compensating for high-frequency disturbances. As usual, caution is needed with RHP zeros in plant model G.

Steady-state rejection in TF representation amounts to just calculating the DC gain of the above formulae:

$$u_{ff} = -G(0)^{-1}G_d(0) \tag{4.21}$$

In some cases, the disturbance measurement itself contains some dynamics $d_m = G_m d$, where d_m stands for disturbance measurement. If this is the case, the compensation should be replaced by $u_{ff} = -G^{-1}FG_dG_m^{-1}d_m$.

There are other cases of disturbance effects that have not been considered:

[7] In some cases, if the record of the effect of a sudden disturbance is available, step-response experimental identification methods (Appendix A.4) can directly determine this transfer function.

- the effect of the disturbance is measured through a sensor $z = C_d x + D_d d$, combining direct measurement plus state information. For example, an outflow in a system that is directly affected by known connection/disconnection of one downstream pump (d), as well as upstream tank levels (x),
- unmeasurable disturbances.

Typically, feedback control needs to be applied in these cases, trying to reduce the effect of the disturbance. But it can be complemented with a "feedforward" control based on (feedback) estimations of the disturbances, as discussed in Section 6.3.2.

4.7 Two Degree of Freedom Controller

With the combination of feedforward and feedback control, better control performances may be achieved. Feedback control is constrained by:

- *stability* issues in the face of uncertainty,
- tracking/regulation *vs.* measurement noise filtering performance trade-off,

and feedforward control, only applicable to stable systems, is limited by:

- sensor availability (not a problem for references, but for disturbance-rejection tasks),
- *performance* degrading in the face of modelling error.

Thus, a so-called 2-DoF control structure can be designed as shown in Figure 4.4. In this case, the control action is generated by

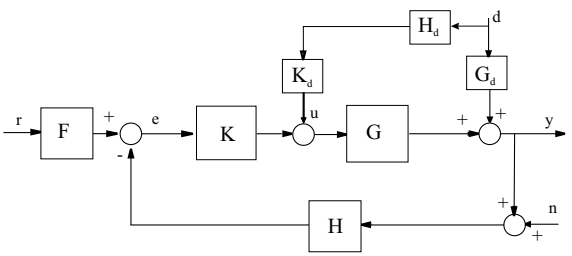

Figure 4.4. Two-degree of Freedom control structure

$$u = K[Fr - H(y + n)] + K_d H_d d$$

where F and K_d are feedforward controllers attached to the reference and measurable disturbances, respectively, K is a forward path controller, H is a feedback controller (and/or a measurement noise filter) and H_d is the disturbance sensor dynamics. The output is expressed by:

$$y = (I + GKH)^{-1} (GKFr + (G_d + GK_dH_d)d - GKHn)$$

So, F can be designed with only tracking in mind, K_d for measurable disturbance rejection, as in (4.20), H for high-frequency noise filtering and K will try to carry out rejection of unmeasurable disturbances and modelling error.

Elementary forms of 2-DoF control appear in industrial PID regulators. Applying the methodologies presented in Section 7.4, a general 2-DoF optimisation framework is able to solve many disturbance rejection problems of engineering significance. A general MIMO 2-DoF structure is further analysed in Section 8.4.3.

4.8 Hierarchical Control

There are many control goals requiring different algorithms and kinds of information from the process, as mentioned at the beginning of this chapter: from the level closest to the process to the plant-wide control. All these activities should be connected and some kind of hierarchy is natural.

Local control. Uses information directly gathered from the process, as fast as possible, to generate the reaction of the control to process changes in order to fulfill the tracking or regulatory control goals. It also includes logic control to handle automata.

Supervisory control. Evaluates the behaviour of the local controls and commands change in parameters, structure or components. It may include adaptation, as well as emergency actions.

Coordinating control. Evaluates the performances of the process and determines objectives (set-points) for the lower level controls. It may include optimisation routines to compute the best operating conditions.

Plant-wide control. Considers the constraints among interconnected processes to optimise the whole plant behaviour. It may include management and policy criteria to define goals for the lower controls.

In digital control, the control algorithm only takes a minimum part of the control code, with a lot of activities around the control to ensure its correct operation. From an algorithmic viewpoint we must consider, among others:

- the control algorithm itself,
- output and state filtering (observers and virtual sensors),
- scaling and monitoring,
- connection with starting up and shouting down procedures,
- alarm treatment,
- data storage and recovery,

and from an instrumental viewpoint, also issues related to:

- data and variables' representation,
- timing, sampling and scheduling of tasks,
- data acquisition systems,

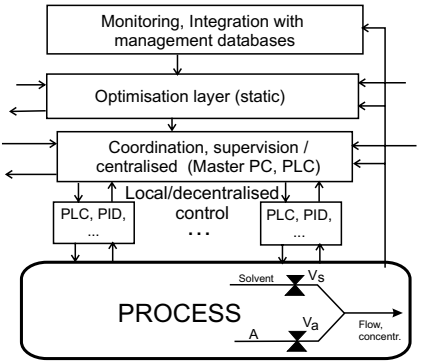

Figure 4.5. Plant-wide control schema

– communication channels and possible delays,
– human/machine interface,
shall be considered for the success of the designed control solution.

4.9 Key Issues in Control Design

Control goals, variable selection and control structure are key decisions in the design of a control system. In any case, once a control system has been designed, a validation process is required to determine:

- the range of validity of the proposed control structure,
- its robustness to modelling errors,
- the limitations in achieving some performances due to the process,
- the limitations in achievable performance due to implementation constraints,
- the need of use of multiple controllers or adaptation,
- the tasks to be allocated to a supervisory control level design.

Some of these issues are introduced in the last chapter of the book, and there is a lot of research to develop new approaches related to:

- integrated design of the process and the controller,
- iterative modelling and control design,
- integrated control algorithm and implementation system design,
- learning controllers.

Given the control goals, and taking into account the limitations above, a suitable control design approach should be followed. This is the matter of the next chapters.

4.10 Case Study: Ceramic Kiln

The production process of ceramic tiles involves a number of operations and subprocesses. Four main activities should be considered: raw materials grinding and milling, pressing and compacting the biscuit, glazing, and firing at the kiln. Most of the final characteristics of the tiles depend on the firing process. The kiln is, therefore, the core of the whole production plant. This case study presents some comments about the definition of the control system, in the way we proceeded in the framework of a practical implementation[8].

Users' goals. The main goal of the production engineers is to get high production of tiles with a high quality, this being defined by the dimensional, tonality and resistance characteristics of the tiles. Direct measurement of some of these variables is not easy. Thus, control objectives should be oriented to control some accessible process variables so as to (indirectly) keep the tiles' properties within some specified bounds (see Section 5.5.1). We will concentrate on the kiln.

After an analysis of the firing process, the main control goal should be to apply a pre-defined temperature firing profile to each single tile. That means not only control of the temperature and time of exposure of each tile during its travel within the kiln, but also to keep adequate ventilation conditions, strongly affecting the surface properties of the tile, maximising the number of tiles per hour.

The process. A kiln is a long tunnel (70–100 m) with a useful section of approximately $1.7\,\mathrm{m} \times 0.5\,\mathrm{m}$, thus, it is a distributed process, where the internal temperature and pressure depends on time and position. To simplify the model, it is divided into sections. Tiles are transported by rollers in one direction and ventilation air is flowing in the opposite direction. In some internal sections, combustion units will heat the tiles whereas in some other sections external air will cool the tiles. We must also distinguish between the upper and lower part of the section (above and below the tiles). The sections are functionally grouped in different zones: pre-kiln, pre-heating, firing, forced rapid cooling, normal slow cooling and final cooling zone, Figure 4.6. The construction of the kiln (number of sections, distribution of burners and fans, rollers, and many mechanical properties) will determine the potential production possibilities, as well as the final product quality. The goal of the control would be to apply the best actions, constrained to the existing kiln hardware.

Process variables. There are many internal variables in the kiln and inside the tiles. The common sensing devices are thermocouples, and pressure, flow and speed meters. The actuators are motors (to manipulate the tiles and fan speed) and valves (to modify the flow of gas and air). At the kiln exit, some tile properties can be measured: calibre, planarity, out-of-square and colour

[8] European ESPRIT project "MARCK", 1997.

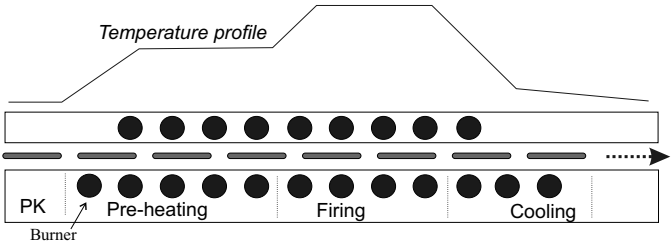

Figure 4.6. Kiln diagram

properties. Mechanical resistance can only be measured out of line and with a great time delay, not being suitable for on-line control of the kiln.

A large number of variables can be considered and measured for monitoring purposes. But from the control viewpoint, the measured variables used to generate the control actions are defined by the number of manipulated variables and their placement. SVD techniques can be used to determine the suitable number of control variables, leaving the remaining inputs as constant or, at least, only used in changes of the operating conditions. It can also provide some hints to select the most relevant temperatures related to each burner (up, down, in the same or contiguous cell, ...).

Setting up a model. As already described, the kiln can be considered as a series of sections (or cells). From the point of view of the thermal behaviour inside it, we may consider the kiln scheme shown in Figure 4.7. The post-combustion gases flow from right to left, while the tiles advance from left to right. A simplified model of the thermal process would be based on first-order TF from:

- a heating burner power to the temperature in the corresponding cell, $G_{i,i}$,
- the temperature of a cell to that of the previous one (due to convection), $G_{i,i+1}$,
- the presence of masses being heated in a cell (tiles, *etc.*).

Experimental identification (Section A.4) could be carried out by introducing pseudo-random binary signals (PRBS). The kiln should be operating under normal conditions, with local PID flow control of the burners. Three kinds of experiments are foreseen:

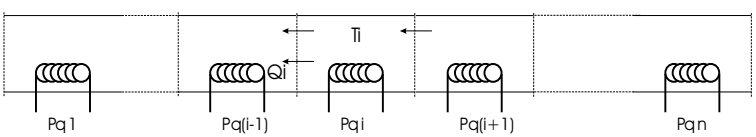

Figure 4.7. Kiln diagram (II)

1. Keeping the temperature in a section $(i+1)$ as steady as possible, introduce a PRBS-like set-point at the local PID of section i. This allows for identifying $G_{i,i}$ using a regular RLS algorithm.
2. Keeping the output of the PID at section i constant, force temperature changes at section $(i+1)$, thus allowing for the identification of $G_{i,i+1}$.
3. a RLS algorithm with two inputs and one output can also be used. In this case, changes both at the i-th control action and at the $(i+1)$-th temperature should be forced.

In Figure 4.8, some of the results on a typical kiln are shown.

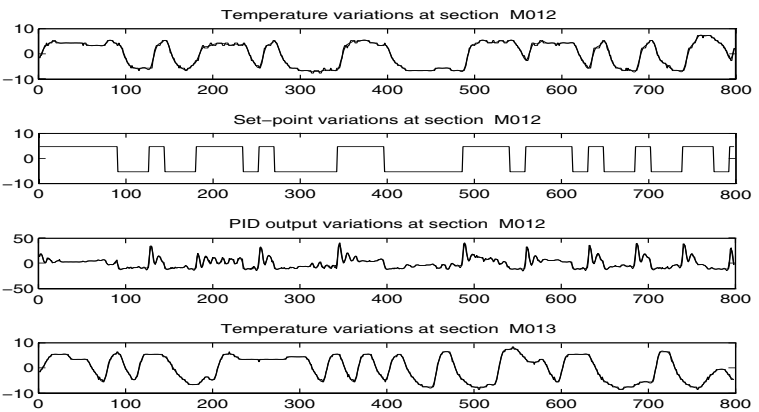

Figure 4.8. Identification of the kiln model. Input/output experimental data

Control structure. A hierarchical control structure seems to be appropriate.

- changes in *production* will determine the set-point of the roller speed. It will also provide a feedforward to the gas burners' flow, as well as changes in the kiln internal pressure and ventilation airflow,
- The final goal, *quality control*, is based on measurements at the exit of the kiln, and the fine-tuning of the temperature profile (and vertical distribution) is defined in a qualitative and heuristic way,
- *coordinating control* to achieve an actual temperature that is equal to the desired reference at each section, $T_i = r_i$. To reach this goal, a decoupling control with decentralised feedback (Section 5.3) was implemented, keeping the controlled variables at their desired value.
- local PIDs installed at the burners will provide a lower-level slave control trying to assure the demanded gas flows (Section 5.4.1).

5

Decentralised and Decoupled Control

In many industrial plants, the basic extension of classical PID controller design, implementation and tuning is the decentralised approach, where structural concepts are used to decouple the interaction between variables. The use of standard equipment and the ease of hand-tuning or understanding by non-specialist technicians are the main advantages of this approach. Nevertheless, the control effort is decomposed into two stages: first to decouple the different subsystems and then to control them. The extra effort rewards consist in simpler subsequent design, implementation and tuning. This chapter addresses some of these issues.

5.1 Introduction

In Section 4.3, selection of suitable inputs and outputs was discussed. Those inputs had significant effect on key internal process variables and the sensors allowed for its determination with acceptable noise sensitivity.

Decentralised (distributed) control tries to control multivariable plants by a suitable decomposition into SISO control loops. It has the advantage of easy implementation and tuning if a sufficient number of sensors and actuators is available. However, with a limited number of them, a centralised controller carrying out matrix operations will be able to squeeze better performance out of processes with strong couplings or conditioning problems.

The key issue behind the use of a centralised strategy is, however, the availability of a precise enough model. If it is not available, then lower performance can be targeted with enough guarantees of success, as discussed in Chapter 8. In this case, a wise combination of standard SISO regulators may solve the problem with comparable results, with implementation advantages.

Let us assume that a first selection of controlled variables, actuators and measurements is available and let us discuss some possibilities to try to solve the problem by use of multiple SISO controllers, first on an independent basis and then on a coordinated structure.

The design of independent loops may have, if the sensor and actuator configuration is adequate, a significative advantage regarding *reliability*: in the event of a fault in another regulator or actuator, the "independent" controller will still try to drive its associated variable to its set-point. Note that in a system heavily dependent on the coordination of all loops, the failure of one component may lead to failure of the overall structure. This may be, particularly, the case with the *centralised* control approach. The reliability advantage is one of the multiple appeals to practitioners regarding the use of separate SISO loops instead of centralised or heavily coordinated approaches. However, this advantage does not apply to all possible plants and sensor–actuator choices.

5.1.1 Plant Decomposition, Grouping of Variables

In many industrial cases, a complex process can be divided into subprocesses so variables can be grouped into several sets corresponding to each subsystem. This division corresponds in many cases to actually engineered subsystems and, in other cases, it is just a "conceptual" framework for control design. Figure 5.1 depicts the basic idea.

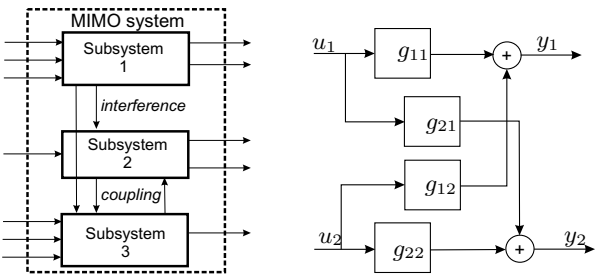

Figure 5.1. A MIMO system as a set of coupled simpler subsystems

The subsystems, however, have some interaction (in the trivial case of totally isolated subsystems, they can be handled independently of the rest). There are two types of interaction:

- *interference (feedforward interaction).* Some variables of a subsystem influence other subsystems. However, these subsystems do not influence variables of the first one.The key characteristic of this type of interaction is that there is no return path to the system originating the interference.
  In the linear case, they can be considered as (additive) disturbances, and they have the form of *triangular* transfer function submatrices. This is the typical case of cascaded stages in an industrial facility,
- *coupling.* In this interaction, there exists a path of cross-influence so that there is a hidden "feedback" loop. Ignoring it in multi-loop strategies can lead to instability.

Example 5.1. In the TITO plant in Figure 5.1 (right), u_1 and y_1 are related by the transfer function g_{11}.

However, using a feedback controller, K_{22}, with u_2–y_2 would inadvertently create an additional feedback path g_{21}–(loop 2)–g_{12}, changing the "apparent" transfer function from u_1 to y_1 to:

$$y_1 = (g_{11} + g_{12} * K_{22} * (1 + K_{22}g_{22})^{-1}g_{21})u_1 \stackrel{\text{def}}{=} \tilde{g}_{11}u_1 \tag{5.1}$$

By $\tilde{g}$, we will denote the *apparent* transfer function produced by coupling. Note that this is a case of the general LFT interconnection in (2.53).

From the background of SISO techniques, two basic approaches can be pursued, depending on the basic requirements:

- **decentralised control** tries to divide the plant and design *independent* controllers for each of the subsystems as a way to handle the control of the overall plant. In this case, two alternatives arise:
  - neglect the coupling. To minimise the risk of bad performance, variables should be grouped so that the strength of the coupling is small. This is the so-called *pairing* problem, and the resulting strategy is called *multi-loop* control. It can be guided by common sense if the system is built to achieve a structure such as the one in Figure 5.1, but there are some approximate model-based tools to help the selection,
  - prior to control design, carry out a *decoupling* operation ("cancelling" coupling by transforming the system into a diagonal or triangular structure via a transformation matrix).
- **cascade control**, adding extra sensors and actuators to improve closed-loop performance of a basic control loop.

Although the grouping can result in MIMO subsystems, for simplicity, the extreme cases will be studied in this chapter: on one hand, decomposing a MIMO plant into SISO subsystems and, on the other, analysing enhancements to a SISO control loop (one set-point) via MIMO techniques.

Many industrial control systems in electrical, chemical plants, *etc.* are actually designed from these ideas and its combination. A steam boiler and distillation column case studies at the end of the chapter will show the application of some of them in a realistic situation.

Of course, apart from these ideas, there always exists the possibility of a centralised matrix-based control calculation, to be discussed in Chapters 6 and 7.

5.2 Multi-loop Control, Pairing Selection

The historically first approach to multivariable control in industry was dividing the sensors and actuators into m subsets and designing control loops using one of the sensor sets and one of the actuator sets (selecting those sets is

usually denoted as input/output *pairing*). In this way, a complex control problem is divided into m supposedly simpler ones, with no information exchange between those controllers. This strategy is called *multi-loop control*.

Multi-loop control usually addresses control of a plant with m sensors and actuators with m SISO loops, although some of the ideas might apply to block-pairing (if centralised control is used for those blocks). For reasons of industrial applicability, only the SISO multi-loop case will be considered, although block-decomposition of a complex plant is indeed done in the project phase, usually based on experience and knowledge of the internals of the process (dividing into independent subsystems to be controlled). Some tools for the block multi-loop control can be found in [88].

In an ideal case, multi-loop control would have the advantages of:

- *flexibility and fault tolerance*, as independent loops can be turned on and off, due to operator decision or faults, without excessive degradation on the performance of the rest of loops,
- *simplicity*, allowing independent design of SISO regulators, with easier on-line tuning.

The multi-loop idea is appealing but it may not work in strongly coupled systems. Only in truly diagonal or block-diagonal transfer function matrices can success be guaranteed (and, under some assumptions, in triangular plants). In any other case, the loops will be interacting due to coupling, even to the point of compromising stability.

In this case, the objective is to achieve a reasonable level of performance in a plant with m SISO controllers, each of them trying to keep under control a particular output y_j by means of manipulating one actuator u_i (from a total of m inputs and m outputs). The number of combinations is $m!$ (factorial of m) so some criteria should be used in deciding the configuration.

In many cases, coupling is neglected, being considered as a "modelling error" when independently designing each of the loops. This fact limits achievable performance for reasonable robustness (see Section 5.2.3 and Chapter 8): "detuning" of some of the SISO regulators (aiming for lower performance objectives) can mask unacceptable interactions.

To successfully apply multi-loop control, a methodology to assess the degree of interaction between the loops is needed. Apart from, of course, trial and error procedures on plant prototypes or fine-tuning and optimising on simulations, some conclusions can be drawn if a plant model is available, even a rudimentary one, such as an approximation to the DC gain matrix.

In fact, as high-performance multi-loop control can be difficult to achieve, the most popular results on stability and pairing are usually referred to very-low-bandwith closed-loops with integral action. In this way, some conclusions based on the DC gain matrix can be drawn. Once this low-performance set-up has been achieved, specifications can be sequentially increased (see Section 5.6).

Niederlinski Criterion

The most immediate result on pairing selection is the Niederlinski criterion [93].

Given a *stable* multivariable system $G_{m\times m}$ (row and column transposition have been done to locate the selected pairings on the diagonal) and a diagonal controller with integral action K/s, the closed loop will be *unstable* if $L = G(0)K(0)$ verifies the *sign condition*[1]:

$$\text{sign}(\det(L)) \neq \text{sign}(\Pi_{i=1}^{m} L_{ii}) \tag{5.2}$$

The Niederlinski criterion can exclude many candidate pairings, assuming that the sign of the integral gain in k_{ii} is the same as that of the DC gain of g_{ii} (as expected in a decentralised design). In this case, L_{ii} has positive gain so the criterion reduces to $\det(L(0)) < 0$ as a sufficient condition for instability.

5.2.1 The Relative Gain Array Methodology

The relative gain array methodology is a widely used screening tool to help determine if a particular input/output pair (say, y_i and u_j) is a wise choice for implementing a SISO control loop, in the sense that coupling and interaction with other loops will be small, refining the Niederlinski result.

Let us detail the basic ideas behind the approach, by considering the TITO system in Figure 5.1 (right), analysed in Example 5.1.

Example 5.2. In the referred example, coupling transforms the relation between u_1 and y_1 from g_{11} to $\hat{g}_{11}$ in (5.1). If g_{11} and $\hat{g}_{11}$ were similar, a controller designed on the open-loop characteristic, g_{11}, would also work when loop 2 is closed. However, we would like to know this possibility *before* actually spending time in designing the controllers, and $\hat{g}_{11}$ depends on K_{22} for which a similar analysis should be made with $\hat{g}_{22}$.

An approximation may assume that output 2 will be "well controlled", this implying $(1+g_{22}K_{22})^{-1} \approx 0$, *i.e.*, K_{22} having a big gain. In this case, with $K_{22} \to \infty$, (5.1) becomes:

$$\tilde{g}_{11} \approx g_{11} + g_{12}g_{22}^{-1}g_{21} \tag{5.3}$$

So, if a controller is designed on $\tilde{g}_{11}$, dependence with the to-be-designed controller for loop 2 is not significant as long as loop 2 performs well.

Let us discuss the general case. In open loop, the transfer function between output y_i and input u_j is g_{ij}, as:

$$y = G(s)u \quad \Rightarrow \quad y_i = \sum_{k=1}^{n} g_{ik}u_k \tag{5.4}$$

[1] It is a generalisation of the well-known SISO criteria: "a process with negative DC gain and positive integral action gain K_I is unstable".

then experiments with all inputs except u_j equal to zero will produce output $y_i = g_{ij}u_j$. It will be assumed that $G(0)$ is non-singular[2].

Now, let us think on the extreme case in which all outputs except y_i are under *"perfect" control*, *i.e.*, very close to its operating point (zero in linearised coordinates), by using *all actuators except* u_j (left for a not-yet-implemented SISO control jointly with sensor y_i). The configuration is depicted in Figure 5.2.

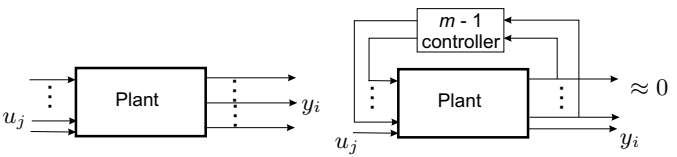

Figure 5.2. Multi-loop control: all loops open (left), all except one closed (right)

Obviously, if a "step" or any other prescribed input were applied to u_j, the control system would try to counteract the effect of this input on the controlled outputs (because, due to coupling, it will act as a "disturbance" on them). So, the control actions $\{u_1, \ldots, u_{j-1}, u_{j+1}, \ldots, u_n\}$ will *not* be zero, and the outputs $\{y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n\}$ will be approximately zero if the control is well designed.

Let us consider the mathematical inverse relation to that in (5.4),

$$u = G^{-1}(s)y \quad \Rightarrow \quad u_l = \sum_{k=1}^{n} t_{lk} y_k \tag{5.5}$$

where t_{lk} is the corresponding element of the inverse transfer function matrix G^{-1}. Particularising for u_j and replacing all controlled outputs by zero, and inverting the result to have the correct causality, we have:

$$u_j = t_{ji} y_i \quad \Rightarrow \quad y_i = \frac{1}{t_{ji}} u_j \equiv \tilde{g}_{ij} u_j$$

The transfer function $\tilde{g}_{ij} = 1/t_{ji}$ is an *apparent* transfer function (it is a closed-loop transfer function with a perfect controller).

So, if g_{ij} is "similar" to $\tilde{g}_{ij}$, the presence of the rest of the controllers will not influence the behaviour of a SISO loop designed on the model g_{ij}, and it will be a "good" candidate for an independent loop. A way of measuring "similitude" between $\tilde{g}_{ij}$ and g_{ij} is checking if its quotient is near 1:

[2] Otherwise, the system would have a zero at $s = 0$ and some combination of outputs would not be controllable in steady-state. An SISO example would be $s/(s+1)$, unable to be driven to a non-zero set-point without an unbounded ramp input.

$$\frac{g_{ij}}{\tilde{g}_{ij}} = \frac{g_{ij}}{1/t_{ji}} \approx 1 \quad \Rightarrow \quad \lambda_{ij} \stackrel{\text{def}}{=} g_{ij} t_{ji} \approx 1 \tag{5.6}$$

Evaluating λ_{ij} for all possible pairings yields the so-called *relative gain* array $\Lambda(s)$. It is straightforward to realise that it is evaluated by *element-by-element* multiplication ($\times$) of G and its inverse transpose:

$$\Lambda(s) = G(s) \times \left(G(s)^T\right)^{-1}$$

The RGA is scaling-independent and controller-independent.

The RGA was introduced in [33] and, for ease of computation and model availability, it is usually evaluated at zero frequency[3] (DC gain). The ideas obviously apply only to regulators without significant gain at higher frequencies (regulators with only integral action).

These coefficients can be interpreted as the ratio between the open-loop SISO static gain and the gain with "perfect" control on the rest of the loops, using the other input variables.

In MATLAB®, the element-by-element product uses notation `.*`, so:

```
g0=dcgain(sys); rga=g0.*inv(g0')
```

Triangular plants. Triangular plants have the identity matrix as the RGA, and they are easier to control. Without loss of generality, let us assume an upper triangular plant. In this case, the deviations in loop j act as disturbances g_{ij} to the upper ones ($i < j$), but the reciprocal does not hold ($g_{ji} = 0$). So, there is only *one-way* interference. Sequential design from $i = m$ to 1 of the control loops will ensure success.

Rules for Pairing Selection

Some rules of thumb from the previous arguments can be stated as [119]:

- $\lambda_{ij} \approx 1$: the pairing y_j–u_i is a good candidate for an independent SISO loop,
- $\lambda_{ij} < 0$: the apparent transfer function has a different *sign* to the open-loop one, so there is a chance of a miscalculation of the sign of feedback, leading to instability. It can be shown [119] that, with integral action controllers, using a negative-RGA pairing will lead to at least one of these three situations: (a) unstable all-closed loop, (b) unstable stand-alone loop (referring to the one with negative RGA), (c) unstable m-minus-one loop (if the loop with the negative relative gain is opened, due to faults or saturation).

[3] In processes with integrators in open loop (infinite DC gain), if integrators can be "pulled out" in such a way that they can be considered part of a regulator (if they appear before a particular input or after a particular output), they can be removed from the plant model and the RGA methodology can still be used.

- λ_{ij} small (positive): The apparent closed-loop "gain" is much higher than that of the open loop. This can cause performance degradation or even instability when closing the loop. It pinpoints an input that has little effect on a particular output in open loop but with significant effect in closed loop due to coupling and feedback. Probably not a good pairing candidate,
- λ_{ij} big (positive): the open-loop gain is higher than that of the closed loop, so control may become *ineffective* when closing the rest of the loops. Furthermore, large RGA elements may indicate difficulties in control due to uncertainty (see Remark 5.6), irrespective of the chosen pairing.

Pairings decided by these rules should be compared against other options from common sense or insights into the process internal mechanisms.

Example 5.3. A system has a DC gain matrix given by:

$$G(0) = \begin{pmatrix} -1 & 1.5 & 2.8 \\ 4 & 1.6 & 4 \\ 0.15 & 2 & -0.1 \end{pmatrix}$$

Evaluating its RGA,

```
g=[-1 1.5 2.8;4 1.6 4;.15 2 -.1]; rga=g.*inv(g')
rga =   0.2600    0.0478    0.6922
        0.7328   -0.0163    0.2835
        0.0073    0.9685    0.0242
```

the recommended pairings would be y_2–u_1, y_3–u_2 and y_1–u_3 if SISO PI controllers are used in first place. For example, the apparent all-loops closed transfer function for y_1–u_2 has a gain of: 31.4 ($1/t_{21}$), 21 times higher than the open-loop one of 1.5!

RGA with full frequency response. The previous considerations have been taken on the basis of the DC gain matrix, and only apply to low-gain integral action. However, for more demanding control specifications, depending on actuator bandwidth and directionality variations with frequency, the full Bode diagram of Λ should be drawn.

In this case, at frequencies important for control stability robustness, *i.e.*, around the peak of sensitivity (4.5) or complementary sensitivity (4.6) (see Section 8.5, in particular Equations (8.11), (F.2)), if $\Lambda(j\omega)$ approaches the identity matrix, indicating that the plant is triangular, stability problems are avoided in multi-loop control [119]. Note that the matrix in this case has complex numbers as elements.

The condition may be a hard one to meet, but, in some cases, it might point out alternative pairings to those from only steady-state considerations if higher performance (with decentralised control) must be attained. As another tool, the achievable bandwidth for different pairings may be assessed using (5.13).

Remark 5.4. Note also that if the process is unstable or non-minimum-phase, G^{-1} will have RHP zeros or poles respectively so the apparent transfer functions may suffer substantial changes with respect to the open-loop ones. In this case, the RGA indications at steady-state will not reflect the whole situation.

Example 5.5 (TITO system). In the 2×2 case (Figure 5.1), the RGA matrix is:

$$\Lambda = \begin{pmatrix} \lambda_{11} & 1-\lambda_{11} \\ 1-\lambda_{11} & \lambda_{11} \end{pmatrix} \qquad \lambda_{11} = \frac{1}{1 - \frac{G_{12}G_{21}}{G_{11}G_{22}}} \tag{5.7}$$

and, with a diagonal controller (K_1, K_2), the closed-loop characteristic equation is:

$$CE(s) = 1 + K_1G_{11} + K_2G_{22} + \frac{K_1K_2G_{11}G_{22}}{\lambda_{11}} \tag{5.8}$$

This shows the relationship between relative gain and closed-loop stability. If $\lambda_{11} \approx 1$, the characteristic equation reduces to $(1 + K_1G_1)(1 + K_2G_2)$.

Remark 5.6. Incidentally, the RGA helps in evaluating the maximum size of perturbations on a particular matrix element so that the matrix does not become singular. In particular, a matrix A becomes singular if a single element a_{ij} is perturbed to $a'_{ij} = a_{ij}(1 - \lambda_{ij}^{-1})$. The result is meaningful in analysing sensitivity to element-by-element uncertainty in experimental identification of ill-conditioned processes. Also, the sum of the absolute value of all RGA matrix elements is, orientatively, close to the minimised condition number (page 292): plants with large RGA elements are usually difficult to control (due to sensitivity to uncertainty). For example, the RGA of the plant in Example B.6 is identity, *i.e.*, probably indicating that the chosen configuration will not pose significant control problems.

Disturbances: RDG

To try to account for possible amplification or attenuation of the disturbance response for a particular candidate pairing, if a rough disturbance model is available, the *relative disturbance gain* (RDG) β_i for a disturbance d can be defined [121] as the ratio between the change in control effort required to counteract d on loop i when the rest of loops are closed *vs.* the case they are open. The pairing is assumed already selected.

In the 2×2 TITO case, for a process with steady-state gain

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} h_{D1} \\ h_{D2} \end{pmatrix} d \tag{5.9}$$

the RDG gains are:

$$\beta_1 = \lambda_{11}\left(1 - \frac{h_{D2}g_{12}}{h_{D1}g_{22}}\right); \quad \beta_2 = \lambda_{11}\left(1 - \frac{h_{D1}g_{21}}{h_{D2}g_{11}}\right) \tag{5.10}$$

where λ_{11} is the $(1,1)$ element of the RGA matrix.

Values of $|\beta| < 1$ indicate a *beneficial interaction* (the loops cooperate) and values bigger than 1 indicate that closing one loop exacerbates disturbance effects on the other (at low frequencies).

5.2.2 Integrity (Fault Tolerance)

As previously commented, an interesting feature of multi-loop control is that in the event of failure of one of the loops, the rest will keep trying to achieve their target set-points. In this way, multi-loop structures might be more fault-tolerant than centralised or decoupled ones.

A multi-loop controlled system has *integrity* if the closed-loop system remains stable as subsystem controllers are arbitrarily brought in and out of service [35].

Integrity allows for any arbitrary order in the connection and disconnection of individual loops. A system not verifying the integrity property may yield instability in intermediate configurations if loops are connected or disconnected in a particular order. It may also have stability problems if some manipulated variables are driven to *saturation*.

As the failure of one of the loops changes the "apparent" transfer functions calculated above, fault tolerance is not guaranteed unless specifically checked.

Usually, conditions on the DC gain allow for checking integrity with a set of very-low-bandwidth integral-action regulators. This is named integral controllability with integrity (ICI).

A necessary condition for ICI is that the RGA elements of the selected pairing are positive [35, 119]. This is one of the main practical reasons for avoiding negative RGA pairings. The condition is not sufficient, however. Additional considerations appear in [35].

In particular, a more restrictive necessary condition, assuming the selected pairings are diagonal (y_i–u_i) (there is no loss of generality as $G(0)$ can be transformed to verify that by straightforward input or output reordering), the RGA of any submatrix of $G(0)$ obtained by elimination of any combination of loops (eliminating the j-th row and column from $G(0)$) must also have positive diagonal terms.

Example 5.7. For a system with DC gain:

$$G(0) = \begin{pmatrix} 56 & 66 & 75 & 97 \\ 75 & 54 & 82 & 28 \\ 18 & 66 & 25 & 38 \\ 9 & 51 & 8 & 11 \end{pmatrix}$$

the RGA matrix is:

```
rga =    6.1559   -0.6948   -7.9390    3.4779
        -1.7718    0.1018    3.1578   -0.4877
        -6.5985    1.7353    8.5538   -2.6906
         3.2144   -0.1422   -2.7726    0.7004
```

so the pairings y_1-u_4, y_2-u_3, y_3-u_2, y_4-u_1 are selected (other three combinations are possible with positive RGA elements). Reordering inputs, the transformed G is:

```
g2 =   97    75    66    56
       28    82    54    75
```

```
38    25    66    18
11     8    51     9
```

Eliminating the column and row corresponding to loop 2, the remaining 3×3 matrix has an RGA calculated by:

```
sel=[1 3 4]; g2=g2(sel,sel); rga=g2.*inv(g2')
```

with diagonal terms $\{-1.167_{(1)}, 0.629_{(3)}, 1.301_{(4)}\}$. So, loop 1 would not be a desirable pairing if only loops 1–3–4 were working: a fault on loop 2 will leave the system in an undesirable state, either because the loops are already unstable or because disconnection or saturation of loop 1 will cause instability.

So, this configuration does not have the ICI property. It can be checked, by inspection, that any other pairing combination also has these problems. There are the following engineering alternatives:

- a redesign of the actuator and sensor locations,
- coping with the possibility of transient instability in start-up and close-down or under a particular combination of faults (maybe establishing a suggested sequencing of the start-up, close-down and fault-handling procedures),
- pursuing a centralised strategy, given that the fault tolerance and flexibility of the decentralised one may not be satisfactory enough,
- try multi-loop control on some combinations of inputs and outputs with lower coupling (however, that may lose the physical interpretation of the signals). This strategy is outlined in Section 5.3.3.

As an exercise, it is left to the reader the simulation of the connection and disconnection of multi-loop controllers with a system such as $G * 1/(s+1)$ and controllers $0.05(s+1)/s$ in SIMULINK®, checking that 1–2–3–4 is a suitable start-up sequence avoiding instability, but loops 3–4 alone are unstable.

Some remarks on pairing selection

As real plants are non-linear, the best pairing selection may depend on the operating point, as the linearised models do change. For an example, see the mixing process case study (Section 5.8.2).

The best performing multi-loop control system is not always the one giving the least steady-state RGA interaction. This issue arises from possible changes in interaction direction at higher frequencies and also due to directionality effects on the action of disturbances: the best performing pairing may depend on the disturbance being considered, as discussed in Section 5.2.1.

So, the numerical methods of pairing selection are far from conclusive and many other considerations might need to be made by the control engineer. Good designs with negative RGA pairings occur very infrequently but applications are reported where those ones might be the best choice [12, 89], although losing integrity.

5.2.3 Diagonal Dominance (Stability Analysis)

Given a transfer function G, let us denote by $\tilde{G}$ the one formed by its diagonal elements. These diagonal elements are used for tuning a multi-loop regulator K, if suitable pairings are found, to obtain a diagonal $\tilde{T}$ closed-loop function. When the regulator is put into the real loop, denoting the relative error:

$$E = (G - \tilde{G})\tilde{G}^{-1} \tag{5.11}$$

it can be shown that:

$$(I + GK) = (I + E\tilde{T})(I + \tilde{G}K) \tag{5.12}$$

so "multi-loop behaviour" = "interaction analysis" * "nominal individual loops". As the nominal loops will be stable, the overall loop will be stable if $(I + E\tilde{T})$ does not have RHP zeros. If the plant is diagonal-dominant, then E is small. To be more precise, Gershgorin's theorem (Section B.3) enables us to derive the following limitation on the designed-for performance $\tilde{T}$ [119] from the Gershgorin radius ϱ_i in (3.37):

$$|\tilde{t}_i(j\omega)| \leq \frac{|g_{ii}(j\omega)|}{\sum_{j\neq i} |g_{ij}(j\omega)|} = \frac{1}{\varrho_i(\omega)} \quad \forall i,\ \forall \omega \tag{5.13}$$

These conditions refers to row sums. Substituting g_{ij} for g_{ji} results in a different condition based on column sums. This is a sufficient condition for closed-loop stability. If g_{ij}, $j \neq i$ are small (diagonal-dominant plant), the above bound is a large number, indicating that good control can be achieved at that frequency.

The frequency for which $\varrho_i \approx 0.5$ can be considered as an orientative designed bandwidth limit for multi-loop control[4]. If it is too low for a practical application, decoupling may be considered, at least at low frequencies (discussed in next section).

Note that (5.13) guarantees multi-loop stability but, as it is a conservative bound, stabilising regulators yielding faster loops may exist. Other sufficient, less conservative conditions can be derived from structured-uncertainty analysis (Section 8.5.4) [119], but they are beyond this book's scope.

5.3 Decoupling

In cases when multi-loop control is not effective in reaching the desired specifications, a possible strategy for tackling the MIMO control could be to transform the transfer function matrix into a diagonal one. This strategy is called *decoupling*.

[4] Assuming the usual SISO criteria (for second- and higher-order plants) regarding design of $\tilde{t}_i$ so its peak is less than 6 dB.

If the resulting transfer function matrix were diagonal (or, approximately, diagonal-dominant, (5.13)), then MIMO control would be equivalent to a set of independent control loops. Decoupling can be achieved in two ways:

- feedforward cancellation of the cross-coupling terms,
- based on state measurements, via a feedback law. The advantage of this second approach is that the decoupler can be cast as a gain matrix (static system).

5.3.1 Feedforward Decoupling

As seen in Section 4.6.2, adding a pre-compensator transforms the open-loop characteristic into $M(s)$, at the will of the designer (except RHP pole-zero cancellations and realisability considerations). Then, any SISO design methodology will finish the overall design if M is chosen to be diagonal.

Once a $K_D = G^{-1}M$ open-loop controller is available, it can be used as a *decoupler* in an enclosing feedback loop (see Figure 5.3).

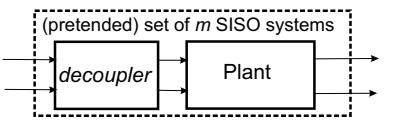

Figure 5.3. Decoupler pre-compensator block-diagram

Example 5.8. A transfer function matrix such as:

$$G(s) = \begin{pmatrix} \frac{6}{s+7} & \frac{5}{s+14} \\ \frac{3}{s+7} & \frac{-0.25}{s+4} \end{pmatrix}$$

can be expressed, converting rows to a common denominator as:

$$G(s) = M(s)N(s) = \begin{pmatrix} \frac{1}{(s+7)(s+14)} & 0 \\ 0 & \frac{1}{(s+7)(s+4)} \end{pmatrix} \begin{pmatrix} 6(s+14) & 5(s+7) \\ 3(s+4) & -0.25(s+7) \end{pmatrix}$$

Hence, inverting the $N(s)$ matrix containing the numerators:

$$N(s)^{-1} = \frac{1}{16.5s^2 + 196.5s + 567} \begin{pmatrix} -0.25(s+7) & -5(s+7) \\ -3(s+4) & 6(s+14) \end{pmatrix}$$

if the process is multiplied by $N(s)^{-1}$, a diagonal regulator, $K(s)$, can be designed for the "transformed" plant $M(s)$ (note that the regulator to be implemented would be a non-diagonal $N(s)^{-1}K(s)$, contrary to the multi-loop case).

If G were non-minimum-phase, N^{-1} would be an unstable system so this simple strategy would not work. In this case, the desired transfer function M must include the model RHP zeros for internal stability of the resulting decoupled loop. Let us show this idea with an example:

Example 5.9. Given a plant and its inverse:

$$G(s) = \frac{1}{(s+1)(s+2)} \begin{pmatrix} s+2 & 4 \\ -2 & -1 \end{pmatrix}; \quad G^{-1}(s) = \frac{(s+1)(s+2)}{6-s} \begin{pmatrix} -1 & -4 \\ 2 & s+2 \end{pmatrix}$$

the target decoupled dynamics must have the factor $(6-s)$ in *all numerators* in the diagonal so the decoupler $G^{-1}M$ is stable and RHP cancellations are avoided: spreading RHP zeros into several channels is a trade-off inherent to most decoupling cases[5]: the decoupled plant must have *two* RHP zeros (at the same value of s, but with different directions).

However, non-diagonal target models allow avoiding duplication of the zero and, furthermore, assigning it to a particular output.

In the case under consideration,

$$G(6) = \frac{1}{56} \begin{pmatrix} 8 & 4 \\ -2 & -1 \end{pmatrix}$$

effectively loses rank: the output zero direction, $y_z^T = (1, 4)$, is "unaffected" by input components of the form e^{6t}. So, the zero at $s = +6$ is "near" output 2.

To avoid cancellation, the target behaviour $M(s)$ must keep this "zero-gain" mode, *i.e.*, it must also fulfill $y_z^T M(6) = 0$. The diagonal factor M_a below indeed fulfills the restriction, but M_b and M_c do as well (controller $G^{-1}M_c$ shown as an example):

$$M_a = \begin{pmatrix} \frac{6-s}{6+s} & 0 \\ 0 & \frac{6-s}{6+s} \end{pmatrix}; \quad M_b = \begin{pmatrix} \frac{6-s}{6+s} & \frac{-8s}{(6+s)} \\ 0 & 1 \end{pmatrix}; \quad M_c = \begin{pmatrix} 1 & 0 \\ \frac{-0.5s}{6+s} & \frac{6+s}{6+s} \end{pmatrix} \tag{5.14}$$

$$G^{-1}M_c = \frac{(s+2)(s+1)}{s+6} \begin{pmatrix} -1 & -4 \\ 0.5(s+4) & (s+2) \end{pmatrix}$$

So, with a bit of additional straightforward work, a feasible M can be built so that decoupling is achieved only at low frequency ($M_a(0) = M_b(0) = M_c(0) = I$), but the RHP zero manifests on only one of the outputs. As the zero direction is more "aligned" with output 2, the dynamic interaction is lower with M_c (compare the numerators in the off-diagonal terms in M_b and M_c). The resulting triangular plant is easier to control than the original one.

The procedure in the example can be easily extended to systems with more than two outputs [119]. The RHP-related ideas here discussed apply to decoupling, feedforward control and also to achievable performance in feedback control. Similar issues would arise when decoupling unstable systems, regarding the spreading of the unstable dynamics to several channels.

Approximate decoupling. To design low-bandwidth loops, insertion of the inverse DC-gain before the loop ensures decoupling at least at steady-state. In some cases, a simple DC decoupling may significatively increase the bandwidth limit determined by (5.13) for a subsequent multi-loop design. If further bandwidth extension is desired, an approximation of G^{-1} valid in low frequencies can be used (such as inversion of slow-timescale reduced-order models, Section 3.10).

[5] In the case when the RHP factor does not appear in all factors in the inverse, spread to all channels can be avoided. Those plants are said to have *pinned* zeros.

5.3.2 Feedback Decoupling

Some decoupling strategies can be built for linear systems (and some non-linear ones, see Section 9.5.3) if the state is measurable.

Let us assume a system with m inputs and m outputs to be decoupled, whose equations are in the usual form :

$$\dot{x} = Ax + Bu; \quad y = Cx + Du$$

Let us consider a particular output y_i, to obtain the "shortest" direct relation to input u. Perhaps y_i does not depend on u if the i-th row of D, denoted as ${}_iD$, is zero. In this case, its time derivative, by applying the chain rule, can be shown to have the form (${}_iC$ denotes i-th row of C):

$$\dot{y}_i = {}_iC\dot{x} = {}_iCAx + {}_iCBu \tag{5.15}$$

If ${}_iCBu = 0$, the second ($\ddot{y}_i = {}_iCA^2x + {}_iCABu$) and successive derivatives will be taken until there exists an r_i so that u appears on its expression ($CA^{r_i-1}B \neq 0$). This r_i is denoted as the *output relative degree*, and in CT linear systems it is the minimum pole-zero excess, so:

$$\frac{d^{r_i}y_i}{dt^{r_i}} = {}_iCA^{r_i}x + {}_iCA^{r_i-1}Bu \tag{5.16}$$

By carrying out the same operation for all outputs, and stacking all output derivatives into one vector, the result is:

$$\tilde{y} = \begin{pmatrix} \frac{d^{r_1}y_1}{dt^{r_1}} \\ \vdots \\ \frac{d^{r_m}y_m}{dt^{r_m}} \end{pmatrix} = \begin{pmatrix} {}_1CA^{r_1} \\ \vdots \\ {}_mCA^{r_m} \end{pmatrix} x + \begin{pmatrix} {}_1CA^{r_1-1}B \\ \vdots \\ {}_mCA^{r_m-1}B \end{pmatrix} u = \tilde{H}x + \tilde{Q}u \tag{5.17}$$

where $\tilde{H}$ and $\tilde{Q}$ are constant matrices that can be easily calculated. Then, if x is measurable and the control action applied is the state feedback:

$$u = \tilde{Q}^{-1}\left(v - \tilde{H}x\right) = -K_Dx + Fv \tag{5.18}$$

$$K_D = \tilde{Q}^{-1}\tilde{H}; \quad F = \tilde{Q}^{-1} \tag{5.19}$$

where v is, at this moment, an arbitrary auxiliary input, the process equations get transformed into:

$$\tilde{y} = v$$

and, explicitly writing the components of $\tilde{y}$, the result is:

$$\frac{d^{r_i}y_i}{dt^i} = v_i \quad \Rightarrow \quad y(s) = \begin{pmatrix} \frac{1}{s^{r_1}} & 0 & \cdots & 0 \\ 0 & \frac{1}{s^{r_2}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{s^{r_m}} \end{pmatrix} v(s) \tag{5.20}$$

So, the system is transformed into a diagonal one in terms of the auxiliary variable.

Example 5.10. Let us have a TITO linear system $\dot{x} = Ax + Bu$, $y = Cx$ with:

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 3 \end{pmatrix}; \quad B = \begin{pmatrix} 1 & 1 \\ -1 & 1 \\ 0 & 0 \end{pmatrix}; \quad C = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Let us consider output $y_1 = x_1$. As it does not directly depend on u ($D = 0$), its first derivative will be taken:

$$\frac{dy_1}{dt} = {}_1C\dot{x} = \dot{x}_1 = x_1 + x_2 + u_1 + u_2 \tag{5.21}$$

so the relative degree of y_1 is $r_1 = 1$ as inputs directly affect its derivative.

Let us now consider $y_2 = x_3$ (not depending on u). Its derivative is:

$$\frac{dy_2}{dt} = \dot{x}_3 = x_2 + 3x_3$$

still not depending on u, so further derivatives must be taken:

$$\frac{d^2y_2}{dt^2} = \dot{x}_2 + 3\dot{x}_3 = 5x_2 + 9x_3 - u_1 + u_2 \tag{5.22}$$

Finding that the relative degree is $r_2 = 2$, joining (5.21) and (5.22), the resulting expression for (5.17) is:

$$\begin{pmatrix} \frac{dy_1}{dt} \\ \frac{d^2y_2}{d_t^2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 5 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{5.23}$$

so, as $\tilde{Q}$ is invertible, the state feedback law (5.18) is:

$$u = \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}^{-1} \left(\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} - \begin{pmatrix} 1 & 1 & 0 \\ 0 & 5 & 9 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \right)$$

and the system is transformed to a new state space realisation:

$$\bar{A} = A - B\tilde{Q}^{-1}\tilde{H} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -3 & 9 \\ 0 & 1 & 3 \end{pmatrix} \quad \bar{B} = B\tilde{Q}^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \tag{5.24}$$

$$\bar{C} = C \qquad \bar{D} = D \tag{5.25}$$

with, indeed, a third-order transfer function:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{s} & 0 \\ 0 & \frac{1}{s^2} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \tag{5.26}$$

State feedback changes the position of the poles of the system (see Section 6.1.1). Looking at (5.20), the zeros of the system, if any, are cancelled. The most important condition for feedback decoupling is that the system must be minimum-phase. Otherwise, unobservable "cancelled" dynamics will be unstable[6].

[6] In the example above, the original system had no transmission zeros, so there is no cancelled dynamics (the decoupled system is still third-order).

Remark 5.11. Decoupling transforms the system into independent integrator transfer functions. However, as well as in SISO control, integrators in the *plant* do *not* guarantee offset-free disturbance rejection in all cases. Furthermore, regarding offset-free tracking, it is (ideally) guaranteed, but the integrators will "disappear" under modelling errors. So, integral action in the controller may still be needed for the decoupled loops.

Discrete systems. The above procedure must be implemented in continuous-time to achieve a decoupled regulator. Otherwise, discretisation errors do occur. So, to implement this strategy, fast sampling must be used.

The procedure can be applied directly to discrete models if, instead of derivation of y, the magnitudes y_{k+1}, *etc.* are calculated until u_k appears. In the linear case: $y_k = Cx_k$, $y_{k+1} = CAx_k + CBu_k$, if $CB = 0$ then $y_{k+2} = CA^2x_k + CABu_k$. By an analogous methodology, a diagonal transfer function with elements z^{-r_i} can be built[7].

Some Remarks on Decoupling

Although at first glance, decoupling seems an appealing idea, there are some drawbacks:

- as decoupling is achieved via the coordination of sensors and actuators to achieve an "apparent" diagonal behaviour, the failure of one of the actuators (stuck valve, saturation) may heavily affect all loops. In contrast to this, multi-loop control (Section 5.2) might have a certain degree of fault tolerance as each loop will try to keep its operating point irrespective of other actuator or sensor failures,
- decoupling cannot be, in general, achieved with standard industrial regulators so the advantage over centralised control is only the ease that separation provides in design: decoupling is usually a design strategy for centralised, computer-controlled implementations,
- a decoupling design (inverse-based controller) may not be desirable for all disturbance-rejection tasks, unless the disturbances enter independently at each output as well. With other disturbance models, a consideration of the correlation (coupling) in the disturbance channel could lead to better designs, achieving similar levels of disturbance rejection with lower input usage [119],
- many MIMO non-minimum-phase systems, when feedforward decoupled, increase the RHP-zeros multiplicity so performance limitations due to its presence are exacerbated. Feedback decoupling does not work in that case,

[7] In most sampled-data systems with no pure delay, the first sample already depends on u_k. However, if the underlying continuous system has a high relative degree or sampling time is small, matrix $\bar{Q}$ will be ill-conditioned, so decoupling will lead to large control actions and sensitivity to model errors. Note also that minimum-phase continuous systems may exhibit zeros outside the unit circle when discretised.

- decoupling may be very sensitive to modelling errors, specially for ill-conditioned plants (although that kind of plant is difficult to control by any method, due to its big sensitivity to actuator imprecision),
- feedback decoupling needs full state measurements. Full state feedback has good robustness properties but stability and performance robustness margins of decoupling-based loops using estimated states (Section 6.2) must be explicitly checked (see Chapter 8).

5.3.3 SVD Decoupling

A matrix M can be expressed, using the singular value decomposition as $M = U\Sigma V^T$, where U and V are orthogonal matrices and Σ is *diagonal*.

The SVD can be used to obtain decoupled equations between linear combinations of sensors and linear combinations of actuators, given by the columns of U and V, respectively. In this way, although losing part of its intuitive sense, a decoupled design can be carried out even for *non-square* plants!

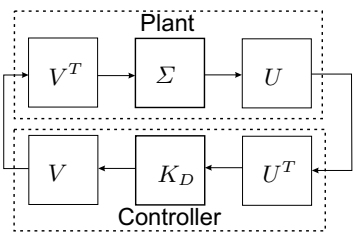

Figure 5.4. SVD decoupling: K_D is a diagonal controller designed for Σ.

If sensors are multiplied by U^T and control actions are multiplied by V, as in Figure 5.4, then the loop, in the transformed variables, is decoupled, so a diagonal controller K_D (such as a set of PIDs) can be used. Note that set-points must also be multiplied by U^T to be expressed in the same units as the transformed outputs. Usually, the sensor and actuator transformations are obtained using the DC gain, or a real approximation of $G(j\omega)$, where ω is around the desired closed-loop bandwidth.

The transformed sensor–actuator pair corresponding to the maximum singular value is the direction with biggest "gain" on the plant, that is, the combination of variables being "easiest to control". The lowest singular value is the most difficult combination to control.

In ill-conditioned plants, the ratio between the biggest and lowest singular value is large (for reference, greater than 20). They are very sensitive to input uncertainty as some "input directions" have much bigger gain than other ones. If actuators are imprecise, when trying to control the "low-gain" output combinations, high actuator amplitudes are needed, so actuator imprecision

causes big disturbances in the "high-gain" ones. An exception to this is if V (or a reordering of it) is diagonal-dominant.

If the "difficult" combinations (at crossover frequencies) do not have a high-gain regulator, this design gives reasonable controllers in terms of robustness [59, 119]. On the contrary, if the same requirements are posed for all directions, sensitivity to actuator uncertainty may be high. For reasonable robustness, it is advisable to have similar controller gains in all the loops.

SVD and the pairing selection

SVD decoupling produces the most suitable combinations for independent "multi-loop" control in the transformed variables, so its performance may be better than RGA-based designs (at the expense of losing physical interpretability).

If some of the vectors in V (input directions) have a significant component on a particular input, and the corresponding output direction is also significantly pointing to a particular output, that combination is a good candidate for an independent multi-loop control, so SVD decomposition may also help in the pairing selection problem for multi-loop control.

If, as usual, SVD and RGA are only evaluated at steady-state (zero frequency), the conclusions will apply only to low-gain proportional or integral-action regulators.

5.4 Enhancing SISO Loops with MIMO Techniques: Cascade Control

Cascade control refers to the design of a control loop for one primary variable by means of multiple sensors and/or actuators and, in its basic configuration, it consists of two cooperative SISO control loops with different time constants. This difference allows for separate design using conventional techniques. It is widely used in practice, sometimes in a transparent way (embedded into the electronics of a servoactuator, controlled power supply, *etc.*).

There exist two basic configurations:

- the output from a regulator is the reference to another one,
- the output from a regulator is the "sensor" to another one.

These configurations can be freely combined in a general situation. Their importance lies in the fact that they allow handling control of non-square systems with SISO techniques or, as an alternative interpretation, they allow improving performance by adding extra sensors and actuators while still using SISO techniques.

If there is not a substantial difference in time constants, although this strategy can still be pursued, the loop design cannot be made independent and based on SISO techniques, so tuning is not intuitive: centralised configurations might be preferable.

5.4.1 Case I: Extra Measurements

In many cases, a faster extra measurement of a secondary output is available. This is the case in speed measurements in mechanical systems, current measurements in switched power supplies, flows in process control, or readings of the actual position of an electro-pneumatic valve.

An example of that situation and the cascade-control loop appears in Figure 5.5, where a flow meter outputs a reading, q, and this flow fills a tank. The tank level, h, is the primary variable for which a reference exists.

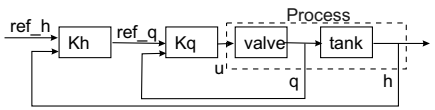

Figure 5.5. Two-sensor, one actuator, cascade control

On a first trial, a SISO control could be pursued by using a regulator for the whole valve + tank plant. However, as valves are significantly non-linear and feed pressure can vary depending on upstream connection–disconnection of additional equipment, the performance with this first controller structure may be far from optimal.

Of course, if a non-linear model of the valve (and its hysteresis, *etc.*) were available, a non-linear controller could try to invert it. If the valve were linear, all the state variables from the plant could be observed with one sensor and then control could act by reconstructing the flow measurement, if needed, from the observer. However, the approach has two significant drawbacks:

- the need of a model of the valve (possibly non-linear),
- the sensitivity to measurement noise of the estimations of internal variables based on the readings of h: estimates need to be filtered for a significant time and by then, the effect of pressure drops becomes significant so that higher control actions are needed. This issues will be discussed further in Chapter 8.

In practice, these drawbacks are solved by adding a flow sensor and using cascade control.

> *The best way to improve performance in practice and reduce sensitivity to modelling errors is to use additional sensors and/or actuators.*

This extra measurement will provide disturbance and modelling error rejection before its effects are manifested on the "slow" subsystems. In this way, the use of the extra sensor can provide significant performance improvement over any alternative using only one sensor, as well as reducing sensitivity to modelling errors on the valve (both regulators, for simple systems, can be tuned using model-free techniques). Furthermore, dividing into two "tasks"

the control action, each of the transfer functions may have a lower order than one overall controller: PIDs can be successfully used for high-order uncertain plants.

Cascade control in this configuration is especially useful if the inner loop is significantly faster than the outer one. The outer controller is named as the *master controller*, and the inner one is the *slave controller*. The equations are:

$$u = K_{slave} * (r_{slave} - y_{inner}); \quad r_{slave} = K_{master} * (r_{outer} - y_{outer}) \quad (5.27)$$

If time constants are similar then a centralised implementation and joint calculation can be given:

$$u = K_1 y_{inner} + K_2 y_{outer}$$

where K_1 and K_2 must be calculated together (for example with the observer + state feedback methodologies in Section 6.2). No master–slave interpretation can be given in this case. However, performance or robustness improvement can be achieved with two sensors.

After presenting the basic ideas for robustness analysis in Chapter 8, Case Study 8.8.1 on page 244 outlines a simple justification on why cascade control does work. Example F.6 on page 329 also presents another case of improving robustness margins by addition of an extra sensor.

5.4.2 Case II: Extra Actuators

In some cases, there are more manipulated variables (actuators) than inputs to the controller and those actuators have different bandwidth and saturation limits.

In Figure 5.6, a schematic block-diagram of the cascade structure is shown. In the diagram, u_1 is a fast actuator, able to achieve tight control on the primary variable measured by the sensor. However, it is either limited in amplitude or it is too costly in terms of efficiency for long-term control at high deviations: as fast, powerful actuators are expensive, cascade control can solve some practical problems with two cheaper actuators (one slow but powerful, another fast but with limited amplitude).

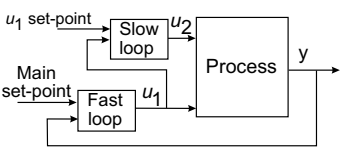

Figure 5.6. Cascade control with 2 actuators and 1 sensor

The cascade controller structure allows a fast disturbance rejection due to the fast actuator with a gradual return of it to a desired "actuator set-point" when the slower one takes charge.

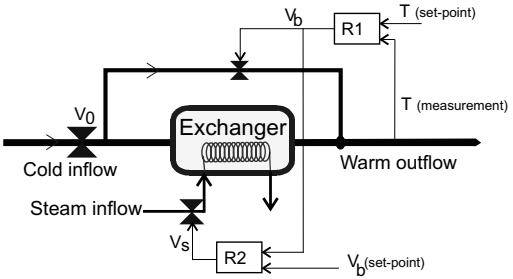

Figure 5.7. Heat exchanger

Example 5.12. A typical example is a heat exchanger with a valve V_s regulating steam flow, whose primary reference is outflow temperature (Figure 5.7). It has a high-order, slow dynamics (time constant of about several minutes plus maybe a significant delay). Let us assume that experimental modelling fits a model such as:

$$G_1(s) = \frac{60}{(120s+1)^4}$$

where output units are degrees celsius and input units are valve opening $[-0.5, +0.5]$ around a nominal 50% set-point.

From the Bode diagram, for target time constants below 120 s, achieving a tenfold increase in speed would require a control effort roughly 10^4 times that of the step response. Achieving that control bandwidth is impossible in practical terms whichever control design methodology is used (a 10^4 times more powerful actuator cannot even be considered). However, the DC gain is reasonable in the sense that operating the valve can cause temperature variations of ± 30 units, and this is deemed enough for compensating for the expected disturbances.

A bypass valve V_b (in a pipe having a reasonably small section), on the contrary, has a fast action (a time constant of seconds), such as, roughly:

$$G_2(s) = \frac{30(0.2s+1)}{(s+1)}$$

and, furthermore, the smaller valve can be operated with more precision. However, it can only compensate temperature variations of ± 15 units, and operating it at wide apertures is inefficient (as the exchanger is heating fluid only to mix it with cool water afterwards).

A cascade controller will successfully combine the advantages of each of the actuators. The bypass valve will be in charge of the fast dynamics (quickly cancelling disturbances or achieving set-point changes with controller $R1$) and a slow loop ($R2$) will drive the bypass valve to a reasonably efficient operating point at 50% opening of valve V_s, for example, by adjusting the steam valve. As there exists an additional actuator (the main inflow valve V_0) the proposed cascade structure may be used as a part of a multi-loop control strategy, if so wished, to follow temperature and an additional flow rate set-points (although some coupling does occur between flow and temperature). Combined strategies are discussed in Section 5.6.1.

5.5 Other Possibilities

Several additional alternatives (and combinations of them) are widely used in industrial multivariable control structures. Some of these will be briefly outlined.

5.5.1 Indirect and Inferential Control

Indirect control refers to controlling a set of variables, z, by controlling other secondary ones, y, with best dynamic characteristics and/or cheaper sensors. Set-points for the secondary variables are obtained by means of a suitable model relating y with the primary variables z, for which control objectives are fixed.

Of course, the viability of the indirect approach will depend on the quality of the model and the influence on it of unmeasurable disturbances and plant variations.

Indeed, indirect control is implicitly used in practically all control systems, as the controller is trying to regulate the electrical signals coming from the sensors, on the basis that they are closely and disturbance-free related with the actual plant outputs. However, by realising that fact, the control designer can use the same idea in more general situations. Figure 5.8 depicts the set-up.

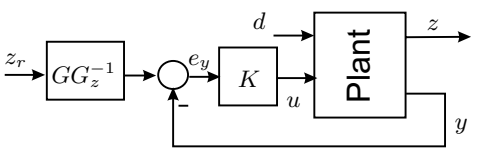

Figure 5.8. Indirect control: use y to control z

If a model is available, some tests [119] can be carried out to determine if y is a good indirect measurement to control z. Let us express the open-loop equations as:

$$y = Gu + G_d d \tag{5.28}$$

$$z = G_z u + G_{zd} d \tag{5.29}$$

where d is an unmeasurable disturbance vector. If a desired z_r should be achieved, the ideal control action would be $u^* = G_z^{-1}(z_r - G_{zd}d)$. Substituting into the first equation, one gets as an ideal target output:

$$y^* = GG_z^{-1} z_r + (G_d - GG_z^{-1}G_{zd})d \tag{5.30}$$

However, as d is unmeasurable, the set-point should be $y_r = GG_z^{-1}z_r$ and $G_d^* = G_d - GG_z^{-1}G_{zd}$ should be small enough (evidently, this is not needed for the measurable components in d, if any).

Furthermore, the controller may not perfectly follow the above set-point y_r, due to disturbances, modelling errors, stationary errors, measurement noise, *etc.* So, the output y should also be chosen in such a way that the loop error, $e = y_r - y$, does not have an important effect on the achieved z. In particular, if the achieved y were $y = y_r - e$, the control action would be, by (5.28), $u = G^{-1}(GG_z^{-1}z_r - e - G_d d)$, so, replacing it in (5.29), the resulting deviation on the achieved z would be:

$$z - z_r = (G_{zd} - G_z G^{-1} G_d)d - G_z G^{-1} e = -G_z G^{-1}(e + G_d^* d) \qquad (5.31)$$

So, indirect control will be successful if both control error in y is small and $G_z G^{-1} G_d^* = G_z G^{-1} G_d - G_{zd}$ is small enough in the frequency ranges of interest[8]. Different choices of measurements will lead to different achievable loop precision (e) and disturbance effects (G_d^*).

Example 5.13. A controlled variable $z = u + d$ is in itself inaccessible, and two candidate measurements $y_1 = 6u + 2d + p$, $y_2 = u + 1.1d + q$ must be evaluated for indirect control. Disturbance d has an expected worst-case size of 1 unit. p and q are other perturbations, sensor non-linearities and measurement noise. Experience of the control designer assesses that controlling y_1 would achieve an error of about 5 units, and controlling y_2 the loop error would be about 3 units. Evaluating $G_z G^{-1}$ yields 1/6 and 1 respectively, and G_d^* yields -4 and $+0.1$. Hence, the worst case error in z may, approximately, be calculated as: $e_{z1} \approx -1/6 * (5 + 4 * 1) = 1.3$ and $e_{z2} \approx 1 * (3 + 0.1 * 1) = 3.1$. Hence, measurement y_1 seems a better choice.

A variation of the idea is the so-called *inferential control* [123], where measurements of y and the model are used to reconstruct an estimation of z as sensor input to the regulators. So, in Figure 5.8 *after* measurement of y, a block with equation $\hat{z} = G_z G^{-1} y$ is inserted and a regulator is designed for the G_z subsystem, with no set-point transformation. Note that in this case, the correction is inside the feedback loop, contrary to the indirect control framework, where the correction is applied to the set-point in a feedforward operation.

In many cases in practice, the availability of these transfer functions is difficult. Most cases of indirect control are decided on the basis of engineering common sense and estimates of the DC gain values of $G_z G^{-1}$ and $G_z G^{-1} G_d - G_{zd}$.

Remark 5.14. If the DC gain characteristics of G, G_z, G_{dd} or G_{zd} are not correctly estimated, there will be a DC offset on the indirectly controlled variable z. Zero steady-state on that variable is possible if it is measured, perhaps infrequently, and the deviations are used to correct the set-point of the measured variable. Indeed, this is the case in many composition measurements in chemical processes, where on-line control is carried out on an easily available physical property (light absorption, conductivity, density) and its set-point is

[8] $G_z G^{-1}$ will not usually be "small": for example, an ideal case would have $G = G_z$.

updated based on more precise product analysis (taken with an interval of minutes to hours) in a sort of outer integral-action loop (perhaps manual). Generalisation of these ideas gives rise to the so-called non-conventional-sampling control (Section 9.4).

Example 5.15. Indirectly controlling the concentration of a particular compound or microorganism via measurement (and feedback control) of light absorption in a particular spectrum is common practice in some chemical, pharmaceutical and food industries. This strategy needs careful calibration of the sensors so that a (in most cases, non-linear) static function relating absorbances and concentrations can be used to set absorbance set-points. However, that model may be affected by disturbances such as ambient temperature (affecting the photodetector sensitivity), accumulation of dirt on the pipes, impurities, *etc.* so either a cleaning and maintenance schedule or a re-calibration one should be carefully followed.

Example 5.16. A flash separator heats light hydrocarbons (mainly methane, ethane, butane plus three other components), lowers the pressure and separates liquid and vapour phases inside a vessel, and the primary controlled variable is the ethane concentration in the product liquid phase.

The product composition is a function of the feed composition, pressure and temperature in the vessel. If on-line analysis of ethane concentration is discarded, temperature and pressure set-points are calculated based on suitable liquid-vapour thermodynamic tables.

To assess the validity of, for example, temperature (assuming good control on pressure) as an "inferential" variable, experiments or calculations are carried out to determine, at least at steady-state:

1. How temperature variations relate to composition variations. This, multiplied by the expected temperature loop error, will indicate composition errors ($G_z G^{-1} e$).
2. How expected variations in feed composition (d) would influence the desired temperature set-point ($G_d^* = G_d - G G_z^{-1} G_{zd}$).

If any of these quantities is deemed excessive for a desired ethane concentration precision, there would be a need for an on-stream analyser. Matrix (singular value) analysis would be required if both temperature and pressure errors were considered.

Anyway, the temperature set-point must be corrected to achieve zero ethane concentration offset. This can be done either manually by an operator, after results from laboratory analysis, or automatically if a downstream analyser (operating at a lower latency than the control loop due, for example, to delays or to being a shared sensor also used in distillation control) is used.

These ideas are closely tied to the problem of secondary sensor and actuator selection for achieving primary objectives, being relevant even in the project phase.

5.5.2 Override, Selectors

In some situations there are several outputs to be controlled with only one available actuator. Of course, there is the option of controlling, in a SISO

set-up, a weighted sum of them or any other combination, but in many cases the control objectives on these outputs can be "prioritised" in some way.

A typical example of this case is given by a combination of performance and fault-tolerance objectives. For example, a central heating system where not only is a particular water temperature desired as a primary reference but also the requirement to keep pressure under a maximum safe value.

These issues are dealt with by using *selector blocks* in the control structure, usually carrying out *maximum* and *minimum* operations on a set of signals. There are two basic configurations:

- *Output selectors.* The sensor to the control loop is the maximum or minimum of a set of similar variables (able to be controlled with the same regulator).
  For example, this would be the case in a fired heater if a main heating power input, u, is used to keep the maximum temperature of a *set* of sensors at a prescribed value. This block is sometimes named as the "*auctioneer selector*".
- *Input selectors.* In this configuration, several controllers compute a control action value for individual sensors and set-points. Afterwards, the selector chooses the minimum or maximum value to be actually applied to the plant via a single actuator. This configuration is usually denoted as *override selector.*
  An example of this case is using the heat input, u, attached to a temperature loop in normal conditions except when the pressure is too large and a pressure controller outputs a lower heating power command, overriding the temperature controller. In this case, the selector would carry out a *minimum* operation. Similar situations arise, for example, in mixing tanks with a primary concentration reference and an override level control for tolerance to unexpected disturbances or faults, to avoid tank overflow.

Override selectors are indeed a common configuration in complex plants with strong safety requirements. In fact, most control systems incorporate *operator override* (manual control) for direct actuation under unexpected circumstances. The issue of automatic/manual bumpless transfer is further discussed in Section 9.3.

Although these issues have been discussed in the framework of decentralised control, the control engineer may need to enforce these requirements also into centralised control subsystems.

5.5.3 Split-range Control

In some cases, actuator constraints or non-linearity make it necessary to add an extra input. These configurations are denoted as split-range control. In this case, the selectors act as logic-switches, carrying out other operations than the minimum and maximum previously discussed.

For example, amplitude constraints can be overcome by using an extra second actuator if the control command is above a particular limit (if the use of two individual actuators is justified with respect to the use of a single but more powerful unit in terms of achieved precision and price).

Other non-linearities apart from saturation may force the use of split-range actuators or sensors:

- different sensors may be advisable for measuring a particular variable with improved precision in different operating regimes. This would also be the case if the linearised state space model becomes weakly observable for a particular sensor configuration, C, in a certain operating point but an alternative C' is available.
- different actuators may be needed in different operating regimes. For example, if a particular configuration of manipulated variables B on the linearised model yields a nearly uncontrollable system (and different actuator locations or physical phenomena B' can provide a more efficient strategy). Also, precision of the actuator at different power levels can drive towards this kind of decision.

The implementation of this scheme can be based on ON/OFF switching between different sensors, actuators and controllers or interpolated in a more smooth way in a gain-scheduling framework (see Section 9.5.2).

The most straightforward and widely used example of this strategy is the use of different actuators for heating/cooling in temperature control systems, but the idea may apply to other kinds of processes.

5.5.4 Gradual Control, Local Feedback

In some kind of manufacturing processes, several processing stages can be set up to gradually improve specifications over the same variable, using actuators with different power and precision and sensors with different precision and range.

Example 5.17. Neutralising a significant flow of liquid with precision is not easy. A first neutralisation mixing tank may achieve a pH in the interval (5.5,8.5) for its outflow product (coarse control), and then a second stage with smaller pipes and more precise valves could achieve a final pH in the range (6.5,7.5). This could be a solution with cost advantages with respect to a unique neutralisation stage with high flow rate, high-precision valves.

The issue in this case is, of course, the trade-off between the performance improvement against the cost of the extra equipment.

5.6 Sequential–Hierarchical Design and Tuning

In cascade and some multi-loop structures, loops are designed sequentially, giving rise to hierarchical control designs.

First, a lower-layer (local, inner) control system is designed for controlling a set of outputs, y_2. Next, with this inner control in place, the design is continued to achieve control of other outputs, y_1. Table 5.1 summarises some of the previously discussed possibilities (y_2 is assumed to be measured) [119].

Table 5.1. Summary of decentralised alternatives. Primary variables (y_1, with control objectives), secondary variables (y_2, always measured)

	Measurement of y_1	Control objectives for y_2
Sequential multi-loop control	Yes	Yes
Cascade control	Yes	No
Partial control	No	Yes
Indirect Control	No	No

Some simplifying assumptions in hierarchical decompositions are usually made:

- use simple regulators for the inner loop,
- use longer sampling intervals for master loops (see Section 9.4),
- use simple models to design the outer loops (assuming the inner loop is in control, its deviations from the reference model – the simplest one would be a gain – are small, if the inner loop bandwidth is higher than the outer one),
- use inner loops to stabilise or diminish some particular disturbances on internal variables (local disturbance rejection) and leave outer loops to open-loop or manual control (true "partial" control),
- variables in y_2 and actuators must be chosen so once under good control they do not impose unnecessary control limitations (RHP poles and zeros, ill-conditioning) on the remaining problem.

The considerations here have been restricted to a two-layer set-up but extensions on similar grounds can be made in a multi-layer hierarchical structure.

Cascade control. In cascade control, y_2 usually denotes secondary outputs, and y_1 denotes primary variables for which there is a set-point and whose regulators provide set-points for y_2.

Multiloop control (sequential loop closing). In sequential multi-loop control, there are control objectives for both sets of variables.

Of course, if a model is available, sequential loop closing (SLC) method is one well-known method for tuning multi-loop control systems for multivariable processes: each controller is designed sequentially with SISO methods by finding the transfer function for the paired input and output after former loops have been closed [39, 60]. As further loops are closed, the transfer function of the previous ones changes, so the tuning sequence must be repeated until convergence is reached [60].

Ideally, in a *triangular* plant, the variables to be first tuned would be those not influenced by the rest, and sequentially, the loops on upper variables will be closed one by one. If a triangular structure exists, control errors on lower-hierarchy loops will affect the current one as *disturbances*. In a general case, as a rule of thumb, faster loops are tuned first[9]. To reduce the effect of later loops, in the tuning of the lower ones approximate transfer functions of the to-be-closed loops may be included [39].

If the system exhibits integral controllability with integrity (ICI), then the order in the sequence will not affect the final result for *slow* target bandwidth. In that case, the controllers can be *iteratively tuned*, even on-line: first, all regulators are tuned to slow specifications (low-gain PI) and subsequently, specifications are increased in the loops, one at a time, until coupling effects start to appear, hindering further improvement. If the system does *not* exhibit integrity, then the order of start-up and tuning of the several loops is indeed relevant.

In a general case, the final performance achieved depends on the sequence of loop closing and the suitability of the selected pairings, and there is no integrity guarantee. This will not happen if the plant is approximately diagonally-dominant.

Example 5.18 (2×2 tuning.). Based on (5.8), if the bandwidth of loop 1 is much faster than that of loop 2, K_2G_{22} will be small at crossover for loop 1. So, $CE(s)$ will approximate to $1 + K_1G_{11}$ if λ_{11} is not small either. So, the faster loop can be closed first, disregarding interaction.

If loop 1 is much slower, then K_2G_{22} will still be "big" at the loop 1 crossover frequency (as, for example, integral controllers have high gain at low frequencies). In this case,

$$CE(s) \approx 1 + K_1G_{11}\frac{1}{\lambda_{11}}$$

so the slower loop must be designed taking into account the change of dynamics (at least, approximating by the gain $\lambda_{11}(0)$) given by closing loop 2.

Example 5.19. The idea in the above example is implicitly applied in the combined multi-loop level and pressure control in the steam boiler case study: strong coupling and RHP zeros do not easily allow fast control on both loops, so a decision is made aiming for a fast control on the pressure loop and a slow, sluggish loop on the level control. See details in Section 5.8.1 on page 159.

5.6.1 Combined Strategies for Complex Plants

Note that mixed strategies, incorporating several of the possibilities discussed in previous sections, are possible. They are, in fact, necessarily implemented in the overall design of a complex control structure. The division on subsystems and subtasks, jointly with supervision, diagnosis and communications constitute the basis of the so-called *plant-wide control* [46].

[9] If no model is available, the Ziegler-Nichols ultimate frequency (stability limit with proportional control) may be used for ranking.

Example 5.20. In the heat exchanger in Example 5.12, if performance needs to be improved, a good solution would be to control the two exchanger valves with an additional *slave* cascade loop, by adding mass-flow sensors, for instance. If the heated fluid needs to be stored in an insulated tank, this control system would be a decentralised subsystem that would control temperature simultaneously with another flow control system governing a main valve, in a multi-loop structure. Flow commands will come from a *master* cascade controller carrying out level measurements in a downstream tank.

In a general situation, for a complex production facility, these subsystems are also a slave cascade control of other chemical process that requires different temperatures of a fluid to compensate for purity/flow/temperature variations of other materials in a downstream reactor.

The expansion is endless, and a whole plant can be controlled with multiloop + cascade + decoupling structures given the sufficient amount of sensors and actuators is available. This is, in fact, the *de facto* way of designing control systems in process industry, and only some subsystems (if any) are "upgraded" to predictive control or other (robust-control) centralised techniques.

Example 5.21. A typical boiling water reactor nuclear power plant has a large number of sensors (operators like to be informed about what is happening), and the operators in the control room are the "masters" of about a dozen "key" loops that themselves are masters of more than 120 "important" regulators controlling flows, temperatures, voltages, flow levels, *etc.* embedded into multi-loop, cascade and feedforward structures, conformed into several main subsystems: nuclear reaction control, steam quality control, control of output electrical magnitudes, *etc.* Operators are also in charge of multitude of diagnostic and supervision tasks, of course. A wise design on the plant-wide level enables coordination of hundreds of elementary regulatory loops towards the overall objectives.

5.7 Key Conclusions

In this chapter, the control of systems with multiple inputs and outputs is discussed using SISO-based tools, either directly (such as in multi-loop or cascade control) or after some multivariable decoupling transformation.

Multi-loop strategies, if suitable, may present the advantages of integrity (fault tolerance), as well as striking simplicity. However, in some cases, tuning may be difficult and coupling may severely limit their performance.

Decoupling is based on mathematical transformations of the system models into diagonal form. Feedforward decoupling can be used in many cases. Feedback decoupling achieves its objective if state is measurable and system is minimum-phase. However, decoupling may be very sensitive to modelling errors and it is *not* the optimal strategy for disturbance rejection tasks.

Cascade control is widely used in industry to improve the behaviour of basic SISO loops via the addition of extra sensors and actuators. However, ease of tuning requires that different time constants are involved in different

subsystems. In general, addition of extra sensors and actuators in a SISO or MIMO loop, even if no primary set-point needs to be fixed in principle for them, will improve achievable performance and/or tolerance to modelling errors (see Chapter 8). The level of improvement must be traded off against the cost of additional instrumentation.

Combining the above strategies, plus override selectors, split-range *etc.* is usually required in most real-world, industrial-scale projects. Division in subsystems with independent control for them is key to the success of a large-scale design. Each subsystem can itself be controlled in a decentralised or a centralised way, depending on the suitability of the above techniques and the availability/price of the instrumentation.

5.8 Case Studies

In this section, some case studies on the topics of the chapter will be presented. The interested reader can find further control-structure selection case studies in, for example, [119, 126, 38, 129, 105], mostly in the chemical engineering framework. Feedback decoupling is illustrated in the case study in Section 6.5.

5.8.1 Steam Boiler

A case study on a decentralised control structure for a steam boiler will now be discussed. The implemented structure combines multi-loop, cascade, override and ratio control set-ups.

Steam has been, over two centuries, the most widely used heat transport fluid. It is used for power generation and process operations. Power generation requires high-temperature, superheated steam for optimum efficiency. Process applications require lower steam temperature, closer to saturation. If several steam characteristics are needed from a single boiler, downstream superheating/desuperheating stations may be needed.

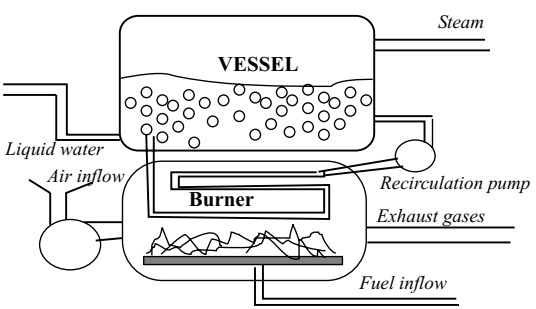

Figure 5.9. Process to be controlled

In industry, control instrumentation in boilers must provide efficient, reliable and safe operation for long periods of time. A possible schematic representation of a typical steam boiler is depicted in Figure 5.9. The objective is to design a low-cost control system ensuring suitable quality (temperature and pressure) of the produced steam, with the basic energy source of a propane or fuel burner to bring to boil the water inside the main vessel. A recirculation pump is used to achieve higher heat-transfer constants. More complex systems can have several take-off lines operating at different pressures.

The control system must act in open or closed loop on a series of pumps and valves to attenuate the effect of perturbations caused by changing demands on the outflow steam characteristics (mass flow, set-point temperatures, pressures) and changes in fuel composition or pressure, *etc.*

Preliminary study. A detailed analysis of drum boiler dynamics and models is made in [18]. Some basic ideas will be discussed below, mainly from the referred source.

The burner provides the heating power to heat the drum boiler. The mass and thermal capacity of the metallic elements is indeed comparable (even bigger) to that of the water stored in the vessel (typically from one to four times). Some low-order models can be devised under reasonable engineering assumptions with satisfactory results [18, 19]. More detailed models exist, based in finite-element heat transfer equations.

Pursuing a model-based approach will require a significant cost in the modelling and experimental parameter identification phases: a MIMO model with second- or third-order temperature dynamics, plus valve dynamics, nonlinearities, radiation, level dynamics (complex, non-minimum-phase due to the steam bubbles inside the liquid phase) *etc.* make a hard task in achieving a faithful model from first-principle knowledge.

Of course, a model-based design could be pursued anyway and, indeed, even with the decentralised PID-like controller to be designed, extensive simulation studies with a good non-linear model are advantageous, if possible, before actual implementation. However, the decentralised structure design will be based on the main characteristics and physical insight instead.

Steam inside the vessel is approximately saturated, as there is liquid–steam equilibrium at least in steady-state. The main heat transfer phenomenon is the use of the heating power to supply the needed latent vaporisation heat to transform liquid into steam, and also to store energy in the form of gas pressure and temperature. The expansion after the outflow valve yields a superheated steam so a certain amount of energy loss during steam transport can be tolerated without condensate deposits on horizontal pipe segments; these are able to cause significant perturbations and damage to the installation. Condensate in steam lines and cavitation in condensate liquid return lines require a careful piping design.

Besides issues relating to outflow steam quality, there are safety-related specifications regarding allowable pressures and liquid level inside the boiler

vessel. The reasons for pressure limits are clear. Regarding level control, a too-high level may drag liquid into the outgoing stream line, and a too-low one may cause overheating of some boiler parts. Some studies (see [18] for reference) report that 30% of the emergency shutdowns in French pressurised water nuclear reactors are caused by poor level control.

Selection of Variables to Control

In the boiler in Figure 5.9 there are several possible sensors: outflow pressure and temperature, vessel pressure and temperature, liquid level, mass inflow and outflow, *etc.* Actuators are a set of valves and pumps regarding air, fuel and cold water feed, recirculation pumps, output steam line, *etc.* The problem of sensor selection will be discussed simultaneously to the selection of the control structure, as the regulators used will be mainly SISO ones. Different sensors and actuators could be needed depending on precision and efficiency requirements.

In practice, besides choosing which variables to measure and manipulate, a specific manufacturer should be chosen based on its reported instruments characteristics: price and linearity, calibration requirements, bandwidth, physical operating principle, reliability and maintenance, operating range, and possibilities of on-line signal processing, communications and self-test for faults.

Main Steam Control Set-up

The basic structure will be based on a multi-loop approach, depicted in Figure 5.10, on the following pairings:

- outflow pressure loop: outflow stream pressure, governing the outflow valve,
- level loop: level sensor, governing inflow valve (see below for details),
- heat loop: outflow enthalpy (from one or several candidate readings of mass flow, pressure and temperature), governing heating power command.

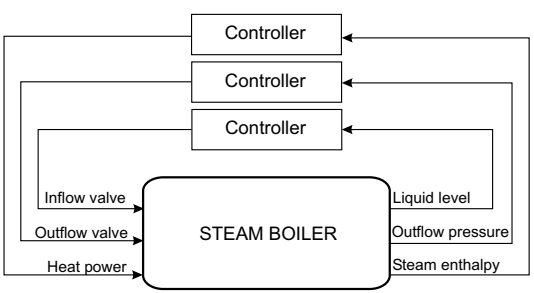

Figure 5.10. Basic multi-loop control structure

The basic structure has been selected, among other possibilities, on the basis of physical insight as no model is available: we are "forced" to pair the outflow pressure to its valve for a quick response and, then, level to liquid inflow. Note, however, that pressure and temperature are expected to be significantly coupled (P–V–T relationships in steam equations), and level will also be deeply influenced by pressure changes (see below). The structure needs to be further refined, as discussed next.

Heat Loop

If outgoing steam pressure is controlled by the output valve, a fully decentralised approach would only need a temperature or outgoing steam mass-flow reading to govern heat power command towards the combustion subsystem for successful control.

To improve performance, however, mass-flow, temperature and pressure readings could be combined to calculate an enthalpy balance to produce a more correct heat command. This heat command (to the water) should be initially overshooted to approximate invert the metallic plant dynamics.

Of course, a simpler approach using a set of adjustable linear gains to approximate the basic idea could be used. For example, in Figure 5.11 a combination of temperature and mass-flow reading is used as a candidate structure using linear regulators.

The mass-flow reading is multiplied by the vaporisation latent heat to obtain the needed increment of heat power. Dividing by the fuel calorific power (joules per burnt kilogram), a conversion to a flow command is achieved.

A PID temperature controller would try to achieve a reasonable profile for desired temperature changes, but it is expected to have a slow response. The proposed structure might not be the optimal one.

Vessel Pressure Control

As the outflow valve governs outgoing steam pressure, nothing prevents at this stage an internal pressure build-up if, for example, downstream machinery is disconnected and outflow is suddenly substantially reduced.

The important safety requirements regarding vessel pressure strongly advise the presence of a pressure regulator (with a vessel pressure sensor), which will command a reduction of heat power if pressure is above a user-defined safe limit. However, it is desired that the controller should not operate if pressure is below the limit.

So, both the steam condition and pressure controllers must be suitably combined by means of a *minimum selector*: the final heat power command to the combustion system will be the minimum of that from the pressure loop and the one from the outgoing steam temperature (enthalpy) controller. If pressure is well below the limit, the pressure loop will command a saturated control action, so heating to just achieve the desired steam conditions will be selected.

Level Control

The main problem in boiler level control is that the presence of steam bubbles below the liquid level causes shrink-and-swell phenomena. For example, addition of cold water produces a momentary lowering of the level reading, although, of course, in the long term the water level must increase: the level dynamics is non-minimum-phase.

Changes of internal pressure also affect level, so there is a significant coupling between those variables: a pressure decrease (opening the outflow valve) produces swelling of the bubbles, yielding rising level readings. However, as liquid is then in an overheated state, the steam production rate quickly increases, so in the long term, the level diminishes if no control is applied.

Due to the coupling and non-minimum-phase behaviour, plus the non-linearities from valves and steam tables, the specifications that can be obtained in practice are very poor indeed, further worsened by the high amount of noise present in sensor readings (caused, again, by the bubbles and turbulence).

Noise forbids derivative action, and non-minimum-phase limits bandwidth. A common choice is a low-gain feedback PI controller. Any improvement would need a substantial modelling effort, non-linearity cancellation, *etc.* and will also meet with the noise and non-minimum-phase fundamental limitations (see Sections 8.3.1 and 8.3.2) so the model-based approach will not, in many cases, significantly improve the achieved results in practice.

To, at least, overcome cold water feed valve non-linearity, a cascade controller can be used (so the level controller provides a master inflow reference) if a flow sensor is placed at the input pipe.

However, in this way, disturbances are poorly cancelled. As level control is usually carried out mainly for safety reasons, level specifications are not strict and significant transient deviations are allowed.

To improve overall performance, requiring lower PI gains, a 2-DoF structure may be used, by adding a feedforward control measuring output steam mass flow, and directly incrementing the inflow reference to the cascade regulator by the same amount. In this way, disturbances are cancelled in the long term with very low feedback gains.

With the presented "final" configuration, depicted in Figure 5.11, the fast shrink-and-swell dynamics is intentionally, *not* controlled (anyway, even if it were modelled, the actuator bandwidths might not be enough).

Control of the Combustion Subsystem

The combustion subsystem will implement a common structure usually denoted as *ratio control*. The objective is to control both the fuel flow and the air-fuel ratio. Similar ratio-control situations arise in mixing processes (such as addition of colourants or reactives).

In the boiler case, the characteristics of the produced steam will in principle point out the need to increase or decrease fuel flow.

The optimal air-fuel ratio can be calculated from stequiometric calculations. For example, for every litre of propane, 5 litres of oxygen (25 of air) are needed for perfect combustion. However, as the mixing and combustion processes are not ideal, the desired air–fuel ratio is slightly incremented to diminish the percentage of unburnt combustible feed and increasing efficiency. If the air–fuel ratio is excessive, however, the combustion is inefficient as well, as the temperature achieved will be lower and part of the heating power is being used to heat air that will be discarded on the exhaust outlets (unless some kind of hot-air management is set up for energy saving).

The maximum efficiency point is thus determined by striking a balance between the percentage of unburnt products and the percentage of energy not transferred to the vessel. In fuel burners, unburnt components also produce significant soot depositions.

The fuel subsystem could be designed on the basis of one sensor (temperature or pressure of the steam in some location) and two actuators (airflow, fuel flow). However, that would have two significant drawbacks:

- the efficiency will not be guaranteed in "closed loop" as there is no measurement of the combustion parameters. As in large-scale installations, fuel properties (mainly in oil or coal burners) vary significantly, so does the maximum efficiency ratio.
- The effect of disturbances on the fuel subsystem (regarding air and fuel temperature, pressure and composition) would be apparent only when, after a significant delay, they affect the steam characteristics. In that way, they will be difficult to compensate for.

So, based on these considerations, results improve significantly if additional sensors on the fuel subsystem are put into operation. Control structures that handle them will be:

- cascade control: steam quality will determine the heat power needed (directly translated into fuel flow), and subordinate regulators will determine the rest of the combustion subsystem actions. In particular, a *fuel flow sensor* will be used for cascade control in the fuel valve,
- Ratio control: the achieved fuel flow (sensor reading) will entail, given a pre-fixed ratio, following an airflow reference by a subordinate cascade controller using an *inlet airflow sensor*.

If efficiency is a concern, as fuel characteristics produce significant variations on the optimal air–fuel ratio, an additional *oxygen sensor* may be put in place: efficient combustion implies that the burnt air should contain around 4–5% oxygen left[10]. So, the ratio reference (and, subsequently, the air inflow set-point) is increased or decreased in closed loop based on the oxygen reading. As an alternative, a sensor for carbon monoxide (CO) can be set up to serve essentially the same purpose.

[10] Ideally, it should be zero, but inefficient mixing produces significant soot and carbon monoxide if exhaust oxygen is below the above stated rate.

As a cheaper variation, an exhaust gas temperature sensor could serve determining the correct combustion point. However, as its set-point is determined by the inlet air temperature and humidity, and also depends on fuel composition, its capability for improving combustion efficiency is quite limited, but it could marginally improve rejection of some disturbances. A similar function to the oxygen sensor could be achieved by an smoke opacity sensor (opacity increases with the amount of soot and unburnt compounds). Inserting several of these sensors will help in detecting faults on the system (see Section 9.6).

So, to achieve a heat power command from a hierarchically higher main control loop, a moderately sophisticated combustion control subsystem uses two flow sensors (air, fuel), two servovalves (air, fuel), and one oxygen or carbon monoxide sensor on exhaust gases, in a mixed cascade and ratio-control setup.

Other Subsystems and Conclusions

Large-scale boilers have additional subsystems to be controlled, further complicating the overall control structure. These subsystems are, for instance:

- fuel pre-processing in oil burners. Variations in fuel viscosity, humidity, *etc.* may need to set up a control system for:
  - pressure in the fuel supply line,
  - fuel temperature, using the residual heat of the exhaust gases,
  - water/fuel ratio (mixing and emulsion control). Addition of a small quantity of water makes viscosity and the spraying and atomisation characteristics uniform (better flame quality),
- feed water treatment, in two lines:
  - to avoid impurities that, deposited on the exchanger pipes, will diminish heat transfer rate, causing overheating and deformations on metallic components. Obstructions due to impurities may entail damage to the system and safety risk for operators. The treatment is carried out by filtering and monitoring water resistance to divert water, if needed, to decalcification or demineralisation units,
  - pre-heating the incoming water causes much lower disturbances on the level loops, so a further temperature control can be put in place.

So, to conclude, a decentralised control structure has been proposed based on physical insight into the underlying process, using cascade, feedforward, feedback and ratio control blocks, to achieve reasonable disturbance rejection and robustness as well as minimum selectors on heat power for safety. The overall control system is depicted in Figure 5.11.
The final design uses:

- eight sensors: (outgoing steam temperature, pressure and mass flow), oxygen in exhaust gas, flow sensors for air, fuel and feed-in liquid water and a vessel pressure sensor,

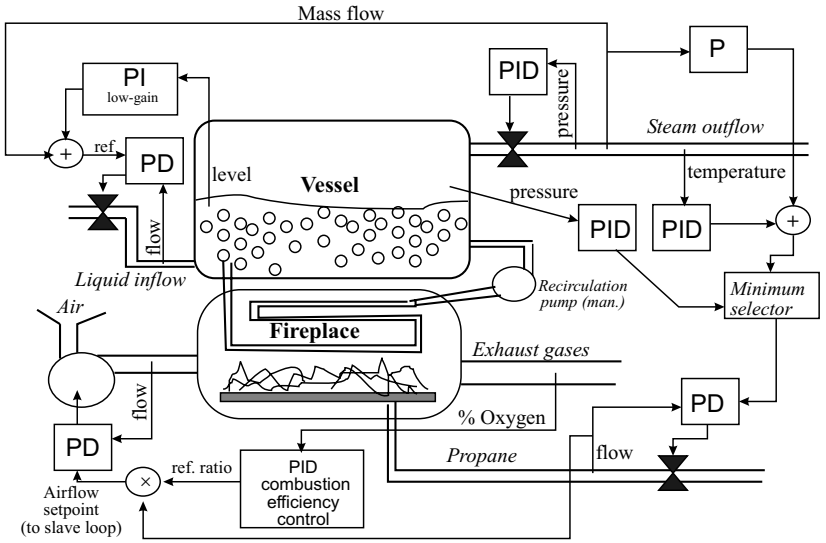

Figure 5.11. Proposed control structure

- four actuators: valves or pumps governing fuel, steam, liquid and air flows.

Further sensors in the additional subsystems could be needed for large-scale plants. Furthermore, additional sensors such as temperature of critical metallic parts or exhaust gas, *etc.* could be installed for safety and monitoring reasons or to further refine the control structure. Reliability issues would also advise incorporation of redundant sensors or actuators.

A wise choice of sensors, actuators and control structure allows easy comprehension of the overall regulation strategies by operators, significant reliability and the use of standard equipment. A centralised regulator (an 8×4 transfer function matrix) tuned with a methodology such as the ones in Chapters 6 and 8 would have the proposed structure as a "particular case". However, improving performance with reasonable model error tolerance would require, in most cases, a substantial effort.

5.8.2 Mixing Process

Now, a pairing selection and decoupling problem will be illustrated for a mixing process [89].

A solvent and a pure component A are mixed in two converging pipes. Actuators are a valve on each flow (V_s, V_A). Outflow q and concentration x_A sensors are in place (Figure 5.12).

The basic equations at steady-state, and its linearisation around a desired operating point (x_{a0}, q_0) are:

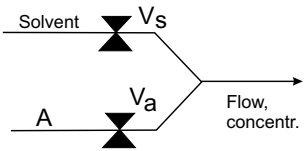

Figure 5.12. Mixing process

$$\begin{array}{ll} q = m_1 V_s + m_2 V_A; & \bar{q} = m_1 \bar{V}_s + m_2 \bar{V}_A \\ x_a = \frac{m_2 V_a}{m_1 V_s + m_2 V_A}; & \bar{x}_a = -\frac{x_{a0}}{q_0} \bar{V}_s + \frac{(1-x_{a0}) m_2}{q_0} \bar{V}_a \end{array} \tag{5.32}$$

We will assume $m_1 = m_2 = 1$. For a first model, dynamics of the valves have been neglected, as well as conditions regarding pressure drops (q is a non-linear function of the valve openings).

Pairing

If the set-point is at low concentrations of A, then V_A to control composition and V_s for total flow are advisable, as composition varies greatly with the amount of flow of pure A. For example $x_{a0} = 0.05$, $q_0 = 1$ leads to a DC gain and RGA given by:

$$G = \begin{pmatrix} 1 & 1 \\ -0.05 & 0.95 \end{pmatrix}; \quad \Lambda = \begin{pmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{pmatrix} \tag{5.33}$$

confirming the intuition. On the other hand, if the set-point is at high concentrations of A, the roles are reversed and now strong influence on composition changes is exerted by valve V_s. In this case, the gain and RGA matrices are:

$$G = \begin{pmatrix} 1 & 1 \\ -0.95 & 0.05 \end{pmatrix}; \quad \Lambda = \begin{pmatrix} 0.05 & 0.95 \\ 0.95 & 0.05 \end{pmatrix} \tag{5.34}$$

confirming the suitability of the alternative pairing. If a wide range of set-points must be attained, maybe multi-loop control is not the optimum choice.

Decoupling

To achieve approximate decoupling at steady-state, the inverse DC gain is (with $m_1 = m_2 = 1$):

$$\begin{pmatrix} \bar{V}_s \\ \bar{V}_A \end{pmatrix} = \begin{pmatrix} 1 - x_{a0} & -q_0 \\ x_{a0} & q_0 \end{pmatrix} \begin{pmatrix} \bar{q} \\ \bar{x}_a \end{pmatrix} \tag{5.35}$$

So, that is the matrix that should be used to achieve decoupling. Intuitively, if flow needs to be increased, both valves must be opened proportionally to the composition, to keep it unchanged. If composition needs to be increased, V_A should be increased and V_S decreased, to keep flow unchanged.

If wide set-point variations are expected, two alternatives are possible[11]: setting a fixed decoupler for an intermediate point ($x_{a0} \approx 0.5$, for example) or changing the decoupling matrix as a function of the known operating points. In this way, a partially non-linear controller is designed (linear in the sensors, non-linear – gain-scheduled, see Section 9.5.2– in set-point processing).

[11] Another alternative is to directly invert the non-linear equations in (5.32). Indeed, with $V_S = (1 - x_a)q$ and $V_A = x_a q$, suitable mixing does occur. This achieves decoupling in all operating ranges, but the product is not a linear operation. This is an example of straightforward non-linearity cancellation (see Section 9.5).

6

Fundamentals of Centralised Closed-loop Control

In this chapter, some now-classical centralised state and output feedback strategies are described, referring to dedicated books for further information.

Centralised control is implemented via computer code, having as inputs all the available sensors and producing signals for all available actuators in the system. This is the most powerful strategy, at least in theory, capable of extracting "optimal" performance (for some definitions of optimality). In practice, however, it requires non-standard equipment (industrial computer, data acquisition cards, communications), its tuning is usually non-intuitive (involves matrix computations) and in case of a fault the whole system may break down. This is one of the reasons why it is not widely used in industry. However, for complex strongly coupled plants, this one may be the only solution with a limited set of actuators and sensors.

Centralised control can be implemented either in open or closed loop. As the open-loop case (set-point tracking) has been considered in previous chapters, in this chapter emphasis will be placed on the solution of the regulation problem. Combined set-point tracking and regulation will be dealt with in Section 6.3 (1-DoF configuration) and Chapter 8 (2-DoF configuration).

Let us analyse the feedback loop using the state space representation, first in a simplifying framework and afterwards in a more general case.

6.1 State Feedback

The first case to be studied will be the one in which the number of independent sensors is equal to the order of the system. This is denoted as *state feedback* or *full-information* control. The system is described by (2.17):

$$\dot{x} = Ax + Bu \tag{6.1}$$

where x is assumed accessible (via n independent sensors so that matrix C in $y = Cx$ is invertible) and, without loss of generality, we will assume that

$C = I$. In Section 6.2, other alternatives will be considered, under the name of *output feedback*, when the number of sensors or deficient signal-to-noise ratio do not allow for a full-information implementation.

Linear time-invariant state feedback control is expressed[1] as:

$$u \stackrel{\text{def}}{=} -Kx + r = -\begin{pmatrix} k_{11} & k_{12} & \dots & k_{1n} \\ \vdots & \vdots & & \vdots \\ k_{m1} & k_{m2} & \dots & k_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + v \tag{6.2}$$

where v is an additional $m \times 1$ input term to be used, for example, to position the system around a nominal set-point where the linearisation is valid (if v is a constant $v = G(0)^{-1}r$, being r a set-point vector) or to carry out additional feedforward control tasks (Section 4.6).

Remark 6.1. State feedback is the natural generalisation of conventional proportional and proportional-derivative control for low-order SISO systems. Indeed, under particular assumptions (see Section 3.7.5), there is a canonical representation where a state vector can be constructed as the reading of a certain sensor and its successive derivatives (generalising, for example, the position + speed feedback in a 2nd-order mechanical system). In many cases, a PD $u = K_P e + K_D \dot{e}$ can be immediately cast as a state feedback law for a dominantly second-order process.

Note that, by sheer definition of state, any state or output derivative can be expressed as a (static) function of the present state: the state contains all past information relevant to the future so a *static* $u = K(x)$ full-information controller has the same capabilities than a dynamic one, $u = K'(x, \dot{x}, y, \dot{y}, \ddot{x}, \dots, ...)$, as the latter can be cast into a form $u = K(x)$. In the linear framework, there is no advantage in further complicating[2] expression (6.2).

Applying the control law, (6.2), to (6.1), the following closed-loop equation results:

$$\dot{x} = Ax + B(-Kx + v) = (A - BK)x + Bv \tag{6.3}$$

So that, by suitable selection of K, a prescribed set of properties for (6.3) can be achieved. Note that disturbances have not been considered at this moment.

For discrete-time systems, using the same control strategy (6.2), $u_k = -Kx_k + v_k$, leads to a closed-loop equation:

$$x_{k+1} = Ax_k + B(-Kx_k + v_k) = (A - BK)x_k + Bv_k \tag{6.4}$$

[1] As usual, negative feedback is assumed.

[2] In an ideal, perturbation-free and model-error free environment.

6.1.1 Stabilisation and Pole-placement

Assuming $v = 0$, state feedback changes the open-loop initial dynamics, $\dot{x} = Ax$, into $\dot{x} = (A - BK)x$. As stability and settling time are associated with the eigenvalues of the system matrix, $A_{cl} = A - BK$, these characteristics can be changed with state feedback control. In fact, the following powerful result [11] does hold:

> emphIf a system is fully reachable, the closed-loop eigenvalues can be assigned to any arbitrary desired position by appropriately selecting K.

The *pole-placement* problem consists in assigning all reachable eigenvalues to prescribed values inside the stable region of the complex plane. If the unreachable modes of the system already lie there, *i.e.*, the system is stabilisable, this is a *stabilisation* problem. The prescribed pole positions can be used to specify the *settling time* by means of the approximate expressions in Appendix A.2.1 or, in general, to target a particular closed-loop dynamics.

This casts in a general framework the design of stable loops with a prescribed settling time that, for first and second-order SISO loops, can be shown to be equivalent to P and PD controllers, respectively.

There are several methods for accomplishing the pole-placement task. The use of canonical forms allows for a simple demonstration, although efficient algorithms can be derived and proved for the general case. An example of a so-called *direct* method will now be given.

Direct Method

The direct method refers to symbolic calculation of the polynomial:

$$\det(sI - A_{cl}) = \det(sI - (A - BK))$$

whose zeros are the closed-loop eigenvalues, and calculating the coefficients of K so the referred polynomial is identically equal to a desired one. Let us illustrate the procedure by means of a couple of examples.

Example 6.2. Let us have a third-order process, with two inputs, with a normalised state-space representation given by matrices

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}; \quad B = \begin{pmatrix} 1 & 0 \\ 2 & 0 \\ 0 & 1 \end{pmatrix}$$

Availability of three sensors for the state variables is also assumed. The desired settling time is about 1 second (poles about -3 or -4) so the desired closed-loop characteristic polynomial could be $(s+3)(s+4)^2 = s^3 + 11s^2 + 40s + 48$. The control strategy (6.2), written as

$$u_1 = -k_1 x_1 - k_2 x_2 - k_3 x_3 \tag{6.5}$$
$$u_2 = -k_4 x_1 - k_5 x_2 - k_6 x_3 \tag{6.6}$$

without considering an extra signal, v, will allow a closed loop $\dot{x} = (A - BK)x$ whose eigenvalues will be the roots of $\det(sI - (A - BK)) = 0$. In this case,

$$\Gamma = \det(sI - (A - BK)) = \det\left[\begin{pmatrix} s-(1-k_1) & -2+k_2 & -3+k_3 \\ -4+2k_1 & s-(5-2k_2) & -6+2k_3 \\ -7+k_4 & -8+k_5 & s-(9-k_6) \end{pmatrix}\right]$$

$$\begin{aligned}\Gamma = {} & s^3 + (k_6 + 2k_2 + k_1 - 15)\, s^2 \\ & + ((k_1 - 6 + 2k_2)k_6 + (6 - 2k_3)k_5 + (3 - k_3)k_4 + 23k_3 - 10k_1 - 16k_2 - 18)\, s \\ & + (-3 + 2k_2 - k_1)k_6 + (6 - 2k_3)k_5 + (-3 + k_3)k_4 + 9k_1 + 9k_3 - 18k_2\end{aligned}$$

So, equating polynomial coefficients, there are three equations and six unknowns[3]. Three of these can be arbitrarily fixed, for example $k_4 = k_5 = k_6 = 0$ (deciding not to use input 2, quite counter-intuitive indeed and only possible because the system is fully reachable by the first control variable, otherwise, this assumption will not work). The remaining equations are:

$$\begin{aligned} k_1 + 2k_2 - 15 &= 11 \\ -10k_1 - 16k_2 + 23k_3 - 18 &= 40 \\ 9k_1 - 18k_2 + 9k_3 + 0 &= 48 \end{aligned} \tag{6.7}$$

whose resolution results in the controller gains ($k_1 = 9.47,\ k_2 = 8.26,\ k_3 = 12.39$).

Example 6.3. Let us now consider the headbox canonical representation (A_L, B_L) obtained in Example 3.11 on page 75. Due to the special structure of the matrices it appears clear that a control law (6.2) with feedback matrix

$$K = \begin{pmatrix} k_{11} & k_{12} & k_{13} + 0.6065 & 0 & 0 \\ 0.7711 & -2.0961 & 0 & k_{24} - 0.2865 & k_{25} + 1.1467 \end{pmatrix}$$

leads to the closed-loop system matrix

$$A_L - KB_L = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -k_{11} & -k_{12} & -k_{13} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -k_{24} & -k_{25} \end{bmatrix}$$

and the characteristic polynomial is $(z^3 + k_{13}z^2 + k_{12}z + k_{11})(z^2 + k_{25}z + k_{24})$, showing the possibility of straightforward pole assignment.

It has been shown that, in the general case, there are many solutions to the pole assignment problem in MIMO systems. To get a canonical form and use the procedure above is rather tedious if carried out manually. Software tools can ease the work.

[3] In the SISO case, we would have had only one column in B and three constants (k_1, k_2, k_3) so the solution would have been unique. SISO pole-placement has a unique solution (if it exists).

Example 6.4. MATLAB® code to carry out the symbolic determinant in Example 6.2 would be:

```
a=[1 2 3;4 5 6;7 8 9]; b=[1 0;2 0;0 1]; syms k1 k2 k3 k4 k5 k6 s
k=[k1 k2 k3;k4 k5 k6]; acl=a-b*k; de=det(s*eye(3)-acl);
de2=collect(de,s)
```

However, the MATLAB® command 'place' directly implements the pole-placement solution:

```
kk=place(a,b,[-3 -4 -4])
kk =   -1.0000    7.0000    3.0000
        7.0000    8.0000   13.0000
```

providing a different feedback gain (same eigenvalues, different eigenvectors) to the one in Example 6.2. Obviously, choosing the "best" one depends on geometric considerations or optimality criteria (Chapter 7). However, they will not be pursued any further in this context. The reader is referred to [11] for detailed analysis.

The importance of the state feedback results in engineering terms is that:

> *the more independent sensors are available, the less sophisticated the control strategy is for a given desired specification: with enough number of sensors, transient specifications can be achieved using several "proportional" controllers.*

Discrete controllers. For the DT case, the pole-placement problem is totally analogous, except that the desired positions for the poles now lie inside the unit disk.

Minimum-time Control

In DT control, there is an interesting option, at least from a conceptual viewpoint. If the poles are all assigned to the origin, the controlled system can reach the equilibrium point in the minimum reaching time, (3.45). These controllers are denoted as *deadbeat* controllers. However, when this strategy is applied to a sampled-data controlled system, caution should be taken with the behaviour of the signals in the intersampling period, because hidden oscillations may appear.

Example 6.5. In Example 6.3, for $k_{13} = k_{12} = k_{11} = k_{25} = k_{24} = 0$ in the feedback matrix K, the closed-loop system transfer matrix from v to y is:

$$M(z) = \begin{pmatrix} \frac{-0.9315}{z^3} & \frac{-0.6321z+0.4923}{z^2} \\ \frac{0.6585}{z^3} & \frac{0.2908z-0.2628}{z^2} \end{pmatrix}$$

It is worth noting that the CT model response to some initial conditions will not reach and *stay* at the equilibrium in three sampling periods, as the DT model step response suggests, and some intersample oscillations will occur.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `place,acker`.

Remark 6.6. Pole assignment strategy implies applying a control signal to move the poles from the open-loop positions to the closed-loop ones. In this framework, the concept of *control effort*, related with this pole shifting [3], points out the magnitude of the control action to achieving this placement. In DT systems, as the control is open loop in between the sampling times, large control efforts will warn for unexpected delays and, in any case, CT or DT, for model uncertainties.

6.1.2 State Feedback PI Control

As already mentioned, the state feedback control law allows us to place the reachable poles in any desired position, assuring a closed-loop dynamic behaviour, but nothing is said about the input/output behaviour. In particular, in servosystems, a common requirement is to be able to follow changes in the references and the classical solution is to introduce the integral action. Another option is to apply a feedforward action, to drive the system in the desired direction.

Let us consider the introduction of an integral action based on the tracking error. Assume that the controlled variables are the system output vector components, $y(t)$. To achieve zero steady-state error for constant or ramp-like disturbances, a number of integrators should be added. For step changes in the references, the control action will be:

$$u = -Kx - K_I \int (y - r)\, dt \tag{6.8}$$

where $K_I = \gamma G(0)^{-1}$, $G(0)$ being the plant's DC gain and γ a small scalar (or a diagonal matrix in case different "integral gains" are wished for different inputs). This is a straightforward way of generalising SISO PID control. However, if K_I is not very small, it may significantly distort the closed-loop dynamics.

Plant augmentation. A wiser alternative is the addition of integrators at the process output and then designing a controller for the augmented process:

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ I \end{bmatrix} y_d; \quad \begin{bmatrix} y \\ v \end{bmatrix} = \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}$$

where the additional auxiliary state variables, v act as "accumulated errors" generalising the concept of the basic control action ($K_I \int e\, dt$).

A stabilising controller for the augmented plant must be designed, for instance, by pole assignment, as before. But again the interpretability of the result is not clear because we can define a set of closed-loop poles, but not to assign them to some specific state variables. There is no simple way, at this moment, to specify that the integral action may have a "slow" pole as in SISO PID control if the regulator is intended to compensate for very slowly-varying drifts. The pole-placement design may assign the slow pole to any of the process modes and not to those exclusively associated to e. In the next chapter, in the context of Kalman filtering, the idea will be further developed.

6.2 Output Feedback

In a general situation, the number of sensors is not necessarily equal to the number of state variables, and there is a presence of disturbances (process and measurement noise) so that an accurate enough reconstruction of the state by numeric derivation is not possible. In this section, a framework to deal with closed-loop control in this case will be presented. It is based on a two-step procedure: first obtaining an estimation of the plant state and then using this estimate in state feedback laws. A fundamental result (separation theorem) will show that the combination of these two stages yields a stable closed loop.

Estimation of the state from a set of output sensors will now be discussed. Let us start with the discrete-time implementations.

Indirect state measurement. The most elemental form of state estimation is to set up a state representation where the outputs and some of their increments (numerical derivatives) conform the state vector. This requires a fine model and sensors allowing the computation of numerical derivatives and/or filtering. However, numerical derivation amplifies quantisation errors (due to finite word-length) and high-frequency measurement noise effects (see example E.1), as $(y_k - y_{k-1})$ is small if sampled at a high frequency, so it is masked by noise.

Subsequent low-pass filtering is usually needed to avoid high-frequency actuator activity. As the filter introduces some poles and delays, the dynamics of the overall system becomes more complex and hence there is a greater risk of instability. Furthermore, filtering introduces *delays* in the detection of disturbances affecting the real unfiltered variables.

Hence, the approach is usually valid only with enough sensors. For example, assuming that *only one derivative* can be safely taken[4], to implement

[4] Taking numerical derivatives of a sensor reading leads to unacceptable measurement noise amplification when second and further derivatives are calculated. Further derivatives might be taken if the signal-to-noise ratio were *very* good and the desired specifications were slow enough so that the dynamics of noise filters are still "fast" compared to that of the closed-loop plant. This situation is rare in practice: sometimes even taking just *one* derivative is impractical.

"pure" state feedback, in practice, a number of sensors *greater than half the order* of the system is needed. In essence, constant state feedback from these estimates is equivalent to setting up a combination of PD controllers with noise-filtering. This is an appealing solution, related to decentralised control structures.

Other model-based solutions will now be presented for the case where the number of sensors (or its signal-to-noise ratio) is low, so "tuning" of the low-pass filtering is adapted to the process characteristics in a "transparent" way. These solutions, however, require a reasonably accurate *model*.

State reconstruction. The definition of observability (Section 3.7.2) implies the ability to compute the process state from a finite amount of past measurements, by taking the inverse of the observability matrix. However, this calculation may be ill-conditioned (see Appendix B.4.1). Extending the dimension of the so-formed observability matrix (3.47) beyond the system order, stacking C, CA, CA^2, ..., CA^N, with $N > n$, an old state at $x(k-N)$ can be computed. Conditioning *always* improves as more information (larger N) is gathered. Afterwards, the current state should be computed based on the process model, (6.1). This is a sort of "averaging" over a user-defined number of past samples, leading to a model-based finite impulse response (FIR) filtering.

The number of past samples for a particular noise amplification depends on the sampling time, T (as A itself depends on T for a discrete model). However, averaging over a significant number of past samples to filter *measurement* noise increases data corruption by *process* noise and modelling errors, so there is a trade-off. It can be shown that the faster the sampling rate the better the estimations are for a particular time interval, NT_s, to be averaged, as intuitively expected.

A drawback of the approach is that matrix dimensions increase with the number of averaged samples. To take advantage of some of the features above without increasing computational load, a recurrent (infinite-memory) estimator will now be developed.

6.2.1 Model-based Recurrent Observer

Let us have a DT system modelled by:

$$x_{k+1} = Ax_k + Bu_k; \qquad y_k = Cx_k \tag{6.9}$$

so that if an initial accurate state estimate, $\hat{x}_k$, were available, the next state should be:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k; \qquad \hat{y}_k = C\hat{x}_k \tag{6.10}$$

Signal symbols with a hat will denote estimations, and those variables without it will denote the trajectories on the actual plant. The pair (A, C) is assumed to be observable (Section 3.7.2), otherwise the unobservable states will not

affect the output vector so estimating them from output readings would not be feasible.

Note that even if $\hat{x}_k$ is accurate, process noise and modelling errors will decrease the reliability of $\hat{x}_{k+1}$ unless on-line corrections are made, based on incoming measurements. This is the idea behind the so-called *observer* equation:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - \hat{y}_k) \tag{6.11}$$

where $\hat{y}_k$ denotes the current estimation of the output based on the knowledge of $\hat{x}_k$ and L is a constant matrix (dimension $n \times p$) usually denoted as *observer gain* whose calculation will be discussed next, weighting the output estimation error, $e_{y_k} = y_k - \hat{y}_k$. In fact, y_k is our actual information from the process and $\hat{y}_k$ is our estimate. Thus, the observer basic equation can be written as:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_k - C\hat{x}_k) \tag{6.12}$$

$$\hat{x}_{k+1} = (A - LC)\hat{x}_k + Bu_k + Ly_k \tag{6.13}$$

Subtracting the last equation from (6.9) yields:

$$(x_{k+1} - \hat{x}_{k+1}) = (A - LC)(x_k - \hat{x}_k) \tag{6.14}$$

$$e_{k+1} = (A - LC)e_k \tag{6.15}$$

where e_k is the *state estimation error*. By calculating L so that the eigenvalues of $(A - LC)$ lie in a prescribed location inside the unit circle, the designer can ensure that an initial estimation error will converge to zero in a prescribed time[5].

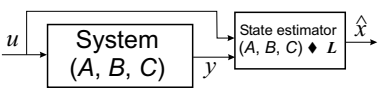

Figure 6.1. Observer structure

Approximately, the longer the prescribed time is, the less sensitivity to measurement noise the estimate will have but the sensitivity to process noise will be higher. With a perturbed model $x_{k+1} = Ax_k + Bu_k + v_k$, $y_k = Cx_k + w_k$ where v_k is the process noise and w_k is the measurement one, Equation (6.15) is transformed into:

$$e_{k+1} = (A - LC)e_k + (I - LC)v_k - Lw_k \tag{6.16}$$

[5] The settling time associated with the dynamics of these observer poles is, approximately, an indication of the number of past samples that the observer averages to obtain the current estimate, to compare with the previously described non-recurrent approach.

Remark 6.7. Note that the lowest effect of measurement noise is achieved with $L = 0$ (open-loop estimation, error dynamics given by $(A - 0 * C) = A$) and, ideally, the lowest effect of process noise is achieved with $L = AC^{-1}$, *i.e.*, full state measurement (error dynamics $A - AC^{-1} * C = 0$ with poles at the origin[6]). In this case, the observer would be better considered as a state filter, as all the state variables are available.

In practice, an intermediate set-up would be used. The designer would decide the position of the poles depending on the ratio between process noise and measurement noise, and the number of sensors. Under certain assumptions, an "optimal" observer can be constructed (see Section 7.2). Usually, a reasonable settling time for tuning the observer poles could be in the range 33–66% of the target closed-loop settling time (when a controller is finally put into effect), assuming it is "reasonably" decided considering the magnitude of modelling errors and sensor signal-to-noise ratios.

The observer general layout is depicted in Figure 6.1.

Remark 6.8. Note that there is a duality between the pole assignment problems in control and estimation. In the first case, (6.3), we looked for the matrix K to place the eigenvalues of $A - BK$. Now, L should be computed to place the eigenvalues of $A - LC$, (6.15). As the eigenvalues of a matrix and its transpose are the same, the second problem is equivalent to computing L to assign the poles of $(A - LC)^T = A^T - C^T L^T$.

Example 6.9. Observer design in MATLAB® is carried out with:

```
L=place(A',C',p);L=L'
```

Transposition is carried out as MATLAB® places the eigenvalues of $A - BK$ where the unknown K is at the right-hand side.

Continuous-time observers. In the CT case, for a model

$$\dot{x} = Ax + bu; \qquad y = Cx \tag{6.17}$$

an observer equation such as:

$$\dot{\hat{x}} = A\hat{x} + Bu + L(y - \hat{y}); \qquad \hat{y} = C\hat{x} \tag{6.18}$$

leads to $\dot{\hat{x}} = A\hat{x} + Bu + L(Cx - C\hat{x})$. So, as before, subtracting from (6.17)

$$\frac{d}{dt}(x - \hat{x}) = A(x - \hat{x}) - LC(x - \hat{x})$$

and the state estimation error dynamics is given by:

[6] The observer such that the eigenvalues of $A - LC$ are at the origin is called the *minimum-time observer*: it is the best one for detecting sudden perturbations in process state, at the expense of high measurement noise amplification if the sampling period is small.

$$\frac{de}{dt} = (A - LC)e \tag{6.19}$$

so that assigning the eigenvalues of $A - LC$ in the location specified by the designer can be used as a criteria to obtain a suitable value for the gain matrix L.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `place, estim, destim`.

6.2.2 Current Observer

In the basic setting, the output estimation error, e_{y_k}, is used to correct the next state estimation, $\hat{x}_{k+1}$, (6.11). In order to improve the current state estimate, using the current output measurement, the following strategy can be used:

Using the *predicted value* of the output, the observer equation, (6.11), is written as:

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + L(y_{k+1} - \hat{y}_{k+1|k}) \tag{6.20}$$

where $\hat{y}_{k+1|k}$ denotes the estimation of y_{k+1} with the information available before measuring it, *i.e.*, with knowledge of $\hat{x}_k$ and input u_k, based on the process model:

$$\hat{y}_{k+1|k} = C(A\hat{x}_k + Bu_k)$$

So, operating as before, the final observer equation is:

$$(x_{k+1} - \hat{x}_{k+1}) = A(x_k - \hat{x}_k) - LCA(x - \hat{x}) \tag{6.21}$$

$$e_{k+1} = (A - LCA)e_k \tag{6.22}$$

An alternative, conceptually different but leading to exactly the same result, is to predict the current state, $\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_{k-1}$, and then update it with the measurement error, $\hat{x}_{k|k} = \hat{x}_{k|k-1} + L(y_k - Cx_{k|k-1})$. Operating on these expressions, (6.22) is also obtained, taking $\hat{x}_{k|k}$ as $\hat{x}_k$.

Remark 6.10. For pole-placement, the computation algorithm requires the observability of the pair (A, C), in the basic setting, but in the current observer, the observability of the pair (A, CA) is needed.

6.2.3 Reduced-order Observer

It may be argued that some state variables are directly measured and thus they do not need to be estimated. Nevertheless, as previously mentioned, even in the case of full state measurement, the estimator may be also considered as

a measurement-noise filter in such a way that better information can be used for control purposes.

If the measurement noise is not relevant and there is a limitation on computing resources, a reduced-order observer can be pursued. To better follow the reduced-order observer design, let us take a similarity transformation T, (2.15), to have the output as the first part of the state vector, completing it with any $F_{(n-m)\times n}$ matrix so T is invertible and applying (2.20):

$$T = \begin{pmatrix} C \\ F \end{pmatrix}; \quad \overline{A} = T\,A\,T^{-1}; \quad \overline{B} = T\,B; \quad \overline{C} = C\,T^{-1}$$

leading, in the CT case, to the equations

$$\begin{bmatrix} \dot{\bar{x}}_1 \\ \dot{\bar{x}}_2 \end{bmatrix} = \begin{bmatrix} \overline{A}_{11} & \overline{A}_{12} \\ \overline{A}_{21} & \overline{A}_{22} \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix} + \begin{bmatrix} \overline{B}_1 \\ \overline{B}_2 \end{bmatrix} u; \qquad y = \bar{x}_1 \tag{6.23}$$

Thus, we must get an estimate of $\bar{x}_2$, based on the subsystem equation:

$$\dot{\bar{x}}_2 = \overline{A}_{22}\bar{x}_2 + \overline{A}_{21}\bar{x}_1 + \overline{B}_2 u = \overline{A}_{22}\bar{x}_2 + \overline{A}_{21} y + \overline{B}_2 u \tag{6.24}$$

The information about this subsystem that we can indirectly measure, as provided by the first subsystem equation, is:

$$breve{y}_2 = \overline{A}_{12}\bar{x}_2 = \dot{\bar{x}}_1 - \overline{A}_{11}\bar{x}_1 - \overline{B}_1 u = \dot{y} - \overline{A}_{11} y - \overline{B}_1 u \tag{6.25}$$

where $breve{y}_2 = \overline{A}_{12}\bar{x}_2$ is a fictitious output that can be "measured" to be used in an observer equation (6.18) for (6.24):

$$\dot{\hat{\bar{x}}}_2 = \overline{A}_{22}\hat{\bar{x}}_2 + \begin{pmatrix} \overline{A}_{21} & \overline{B}_2 \end{pmatrix} \begin{pmatrix} y \\ u \end{pmatrix} + L(breve{y}_2 - A_{12}\hat{\bar{x}}_2) \tag{6.26}$$

So, replacing $breve{y}_2$ by its value in (6.25):

$$\dot{\hat{\bar{x}}}_2 = \left(\overline{A}_{22} - L\overline{A}_{12}\right)\hat{\bar{x}}_2 + \left(\overline{B}_2 - L\overline{B}_1\right) u + \left(\overline{A}_{21} - L\overline{A}_{11}\right) y + L\dot{y} \tag{6.27}$$

In fact, the term $L\dot{y}$ must be forwarded to the left-hand side, giving:

$$\frac{d(\hat{\bar{x}}_2 - Ly)}{dt} = \left(\overline{A}_{22} - L\overline{A}_{12}\right)\hat{\bar{x}}_2 + \left(\overline{B}_2 - L\overline{B}_1\right) u + \left(\overline{A}_{21} - L\overline{A}_{11}\right) y$$

And, making the change of variable $w = \hat{\bar{x}}_2 - Ly$, after straightforward operations, the result is:

$$\begin{aligned} \frac{dw}{dt} &= (\bar{A}_{22} - L\bar{A}_{12})w + \left((\bar{A}_{22} - L\bar{A}_{12})L + \bar{A}_{21} - L\bar{A}_{11} \;\; \bar{B}_2 - L\bar{B}_1 \right) \begin{pmatrix} y \\ u \end{pmatrix} \\ \hat{x}_2 &= w + Ly \end{aligned} \tag{6.28}$$

Undoing the state transformation T, going back to the original state variables, and stacking the first set of measured states, the output equation would be replaced by:

$$\hat{x} = T^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix} w + T^{-1} \begin{pmatrix} I & 0 \\ L & 0 \end{pmatrix} \begin{pmatrix} y \\ u \end{pmatrix} \tag{6.29}$$

A similar equation can be derived for a DT reduced-order observer.

The observer dynamics is determined by the eigenvalues of:

$$\overline{A}_{22} - L\,\overline{A}_{12} \tag{6.30}$$

Note that this reduced-order observer is a "current" observer, as the current output measurement is used to estimate the partial state ($D \neq 0$).

6.2.4 Separation Principle

A two-stage algorithm applying pole-placement techniques to observer and state feedback controller design yields a closed-loop system with poles located at the union of those from each of the designs (under the assumption of negligible model error). The proof will be presented for the CT case, although an analogous one can be provided for the DT case.

Let us connect a plant with equations ($\dot{x} = Ax + Bu$, $y = Cx$) in feedback with a controller (observer + state feedback replacing state by its estimation):

$$\dot{\hat{x}} = (A - LC)\hat{x} + Bu + Ly \tag{6.31}$$

$$u = -K\hat{x} \tag{6.32}$$

The normalised state space representation of the controller, including the observer[7], is the n-order system:

$$\begin{aligned} \dot{\hat{x}} &= (A - LC - BK)\hat{x} + Ly \\ u &= -K\hat{x} \end{aligned} \tag{6.33}$$

The overall closed-loop system is $2n$-order, with a state vector x_c: $\{x, \hat{x}\}$, whose dynamics is given by:

$$\frac{d}{dt} \begin{pmatrix} x \\ \hat{x} \end{pmatrix} = \begin{pmatrix} A & -BK \\ LC & (A - BK - LC) \end{pmatrix} \begin{pmatrix} x \\ \hat{x} \end{pmatrix} \tag{6.34}$$

Defining a new state vector x_c^n: $\{x, x - \hat{x}\}$ by the similarity transformation

$$x_c^n = \begin{pmatrix} I & 0 \\ I & -I \end{pmatrix} x_c$$

applying (2.20), it results

$$\frac{dx_c^n}{dt} = \begin{pmatrix} A - BK & BK \\ 0 & A - LC \end{pmatrix} x_c^n \tag{6.35}$$

[7] Note that it is a system whose outputs are the control actions and the inputs are the sensor readings, so it is a usual feedback controller, anyway.

Thus, due to the upper block triangular structure, the eigenvalues of the plant + controller are those of $A - BK$ and $A - LC$. This justifies the *independent design* of observer and controller gain matrices, and (6.35) is termed the *separation principle*. As previously commented, a usual rule of thumb is to design observer poles to be somehow faster than controller ones. The suitability of the approach depends on having a good signal-to-noise ratio.

This is also applicable to the DT observer, as well as to the current and reduced-order observers, with the appropriate change of state vector.

Instead of (6.33), the normalised state representation of a controller using a current observer is, denoting $\psi_k = \hat{x}_{k-1}$:

$$\begin{aligned} \psi_{k+1} &= (I - LC)(A - BK)\psi_k + Ly_k \\ u_k &= -K(I - LC)(A - BK)\psi_k - KLy_k \end{aligned} \tag{6.36}$$

Design code. The MATLAB® command `Kss=reg(sys,k,l)` sets up the controller in (6.33). An example of CT design, given a plant, `gs`, and a desired pole location, `pd`, is:

```
k=place(gs.a,gs.b,pd); l=place(gs.a',gs.c',pd);
rg=reg(gs,k,l');
```

the last line being equivalent to: `rg=ss(gs.a-l'*gs.c-gs.b*k,l',-k,0)`. The same commands carry out the DT design (including the sampling time, `Ts`, as a last argument to `ss`, as done below).

The command `dreg` does the same with the current discrete observer. For a DT plant `gsd` and desired poles `pdd`, the code synthesising the regulator is:

```
kd=place(gsd.a,gsd.b,pdd); ld=place(gsd.a',gsd.c',pdd);ld=ld';
[a2 b2 c2 d2]=dreg(gsd.a,gsd.b,gsd.c,gsd.d,kd,ldc);
rgd2=ss(a2,b2,c2,d2,Ts);
```

or, alternatively, carrying out (6.36) directly:

```
pp=(eye(2)-ldc*gsd.c)*(gsd.a-gsd.b*kd);
rgd2=ss(pp,ldc,-kd*pp,-kd*ldc,Ts);
```

Computer code for real-time control will be discussed in Chapter 9.

6.3 Rejection of Deterministic Unmeasurable Disturbances

The present observer + state feedback equations are designed to modify the system dynamics, but they do not explicitly solve the "disturbance rejection" problem: the methodology only provides tools to ensure that, departing from an off-equilibrium situation, equilibrium will be approximately reached in a specified time (given by the prescribed poles).

Enhancements to the approach will be presented in this section that are specially focused to tackle the *deterministic* disturbance problem.

6.3.1 Augmented Plants: Process and Disturbance Models

As discussed in Section 2.10.1, in some cases a deterministic disturbance can be expressed as the output of an autonomous generator system. In engineering practice, constant, ramp and sinusoidal disturbances can be expressed as the output of a marginally unstable unexcited linear system with non-zero initial conditions:

$$\dot{x}_d = A_d x_d; \qquad d = C_d x_d$$

Process disturbance. In most cases, the disturbance, d, generated in another subsystem, is not directly measurable. Instead, its effects modify the process state

$$\dot{x} = Ax + Bu + B_d d$$

and these modifications are measured by the sensors measuring the output variables of interest ($y = Cx + Du$).

Output disturbance. If, on the contrary, the disturbance is pure measurement noise, it is added to the sensor readings and does not modify the plant state:

$$z = Cx + Du + d$$

However, the output variable which has to be controlled is still y.

Of course, mixed situations can be thought of ($\dot{x} = Ax+Bu+B_d d$ and $z = Cx + Du + D_d d$). Writing the disturbance and plant equations in normalised form, the result is an *augmented plant* model:

$$\frac{d}{dt}\begin{pmatrix} x \\ x_d \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{x}_d \end{pmatrix} = \begin{pmatrix} A & B_d C_d \\ 0 & A_d \end{pmatrix}\begin{pmatrix} x \\ x_d \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} u \tag{6.37}$$

$$z = \begin{pmatrix} C & D_d C_d \end{pmatrix}\begin{pmatrix} x \\ x_d \end{pmatrix} + Du \tag{6.38}$$

where, for process disturbance $D_d = 0$ and $B_d \neq 0$ and for output disturbance $D_d \neq 0$ and $B_d = 0$. In any case, it can be easily checked that the disturbance state is *unreachable* by input u, as intuitively expected.

Disturbance estimation. Let us consider how to cancel the effect of the disturbances on the true output $y = Cx + Du$. In the following, for simplicity, $D = 0$ will be assumed.

The first assumption is that the *whole* augmented state (x, x_d) must be observable[8]. Then, an observer can be designed to estimate the process and disturbance states. Thus, $(\hat{x}, \hat{x}_d)$ tends to its actual value on the real plant (in the absence of additional sources of perturbations and modelling errors) so that the designer can specify the time for convergence.

[8] This algebraic requirement has a clear engineering interpretation: the disturbance must, somehow, affect the outputs (if it does not, there is no need to deal with it). Thus, the assumption holds in most practically relevant cases, with at least as many sensors as actuators.

6.3.2 Disturbance rejection

Once the estimate $\hat{x}_d$ is available, the state feedback has to be extended so that:

$$u = -K_{ext} \begin{pmatrix} \hat{x} \\ \hat{x}_d \end{pmatrix} = - \begin{pmatrix} K & K_d \end{pmatrix} \begin{pmatrix} \hat{x} \\ \hat{x}_d \end{pmatrix} = -K\hat{x} - K_d\hat{x}_d$$

At first sight, this is another state feedback problem that can be solved by, for example, pole-placement as in Section 6.1.1. However, the disturbance state is *unreachable* so the poles corresponding to A_d *cannot* be changed. The procedure to be followed is:

- first, a state feedback law, $u = -Kx$, is calculated for the *non*-augmented plant, $\dot{x} = Ax + Bu$,
- second, K_d is calculated to cancel the perturbation effect.

Let us discuss the second step of the procedure.

Output disturbance. If the disturbance is pure measurement noise, then $K_d = 0$ as the observer, thanks to the perturbation model, is able to discriminate between signal and deterministic noise and the right control action is $u = -K\hat{x}$, discarding $\hat{x}_d$.

Process disturbance. If the disturbance is acting on the process, the basic ideas are the same as in the measurable disturbance case discussed in Section 4.6.3, replacing measured values by the estimated ones:

- if there exists a matrix M so that $B_dC_d = BM$, it is said that disturbances enter by the *input channels*. After convergence of the observer[9], a setting

$$K_d = M; \quad u = -K\hat{x} - M\hat{x}_d$$

  produces a closed loop cancelling d as $B_dd = B_dC_dx_d \approx BM\hat{x}_d$.
- if the input channel condition does not hold, a matrix K_d could be sought so that $B_d - BK_d$ is small in a suitable sense.

Example 6.11. Linearising and discretising a first-order plant, and adding an input disturbance, the following model is obtained:

$$\begin{aligned} x_{k+1} &= 0.97x_k + 0.06(u_k + d_k) \\ y_k &= 2x_k \end{aligned}$$

The augmented plant will be formed by adding a disturbance model assuming a constant d, *i.e.*, $d_{k+1} = d_k$, so the augmented state space representation is:

$$\begin{aligned} m_{k+1} &= \begin{pmatrix} 0.97 & 0.06 \\ 0 & 1 \end{pmatrix} m_k + \begin{pmatrix} 0.06 \\ 0 \end{pmatrix} u_k \\ y &= \begin{pmatrix} 2 & 0 \end{pmatrix} m_k \end{aligned} \tag{6.39}$$

[9] Proof of the joint convergence of controller + observer would need a procedure analogous to that in Section 6.2.4.

where m_k is the augmented state vector (x_k, d_k). If the "true" d does not behave as a constant, the above representation will have a *modelling error*.

A regulator (for the non-augmented plant) achieving a settling time of 16 samples will be designed, *i.e.*, with regulator poles at $p_d = 0.05^{\frac{1}{16}} \approx 0.83$. The result is $K_1 = 2.333$.

Let us design an observer for the augmented plant so that the overall design can cope with constant disturbances achieving zero steady-state error. Observer poles will be placed at $p_d = 0.05^{\frac{1}{11}} \approx 0.76$ (45% faster than the controller one), by solving the equations of eigenvalue placement for $A_{ext} - LC_{ext}A_{ext}$. After operation, the result is $L = (0.2023\ 0.48)^T$. So, as disturbances enter by the input channel ($B_d = B$, so $M = 1$), the overall regulator gain matrix is $K = (2.333\ 1)$ and the joint observer + regulator have equations (6.36):

$$\begin{aligned} \psi_{k+1} &= \begin{pmatrix} 0.4942 & 0 \\ -0.7968 & 1 \end{pmatrix} \psi_k + \begin{pmatrix} 0.2023 \\ 0.48 \end{pmatrix} y_k \\ u_k &= \begin{pmatrix} -0.3565 & -1 \end{pmatrix} \psi_k - 0.9519 y_k \end{aligned}$$

and the resulting regulator transfer function (SISO) is:

$$K(z) = \frac{u(z)}{y(z)} = -\frac{0.9519z(z - 0.9143)}{(z-1)(z-0.4942)}$$

As expected, it includes an integrator. It is left to the reader to check that the SISO transfer function from the disturbance to the output is:

$$\frac{y(z)}{d(z)} = \frac{G}{1 - GK} = \frac{0.12(z - 0.4942)(z-1)}{(z-0.76)^2(z-0.83)}$$

with zero DC gain and poles at the required positions.

The use of a reduced-order observer could have allowed the design of a closed loop of order 2, with a first-order controller.

Disturbance models in practice. Note that, in many cases in practice, the true physical source of all disturbances and their mechanism of action is unknown so that first-principle equations cannot yield a determination of B_d. Anyway, as disturbances must be cancelled by manipulation of the available inputs, it is reasonable to assume that disturbances are input disturbances, $B_d = B$ and $D_d = 0$, and "invent" one source of disturbance for each actuator to implement the observer ($M = I$). In this way, the observer will calculate which value of "input disturbance" explains the deviations in sensor readings (irrespective of the true disturbances being applied at the input or at any other location in the plant). In this case, the control action is, directly:

$$u = -K\hat{x} - C_d\hat{x}_d$$

Indeed, for input disturbances ($B_d = B$ in (6.37)) the regulator state matrix is given by (6.33), *i.e.*:

$$A_{reg} = \begin{pmatrix} A & BC_d \\ 0 & A_d \end{pmatrix} - \begin{pmatrix} B \\ 0 \end{pmatrix} \begin{pmatrix} K & C_d \end{pmatrix} - \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} \begin{pmatrix} C & 0 \end{pmatrix} = \begin{pmatrix} A - BK - L_1C & 0 \\ -L_2C & A_d \end{pmatrix}$$

and, as it is a triangular matrix, the poles of the disturbance generator A_d are poles of the regulator.

If constant disturbances are assumed, that is $A_d = 0$, it is interesting to realise that the estimator involves the addition of integrators. This is in accordance with the classical PI control technique to cancel constant disturbances. The previous example also illustrates these ideas.

6.4 Summary and Key Issues

State space representation allows for an easy extension of PD control laws to systems of arbitrary order via state feedback. If not enough sensors are available, state estimation needs to be carried out. Combination of both approaches leads to the so-called "output feedback control" whose analysis constituted a major theoretical and practical breakthrough in the 1950's and 1960's.

Regarding practical implementation, it allows the design of controllers with a prescribed settling time. The key issue is the existence of multiple solutions in the MIMO case so some may be the best with regards to some criteria, posing the optimisation problems to be solved in next chapter. Criteria that should be considered in practice are I/O pairing, sensitivity to measurement noise and limited control action. At this moment, the only available guideline is that positioning the poles near their open-loop position results in low-gain controllers and low-gain observers having reasonable applicability. In Chapter 9, implementation issues are discussed.

Integral action and, in general, deterministic disturbance cancellation can be easily cast into the state feedback + observer framework.

6.5 Case Study: Magnetic Suspension

A floating platform is controlled by magnetic drives acting on the three extreme positions by means of electromagnets. In Figure 6.2, a schema of the system is depicted [50], where:

- x_v: vertical displacement of the centre of gravity (G) of the platform,
- x_p: pitch angle,
- x_r: roll angle,
- r_i: distance of the i-extreme to the corresponding actuator.

The non-linear electromagnetic forces acting on the platform are:

$$F_j = k_j \left(\frac{u_j}{r_j} \right)^2 \qquad \text{with } j = 1, 2, 3$$

where u_j is the voltage applied to the powerful j-electromagnet, and k_j is a magnetic constant.

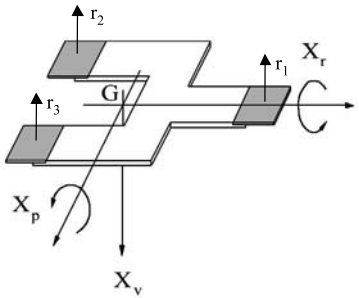

Figure 6.2. Magnetic suspension system

To complete the model, force and torque balances are applied to the platform[10], leading to the mechanical equations:

$$\begin{cases} M\,\ddot{x}_v = Mg - (F_1 + F_2 + F_3) \\ J_p\,\ddot{x}_p = l_{1g}F_1 - l_{2g}\,(F_2 + F_3) \\ J_r\,\ddot{x}_r = l_{3g}\,(F_2 - F_3) \end{cases}$$

The distance from each platform extreme to the corresponding electromagnet, r_i, is geometrically related to the position variables previously defined, x_v, x_r, x_p. The full model involves nine equations with nine internal variables.

In order to derive a linear internal model, the position (x_v, x_p, x_r) and their derivatives ($\dot{x}_v$, $\dot{x}_p$, $\dot{x}_r$) are taken as state variables and the force and geometrical equations, which do not involve dynamics, are linearised around an operating point, usually defined by the positions without movement. The input vector is composed of the voltages (or the currents) applied to the corresponding coils. The extreme positions are taken as components of the output vector, also requiring the linearisation of the output equations.

$$x = \left[\, x_v \; x_p \; x_r \; \dot{x}_v \; \dot{x}_p \; \dot{x}_r \,\right]^T; \quad u = \left[\, u_1 \; u_2 \; u_3 \,\right]^T; \quad y = \left[\, r_1 \; r_2 \; r_3 \,\right]^T$$

Linear model. A laboratory platform has been modelled and linearised, leading to the model:

$$\Delta\dot{x} = A\;\Delta x + B\;\Delta u; \quad \Delta x = x - x^* \quad \rightarrow \quad x = \Delta x + x^*; \ldots$$

denoting with $()^*$ the steady-state variables:

$$x^* = \left[\, x_v^* \; 0 \; 0 \; 0 \; 0 \; 0 \,\right]^T; \quad u^* = \left[\, u_1^* \; u_2^* \; u_3^* \,\right]^T; \quad y^* = \left[\, r_1^* \; r_2^* \; r_3^* \,\right]^T = \left[\, x_v^* \; x_v^* \; x_v^* \,\right]^T$$

and

[10] Assuming small deviations over the main inertia axes.

$$A = \begin{bmatrix} 0 & 0 & 0 & 1\,0\,0 \\ 0 & 0 & 0 & 0\,1\,0 \\ 0 & 0 & 0 & 0\,0\,1 \\ 1089.71 & -16.4811 & -0.0424 & 0\,0\,0 \\ -494.935 & 2029.55 & -0.2332 & 0\,0\,0 \\ -4.4942 & -0.8224 & 1016.34 & 0\,0\,0 \end{bmatrix}; \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -2.4038 & -1.9050 & -1.9718 \\ 23.605 & -10.4689 & -10.8362 \\ 0 & 24.2115 & -25.0619 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & -0.327 & 0 & 0\,0\,0 \\ 1 & 0.183 & -0.12 & 0\,0\,0 \\ 1 & 0.183 & 0.12 & 0\,0\,0 \end{bmatrix}$$

Structural properties. It is easy to apply the analysis techniques developed in Chapter 3 and check that:

– the system is open-loop unstable:

$$eig(A) = \begin{bmatrix} -45.1459 & 45.1459 & 32.8804 & -32.8804 & 31.8801 & -31.8801 \end{bmatrix}$$

– the system is state- and output-reachable. `[u s v]=svd(ctrb(A,B))` points out that x_p and $\dot{x}_p$ are the most controllable direction, x_r and $\dot{x}_r$ the second ones (3 times less), and the vertical position is 27 times less controllable,

– The system is state observable,

– With `[u s v]=svd(ctrb(A,B(:,1)))`, *etc.* each input only reaches four state variables, and with a bound on condition number of 10^5, only two,

– The observability space is dimension 4, for any single output,

– The relative degree matrix has 2 in all elements.

Centralised control design. Let us go for a multivariable control system with the following requirements:

1. No steady-state error in position.
2. Poles around -100.
3. Output feedback, using a reduced-order observer.

A centralised optimal control strategy for this system is reported in [51].

1. To fulfill the first requirement, an integrator should be added to each output (controlled variable):

$$v_i = \frac{1}{s}\left(y_{d,i} - y_i\right) \quad \rightarrow \quad \dot{v}_i = y_{d,i} - y_i, \qquad i = 1, 2, 3$$

Thus, the resulting augmented system (dimension 9) will be (6.1.2) with augmented matrices:

$$Aa = \begin{bmatrix} A & 0 \\ -C & 0 \end{bmatrix}, \quad Ba = \begin{bmatrix} B \\ 0 \end{bmatrix}, \quad Ca = \begin{bmatrix} C & 0 \\ 0 & I \end{bmatrix}$$

Any stabilising state feedback will match the first requirement.

2. To fulfill the second requirement, a pole-placement problem (using the MATLAB® command `place`, for instance) will give the state feedback matrix:

$K = \mathsf{place}(Aa,\ Ba,\ [\text{-}100,\ \text{-}100,\ \text{-}100,\ \text{-}102,\ \text{-}102,\ \text{-}102,\ \text{-}104,\ \text{-}104,\ \text{-}104])$

according to the desired pole position. It yields:

$$K = 10^5 \begin{bmatrix} -0.048 & 0.009 & -0.000 & -0.0005 & 0.0001 & -0.0000 & 1.133 & 0.225 & 0.225 \\ -0.054 & -0.006 & 0.007 & -0.0005 & -0.0001 & 0.0001 & 0.284 & 1.663 & -0.162 \\ -0.052 & -0.005 & -0.006 & -0.0005 & -0.0001 & -0.0001 & 0.274 & -0.157 & 1.607 \end{bmatrix}$$

The SIMULINK® diagram is shown in Figure 6.3, assuming all of the state vector is measurable. Block F is in charge of generating feedforward (2-DoF) actions, if so desired, to modify the tracking response, *i.e.*, $v = F(s)r$ in (6.2).

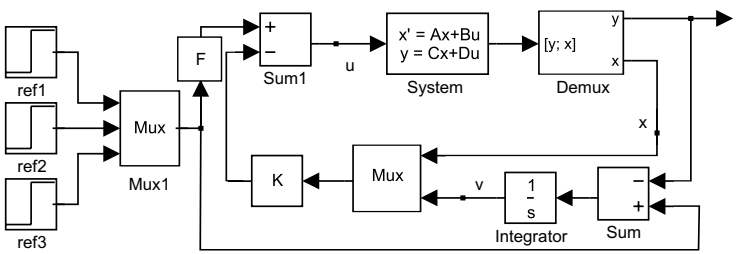

Figure 6.3. Integral control by augmented state feedback

3. To implement output feedback, velocities should be estimated. Note that the incremented state variables, the integrators, are obtained from the outputs and so they are accessible. Let us concentrate on the non-augmented system.

To better follow the reduced-order observer design, let us take a similarity trasformation as proposed in (6.2.3). In this case, with $F = 0_{3\times 3}\ I_{3\times 3}$, Equations (6.23) result in:

$$\begin{bmatrix} \dot{\overline{x}}_1 \\ \dot{\overline{x}}_2 \end{bmatrix} = \begin{bmatrix} 0_{3\times 3} & \overline{A}_{12} \\ \overline{A}_{21} & 0_{3\times 3} \end{bmatrix} \begin{bmatrix} \overline{x}_1 \\ \overline{x}_2 \end{bmatrix} + \begin{bmatrix} 0_{3\times 3} \\ \overline{B}_2 \end{bmatrix} u; \qquad y = \overline{x}_1$$

$$\overline{A}_{12} = \begin{bmatrix} 1 & -0.327 & 0 \\ 1 & 0.183 & -0.12 \\ 1 & 0.183 & 0.12 \end{bmatrix}; \overline{A}_{21} = \begin{bmatrix} 423.3 & 333.4 & 333 \\ -4157 & 1832 & 1830 \\ 0 & -4237 & 4233 \end{bmatrix}; \overline{B}_2 = \begin{bmatrix} -2.403 & -1.905 & -1.972 \\ 23.6 & -10.47 & -10.84 \\ 0 & 24.21 & -25.06 \end{bmatrix}$$

Remember that the reduced-order observer equation is (6.27), and the observer dynamics is determined by the eigenvalues of $\overline{A}_{22} - K_0\overline{A}_{12}$. To compute the observer gain, its poles should be faster than those of the controlled system. Again, using the MATLAB® command `place`:

$$K_0^T = \mathsf{place}(\overline{A}_{22}^T,\ \overline{A}_{12}^T,\ [\text{-}150,\ \text{-}150,\ \text{-}150])$$

it yields:

$$K_0 = \begin{bmatrix} 53.8235 & 48.0882 & 48.0882 \\ -294.1176 & 147.0588 & 147.0588 \\ 0 & -625 & 625 \end{bmatrix}$$

The original state vector is obtained by the inverse transformation:

$$\hat{x} = T^{-1}\overline{x}_{est} = T^{-1} \begin{bmatrix} y \\ \overline{x}_{2,est} \end{bmatrix}$$

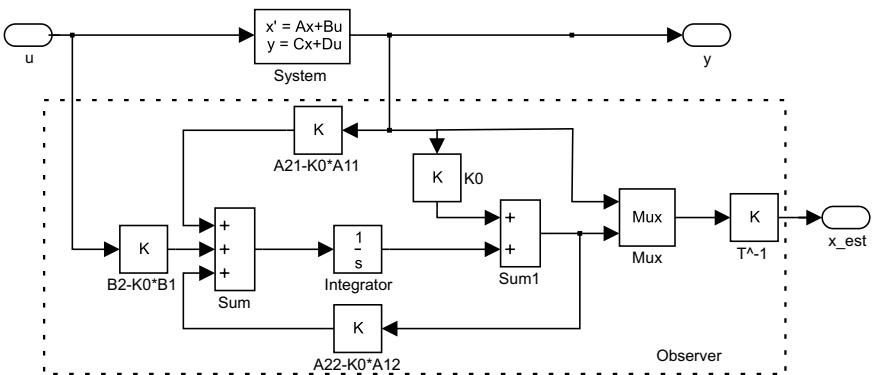

Figure 6.4. Reduced-order observer

The SIMULINK® diagram of the observer is shown in Figure 6.4. The MATLAB® code that yields a normalised internal representation, (6.28)–(6.29), is:

```
Aro=A22-K0*A12; Bro=[(A22-K0*A12)*K0+A21-K0*A11 B2-K0*B1];
Cro=inv(T)*[zeros(3,3);eye(3)];
Dro=inv(T)*[eye(3) zeros(3,3);K0 zeros(3,3)];
rdobs=ss(Aro,Bro,Cro,Dro);
```

The observer has six outputs (six states, in original coordinates), six inputs (three plant outputs + three plant inputs) and three internal states. Altogether, the output-feedback integral control is implemented as shown in Figure 6.5. The controlled plant behaviour, under step changes in the references ($F = I$), can be seen in Figure 6.6, where a settling time of 0.06 s can be realised.

In order to show the performance degrading due to the observer, in Figure 6.7 the free response (output 1) of the controlled system with observer and with direct state access can be compared. An initial condition vector $x_0 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$ is assumed. Similar responses occur in the other output variables. A faster observer could have been designed to achieve less difference in the nominal responses.

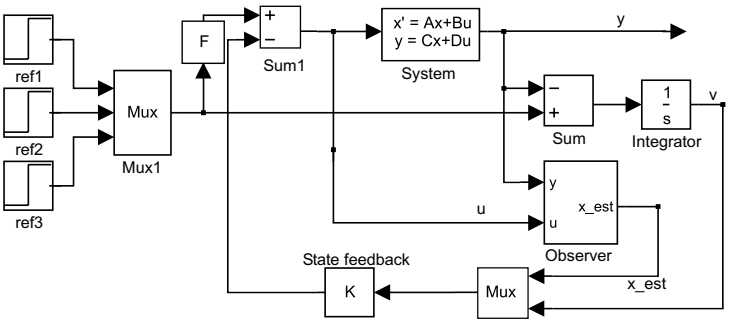

Figure 6.5. Output-integral feedback control

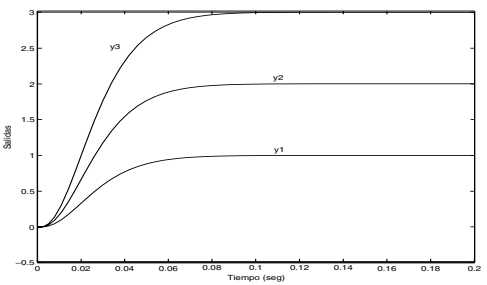

Figure 6.6. Step response: integral-output feedback

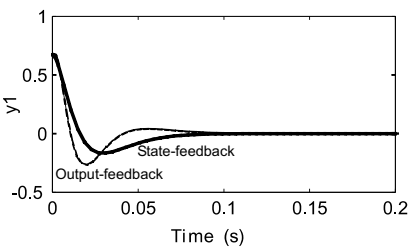

Figure 6.7. Free response of the controlled plant with and without observer (y_1)

Decoupled multi-loop control design. Another control option, as developed in the previous chapter, is to decouple the three subsystems (attached to each position and its derivative) and to implement a multi-loop control on the decoupled system, by designing a PID control, for instance. The decoupling may be achieved by a feedforward transfer matrix or by state feedback. Assuming the state is accessible for measurement (otherwise using the observer), the state feedback solution in Section 5.3.2 on page 139 can be implemented, as the plant has no zeros (check). In particular, it can be checked that all three relative degrees, r_1, r_2, r_3, are equal to 2, and the matrices $\tilde{Q}$ and $\tilde{H}$ in (5.17) are:

$$\tilde{Q} = \begin{bmatrix} -10.1226 & 1.5184 & 1.5716 \\ 1.9159 & -6.7261 & -0.9475 \\ 1.9159 & -0.9154 & -6.9621 \end{bmatrix}; \quad \tilde{H} = \begin{bmatrix} 1251.558 & -680.143 & 0.0338 & 0\ 0\ 0 \\ 999.681 & 355.025 & -122.046 & 0\ 0\ 0 \\ 998.602 & 354.827 & 121.876 & 0\ 0\ 0 \end{bmatrix}$$

so, the decoupling feedback and feedforward terms in (5.19) are:

$$K_D = \begin{bmatrix} 176.1111 & -57.5883 & 0.0000 & 0\ 0\ 0 \\ 175.0000 & 32.0250 & -21.0000 & 0\ 0\ 0 \\ 168.8889 & 30.9067 & 20.2667 & 0\ 0\ 0 \end{bmatrix}, \quad F = \begin{bmatrix} -0.1068 & -0.0212 & -0.0212 \\ -0.0268 & -0.1568 & 0.0153 \\ -0.0259 & 0.0148 & -0.1515 \end{bmatrix}$$

It can be checked that the decoupled transfer matrix of the system is:

$$G_{dec}(s) = C(sI - A - BK_D)^{-1}BF = \begin{bmatrix} \frac{1}{s^2} & 0 & 0 \\ 0 & \frac{1}{s^2} & 0 \\ 0 & 0 & \frac{1}{s^2} \end{bmatrix}$$

so, three straightforward PD control loops will now suffice for decoupled pole-placement. Indeed, with a controller with transfer function $G_R = K_d s + K_p$, each loop will have a characteristic equation $s^2 + K_d s + K_p = 0$, so to place the poles at -100, the gains $K_d = 200$, $K_p = 100^2$, will suffice.

The decoupled control schema can be seen in Figure 6.8.

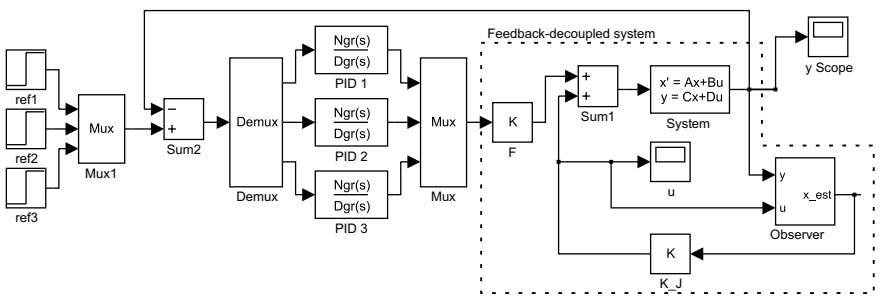

Figure 6.8. PD decoupled control

Remark 6.12. The solution in Figure 6.8 can be improved. Indeed, if the state is accessible (or a state observer is implemented) the output derivative can be directly estimated from the observer output ($\dot{y}_i = {}_iCA\hat{x}$), not depending on u as the relative degrees are $r_i = 2$. So, the PD controller can be implemented as an outer static state feedback, without carrying out additional numerical derivatives. Remark 5.11 about steady-state errors applies here.

7

Optimisation-based Control

The objective of this chapter is to present various controller design techniques based on the optimisation of a (scalar) cost index. First, an optimality-based approach to state feedback and observer design will be presented. Then, the popular multivariable predictive control will be briefly discussed. The chapter will end with a norm-optimisation framework that is able to include some of the previous methodologies as particular cases.

Motivation. The pole-placement framework discussed in the previous chapter accounts for settling-time specifications, but in multivariable systems there is no way of specifying overshoot requirements usual in SISO cases; in fact, it is difficult to even define them. Furthermore, the poles are, from an engineering view, a result from the roots of a determinant that has an obscure relation to practical requirements such as:

- tighter control of output y_1 is desired, either because variable y_2 is unimportant and its deviations can be much larger or because disturbances happen to show a minor effect on y_2,
- with a particular configuration, actuator u_1 often saturates whereas u_2 seldom does: a redistribution of the control authority is desired.

Indeed, in MIMO systems there are multiple solutions to the pole assignment problem, and different actuator amplitudes and precision requirements for each controlled variable: it is difficult for a pole-placement design to be modified to improve control on a particular output or to force lower commands to a particular actuator. Furthermore, some of the states may have no clear physical meaning and may not be quite controllable or observable, so using high gains to assign those poles may not yield a significant input/output improvement.

To account for these situations, a new design paradigm was developed in the 1950s that was based on optimisation of a cost index, flexible enough to deal with the above-mentioned situations. It was conceived as a generalisation of least-squares problems. Optimisation-based control of dynamical systems

reaches significance after the works by Bellmann [24], Pontryagin [103] and Kalman [70], originated by an earlier paradigm shift from transfer functions and frequency response to state space formulations. A general framework, able to accommodate robustness considerations, was developed in the 1980s [44].

7.1 Optimal State Feedback

Many control problems can be expressed, in plain terms, as minimising the "deviation" of the controlled variables, y, from their prescribed set-points, r. The meaning of deviation can be formalised as, for example, $(y-r)^2$. In this way, squaring the deviations indicates that positive and negative ones are equally undesirable. Initially, $r = 0$ will be assumed, as it denotes any constant equilibrium point on a linearised model.

The Cost Index

To define the optimisation problem, a cost index reflecting the engineering specifications must be set up. As signals change over time, the control objective should be oriented to minimise the "accumulated deviation":

$$J_c = \int_0^\infty y^2\, dt; \qquad J_d = \sum_{k=0}^{\infty} y_k^2$$

where J_c refers to CT designs and J_d to DT ones.

These measures are a common performance indicator in SISO control (ISE, integral of squared error). Indeed, they are also of extraordinary importance to general control theory. This accumulated deviation is named as the 2-norm[1] of the signal and is usually denoted by $\|y\|_2$.

Transient deviations may originate from two sources: (a) off-equilibrium initial state, (b) disturbances. The problem to be addressed at this moment will regard the first one: the objective will be reaching a desired operating point with minimal accumulated deviation. The second one will be the objective of the next section.

Control cost. Large amplitudes of the manipulated actuators also imply a "cost": risk of saturation, energy consumption, *etc.* Indeed, a "too-aggressive" controller would make the output norm small by injecting control signals with large amplitude to quickly drive the system to equilibrium.

To avoid saturation and extend actuator life (and to decrease sensitivity to modelling errors, see Chapter 8), the "optimal" regulator must take into account penalties on u as well. To achieve that, the term $\int u^2$ is usually included in the index to be minimised; otherwise, the problem is *ill-posed*.

[1] There are another possibilities for measuring the "size" of a signal, such as the integral of the absolute value. In Appendix C, formal definitions and a summary of other choices for signal and system norms is given.

Weights. In MIMO settings, there will be different relative importance between the variables involved: some form of relative weighting must be set up.

In principle, minimising a weighted sum of the 2-norm of inputs and outputs (such as $\sum_{i=1}^{p} \int_0^\infty q_i y_i(t)^2\, dt + \sum_{i=1}^{m} r_i \int_0^\infty u_i(t)^2\, dt$,) might be a better approach to obtaining a good-performance regulator, with low energy usage and greater ease of tuning than a pole-placement approach.

Example 7.1. Let us consider the simplified headbox model, 2.12.1, presented in Chapter 2. If a sudden disturbance causes the process to depart from its prescribed set-point, there may be different practical requirements regarding the transient deviations of each of the outputs, and the use of each actuator. Let us think on allowing four times less deviation to the stock exit speed, v, than in the stock level, h, and desiring to use three times less the stock flow, q, than the air pressure, p. Those requirements can be approximately expressed as the following optimisation problem: given a starting state $x(0) \neq 0$, minimise:

$$J = \int_0^\infty (4v(t))^2 + h(t)^2 + \rho(p(t)^2 + (3q(t))^2)\, dt \tag{7.1}$$

where ρ is a user-defined small number limiting control activity. In this way, to make a particular variable smaller, its weighting in the control cost index is increased. By denoting process variables with the usual notation (y, u), taking into account that:

$$16y_1^2 + y_2^2 + \rho(u_1(t)^2 + 9u_2(t)^2) = (y_1\ y_2) \begin{pmatrix} 16 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \rho\, (u_1\ u_2) \begin{pmatrix} 1 & 0 \\ 0 & 9 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

the index can be cast into a matrix form (W_1 and W_2 are weighting matrices):

$$J = \int_0^\infty (y^T W_1 y + \rho u^T W_2 u) dt$$

and, replacing the output by its expression $y = Cx$, index J is an expression depending on future states and inputs:

$$J = \int_0^\infty (x^T C^T W_1 C x + \rho u^T W_2 u) dt$$

General linear quadratic regulator problem statement. From the previous example, in a general case, the optimisation problem can be formulated as follows:

Given the process defined by (2.17), (or (2.29) in DT), with initial state $x(0) = x_0$, find the control input minimising the index:

$$J_c = \frac{1}{2} \int_{t=0}^{t_f} (x(t)^T Q x(t) + u(t)^T R u(t))\, dt + \frac{1}{2} x(t_f)^T S_N x(t_f) \tag{7.2}$$

$$J_d = \frac{1}{2} \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k) + \frac{1}{2} x_N^T S_N x_N \tag{7.3}$$

where Q, and S_N are positive semi-definite $n \times n$ matrices, R is a positive definite $m \times m$ matrix[2] and t_f (N in the DT case) is a user-defined "control horizon".

Note that, as we are working on linearised models, the variables are incremental, *i.e.*, the index refers to deviations or transient errors when trying to achieve a desired set-point: Q is the cost of the transient path, and S_N is the "final" cost of not "hitting" the target state[3].

The previous index refers to control tasks lasting a fixed time (such as positioning a robotic arm). In process control, running continuously, the indexes are usually defined with an infinite final time, as in the previous example:

$$J = \frac{1}{2}\int_0^\infty [x(t)^T Q x(t) + u(t)^T R u(t)]dt \tag{7.4}$$

$$J = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T Q x_k + u_k^T R u_k) \tag{7.5}$$

In this way, the control engineer casts the specifications into suitable matrices Q and R, as it was done in the example.

7.1.1 Linear Regulators

It can be shown (see a detailed problem statement in Appendix D) that the solution to the minimisation of the cost index (7.4), where $x(t)$ is the trajectory of the CT system $\dot{x} = Ax + Bu$, is a *linear time-invariant state feedback*, similar to the solution of the pole-placement problem, (6.2), being:

$$u_{opt}(t) = -R^{-1}B^T S\, x(t) = -Kx(t) \tag{7.6}$$

Matrix S is the positive definite solution of the *Riccati equation* [133]:

$$0 = A^T S + SA - SBR^{-1}S + Q \tag{7.7}$$

Similarly, for a DT system $x_{k+1} = Ax_k + Bu_k$, the minimisation of the cost index (7.5) leads to the linear state feedback

$$u_{k,opt} = -R^{-1}B^T S\, x_k = -Kx_k \tag{7.8}$$

where matrix S is the solution of the DT Riccati equation:

$$S = Q + A^T SA - A^T SB(B^T SB + R)^{-1}B^T SA \tag{7.9}$$

[2] Otherwise there will be a non-penalised control direction so the optimal controller would have infinite gain on it: the problem would be ill-posed.

[3] Other optimisation problems with significance in control engineering are the fixed final state (S_N tending to infinity), and the CT minimum-time control (in which a fixed final state must be reached in a minimum time with a particular actuator saturation limit). The reader is referred to [81, 122] for details on these.

A simplified derivation of the optimal controller, for the DT case, is briefly presented in Appendix D.2. The reader is referred to [122, 81] for ample discussion. As the cost index is quadratic and the system model is linear, the resulting regulator is named in the literature as linear quadratic regulator .

In the LQR design approach, the only design parameters are the weighting matrices, Q and R, and some other properties, like closed loop bandwidth, pole position or response speed and damping, are not directly pursued. As a result, achieving a particular settling time (pole positions) must be made by trial and error on these matrices. In any case, the LQR solution has a number of properties and options that are rather interesting from the control user viewpoint.

MATLAB® implements the CT optimisation routines in its command `lqr`, with the syntax: `[K,S,E]=lqr(A,B,Q,R)`. It returns the optimal state feedback gain, K, the Riccati matrix, S, and the eigenvalues, E, of the closed-loop system $(A - BK)$. The command `dlqr` is used for the DT regulator.

Once the LQR controller is obtained, the dynamic behaviour of each controlled variable can be checked and the closed-loop poles can be evaluated. If we want to modify the system behaviour, the tuning parameters are the weighting matrices, Q and R, and some basic interpretation and rules of thumb are available. The following example outlines the procedure.

Example 7.2. Let us have a third-order system with state space representation:

$$\frac{dx}{dt} = \begin{pmatrix} -2 & 1 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -4 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 1 & -0.5 \\ 1 & 0.2 \end{pmatrix} u$$

$$y = \begin{pmatrix} -17 & 4 & -10 \\ 5 & 8 & -6 \end{pmatrix} x$$

where y_1 must have control four times tighter than y_2 and actuator u_2 may have three times more deviation than actuator u_1, when an initial off-equilibrium situation caused by a sudden disturbance or set-point change is corrected.

The solution for the problem in MATLAB® language is:

```
[K,S,E]=lqr(A,B,C'*[16 0;0 1]*C,rho*[9 0;0 1]);          (7.10)
```

where ρ can be adjusted (by trial and error) to achieve a particular settling time. For example, a controller achieving the desired input and output deviation objectives and a settling time of 0.4 s can be found with $\rho \approx 2$. Figure 7.1 plots the result of the simulation code below (lines labelled as `(a)`), where the output vector has been augmented to include $u = -Kx$:

```
sscl=ss(A-B*K,B,[C;-K],0); [y t x]=initial(sscl,[1 1 1]',0.59);
figure(1), plot(t,y(:,1:2)), figure(2), plot(t,y(:,3:4))
```

Note that the settling time for y_1 is faster than that for y_2, as designed. However, in spite of our decision to "punish" deviations in u_1, the optimiser still uses u_1 quite a lot. That occurs because u_1 has more than double the "gain" in terms

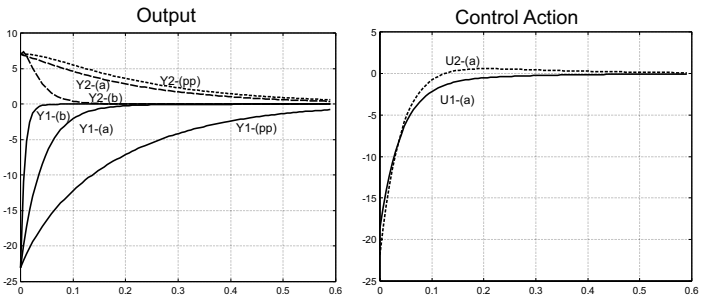

Figure 7.1. Non-zero initial condition response of optimal controller (3 designs).

of input/output controllability[4], so to achieve the required performance in y the optimal seems to be using actuator u_1. Based on these results, matrices Q and R can be fine-tuned to suit the engineer's needs. For example, to increase control requirements on y_2, $Q(2,2)$ can be changed to 3. To decrease amplitude of u_1, $R(1,1)$ is changed to 15. To make the loop faster, ρ is decreased to 0.1. Figure 7.1 plots the output of this second case, labelled as `(b)`. Control actions (not plotted, to avoid messing up the figure) are five times bigger: the approximate starting value of u_1 is -60, and that of u_2 is -120. Further refinements can be made if so wished.

Example 7.3. If a pole assignment solution is searched using MATLAB® commands and having the same poles as those resulting from the LQR design (vector E in (7.10)), a different feedback law will be obtained and a different dynamics will relate the input/output variables in a MIMO case: assigning the poles at the same place than the LQR design `(a)` in the previous example yields, for example, the outputs labelled as `(pp)` in the above figure. Control amplitudes (not plotted) are one third of those in the LQR design.

Note that the individual adjusting of input and output deviations from an initial non-zero state (by trial and error, however) is usually more related to "real engineering" MIMO specifications than pole-placement methodologies.

Stability. The (infinite time horizon) LQR controller always yields stable closed-loop systems if some conditions are satisfied:

- the system (A, B) is controllable or at least stabilisable (*i.e.*, the uncontrollable modes, if any, are stable)
- Q is positive definite so there does not exist a non-penalised direction in state space. A softer condition requires all the state variables (at least the unstable modes) to be weighted in the cost index, in such a way that if $Q = N^T N$ (*i.e.*, a "fictitious" output, $\breve{y} = Nx$, appears as $\breve{y}^T\breve{y}$ in the index), the pair (A, N) is *observable*.

Indeed, in this case, a controller K exists so that all closed-loop poles of $A-BK$ can be placed in a prescribed stable location. This controller will have

[4] As an exercise, check the three singular value plots of the open-loop transfer matrix with one and two actuators, to verify that input 1 is responsible of almost all the "maximum gain" all over the frequency range.

a *finite* value of the index (as its response will be exponential and $\int_0^\infty (me^{at})^2$ converges for $a < 0$). Hence, as the optimal regulator must have a cost below or equal to any of the pole-placement ones and all unstable loops would attain an infinite cost, necessarily the LQR controller must be stable[5].

It can be also proved (see, for instance [9]) that the LQR design presents good robustness against uncertainties in gain and delays[6]. Nevertheless, specific design techniques have been developed to deal with model uncertainty, as discussed in next chapter.

Settling-time Specifications: Combined LQR + Pole-region Specification

Settling-time specifications cannot be directly reflected in values for the design matrices. The usual procedure is a trial and error one over parameter ρ in (7.1): the smaller the value of ρ is made the faster the transient will be. In that way, a kind of regulation achieving a desired settling time with minimum control effort is achieved. For nearly-uncontrollable modes, changing its dynamics would require a very large control energy (low ρ).

Anyway, if a desired minimum exponential stability $e^{-\alpha t}$ ($\alpha > 0$) is desired, the cost index (7.4) can be modified to:

$$J = \frac{1}{2} \int_{t=0}^{\infty} e^{2\alpha t} [x(t)^T Q x(t) + u(t)^T R u(t)] dt \tag{7.11}$$

In this way, for the index to be finite, a solution must be obtained so that x and u are bounded in norm by $Me^{-\alpha t}$ (M an unknown constant), as specified.

These index modifications amount to setting time-varying Q and R but do not change the fundamental assumptions about the problem. To transform the problem to the original setting, let us define the new state and input vectors:

$$\bar{x} = e^{\alpha t} x(t); \qquad \bar{u} = e^{\alpha t} u(t)$$

Using (2.17), the new state equation is

$$\dot{\bar{x}}(t) = \alpha e^{\alpha t} x(t) + e^{\alpha t} \dot{x}(t) = (\alpha I + A)\bar{x}(t) + B\bar{u}(t)$$

Thus, the new LQR problem can be solved with data: $(\bar{A}, B, Q, R)$, being $\bar{A} = \alpha I + A$. Other changes of variables can be applied to specify a circular region in the s-plane for the poles, if settling time and damping specifications are to be approximately enforced [54].

[5] Formal proofs are based on "energy" considerations (Lyapunov functions), proving that $\frac{dx^T S x}{dt} < 0$ so $x^T S x$ decreases (to zero).

[6] If all sensors are available. Observers may drastically reduce stability margins, see Chapter 8.

Discrete-time case. In the DT case, if the system's response should be bounded by $M\beta^k$ ($\beta < 1$), the index must be changed to:

$$J = \frac{1}{2}\sum_{k=0}^{\infty} \alpha^{2k}(x_k^T Q x_k + u_k^T R u_k) \qquad \alpha = \frac{1}{\beta} > 1 \tag{7.12}$$

Now, with the change of variable:

$$\bar{A} = \alpha A; \qquad \bar{B} = \alpha B$$

designing an optimal controller for the modified syste $(\bar{A}, \bar{B})$, and keeping the same Q and R, will solve the optimisation of (7.12) for the original one (2.29).

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `lqr,dlqr,lqry`.

Remark 7.4. In some applications, *frequency-weighted* cost indexes are useful for stressing, for example, that strong controller gain is desired in a particular frequency range (low frequency, disturbance-dominated frequencies, *etc.*) but there is no need for such a high gain in other frequencies where disturbances or set-point changes are not present. Indeed, the remarkable interest of this idea led to the development of a generalised framework, described in Section 7.4.

Sampled-data systems. The proposed cost index for a discrete system does not take into account inter-sample behaviour. An alternate setup for optimal control of sampled-data systems is minimising the continuous *integral* cost index (7.4) with a discrete regulator (assuming ZOH discretisation). This problem can be cast as a discrete LQR one with a mixed cost index in the form (D.22). For details, see [48]. The resulting regulators, however, do not exhibit significant differences with optimisation of (7.5) if sampling time selection is adequate, if Q and R in the discrete case are equal to the continuous ones multiplied by the sampling period, T_s.

The MATLAB® command `K= lqrd(A,B,Q,R,Ts)` gives the DT feedback law under discussion.

7.2 Optimal Output Feedback

Problem statement. LQR control requires the system state accessibility to compute the current action, as indicated by (7.6) or (7.8). If the set of measured variables is $p < n$, an observer could be implemented, as shown in Section 6.2. But an optimal approach can be also followed to design the estimator, mainly if there are noisy measurements or disturbing inputs (coloured

noise) in the system. In this section, the problem of achieving minimal deviations on the variables of interest, when subject to random disturbances, will be presented.

In this case, the least squares indexes to be minimised may be expressed in terms of *variances*, such as $E(y^2)$ or $E(e^2)$, giving rise to a family of design methodologies under the name of *minimum-variance* estimation and control. For a SISO approach, the reader is referred to, for example, [63]. The MIMO approach is usually denoted as linear stochastic control [14], and its basic ideas will be outlined in this section.

For simplicity, the discussion will be restricted to discrete-time random disturbances. The performance objectives, in the MIMO case, will be to minimise a cost index similar to (7.3). However, the presence of random disturbances makes J a random variable and repeating experiments will result in a variety of cost index values. The stochastic control problem deals with minimising its *expected value*, so the *mathematical expectation* notation should be introduced[7].

The cost index (7.3), in a stochastic setting, will be:

$$J = E[\frac{1}{2}\sum_{k=0}^{N}(x_k^T Q x_k + u_k^T R u_k) + x_N^T S_N x_N] \tag{7.13}$$

The solution to this problem can also be decomposed in two steps: "optimal observer" design and state feedback LQR control.

7.2.1 Kalman Observer

In the following, the design of the above-mentioned optimal observer or filter will be pursued, as a first step towards the solution of the full control problem (minimisation of (7.13)).

Optimal observer. The main idea is designing a convergent observer[8] of the system state, x_k, with *minimum sensitivity to disturbances*, given a model with p outputs and defined by:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Gv_k \\ y_k &= Cx_k + w_k \end{aligned} \tag{7.14}$$

where the "size" of process noise, v_k, and measurement noise, w_k, is specified in terms of their *variance matrices*, $V = E(vv^T)$ and $W = E(ww^T)$, with dimensions $n \times n$ and $p \times p$, which are assumed to be known. Process and measurement noise are also assumed to be uncorrelated, $(E(v^T w) = 0)$.

[7] The basic ideas on discrete multivariable random processes are outlined in Appendix E. The reader may need to browse through this appendix to better comprehend some of the ideas below.

[8] *i.e.*, stable so that under no disturbances, the true state is estimated.

For the optimal observer (or filter), different assumptions can be taken, as seen in the previous chapter, the final goal being to compute an observer gain L, (6.12). In Section 6.2.1, this matrix was computed based on the location of the observer poles and, as a result, it influenced the behaviour regarding measurement noise and process noise (see (6.16)).

If a model of the disturbances is available, as for instance (7.14), for an observer like

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + L(y_k - C\hat{x}_{k|k-1}) \tag{7.15}$$

there exists an "optimal" observer gain, leading to a particular pole location, to achieve the least noise effect on the estimated state, that is, to minimise

$$J = E[(x_k - \hat{x}_{k|k})^T(x_k - \hat{x}_{k|k})] \tag{7.16}$$

This observer is denoted as the *Kalman filter*, introduced in [68, 69]. It minimises the size of the estimation error, $e_k = x_k - \hat{x}_k$, and $\hat{x}_k$ is the *best prediction* of x_k in a statistical sense[9], with the information available in sampling instant k ($\hat{x}_k \stackrel{\text{def}}{=} \hat{x}_{k|k}$).

As with the LQR problem, the optimal observer is time-variant (L depends on k).This is intuitively expected, as initial measurements (with no "past information") cannot use the state equation and the optimal is to "totally believe" the sensor readings. As past readings are gathered, the optimal is a weighted average of current and past data. The averaging coefficients change with time as the number of past data changes. In the observer approach, all past data are averaged onto the state vector, x_k, and current and past data are averaged in the observer equation. After a few samples, a reasonable state estimate is available, so the model equations start to be significant. Although the optimal observer is time-varying, in practice, the so-called time-invariant *stationary* observer, obtained after convergence of the observer gain to a constant value, is the one used in most cases. It is optimal once it has been in operation for some time (in particular, the settling time according to the observer poles). It is suboptimal in the first handful of samples, but that fact is irrelevant in continuously-operating process control.

In Section E.4.2, the full development for obtention of the observer gain is detailed. The stationary solution also involves computing the solution to a Riccati equation.

The stationary observer is the result, for example, of the `dlqe` command in MATLAB®, with the syntax `[L,P,Z,λ]=dlqe(A,G,C,V,W);`, where $P = E(x_{k|k-1} - x_k)(x_{k|k-1} - x_k)^T$, and $Z = E(x_{k|k} - x_k)(x_{k|}) - x_k)^T$ is the error covariance matrix. λ yields the observer eigenvalues and L is the observer gain.

[9] This means that $\hat{x}$ is equal to x *in average*, and that the residual error $x_k - \hat{x}_k$ is uncorrelated with the past measurements. In linear systems, uncorrelation is equivalent to statistical *independence* (no prediction improvement possible).

Design Rules

In this section, the implications in practical designs will be discussed, leaving the details in Appendix E for the interested reader.

Depending on the relation between the sizes of V and W (signal-to-noise ratio), the number of samples averaged (settling time) and the final stationary "gain", L, are variable.

The variance matrices must be known. However, this is a difficult issue in practice, specially regarding the "process noise" variance. Usually, observer design is based on guesswork or trial and error on matrix V.

Sensor noise matrix. A *diagonal* covariance matrix W for outputs is usually constructed, placing in the diagonal elements the noise variance for each of the sensors, experimentally measured[10], neglecting any cross-talk.

Process noise matrix. Regarding construction of V, if a particular source of random disturbances ψ_k is known to affect the process state via G in (7.14), the *dlqe* command can be directly applied. If several disturbance sources are known ($v_k = G_0 v_0 + G_1 v_1 + \dots$), then G is set to the identity matrix and matrix V can be built as $V = \sum G_i * \Lambda_i * G_i^T + \epsilon I$ where Λ_i are scalars or matrices with the variances of the known disturbance sources and ϵ is a small number accounting for disturbances with unknown directionality. A typical model is, for example, input noise where $G = B$ and Λ contains the variances of each of the noisy actuators. If nothing is reasonably known, an identity matrix in G and a diagonal V will do.

Eigenvalues. In any case, once a stationary value of L is calculated, the settling-time formulae with the eigenvalues of $A - LCA$ provide a hint on how many samples are being averaged to provide the filtered state measurement. If that is considered excessive or excessively low for the particular application, then a scaling on V allows the designer to change the observer's settling time so that a reasonable one is achieved. Lower values of V produce a slower filter with lower measurement noise amplification (smaller L), as lower process noise is associated with more confidence on the model's state equation.

Design code. Once the observer gain, L, is calculated, it is handled as any observer gain as discussed in the previous chapter, Section 6.2.1.

With the augmented models (Section 6.3), the Kalman filtering can be easily extended to mixed random + deterministic disturbances, and indeed that is often the case, at least with integral action. The following example shows the main concepts, combining the observer with a pole-placement regulator.

Example 7.5. Let us design an optimal observer for the system in Example 6.11 on page 180, whose augmented model (6.39) is repeated here for easier reference:

[10] Assuming Gaussian distribution, standard deviation is one third of the interval where 99% of sensor measurement noise values are into. It can be estimated from experiment or from the sensor data sheets provided by its manufacturer.

$$\begin{pmatrix} x \\ x_d \end{pmatrix}_{k+1} = \begin{pmatrix} 0.97 & 0.06 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ x_d \end{pmatrix}_k + \begin{pmatrix} 0.06 \\ 0 \end{pmatrix} u_k + v_k$$

$$y = \begin{pmatrix} 2 & 0 \end{pmatrix} \begin{pmatrix} x \\ x_d \end{pmatrix}_k + w_k$$

Sensor variance, W, is fixed to one unit, for reference. The regulator is designed to place one closed-loop pole at 0.8. Let us discuss the optimisation-based alternative to a pole-placement observer. As previously discussed, process noise variance is rarely known. Let us show the result of the Kalman observer design process for different values of V.

Case 1: `V=diag([1e5,1e5])`. Large values of V mean that the equation $x_{k+1} = Ax_k + Bu$ is totally unreliable, so there is no way of using past information: the result will be similar to a minimum-time observer.

Indeed, the optimal observer is `L=[0.5;0.485]`, with observer poles placed at 0 (for x_1) and at 0.942 (for the integral action). State variance[11] x_1 is 0.25, and for x_d, it is 10^6. It is essentially PI control. To obtain the controller transfer function, the observer equation (6.36) is typed into MATLAB®, and converted to transfer function form:

```
Ao=Ae-L*Ce*Ae-(eye(2)-L*Ce)*Be*K; Bo=L; Co=-K*Ao; Do=-K*Bo;
controller=ss(Ao,Bo,Co,Do,Ts);tf(controller)
```

The result is $u(z) = -1.902(z - 0.949)/(z - 1)\, y(z)$.

Case 2: Model fully reliable, noise-free, sensor extremely noisy in comparison. `V=diag(1e-5,1e-5);` yields L =[0.0055; 0.0031], with observer poles placed at 0.98 and 0.97 (very similar to the open-loop response). Many sensor samples are needed for a modification of the model variable. The resulting regulator is a very low-gain one: $u(z) = -0.018991z(z - 0.9669)/(z - 0.7911)/(z - 1)$.

Case 3: `V=diag(0.2 0.05)` depicts an intermediate case. Then, `L=[0.30; 0.15]`, the estimated error variance is 0.14 for x_1 and 1.7 for the disturbance. The resulting regulator is: $-0.973z(z - 0.97)/(z - 1)/(z - 0.33)$, adding a higher-frequency filter to the previous ones. Note that the state x_1 variance is reduced compared to the model-free 0.25, as several measurements are "averaged" using the model.

Of course, the power of the Kalman filter is that the previous ideas in SISO regulators get transparently translated into MIMO systems, via suitable variations of matrices V and W. Another elementary example will show some ideas, and a MIMO case study will be discussed at the end of the chapter.

Example 7.6 (Sensor fusion). The Kalman observer allows improvement of the precision of hardware sensors by implementing an optimal *sensor fusion* jointly with the use of past information (via a dynamic model). The more information sources are available, the smaller is the estimation error variance. Sensor fusion can be carried out with sensors of different quality, at different locations and even at

[11] The state variance of $0.25 = 1/2^2$ means that this approach is essentially equivalent to forgetting the model equation. In this case, with only $Y = CX + W$, the variance figure can be obtained from (E.24).

different sampling rates (for introduction to non-conventional sampling, see Section 9.4).

For example, two identical sensors for the process in the above example (Case 3) yield a 2×2 L matrix, and estimated x_1 state variance of: 0.08, half of than using only one sensor.

```
Ce=[Ce;Ce]; [L P Z E]=dlqe(Ae,eye(2),Ce,diag([0.2 0.05]),eye(2))
```

The obtained regulator:

$$\frac{1.16652z(z-0.9704)}{(z-0.2388)(z-1)}0.5(y_a+y_b)$$

averages the measurements, as intuitively expected, and as measurement noise will be less of a problem, the pole is faster than in Case 3 above, and gain is slightly increased as well.

The Kalman filter is a powerful approach used in advanced estimation applications in control and filtering. In some applications, Kalman estimators are used as such, and not as an intermediate for state feedback. By means of the MATLAB® expression `[KEST,L,P] = KALMAN(SYS,V,W)`, over a system 'SYS' as defined by (7.14), with disturbance covariance matrices W and V, gives a dynamic system, 'KEST', according to (7.15).

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `dlqe,destim,kalman,dkalman,kalmd`.

7.2.2 Linear Quadratic Gaussian Control

It can be proved [14, 122] that in linear MIMO minimum-variance control, a separation principle analogous to that in Section 6.2.4 does hold, in the sense that the optimal rejection of white noise disturbances in v_k and w_k is achieved by combining a Kalman observer and a LQR regulator. That is, under some assumptions,

– linear system, exact model,
– Gaussian disturbances or deterministic known disturbances,
– quadratic optimisation indices,

the optimal control problem can be decomposed into calculating the *state-feedback LQR* regulator minimising (7.3) and an *optimal observer* so that the estimation error, $E(x_k - \hat{x}_k)$, is minimised.This combination is usually termed the linear quadratic Gaussian regulator. For an infinite time optimisation horizon, the control law is a linear state feedback and the optimal filter is a full order linear state observer. A MIMO application is described in Case Study 7.6.

Note 7.7. Optimality as discussed in this section does not imply optimality on a real plant under modelling errors [42]. The singular value plot of the resulting regulators (see Chapter 8) will help on deciding whether the high-frequency noise amplification and robustness to modelling errors are under reasonable bounds or if redesign needs to be carried out (even changing the sensor quality or location).

Design code. The LQR controller gain, K, and the Kalman observer, L, are handled in the same way as in Chapter 6 and Example 7.5. However, there are some additional MATLAB® commands. By means of the MATLAB® expression `lqgr = lqgreg(KEST,K)`, a dynamic system composed of an optimal filter, 'KEST' (designed by MATLAB® command `kalman`), followed by an LQR linear feedback control, 'K' (designed by MATLAB® command `lqr`), is formed according to $u = -K\hat{x}$ and $\hat{x} = KEST(u, y)$. It is worth noting that the LQG regulator has the same dimension as the original system, as in pole-placement. In some cases, model reduction techniques (Section 3.10) will allow for a lower-order controller.

7.3 Predictive Control

Predictive control [40] has become popular in industry, mainly because of the simplicity of the required underlying models (in some cases, a non-parametric step response approximation can be used), the ability to incorporate *future* set-points and the availability of commercial software packages [106], able to solve the optimisation problems with constraints that conform the core of the methodology. Let us detail now the basics of one of the approaches. For detailed reference and alternative solutions, the reader is referred to [107, 34]. Some details and implementation issues are available in [46].

The core of the procedure lies in the ability to separate the future behaviour of the plant into two components:

- the behaviour to be accomplished if no input changes are applied (free response, predictor),
- the modifications to that basic behaviour if input changes are effected (forced response). Input changes in discrete-time control can be envisaged as steps that the regulator should apply to correct the first behaviour if not acceptable. In this way, calculation of the effect can be derived from step-response experiments (or simulations).

Indeed, due to the linearity assumption, the process output can be expressed as a sum of the output of the *predictor* plus the forced response. The calculation of the control actions leading to a desired output profile is the objective of the regulator.

For simplicity and coherence with the previous framework, a state space approach will be presented first. Afterwards, an outline of alternative methodologies and simplifications will be sketched.

7.3.1 Calculating Predictions

Prediction vector, free response. Let us assume a process with p outputs and m inputs, for which a state space model, (2.17)-(2.19), is available. If an observer (pole-placement or Kalman filter) is in place, then an estimate of the current plant state, $\hat{x}_k$, is available.

In predictive control, before calculation of the control action, u_k, a prediction of future output if no changes were effected is carried out, for determining control moves if the prediction does not meet the prescribed objectives.

Let us, for simplicity, assume that the current time is $k = 0$, and the last action, u_{-1}, is kept constant. The predictions will be given by:

$$\hat{x}_1 = A\hat{x}_0 + Bu_{-1}; \quad \hat{x}_2 = A\hat{x}_1 + Bu_{-1} \ \dots$$

$$\begin{pmatrix} x_{k+1} \\ u_{-1} \end{pmatrix} = \begin{pmatrix} A & B \\ 0 & I \end{pmatrix} \begin{pmatrix} x_k \\ u_{-1} \end{pmatrix}$$

$$\hat{y}_N = C\hat{x}_N = \begin{pmatrix} C & 0 \end{pmatrix} \begin{pmatrix} A & B \\ 0 & I \end{pmatrix}^N \begin{pmatrix} \hat{x}_0 \\ u_{-1} \end{pmatrix} \tag{7.17}$$

In the algorithm implementation, predictions are usually stacked in a prediction vector, from a minimum prediction horizon, N_1, to a maximum one, N_2:

$$F(x_0, u_{-1}) = \begin{pmatrix} \hat{y}_{N_1} \\ \hat{y}_{N_1+1} \\ \vdots \\ \hat{y}_{N_2} \end{pmatrix} \tag{7.18}$$

so the dimensions of F are $(N_2 - N_1)p \times 1$.

Effect of input changes. The system response to zero initial conditions (equilibrium) and a change in input is, actually, the step response. Of course, for additional input changes in future instants, the overall effect will be the sum of the step responses suitably scaled and delayed, resulting in a convolution-like formula:

If $\{S_k\} = \{s_0, s_1, s_2, \dots\}$ is the step response of the process under consideration, the response to an input change of amplitude Δu_0 is $\Delta u_0 \cdot \{S_k\}$. If, in sample $k = 1$, a further input change Δu_1 is applied, the overall response will be $\Delta u_0\{S_k\} + \Delta u_1\{S_{k-1}\}$ where $\{S_{k-1}\}$ denotes the delayed response $\{0, s_0, s_1, s_2, \dots\}$. In general, for a sequence of input step increments, $\{\Delta u_0, \Delta u_1, \dots\}$, the response is: $\{Y_k\} = \sum_{i=0}^{k} \{S_{k-i}\}\Delta u_i$.

Example 7.8. The system with step response $\{0, 0.2, 0.4, 0.5, 0.6, 0.6, 0.6, \dots\}$, subject to input $\{2, 2, 3, 4.5, 4.5, 4.5, \dots\}$ has an output given by:

$$\{0, 0.2, 0.4, 0.5, 0.6, \dots\} * 2 + \{0, 0, 0, 0.2, 0.4, 0.5, 0.6, \dots\} * 1 + \{0, 0, 0, 0, 0.2, 0.4, 0.5, 0.6, 0.6, 0.6, \dots\} * 1.5 \tag{7.19}$$

In practice, to limit the size of the summations, the number of allowed future input moves is usually limited to a user-defined *control horizon*, NU.

As shown in the next examples, using the same stacking as in (7.18), the forced response can be expressed in matrix form as:

$$Y = G\Delta U \tag{7.20}$$

Example 7.9. Check that the output of the system in the last example subject to arbitrary step input changes, Δu_k, can be expressed by:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ \vdots \end{pmatrix} = \begin{pmatrix} 0.2 & 0 & 0 & 0 \\ 0.4 & 0.2 & 0 & 0 \\ 0.5 & 0.4 & 0.2 & 0 \\ 0.6 & 0.5 & 0.4 & 0.2 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \Delta u_0 \\ \Delta u_1 \\ \Delta u_2 \\ \Delta u_3 \end{pmatrix}$$

MIMO systems. In multivariable systems, each of the terms s_k is a matrix whose element $s_k(i, j)$ is the response at time k for output i, when a step is applied to input j, and y_k, u_k are column vectors.

With that notation, the response equation can be also expressed as:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \end{pmatrix} = \begin{pmatrix} s_1 & 0 & 0 & 0 \\ s_2 & s_1 & 0 & 0 \\ s_3 & s_2 & s_1 & 0 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} \Delta u_0 \\ \Delta u_1 \\ \Delta u_2 \\ \Delta u_3 \end{pmatrix} \tag{7.21}$$

Where the matrix dimensions are $(p\,(N2-N1)) \times (m\,NU)$. These matrices are usually set up by a computer. They have a close relationship to the Haenkel parameters in (2.61).

Example 7.10. Let us consider the Wood and Berry distillation column [128], with an approximate model, scaled in engineering units [46], given by:

$$\begin{pmatrix} X_D(s) \\ X_B(s) \end{pmatrix} = \begin{pmatrix} \frac{2.56e^{-s}}{16.7s+1} & \frac{-5.67e^{-3s}}{21s+1} \\ \frac{1.32e^{-7s}}{10.9s+1} & \frac{-5.82e^{-3s}}{14.4s+1} \end{pmatrix} \begin{pmatrix} F_R(s) \\ F_S(s) \end{pmatrix} + \begin{pmatrix} \frac{38e^{-8s}}{14.9s+1} \\ \frac{49e^{-7s}}{13.2s+1} \end{pmatrix} F(s) \tag{7.22}$$

where X_D and X_B refer to distillate and bottom concentrations of a volatile compound, F_R and F_S are the reflux and steam flows, F refers to inflow disturbances. Time units are in minutes.

To set up a predictive controller, a constant-input predictor, (7.17), should be set up (obtaining the current state via an observer). Then, for $NU = 3$, $N1 = 4$ and $N2 = 7$, the prediction equation would be completed by the step response matrix[12]. Discretising the response at $T_s = 1$ min, the result is given by:

[12] To keep matrix dimensions small, the values of the design parameters are not following the design guidelines to be later discussed.

$$\begin{pmatrix} X_{D4} \\ X_{B4} \\ X_{D5} \\ X_{B5} \\ X_{D6} \\ X_{B6} \\ X_{D7} \\ X_{B7} \end{pmatrix} = \begin{pmatrix} 0.29 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0.42 & 0 & 0.29 & 0 & 0 & 0 \\ -0.26 & -0.39 & 0 & 0 & 0 & 0 \\ 0.55 & 0 & 0.42 & 0 & 0.29 & 0 \\ -0.52 & -0.75 & -0.26 & -0.39 & 0 & 0 \\ 0.66 & 0 & 0.55 & 0 & 0.42 & 0 \\ -0.75 & -1.09 & -0.52 & -0.75 & -0.26 & -0.39 \end{pmatrix} \begin{pmatrix} \Delta F_{R0} \\ \Delta F_{S0} \\ \Delta F_{R1} \\ \Delta F_{S1} \\ \Delta F_{R2} \\ \Delta F_{S2} \end{pmatrix} \tag{7.23}$$

Superposition. Finally, The overall prediction equation is the addition of the predictor and the to-be-calculated forced response:

$$Y = G\Delta U + F(x_0, u_{-1}) \tag{7.24}$$

7.3.2 Objective Function

The objective function in predictive control is (assuming current sample is $k = 0$) a least squares criterion:

$$J = \sum_{k=N_1}^{N_2} e_k^T Q e_k + \sum_{k=0}^{NU-1} \Delta u_k^T R \Delta u_k \tag{7.25}$$

where e_k is the vector of loop errors (so $e_k^T Q e_k$ is a weighted norm of the error) and Δu_k is a vector with the control moves on each manipulated variable. The errors are calculated as

$$e_k = y_k - r_k$$

where r_k is a sequence of vectors formed with future references to be tracked by each of the controlled process variables.

It is similar to the LQR cost (7.3), but incorporates future set-points and penalises the control *moves* instead of the actual control actions[13]. Note also that only a limited number of control moves, NU, is allowed to achieve the control objective. For example, $NU = 1$ produces something similar to "steady-state control" where a single step is calculated to provide average zero deviation during the prediction window (closely related to DC gain if N_2 is big enough).

The approximate interpretation of the index parameters is:

- Q, R are (usually diagonal) matrices weighting individual outputs and control moves. Their meaning is similar to those in the discrete LQR index, (7.3),

[13] To pose a similar problem in an LQR set-up, it is only needed to add an integrator at the plant inputs and design a controller for the augmented plant, with straightforward weight modifications.

- N_1: minimum time elapsed until reduced error is desired. Usual value is 1 or the plant's dead time, if any. If the plant is non-minimum-phase, this parameter allows elimination of the first samples of inverse response from the control objectives[14],
- N_2: the future time over which the predictions are evaluated. Usual value is around the desired closed-loop settling time (with a correctly chosen sampling time, N_2 should range between 15–40 samples),
- NU is the number of input moves considered in the calculation. Usually, it may range from 1 to 0.2-0.3 times N_2. The lower its value is, the smoother the resulting controllers are (if NU is greater than the plant order, with reduced R the predictive controller computes nearly minimum-time controllers).

Solution. The unconstrained solution to the optimisation is:

$$\Delta U = (G^T Q^* G + R^*)^{-1} G^T Q^* (w - F(x_0, u_{-1})) \tag{7.26}$$

where:

- G and F are obtained from the process model (7.24) and estimated state,
- Q^* is a block-diagonal square matrix with size $p(N_2 - N_1 + 1)$, each block being built by matrix Q in the cost index,
- R^* is a square matrix (size $m\,NU$), formed by block-diagonal stacking of R,
- w is the vector of future set-points, arranged in the same way as the outputs are in (7.18).

Receding horizon policy. Only the first of the optimal actions is applied to the plant. At the next sampling instant, recalculation of a whole batch of control moves is done based on new state estimations, the subsequent predictions and future references[15]. This is called receding-horizon optimisation, as the cost index acts like a window on the near future, displaced as time goes on: u_{k+1} with the information at time k will not be the same as u_{k+1} with the information available at time $k+1$.

7.3.3 Constraints

If the unconstrained solution (7.26) is directly implemented, with constant set-points, the resulting regulator can be cast into a state feedback form (all future predictions depend on current state so Δu will depend linearly on it: $\Delta u_k = -K\hat{x}_k$). With non-constant references, the result is a two degree of

[14] See Section 8.3.2 for limitations regarding required speed of response in non-minimum-phase plants

[15] For computational reasons, only the first move is actually calculated, if possible, as the rest of them are not going to be applied.

freedom controller $\Delta u_k = -K\hat{x}_k + L\rho_k$, where ρ_k is a vector of future set-point values[16].

However, the practical interest of predictive control lies in the possibility of constrained optimisation that many commercial solutions allow: efficient operation of a process often compels operating points lying near technological limits (process constraints).

Constraints may be specified as *hard* (bounds whose violation is not tolerated at any time) or *soft* (bounds that, under certain circumstances, can be violated but incur substantial cost penalties – the cost function becomes non-quadratic but still convex).

Usual constraints are:

- bounds for manipulated variables, u, and process outputs, y,
- slew-rate saturation in actuators,
- forcing monotonic behaviour in outputs,
- overshoot constraints.

A general explicit solution like (7.26) does not exist in the presence of constraints, and usually numeric optimisation algorithms are used, such as quadratic programming (QP), ellipsoid methods, *etc.* The use of generic non-linear optimisation routines allows for non-linear models to be used[17] [8]. For further detail on constrained predictive control calculation, the reader is referred to [90, 130] and the already-mentioned books and reviews.

7.3.4 Disturbance rejection

Note that, in vector F, the effect of disturbances is accounted for if the state space representation of the plant is augmented to include their partially deterministic behaviour (Table 2.2). The most usual disturbance model is a constant, $d_{k+1} = d_k$.

To implement feedforward total rejection of measurable disturbances, their effect into the process state must be known, integrating deterministic, random, measurable and unmeasurable disturbances into a Kalman filter setup for a suitably augmented plant. However, the availability of complex disturbance models is uncommon, and MIMO plant augmentation may be cumbersome. So, some simplified approaches are used in commercial controllers.

[16] The optimisation has also the beneficial side-effect of smoothing the reference trajectories in some processes, by taking into account the control action limitations in the cost index.

[17] Although its complexity hinders real-time implementation in many cases, non-linear models can be used for prediction and then a linearisation around the trajectory can be used to generate the control actions. Those actions can be further refined by taking new linearisations, in an iterative set-up.

Simplifications

The most usual simplification (assuming a constant disturbance) is, by knowing the approximate step response ($\{0, g_1, \ldots, g_N, g_N\}$, with settling time N and DC gain g_N) to carry out the predictions as:

$$y_k = g_N u_{k-n} + g_{n-1}\Delta u_{k-(n-1)} + \cdots + g_1 \Delta u_{k-1} + d_k$$

or, by using a finite approximation to the impulse response (increments of step response $h_i = g_i - g_{i-1}$) and absolute values of manipulated variable:

$$y_k = h_1 u_{k-1} + \cdots + h_N u_{k-N} + d_k$$

so that the disturbance, d_k, can be estimated from input/output measurements at instant k:

$$d_k \approx y_k - (g_N u_{k-n} + g_{n-1}\Delta u_{k-(n-1)} + \cdots + g_1 \Delta u_{k-1})$$

The prediction vector is calculated by using the same equations for future y_{k+p}, assuming d_k will be constant in future samples (*i.e.*, $d_{k+p} = d_k$).

In multivariable predictors, each of the g_k is a matrix where $g_{k\,(i,j)}$ is the k-th term of the step response in input j of output i, and d_k is a column vector.

Remark 7.11. With these simplifying assumptions, step-response experiments are the only experiments needed to put into operation a centralised multivariable predictive control. This is particularly appealing and many industrial implementations are based on variations of this scheme. Some other simplifications are also widely used. The interested reader is referred to the previously cited references for detailed development, examples and applications.

7.4 A Generalised Optimal Disturbance-rejection Problem

Problem statement. In Section 4.5.3, it was shown that many open-loop, closed-loop, 2-DoF and disturbance-rejection tasks can be depicted in block-diagram form as a lower linear fractional transformation (a generalised interconnection, discussed in Section 2.7.2). Figure 4.1 is here repeated as Figure 7.2 for convenience.

Under that framework, the global closed-loop transfer matrix has the set-point commands and disturbances as inputs, and the errors (deviations) of a set of controlled variables as outputs. The generalised plant becomes:

$$\begin{pmatrix} \text{deviations} \\ \text{controller input} \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} \text{set-pt. and dist.} \\ u \end{pmatrix} \tag{7.27}$$

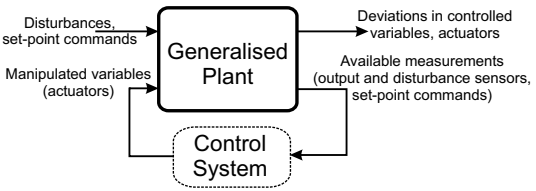

Figure 7.2. A general configuration of a control loop

where P_{ij} are themselves transfer function matrices expressing the process model, the interconnection structure and some frequency weights expressing control specifications (see later).

The closed-loop transfer matrix depends on the generalised plant, P, and the controller, K. A good controller is one that makes the closed-loop transfer matrix (2.53) *small*. Note that for different generalised plants, a number of control problems can be considered, as examples in Section 4.5.3 show.

The objective of optimisation-based solutions to the above problem is clear: it is to make the closed-loop transfer matrix the *smallest* possible.

Of course, a precise meaning for "small" depends on the used system norm (Appendix C).

The solution. In the 1980s, a solution [44] for some optimal control problems in the above framework was found. It encompassed, among others, the LQG paradigm presented in previous sections.

In particular, a state space solution exists for the case of minimising the ∞-norm, (3.38), and the 2-norm of the closed-loop transfer function matrix[18].

From a user's point of view, there may be some confusion regarding which norm should be selected. The next chapter and Appendix F, in particular Remark F.1, will clarify the question. For the moment being:

- $\mathcal{H}_2$ optimal control is the generalisation of the LQR + Kalman filter into the LFT framework, admitting frequency weights. It regards minimising the variance subject to white-noise input,
- the $\mathcal{H}_\infty$ optimal regulator has deep connections with robustness to modelling errors so it is the recommended choice when combining performance-robustness trade-off criteria.

The resulting controller order is equal to that of the generalised plant. However, the $\mathcal{H}_\infty$-regulator state space realisation, (A, B, C, D), is not interpretable as an observer + state feedback in a general case, but as an observer plus a "worst-case" term [119, 133].

The details of the solution require a mathematical development out of the scope of this work and, anyway, it is not needed for grasping the fundamental ideas regarding its application to engineering problems, thinking of them

[18] Theoretical developments in the 1990s also consider some cases of the 1-norm [113].

as a general framework where the now-classical LQR and LQG approaches can be embedded, with a similar interpretation regarding basic features. The interested reader can consult [133] for an in-depth analysis. Some solutions are implemented in the *Robust Control Toolbox* and *μ-Synthesis Toolbox* of MATLAB®, so they can be used by a practising engineer.

For the control problems in the framework under discussion to be *well-posed*, some of the outputs of the generalised plant must be the *control actions*: if no restrictions on the size of the control actions were put in place, the "optimal" regulators would have infinite gain: the block-diagrams in Section 4.5.3 have to be slightly modified (see below).

7.4.1 Design Guidelines: Frequency Weights

The examples in Section 2.7.2 illustrate the block-diagram operations for obtaining the generalised plant in some frequent cases. However, a key issue for applicability is being able to introduce engineering specifications (settling time, bandwidth, *etc.*) in this framework. This is done by means of "weights" in the generalised plant, generalising the state and input weights (Q, R) in LQR control and the disturbance sizes (V, W) in Kalman filtering. Those weights may be full transfer function matrices and its behaviour on all frequencies must be designed. Let us discuss that issue.

In the design of feedback control systems there are, in many occasions, conflicting requirements. For example, the fast detection of process noise *vs.* the insensitivity to measurement noise, as discussed on page 173. Another conflicting requirement is the need for limited control activity (due to saturation and finite actuator bandwidth) *vs.* the desire for fast, tight control of the outputs.

Fortunately, some of these requirements may be thought of as applying to different *frequency bands*:

- usually, the control objectives refer to low-frequency signal components, as high-frequency ones are assumed to be wide-band measurement noise. *Control* must act at low frequencies and *filtering* must concentrate on higher ones,
- tight control is desired at frequencies dominated by the disturbance effects (usually low-frequency process noise) and, on the other hand, a reasonable limitation on high-frequency actuator activity is desired (*i.e.*, the loop should be "opened" at high frequencies).

Based on these ideas, sensible frequency weights to the different exogenous inputs and outputs of the generalised plant can be designed to pose meaningful control problems in practice. The basic building blocks are the low-pass, high-pass and band-pass filters described in Appendix A.2.2.

The following examples will show the fundamentals of the procedure. Full details on weight selection and design methodologies can be consulted in [119].

Example 7.12. The disturbance-rejection problem in example 4.4 will now be enhanced to include a requirement of small errors at lower frequencies, including a low-pass filter W_e, amplifying the lower frequencies to make them more significant, as well as a limitation, ρ, on the control activity. Figure 7.3 shows the block-diagram, detailing the generalised plant structure, leading to the equation:

$$\begin{pmatrix} \text{errors}_{h\times 1} \\ \text{sensors} \end{pmatrix} = \begin{pmatrix} 0 & \rho \\ W_e & W_e G \\ I & G \end{pmatrix} \begin{pmatrix} d \\ u \end{pmatrix} \tag{7.28}$$

where $h = m + p$ is the total number of inputs + outputs.

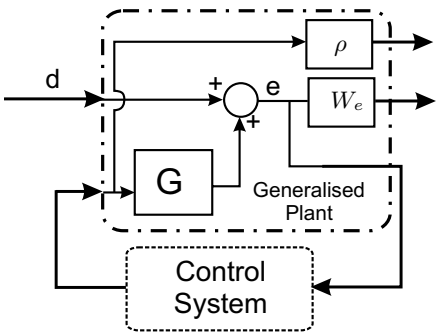

Figure 7.3. Weighted closed-loop disturbance rejection

A possible transfer matrix for W_e is a diagonal one with low-pass filters in it, such as:

$$W_e = \text{diag}\left(\frac{q_i}{\frac{1}{\lambda_i}s + 1}\right)$$

where a larger q_i forces the low-frequency errors in output y_i to be smaller, and λ_i is a target "bandwidth" in which disturbance rejection should be effective in each output (the weight generalises the Q matrix in LQG cost (7.13)). Input weight ρ can be a diagonal matrix specifying different penalisation for each of the available actuators, equivalent to the R matrix in (7.13). If ρ is a high-pass diagonal transfer matrix, it will force control activity at high frequency to be smaller. So, the selection of q_i, λ_i and ρ_i will translate the engineering requirements into the optimisation framework. The case study at the end of this chapter presents an example of weight selection in a distillation process.

The generalised plant framework is a recasting (in block-diagram and LFT language) of the optimisation problems in Sections 7.1 and 7.2 (stationary case). The greater generality of the approach now discussed is due to:

- the ability to incorporate feedback and feedforward (and mixed two degree of freedom cases) control in a unified framework,
- The ability to specify desired loop behaviour in frequency response, generalising classical frequency-response design (Bode diagrams, lead-lag compensators,...),

- the ability to chose which transfer matrix norm has to be minimised by feedback (least squares LQG is a particular case of a 2-norm minimisation problem), and its relationship to robustness to modelling errors (see next chapter).

Frequency weights can be inserted into the generalised plant or, to separate information flow and interconnection from control specifications, set up so that if P is the unweighted plant, the weighted one is:

$$P_w = W_O P W_I \tag{7.29}$$

where W_O are the weights penalising size of the generalised outputs, and W_I are weights denoting the relative size of the generalised inputs (set-points, disturbances, measurement noise). W_O and W_I are usually set up as diagonal transfer matrices, its lower components being equal to 1 (the actual input/output signals connected to the controller must be, of course, unweighted). An example of this approach is later detailed in the case study.

Remark 7.13. As the resulting regulator order is equal to that of the generalised plant, high-order weights result in a high-order regulator[19]. Care must be taken with round-off errors in discretised implementations (see Example 9.1) [73].

The problem, as stated now, does not consider modelling errors, so the resulting design refers to the so-called *nominal performance*.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `lftf,h2lqg,dh2lqg,hinf,dhinf,hinfsyn,h2syn`.

Mixed sensitivity. Let us correctly pose the closed-loop disturbance rejection problem whose generalised plant was set up in (7.28). The partitioning of the generalised plant in (7.27) results in:

$$P_{11} = \begin{pmatrix} 0 \\ W_e \end{pmatrix}; \quad P_{12} = \begin{pmatrix} \rho \\ W_e G \end{pmatrix}; \quad P_{21} = I; \quad P_{22} = G$$

so, using (2.53), the function whose norm should be minimised is:

$$F_l(P,K) = \begin{pmatrix} \rho K(I-GK)^{-1} \\ W_e(I + GK(I-GK)^{-1}) \end{pmatrix}$$

Denoting as $S = I + GK(I-GK)^{-1} = (I-GK)(I-GK)^{-1} + GK(I-GK)^{-1} = (I-GK)^{-1}$, *i.e.*, the closed-loop sensitivity (Section 4.5, unweighted

[19] Note that the order of a diagonal weight is the sum of orders of each transfer function!

transfer function (4.5) between disturbance and output), the objective is to minimise:

$$\| \begin{pmatrix} \rho KS \\ W_e S \end{pmatrix} \| \tag{7.30}$$

This control problem is named in the literature [119] as the *mixed sensitivity* problem, where a weighted combination of the disturbance rejection performance, S, and the transfer function from disturbance to control action, KS, must be made small enough. The effect of the stacked term ρKS is beneficial regarding robustness to modelling errors, and necessary for the problem to be well-posed and meaningful in practice.

7.5 Summary and Key Issues

Optimisation-based control is an appealing approach, as it obtains the "optimal" controller on a particular setting. The issue is how to translate engineering specifications into a single cost index to be minimised.

The signal-based approach (LQG) optimal disturbance rejection requires disturbance models that are difficult to obtain in practice.

The frequency-based approach (norm optimisation) uses as disturbance models some frequency weights approximating the disturbance spectrum, if known, and control specifications are also translated as some prototypical filters.

Once a first controller is obtained, fine-tuning it and changing the design parameters is easier than in the pole-placement framework. Some trial and error is carried out to balance disturbance rejection, settling time and actuator usage. These issues are closer to MIMO practical requirements than the abstract concept of "pole".

7.6 Case Study: Distillation Column

Let us consider the Wood and Berry distillation column [128], with an approximate model given in (7.22), where X_D and X_B refer to distillate and bottom concentrations of a volatile compound, F_R and F_S are the reflux and steam flows and F refers to inflow disturbances and time units are in minutes.

Let us compare the LQG and optimal disturbance rejection with generalised plant set-ups, solved using MATLAB® toolboxes. Furthermore, in the next chapter, the standard mixed-sensitivity procedure is pursued for this plant on page 247, in the context of robustness analysis.

Design 1: Discrete LQG Approach

The model is built by forming the suitable transfer matrix and setting up the delays (`ioDelayMatrix`) in MATLAB® language, then discretising (`c2d`) and incorporating the delays into the state space representation (`delay2z`):

```
s=tf('s'); Ts=1; g11=2.56/(16.7*s+1); ...

g=[g11 g12;g21 g22];g.ioDelayMatrix=[1 3;7 3];gz=c2d(g,1,'zoh');
gzss=ss(gz); sys=delay2z(gzss);
```

rendering a 14-state realisation (no approximation is needed as the delay is a multiple of the sampling period).

Then, integral action is added so no steady-state offset occurs. Fictitious input disturbances are used to augment the state matrix, according to expressions in Table 2.2:

```
Abig=[sys.a sys.b;zeros(2,14) eye(2)]; Bbig=[sys.b;zeros(2,2)];
Cbig=[sys.c zeros(2,2)];
```

Then, a LQR design (`dlqr`), weighting mainly the squared output deviations, is carried out with the non-augmented system ($Q = C'C + 10^{-4}I$, $R = 0.7I$):

```
kl=dlqr(sys.a,sys.b,sys.c'*sys.c+1e-4*eye(14),0.7*eye(2));
 abs(eig(sys.a-sys.b*kl))
```

so the state feedback constant `kl` is obtained. The dominant pole is located at $z = 0.958$, corresponding to about 70 minutes settling time. If the obtained response were not satisfactory, scalings of Q and R would enable a faster/slower loop or reduction on deviations of a particular sensor or actuator.

Afterwards, a Kalman filter is designed, assuming a process-noise to measurement ratio of 5, with the arrangement given by:

```
l=dlqe(Abig,eye(16),Cbig,eye(16)*5,eye(2)*1);
```

having a dominant pole at 0.96. As usual, faster observers would have been obtained with increased "process noise". If lower "variance" is designed for the integral action states, the offset-correction dynamics can be made slower if so wished.

To close the loop, the state feedback constant needs to be enlarged by appending an identity matrix (to carry out cancellation of the estimated input disturbance – Section 6.3). The closed-loop regulator (`dreg(a,b,c,d,k,l)`) is formed and the equations for the overall closed loop (`feedback`) are built for evaluation (choosing as input the disturbance entry point after the plant). As disturbance enters via a transfer function `gdz`, the resulting feedback must be multiplied by it:

```
[Ac,Bc,Cc,Dc]=dreg(sysb.a,sysb.b,sysb.c,sysb.d,[kl eye(2)],1);
re=ss(Ac,Bc,Cc,Dc,1); clp=feedback(eye(2),series(re,sys));
clpd=series(gdz,clp);
```

The disturbance step response is plotted in Figure 7.5 on page 217. The overall controller order is 16 (14 states + 2 disturbance integrators). The frequency response (sigma-plot from d to (y_1, y_2)) appears in Figure 7.4, jointly with the next design, for later comparison.

Design 2: Continuous $\mathcal{H}_\infty$ Optimisation

In this case, for the sake of comparison, a continuous design will be pursued (so the resulting regulator must be implemented at a fast-enough sampling rate) with the generalised plant.

The disturbance model will be also incorporated into the generalised plant. Furthermore, a first-order Padé approximation (`pade`) is carried out on all delays, and transformed to state space (`ss`) and transfer function (`tf`) representation for later use:

```
gd1=38/(14.9*s+1); gd2=49/(13.2*s+1); gd=[gd1;gd2];
gd.ioDelayMatrix=[1;0];
gpss=minreal(ss(pade(g,1))); gptf=tf(gpss);
gdpss=minreal(ss(pade(gd,1))); gdptf=tf(gdpss);
```

Then, the generalised plant is formed. Its inputs are the disturbance, d, measurement noises, n_1, n_2 (arranged in a column vector n), and the manipulated plant inputs, u_1, u_2 (vector u). Its outputs will be the plant outputs, $y = (y_1, y_2)^T$, the plant inputs, u (limitations on its size are needed for well-posedness), and the measurement-noise contaminated output, m_1 and m_2 (to be used as controller input, as vector m). So, it will total six outputs and five inputs:

$$\begin{pmatrix} y \\ u \\ m \end{pmatrix} = \begin{pmatrix} G_d & 0 & G \\ 0 & 0 & I \\ G_d & I & G \end{pmatrix} \begin{pmatrix} d \\ n \\ u \end{pmatrix} \tag{7.31}$$

and the MATLAB® syntax to build it is, for example:

```
GenPlNoWeight= [gdptf zeros(2,2) gptf;zeros(2,1) zeros(2,2)
eye(2);gdptf eye(2) gptf];
```

Weights. Some frequency-dependent weights must be set up to include the control specifications, so the generalised plant to be feeded to the optimiser will have the form $P_w = W_O P W_I$ in (7.29).

Regarding W_O, it will be built as a diagonal matrix, consisting of: true output weights (on y), control action weights (on u) and identity (on m, the signal to be used for closing the loop).

The plant output (first generalised output) size will be penalised by a low-pass diagonal weight (large penalty for deviations at low frequencies, lower penalty for deviations at higher frequencies). Input size (second generalised output) will be penalised by multiplying it by a constant diagonal matrix.

Regarding W_I, the sizes of d and n will be left the same, as significant measurement errors are expected in chemical sensors[20]. This is the "equivalent"

[20] Realising that $\overline{\sigma}(G_d) \approx 1$ at about 5 rad/s amounts to specifying that, above that frequency, sensor errors would be bigger in size to the disturbance-induced deviations trying to be rejected, so W_I acts as a request for the loop to be opened from 5 rad/s onwards.

of deciding a size on the measurement and process noise variances in Kalman filtering. If another signal-to-noise ratio is estimated, a different W_I can be set up, of course. In this case, W_I will be the 5×5 identity (d is dimension 1, n is dimension 2 and u (always weighted by 1) is dimension 2). So, the weighted plant is formed with the following code:

```
wu=[8 0;0 8]; wy=[50/(100*s+1) 0;0 50/(100*s+1)];
wo=[wy zeros(2,4);zeros(2,2) wu zeros(2,2);zeros(2,4) eye(2)];
wi=eye(5); GenPlWeighted= wo*GenPlNoWeight*wi;
```

Calling the optimiser `hinf(plant,nmeasures,ncontrollers,...)` (using the last two plant outputs, m, and the last two inputs, u, plus some search limits and tolerances, *etc.* For detail, see the help in MATLAB® command prompt. The regulator `k2` is formed, in state space and transfer function form[21]. It has order 11:

```
 P2ss=minreal(ss(GenPlWeighted));
Psm=pck(P2ss.a,P2ss.b,P2ss.c,P2ss.d);
 [k cl gamf]=hinfsyn(Psm,2,2,0.05,150,0.005);
 [at,bt,ct,dt]=unpck(k);
k2ss=ss(at,bt,ct,dt); k2tf=tf(k2ss);
```

so the closed loop can now be formed and its behaviour analysed. The controller feedback connection (LFT) is done with the unweighted plant:

```
clnwdesigned=minreal(lft(GenPlNoWeight,k2tf));
```

The step responses to a disturbance d and the maximum singular value (worst-case amplification) as a function of frequency are depicted in Figures 7.5 and 7.4 respectively, for both LQR and $\mathcal{H}_\infty$ designs:

```
figure(1),step(clpd,clnwdesigned([1 2],1));
figure(2)
sigma(clnwdesigned([1 2],1),clpd,GenPlNoWeight([1 2],1))
```

Outputs 1 and 2 from the generalised plant are the true plant outputs[22]. The unweighted open-loop plant response to the disturbances is also plotted to help in comparing what controllers have achieved (adding the step and sigma-plot of `gd`).

Both designs behave in a similar way. Note that control is effective below 0.2 rad/s (the achieved control bandwidth). If flow disturbances have significant components beyond that frequency, the controllers will *amplify* their effect (waterbed effect, see page 225). That fact might be important in practical implementation.

Of course, by suitably modifying the design parameters (weights, *etc.*), more "aggressive" regulators can be pursued. The issue now is how tolerant

[21] The commands `pck` and `unpck` carry out irrelevant housekeeping regarding compatibility between MATLAB® 5.3 and older versions.

[22] Outputs 3 and 4 are the plant inputs (control actions) and they have not been plotted, although they can be if so wished.

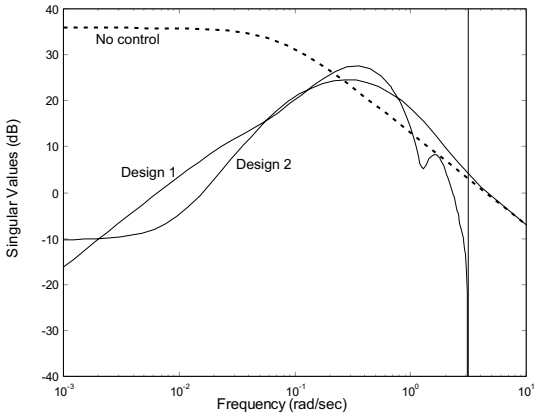

Figure 7.4. Singular value plots of the disturbance effect

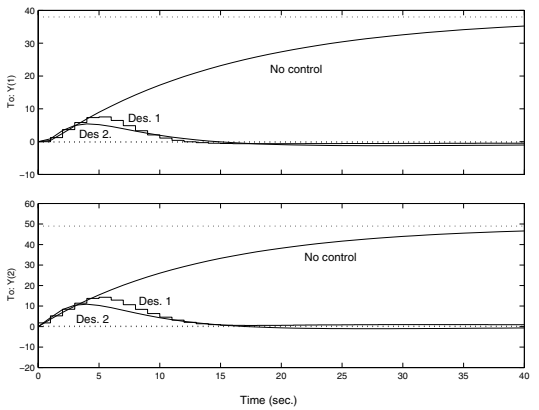

Figure 7.5. Step disturbance responses

to modelling errors those designs are, and which is the quality of the original model with respect to the real plant. This is the topic to be discussed in next chapter.

Remark 7.14. The sensitivity, $S = (I + GK)^{-1}$, is the closed-loop transfer matrix assuming unweighted disturbances affecting both output channels. It can be formed and plotted (left to the reader) with:

```
sens=minreal(feedback(-eye(2),series(k2ss,gpss)));sigma(sens)
```

The result is surprising at first glance, as the maximum singular value is around 1 at all frequencies: there is one direction in which strong disturbance attenuation does occur (associated with the minimum singular value) and no attenuation is pursued at all in the orthogonal direction. That happens because the disturbance transfer matrix has rank one (only one source of

disturbance). If disturbance attenuation is sought in "all directions", a full-rank disturbance model must be used (see the mixed-sensitivity design in page 247).

Order reduction. The eleventh-order regulator can be reduced without significant degradation of its response. This is common practice in the norm-optimisation framework as weights can add up significant order to the resulting controller.

A balanced realisation, `k2ssb`, in decreasing order of observability (= controllability) is obtained by:

```
[k2ssb g T Ti]=balreal(k2ss);
```

where additional outputs are g (controllability–observability singular values) and the balancing transformation, `T`, and its inverse, `Ti`.

Afterwards, some states can be eliminated. Simulation shows that, deleting the last seven states, a fourth-order regulator can be obtained without visible differences in closed-loop step and frequency response. The following code carries out the reduction:

```
k2ssred=modred(k2ssb,[5 6 7 8 9 10 11]);
```

and the final regulator transfer function matrix is:

$$
\begin{aligned}
k_{11} &= \frac{0.013885(s+10.92)(s+1.317)(s-0.4273)(s-0.1462)}{(s{+}0.007945)(s{+}0.003293)(s^2{+}5.114s{+}10.48)} \\
k_{21} &= \frac{-0.040685(s{+}37.94)(s{+}0.5355)(s{-}0.4114)(s{+}0.05037)}{(s{+}0.007945)(s{+}0.003293)(s^2{+}5.114s{+}10.48)} \\
k_{12} &= \frac{-0.027318(s{+}34.25)(s{+}1.048)(s{-}0.2045)(s{+}0.09384)}{(s{+}0.007945)(s{+}0.003293)(s^2{+}5.114s{+}10.48)} \\
k_{22} &= \frac{0.011265(s{+}1020)(s{+}0.6233)(s^2{+}0.1179s{+}0.003607)}{(s{+}0.007945)(s{+}0.003293)(s^2{+}5.114s{+}10.48)}
\end{aligned}
\tag{7.32}
$$

8

Designing for Robustness

In this chapter, an introduction to the fundamental concepts of robust control design is presented at an intuitive, introductory level. The reader must be aware that, by dealing with models, a representation inaccuracy is always present and the control system design must cope with the wider possible range of actual systems.

Although some qualitative facts about robustness and frequency-response SISO robustness were known in the 1940s [29], the advent of pole-placement SISO techniques [47] and "optimal" control in the 1960s disregarded robustness. However, optimal controllers were shown in the 1970s to be sensitive to arbitrarily small modelling errors in some circumstances [42]. In the 1980s a generalised theory of linear feedback control robustness allowing analysis and synthesis reached maturity [131, 43, 44, 86]. Some of the basic ideas will be discussed in this chapter.

8.1 The Downside of Model-based Control

Model-based control design immediately poses a question to the designer: how sensitive is the design to modelling errors? The issues related to this are the subject of this chapter.

The dependence on model quality needs to be analysed in any engineering optimisation problem. In fact, it applies to a wide variety of problems in which an "agent" interacts with an "environment": the more specialised an agent is to a particular environment, the worse its behaviour can be when the environment experiences some change.

The point is: *optimisation in practice* needs additional considerations to those of *model-based optimisation*. The more complex a model is, the more the practical success of optimisations based on it depend on the accuracy of its parameters.

In uncertain environments, there are "risk management" decisions concerning a compromise between increasing expected performance *vs.* decreasing

sensitivity to unpredictable changes. This applies to control design, production scheduling, portfolio management, *etc.*

The "best controller" would be the one that, gathering information from the environment, can adapt its strategy to changes in it. This is the basic idea behind *adaptive control* [22]. Although there are significant theoretical developments, its large-scale implementation in MIMO industrial processes is far from being common, as many underlying assumptions are difficult to verify. There are also wind-up problems with saturations and actuator and sensor faults.

In industrial applications, sometimes a controller is put in place and, after gathering information from its closed-loop behaviour, the model can be updated and a new controller redesigned. This approach is named *iterative identification and control*, and some basic ideas on it are outlined later on.

8.1.1 Sources of Uncertainty in Control

Achieving good plant performance in the face of uncertainty is one of the objectives of control. The sources of uncertainty are twofold.

Unmeasurable perturbations. They produce output deviations. With a controller in place, the achieved deviations must be below a user-defined bound.

Modelling errors. They can be also divided in three categories:

- parameter uncertainty, originated by uncertainty in physical parameters. An example is an interval-uncertain plant described by $k/(\tau s + 1)$, $k \in [0.7, 0.9]$, $\tau \in [1, 1.5]$. Parameter uncertainty may be accommodated with high-gain regulators in some cases,
- unmodelled dynamics (neglected delays and fast time constants): the order of real plants is infinite (partial differential equations) and *lumped parameter* models are only approximations. These approximations are intentionally made in many cases, to keep the problem tractable. For example:
  - multi-loop control (Section 5.2) disregards the non-diagonal terms in tuning of each loop. The cross-coupling is a modelling error that should be suitably withstood,
  - low-order models are often used to avoid modelling cost. For example, modelling a distillation column with experimental first-order + delay models, instead of a detailed one based on thermodynamics, *etc.*
- non-linearity: linear regulators must be at least robust to smooth non-linearities on the process. In fact, practical requirements often require suitable behaviour in the face of non-smooth non-linearities such as hysteresis or backlash (mainly with valves and other mechanical actuators).

Note that modelling errors can change with time, due to plant aging or component replacement: significant modelling errors need to be considered (in the future) irrespective of the initial effort devoted to modelling.

8.1.2 Objectives of Robust Control Methodologies

Given the significant practical importance of uncertainty, there is a need for studying the sensitivity of control systems to it, and ample research in that direction has been carried out in the last 30 years.

The objective of *robust* control is to be able to design controllers that achieve a desired level of performance and, furthermore, they can cope with a collection of uncertainty structures (with particular "sizes") specified by the designer. A usual approach is to try to maximise the tolerated uncertainty bounds. The detailed analysis of these issues is not under the scope of the book, and a full course can be devoted to this. However, the theory developed in the 1980s has significant practical implications.

The methodologies are based on a formal description of the control problem where the stability and performance objectives must be fulfilled for *any* plant, G, belonging to a family $\mathcal{G}$. The family of plants may be described by parameter variations, as bounds in frequency response, *etc.*

Two problems are the core of robust control techniques:

- **analysis.** Determining if a regulator, designed with any methodology, will withstand a known modelling error structure and size, in terms of:
  - *robust stability* (RS): excluding the possibility of an unstable closed loop for the available modelling error estimation,
  - *robust performance* (RP): asserting that the performance objectives will be fulfilled for *any* plant in the uncertain family,
- **synthesis.** Determining a regulator that maximises tolerable modelling error for a given performance level or, conversely, determining the maximum performance level for a given modelling error description.

After a first analysis of the problem from an "intuitive" point of view, some ideas on linear robust control will be briefly outlined from Section 8.5 onwards. Additional information can be found in Appendix F.

8.2 Uncertainty and Feedback

Feedback is needed to solve the problem of achieving design specifications in spite of modelling errors and perturbations: engineers calculate an "optimal" nominal set-point for the plant's actuators but then the variables deviate from their prescribed settings so sensors and feedback control are put in place. There is the need to guarantee that the controller will behave acceptably in an uncertain environment.

A very good controller would be one achieving significant reduction of the effect of disturbances on the plant's outputs, suitable filtering of sensor noise and fast tracking of set-point changes.

These criteria are subject to the requirement of "optimal" efficiency in terms of power consumption or minimal actuator activity. Furthermore, the

specifications must be achieved even if the actual plant undergoes significant changes with respect of the one initially modelled.

Of course, the combined fulfillment of *all* the previous criteria is difficult. Note that to achieve maximum efficiency and throughput, industrial plants should operate near their technological limits. However, to improve tolerance to modelling errors and big disturbances, and to increase the reliability of the overall process, system operation must be reasonably away from the technological limits: *uncertainty limits achievable performance.*

Note that modelling error has two faces (Example 1.4 on page 13, [118]):

- small modelling errors (negligible differences in open-loop responses) can lead to arbitrarily big differences in closed-loop performance,
- big modelling errors (quantified by the differences in open-loop response) can be mitigated by feedback.

So, there is a need to formalise which modelling errors *can* be mitigated by feedback and which ones *cannot*.

8.2.1 Model Validity Range

Models are usually obtained either from first-principle models or from black-box parameter adjustment procedures, or a combination of both. The "perfect" model does not exist, and it is only a "convenient" simulation of the plant for a given set of signals (those used in the model-building experiments) and goals. Hence, the validity range of a linear model is usually limited to input signals with the following characteristics:

- signals of *low amplitude* (as higher-amplitude ones excite non-linearities unaccounted for in a linearised model). Note that, however, too-low amplitude experiments may have a deficient signal-to-noise ratio! So, it is important to decide on the amplitude of input signals in identification experiments, to balance noise contamination and non-linearity excitation.
- signals of *low frequency*, for three reasons:
  - The model is often validated with step-response comparison,
  - even if experiments with higher frequencies have been carried out (such as empirical Bode diagram fitting), the signal-to-noise ratio of the experiment degrades as the plant's response *decreases* with increasing frequency. Sensor bandwidth issues come also into consideration,
  - to properly fit *all data* in an experiment, a high-order model might be needed but maybe that model is capturing spurious artifacts due to noise: increasing the number of parameters of a model does not always entail a better quality of its results. Simplified models are usually calculated for control design.

These limitations in model validity directly translate to limitations in controller performance, to ensure that closed-loop simulations agree reasonably with future experiments on the "real" plant, and that plant variations in time will be consistently tolerated.

8.2.2 High Gain Limitations

There is a fundamental difference between parametric uncertainty and uncertainty coming from unmodelled dynamics. Let us consider the feedback loop in Figure 1.3 on page 10, with Equations (1.3).

A preliminary analysis of its sensitivity to error in the SISO case was given by (1.5). Broadly speaking, many cases of parametric uncertainty can be dealt with by using *high-gain* controllers. Random disturbances can also be cancelled by high-gain feedback because the sensitivity (4.5) tends to zero as K gets larger. Some non-linearities can be cancelled, too, by high-gain feedback (for example, integral action, with high-gain at zero frequency, cancels *static* non-linearities).

However, there is a catch: unmodelled dynamics usually imply gain limitations (see Section F.1), and there is *always* some amount of this uncertainty in any real-world application. Furthermore, "waterbed" effects (page 225) assert that lowering S in a particular frequency range increases it in another, so decreasing sensitivity to errors in one frequency band implies increasing it in another one.

Example 8.1. A second-order system with interval coefficients:

$$G(s) = \frac{1}{s^2 + [-1, 4]s + [1, 10]}$$

can be stabilised using a high-gain PD regulator, $u = K_p e + K_d \dot{e}$, regardless of parameter uncertainty.

Indeed, the characteristic equation for the closed loop is:

$$0 = N_G N_K + D_G D_K = s^2 + ([-1, 4] + K_d)s + ([1, 10] + K_p)$$

where N and D refer to numerator and denominator. In second-order plants, positive coefficients imply stability, so any $K_p > -1$, $K_d > 1$ will stabilise the loop.

For example, with $K_d = 100$, $K_p = 2500$ the poles have a real part lower than -37 (settling time lower than 0.1 s) and an imaginary part with modulus lower than 7.5 (insignificant overshoot) for any parameter combination. This seems to be a good behaviour but neglected dynamics in the process and noise filtering will make the design unviable in practice. If the real system were:

$$G(s) = \frac{1}{(s^2 + s + 8)(0.011s + 1)^2}$$

the proposed regulator would destabilise the loop.

Fortunately, an approximate assessment of robustness to neglected dynamics can be carried out with the tools in Section 8.5.

8.3 Limitations in Achievable Performance due to Uncertainty

The objective of this section is to understand a set of basic engineering common-sense limitations to achieving a sensible design in the face of significant modelling errors.

At this moment, some insight can be acquired by drawing the Bode diagrams of all transfer functions in Example 1.4: G_1 and G_2 are identical at low frequencies but differ at higher frequency ranges; on the contrary G_1 and G_3 are very different at low frequencies but similar at high frequency.

The diagrams lead to the following conclusion, true in a general sense in many situations:

> *low-frequency uncertainty can be accommodated by intense enough feedback; however, high-frequency uncertainty may cause severe problems with high-gain controllers.*

8.3.1 Amplitude and Frequency of Actuator Commands

Limitations in control signal amplitude

Simulating the process against likely disturbances or set-point changes must produce control commands of reduced amplitude. Otherwise, non-linearities will be excited and the results will not match those from simulations[1].

Enforcing smooth set-point transitions. To avoid high-amplitude and abruptly-changing input signals, smooth set-point transitions are enforced in control of complex systems (limiting the operator commands with rate-saturation circuits or algorithms, and sequencing slow start-up and shut-down operations). Also, abrupt set-point transitions causes great power-demand changes that act as disturbances to the own process power sources and nearby equipment (voltage drops, *etc.*) and even to the utilities' power grids.

Of course, non-linear control design techniques are less influenced by the amplitude constraints (with non-linear models, of course). An outline of some possibilities is presented in Section 9.5.

Limitations in frequency

If the desired behaviour is too different to the open-loop step response, loop signals have fast, high-frequency components for which simulations are unreliable: those components can make the "real" plant exhibit a very different response to that of the models: modelling errors impose *designed bandwidth* limitations. For example, Equation (5.13) entailed a bandwidth limitation in multi-loop control.

[1] The most frequent and apparent non-linearity is actuator saturation, and the issue of saturation avoidance is key in control of unstable systems as the plant with saturated actuators behaves similarly to open-loop step response (*i.e.*, in an unstable way).

Limitations due to noise amplification

The presence of noise in the sensors when operating under realistic conditions also limits final closed-loop performance. Even with precise, high-order models, a reduced number of sensors needs slow observers (or filters) for acceptable filtering of measurement noise (reconstructing state vectors amounts to a kind of numerical derivation). So, as slow dynamics is introduced, fast closed loops cannot be achieved in that case. Not surprisingly, perhaps a simpler model can suffice to achieve that limited performance.

Example 8.2. With one sensor of "average" quality, no more than two states (y, $\dot{y}$) can be reconstructed in a "short" time with "reasonable" noise filtering. So, even if a precise high-order model is available, it will not be able to squeeze out "aggressive" performance (fast rejection of disturbances) on the real plant as the estimates of all but two states will be too noisy.

Hence, the real improvement to be obtained when compared to using a simplified second-order model and a PD regulator (even hand-tuned) may not be worth the modelling effort and the greater software complexity. There will be better guarantee of performance improvement on the real plant if a second sensor is added[2].

Waterbed effects. The "optimal" shape of a regulator would have a high gain at low frequencies (to counteract drifts, disturbances) and low gain at high frequencies (to avoid exciting neglected dynamics and amplifying measurement noise). However, an abrupt gain change is not possible with a linear finite-order controller: high gain changes are always accompanied by considerable phase delays [28]. This delay worsens robustness margins.

Developing this idea [134, 133, 119], the following result (intuitively reasonable) holds: control means making the loop insensitive to disturbances and modelling errors at certain frequencies (usually aiming for very small sensitivities in the low-frequency band) at the expense of making the closed-loop *more* sensitive at medium-high frequencies[3]. This is called the "*waterbed*" effect, pointed out already by Bode in the 1940s. The effect is more "intense" for unstable and non-minimum-phase plants. An example of the waterbed result was shown in Figure 7.4 on page 217 and its discussion: the disturbance effect was diminished by feedback up to a particular frequency and increased at higher frequencies.

8.3.2 Unstable and Non-minimum-phase Systems

In previous chapters, the observer + state feedback framework has been introduced jointly with other design techniques. At first glance, no special difficulty

[2] As discussed in Chapter 5, if a clever place for it is found, it might be possible to easily hand-tune an additional PD for it in a cascade or multi-loop arrangement.

[3] It can be shown [134] that, for any feedback controller so that GK has relative degree ≥ 2, $\int_0^\infty \ln|\det S(j\omega)|\, d\omega = \pi \sum Re(p_i)$ where p_i are the unstable poles. So, small sensitivity (negative $\ln|\det S|$) must be compensated for by positive logarithms (sensitivity greater than 1) at other frequency ranges.

seems to arise from the presence of particular characteristics such as unstability or non-minimum phase plants, as closed-loop poles can be moved to arbitrary positions, and input and output weights can be varied at will.

However, the apparent unimportance of this issue applies only to simulation: unstable and non-minimum-phase systems are difficult to control in practice with suitable robustness margins. There are some bounds for the specifications that can be squeezed out from these classes of systems to guarantee a minimal tolerance to modelling errors and satisfactory performance in practice. They can be derived analytically [119, 15] but only an outline with the ideas of practical interest will be presented, based on intuitive grounds.

Unstable systems. Unstable systems are difficult to control for many practical reasons:

- experiments on them are difficult to perform, hence quality of models is lower than in those obtained for stable processes,
- on one hand, the controller must be "aggressive" to be able to stabilise the process but in that case it is sensitive to modelling errors; on the other hand, a non-aggressive (low-gain) controller is unable to stabilise the nominal plant,
- actuator saturation implies the existence of an *unrecoverable space*. For example, the process $\dot{x} = x + u$, with actuator bound $|u| < 1$, cannot be driven to zero if $|x| > 1$. For a detailed view on this aspect, see [61]. Unstable processes, even if stabilised around an operating point by a controller, become unstable for high values of disturbances or set-point variations.

From analytical frequency-response considerations [15, 16, 119], the following rule of thumb is usually asserted: to control an unstable system with acceptable robustness margins the target bandwidth must be *at least double* to the unstable poles' break-down frequency. This implies, roughly, that the closed-loop settling time must be, *at most*, the time the process outputs takes to multiply by three to four initial deviations.

Non-minimum-phase systems. For internal stability, in SISO plants, the loop transfer function $T = (I + GK)^{-1}GK$ must have as zeros the RHP zeros of the plant. This also holds in MIMO systems (see Example 5.9 on page 138), *i.e.*, it is impossible to avoid an initial inverted response for all outputs of an NMP MIMO plant. However, MIMO plants have an advantage because in most cases the NMP response can be directed with relative freedom to the output of choice, as in the referred example.

The presence of RHP zeros in T bends upwards the frequency-response diagram (T controls the sensitivity to relative uncertainty (F.2)), so it worsens the robustness margins.

The rule of thumb in this case [16, 119] is to aim for a bandwidth *less* than *half* of the RHP zero break frequency.

Simultaneous RHP poles and zeros. Plants with both RHP poles, p_{rhp}, and zeros, z_{rhp}, are difficult to control as RHP zeros impose an upper bandwidth (speed of response) limit and stabilisation requires a lower bandwidth limit: if both limits are close, the robustness margins that can be achieved are very low. Unacceptable margins may be achieved if $z_{rhp} < 6p_{rhp}$. Similar difficulties arise with simultaneous RHP poles and delay.

Example 8.3. A system with an RHP zero in +4 should not be controlled requiring reasonable tracking of references quicker than 2 rad/s. On the other hand, a system with an RHP pole in +5 should be controlled aiming for a bandwidth greater than 10 rad/s. A real system with both RHP components will be substantially difficult to model and control (even if simulations work satisfactorily). A famous example is the "unridable bicycle" in [75], discussed in this context in [16].

As mentioned, the actually achieved robustness margins with the above mentioned "rules of thumb" for RHP poles and zeroes can be evaluated with the techniques discussed in Section 8.5. Example 8.9 on page 239 discusses the robustness of a pole-placement design for a process with an RHP pole close to an RHP zero.

8.4 Trade-offs and Design Guidelines

To summarise the previous section, to ensure that closed-loop simulations produce results similar to those to be encountered when the controller is put into operation on the real plant (robustness), a controller design must aim for:

- low amplitude of control increments,
- low bandwidth (limited high-frequency components in system inputs).

However, low-gain, low-bandwidth controllers cannot quickly drive the system to the desired operating point, neither can they compensate correctly for fast-changing disturbances. Then, an *increase* of robustness to non-linearity and undermodelling implies a *decrease* of the sought performance objectives and *vice-versa*. This is termed the performance-robustness trade-off.

8.4.1 Selection of Design Parameters in Controller Synthesis

Without resorting to robust design methodologies, some insight from the previous considerations can be applied to set up design rules in the methodologies in previous chapters to achieve reasonably robust results.

Pole-placement

In pole-placement techniques with *stable* plants, aiming for a closed-loop behaviour closer to that of the open loop usually results in better robustness margins[4].

Regarding observer design, increasing its speed of response entails increased sensitivity of the estimated state to measurement noise (high-frequency amplification). So, increasing observer settling time will lower high-frequency components in the control action (filtering out noise) and hence improve robustness, at the expense of delaying detection of the process-noise (lowering disturbance rejection performance).

The above discussion does *not* apply to unstable plants, as open-loop control for them is not a viable solution. Unstable plants need a minimum control energy to be stabilised, in line with the comments in Section 8.3.2. The *robust stabilisation* problem is defined as obtaining the stabilising regulator that maximises tolerance to modelling error.

Optimisation-based control

In LQR (discussed in Section 7.1), the performance-robustness tuning knob is usually the control-action weight, R: the higher R is, the smoother and lower bandwidth the resulting control actions are and, usually (stable plant), robustness margins do increase as well. Of course, activity of multiple actuators may be individually tuned by suitably modifying each diagonal term in R. Furthermore, modifying matrix Q may be used to relax/enhance a desired performance goal (for example, a particular output or state).

In observer (Kalman filter) design, in order to diminish the high-frequency gain of the resulting controller, decreasing "process noise" variance, V, yields a lower observer gain.

If the resulting regulator is simulated on the *nominal* model, the nominal output variance (model output cost) decreases as control-action weight does so, because control becomes more vigorous and disturbances are better cancelled. However, if tried in the *real* plant, increasing control energy (decreasing ρ from starting high values) first achieves the desired output variance reduction, but there is a *terminal* controller so that increasing nominal performance objectives beyond that point produces an actual reduction in real-plant achievement. The level of terminal performance achieved depends on the accuracy of the model used.

Norm-optimisation. There is a deep relationship between robust control design methodologies and the LFT optimisation problems discussed in Section 7.4, so later sections in this chapter will be devoted to the issue. In principle, penalising control actions yields more robust controllers.

[4] As an extreme case, aiming to place the controller/observer poles at the exact locations of those of the plant's open-loop dynamics gives $K = 0$, $L = 0$ as solutions (*i.e.*, an open-loop controller that will *never* destabilise an open-loop stable plant) for maximum stability robustness and minimal performance.

8.4.2 Iterative Identification and Control

The issues in the previous discussion point out the need for interleaved stages of identification and control to design regulators of increasing performance.

Indeed, on one hand, identification has a cost: setting up models, carrying out experiments (some of them even destructive on prototypes) to determine the model parameters, and qualified human-hours to work on model-building, simulation and validation.

On the other hand, feedback control reduces sensitivity to uncertainty at certain frequencies (but increases it usually at higher-frequency bands). For example, integral action provides steady-state error-free tracking with only knowledge of the *sign* of the DC gain (or, in a multivariable case, a rough estimate of the plant's inverse DC gain).

As feedback reduces sensitivity to errors in certain frequency ranges but increases them in others, the identification and modelling criteria depend on the performance objectives.

Iterative identification and control [5, 79] is a collection of techniques based on the above ideas, exploiting the interplay between modelling, identification and control: as more stringent controllers are put in place and the model's weaknesses start to surface, new identification experiments suitably tailored try to extract information on the relevant frequency ranges critical for robustness to modelling errors. These ideas can also help in building low-order models, as there is no need to fit certain model characteristics that will be made irrelevant by feedback [112].

Reaching *terminal* performance may be caused by two generic situations:

- fundamental limitations have been hit, in particular due to actuator saturation, sensor quality (precision, signal-to-noise ratio) or other non-linearities. To increase performance, additional sensors and actuators would be needed (or replacing the current ones with more precise, linear or powerful devices),
- the model is no longer useful. To increase performance, a new model must be obtained, preferably from the data gathered from past closed-loop experiments, stressing their limitations. If limitations from non-linearity are suspected, non-linear models and design techniques could be needed.

To summarise, in many practical control engineering problems, reaching terminal performance triggers changes in instrumentation configuration and/or plant models to further improve performance, in an iterative process interleaved with controller design and validation stages.

8.4.3 Generalised 2-DoF Control Structure

The previous discussions on performance-robustness trade-off assert that the "gain" on the feedback path must be limited under modelling errors. Hence, the disturbance-rejection performance is limited by uncertainty. Note also that

high gain on the feedback path exacerbates the effect of measurement noise on actuators and plant output, so it is another gain-limiting factor. The issue now is trying to achieve the best possible performance in both set-point tracking and disturbance rejection tasks.

Let us analyse a general control loop depicted in Figure 8.1. A set of controlled outputs, z, is to be kept at a set of desired values, r, using manipulated variables u and measurements y (in many cases coincident with some or all variables in z). Let us first assume linearity in all equations and that the loop is governed by:

$$\begin{pmatrix} z \\ y \end{pmatrix} = \begin{pmatrix} G_z & G_{dz} \\ G & G_{dy} \end{pmatrix} \begin{pmatrix} u \\ d \end{pmatrix}; \qquad u_{fb} = -Ky_{fb}; \quad \begin{pmatrix} y_{ff} \\ u_{ff} \end{pmatrix} = \begin{pmatrix} F_y \\ F_u \end{pmatrix} r$$

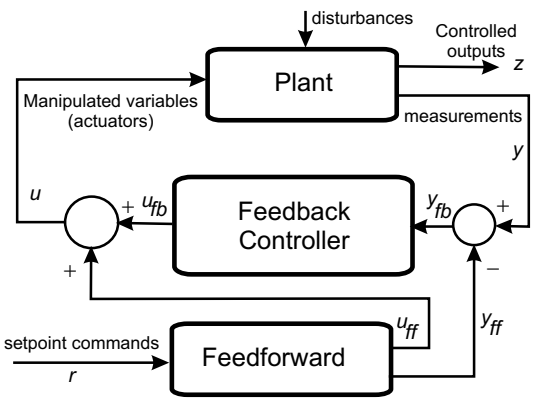

Figure 8.1. General 2-DoF controller configuration

In this case, the achieved output is:

$$z = G_z(I + KG)^{-1}\left(K(F_y r - G_{dy}d) + F_u r\right) + G_{dz}d \tag{8.1}$$

If measurements and controlled outputs ($y \equiv z$) coincide,

$$y = (I + GK)^{-1}\left(G_{dy}d + (GF_u + GKF_y)r\right) \tag{8.2}$$

Let us analyse some common cases, starting from the simplest ones, denoting $S = (I + GK)^{-1}$.

Case 1. $y \equiv z$ and modelling error is reasonably small. In this case, a high-gain regulator K can be designed, and feedforward can be trivial $F_u = 0$, $F_y = I$. This situation is called *error-based* controller (Section 4.5), as control actions are taken based on the deviation from a desired set-point ($y_{ff} = r$). Then, (4.3) is obtained (n and d_u have not been included for simplicity):

$$y = (I + GK)^{-1}\left(G_{dy}d + GKr\right)$$

If the gain is high enough[5] over a wide-enough frequency band, then the deviations from the nominal operating point, $y - r$, will be small (as $(I + GK)^{-1}G_{dy}$ will be small in a reasonable range of frequencies and $(I + GK)^{-1}GK \approx I$). So, the resulting regulator will satisfy to an acceptable level both tracking and disturbance rejection specifications. This configuration is the usual one in SISO PID controller settings, and also in MIMO decentralised and cascade configurations.

In many cases, an inaccurate model hinders gain increase, so tracking performance may be unacceptable, but disturbance rejection may be satisfactory (if disturbances are small or concentrated in a frequency range where robustness constraints are not stringent; otherwise, a better model is needed).

Case 2. In the previous case, to improve steady-state tracking with less requirements for integral action, steady-state calculation of U_{ff} can be carried out. In a linear CT case, setting up $F_u = G^{-1}(0)$, $F_y = I$, yields:

$$y = S\left(G_{dy}d + (G(s)G^{-1}(0) + GK)r\right)$$

so, in steady-state, with a constant disturbance d_{ss}:

$$y = (I + G(0)K(0))^{-1}\left(G_{dy}(0)d_{ss} + (I + G(0)K(0))r\right) = r + SG_D(0)d_{ss}$$

Hence, integral action is not needed for tracking tasks. Some industrial PIDs have this option or a more limited version that provides a user-defined constant u_{ff} (see later in this section).

Case 3: 2-DoF set-up. If feedback gain needs to be limited, and $z \equiv y$, and accurate tracking is desired, the procedure is the following:

1. Design a feedback controller optimising disturbance rejection performance, with reasonable robustness.
2. State a desired reference model for set-point changes, $M(s)$. Use $F_u = G^{-1}M$, $F_y = M$, as in open-loop tracking (see Section 4.6.2).

Then, output will be given by:

$$\begin{aligned} y = S\left(G_{dy}d + (GG^{-1}M + GKM)r\right) \\ = S(G_{dy}d + (I + GK)Mr) = SG_{DY}d + Mr \end{aligned} \quad (8.3)$$

so feedforward achieves the reference model, M, irrespective of the actual value of K (if model G is accurate enough). The input to the controller K, under exact modelling assumptions, can be shown to be:

$$y_{fb} = y - Mr = SG_{dy}d$$

so feedback only acts on disturbance-rejection tasks. Of course, any modelling error makes the above equations inaccurate, so the feedback regulator K acts to compensate for *both* disturbances and modelling errors.

This set-up has two key advantages:

[5] For example, infinite at DC frequency by adding integrators.

- disturbance-rejection and tracking design can be done *independently*,
- the system inversion in F_u (determining the control actions that would yield the desired response if model is approximately correct) can actually be carried out on a more complex (even non-linear) model if available (in some cases), thus diminishing the strength of feedback in significantly non-linear loops (for tracking tasks).

Case 4: Generalised-plant framework. The generalised-plant framework allows norm-optimisation of the (disturbance, set-points) to (controlled outputs) transfer function matrix via designing a controller based on a measurement vector including any known set-point or disturbance. With suitable weights defining the specifications, optimisation-based 2-DoF design for $z \neq y$ can be carried out, so optimal disturbance-rejection and tracking design are carried out *simultaneously*. This set-up is more general than the ones considered before. The problem under consideration would be cast as:

$$\begin{pmatrix} e \\ \begin{pmatrix} r \\ d_m \\ y \end{pmatrix} \end{pmatrix} = \begin{pmatrix} G_{dz} & -I & G_z \\ 0 & I & 0 \\ H & 0 & 0 \\ G_{dy} & 0 & G \end{pmatrix} \begin{pmatrix} \begin{pmatrix} d \\ r \end{pmatrix} \\ u \end{pmatrix} \tag{8.4}$$

where $e = z - r$ is the loop error to be minimised, in $d_m = Hd$, H is the identity or a submatrix of it, denoting which perturbations in the full vector d are actually measurable in d_m, available as controller inputs. If the perturbation sensor had significant dynamics, matrix H should change accordingly to incorporate it. Inputs to the controller are (r, d_m, y) and the resulting 2-DoF control action vector u will be the last of the inputs to the generalised plant. The techniques in Section 7.4 can be directly applied.

2-DoF in Industrial Regulators

The above schemes, as such, are not usually available in industrial regulators, and must be implemented in a customised code for a MIMO case. However, the basic ideas are key in achieving a suitable balance in the performance-robustness trade-off and some simpler implementations are actually widely used: The importance of 2-DoF designs in industrial practice must be stressed. In particular, the implementation of 2-DoF in PIDs will be briefly discussed as they are the building blocks of many multivariable decentralised control systems in practice. Let us discuss two alternatives:

- **Feedforward input set-point** (constant or linear with r):

$$u = K_p e - K_d \frac{dy}{dt} + K_i \int e\, dt + (K_{f1} + K_{f2} r) \tag{8.5}$$

  The linear transformation of the set-point r can be used to amplify or attenuate the steady-state step to compensate position errors.

- **Set-point pre-filtering**: replacing r above by a modified set-point, r^*, result of some kind of (dynamic) pre-processing of the true set-point r.

Regarding the second option, the most usual pre-filtering is *rate saturation*. For instance, a discrete-time implementation can be:

$$r_k^* = r_{k-1}^* + \text{sat}_\beta(r_k - r_{k-1}^*) \tag{8.6}$$

where $\text{sat}_\beta(\cdot)$ stands for identity inside $[-\beta, +\beta]$ and saturation outside. This is useful in two situations:

- to enforce smooth set-point transitions in processes with significant non-linearity, for robustness,
- to diminish overshoot in step set-point changes (without resorting to decrease the feedback gains from those decided for disturbance rejection).

More complex pre-filterings can be put in place, such as ($\alpha < 1$):

$$r_k^* = r_{k-1}^* + \alpha(r_k - r_{k-1}^*) + (1-\alpha)\text{sat}_\beta(r_k - r_{k-1}^*) \tag{8.7}$$

where a step change is initially partially applied and the full step is later reached in a rate-saturated ramp. The use of this set-up is the same as (8.6).

If the speed of the tracking response must be accelerated without changing the disturbance rejection characteristics, the above equation can be implemented with $\alpha > 1$. In that way, the step change is initially amplified and in the nex samples it is gradually reduced to the final value.

A similar effect of initial attenuation/amplification can be achieved using a dynamic pre-filter with $\alpha < \beta$ or $\alpha > \beta$ respectively, such as:

$$r^*(s) = K_e \frac{\alpha s + 1}{\beta s + 1} r(s) \tag{8.8}$$

8.5 Robustness Analysis Methodologies

After devoting some time to discussing modelling errors and control limitations in an "intuitive" way, the basics of modern robust control techniques will be introduced. These techniques use not only a model but also a "model error" bound. Apart from the issues to be now discussed, ill-conditioned plants are difficult to control, as they are sensitive to model and actuator uncertainty, as discussed in Section B.4.1, and also on page 324. RGA evaluation is also related to that issue (see Remark 5.6).

Let us discuss in this section the analysis of an already designed loop. The next section will discuss the controller synthesis.

8.5.1 Sources and Types of Uncertainty

The concept of "modelling error" or "uncertainty" needs to be refined before a formal treatment of the sensitivity to it can be defined. Uncertain subsystems are often drawn in block-diagrams as blocks with a dynamic system Δ, representing linear and non-linear mismodelling and time-variations.

Uncertain Models

Additive uncertainty. Unstructured additive uncertainty, Δ, is defined as any dynamic system (linear, non-linear, distributed, time-variant) adding its effect to that of the nominal model G:

$$G_{true} = G + \Delta \tag{8.9}$$

where G_{true} represents the elusive *true* plant, and only a measure of the "size" (norm) of Δ is assumed known. Figure 8.2 (left) depicts the idea in block-diagram form.

Output-multiplicative uncertainty. Another usual representation of uncertainty is the relative error:

$$G_{true} = G(I + \Delta) \tag{8.10}$$

The expression assumes modelling error at some output subsystems. Figure 8.2 (right) depicts output-multiplicative uncertainty in block-diagram form.

LTI uncertainty. Unstructured LTI additive uncertainty represent mismodelling due to neglected *linear* dynamics (the true plant $G+\Delta$ is assumed LTI). Only in LTI uncertainty, Δ can be considered a transfer function matrix.

Remark 8.4. Uncertainty occurring in real systems belongs to the non-linear, time-variant description as usual models ignore plant non-linearities and plant variations with time. However, the results assuming this uncertainty may be conservative in the sense that linear time-invariant uncertainty is able to explain linear undermodelling (for example, due to the use of finite-order lumped-parameter equations to approximate infinite-dimensional partial differential equations). Furthermore, if only small deviations from the set-point are expected, non-linearities do not become very significant, and also plant variations are assumed to be very slow compared to the closed-loop dynamics. This is the reason because of which, in many cases, a less cautious analysis assuming LTI uncertainty is carried out.

Structured uncertainty. Structured uncertainty is used to denote a description of uncertainty with a more refined detail. The usual form of describing this structured uncertainty is with:

- *parametric* uncertainty: uncertainty in the values of the coefficients of the matrices in the state space description or the coefficients in the transfer function matrix,

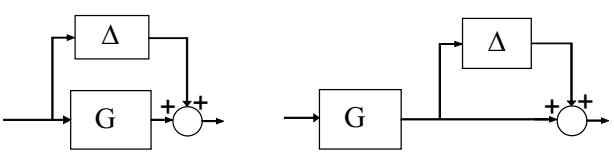

Figure 8.2. Unstructured uncertainty: additive (left), output-multiplicative (right)

- knowledge of the internal structure of the system. The most important cases are:
  - *diagonal* actuator and sensor uncertainty: a reasonable engineering assumption is (usually) the non-existence of couplings between different actuators, and a similar consideration for sensors. So, actuators have a diagonal transfer function matrix with a diagonal uncertainty description. Diagonal input uncertainty is particularly significant in valve-actuated plants,
  - known interconnections and subsystems: for example, existence of internal cascade-control loops, amplifiers, *etc.*, each of these contributing to the overall modelling error in a particular, specialised way.

One advantage of frequency-domain uncertainty descriptions over the parametric approach to robust control [25] is that one can choose to work with a simple model and represent neglected dynamics as an unstructured modelling error. However, in a general case, the pure-parametric case may obtain less conservative solutions (better performance for the same modelling error tolerance) if this is the main source of uncertainty. However pure parametric approaches may obtain unsuitable high-gain regulators, as in Example 8.1.

Real life has a mixture of parameter uncertainty and neglected dynamics. Some structured-uncertainty descriptions are able to handle approximations to this case (see Appendix F.2 for an outline of basic ideas).

8.5.2 Determination of Uncertainty Bounds

The applicability of robust analysis and design techniques depends on the availability of a modelling error bound for a particular control problem. Two approaches can be pursued, depending on the nature of the modelling process.

Black-box models. There exist identification methods that provide an estimated model and an estimated error bound. For details on process identification, the reader is referred to [84], and for details on model error bound identification to [113, 36, 56, 77], and also the literature on iterative identification and control [5].

Grey-box and physical modelling. In this case, the model can be represented as an uncertain physical-parameter one. Random variations of these parameters provide plants whose difference with the nominal one may be bounded by a particular weight. Of course, random variations may skip the "worst case" model. To obtain the worst case model, interval arithmetic can be used [65], or interval-uncertainty design methods can be directly applied [25].

Example 8.5. In multi-loop control, neglecting off-diagonal elements can be considered as a modelling error. Indeed, that was discussed in (5.11) in page 136. The sigma-plot of E will determine a modelling error bound, to apply small-gain inequalities to be discussed next. However, as off-diagonal uncertainty is, in a certain sense, *structured*, other conditions can be derived, such as (5.13).

A further example, dealing with an uncertain-parameter model, is discussed in Example 8.8 on page 238.

8.5.3 Unstructured Robust Stability Analysis

In Appendix F, analysis of the allowed "size" of the *unstructured* modelling error tolerated by a particular closed-loop controller is carried out: if the actual model error is known to be below a bound to be calculated, the system is said to be *robustly stable*.

The bounds on modelling error are calculated as a function of the nominal plant, G, and the controller in place, K, applying the small-gain theorem. See the referred appendix for details. The main results are summarised below.

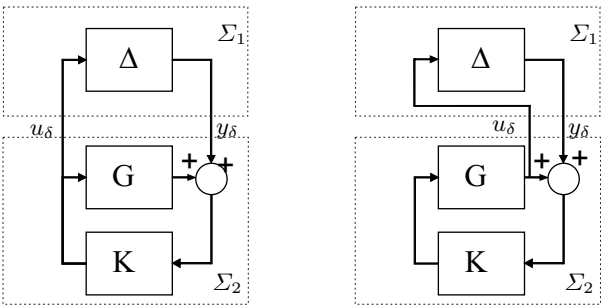

Figure 8.3. Feedback loops with additive and output-multiplicative uncertainty, with small-gain interpretation (cf. Figure C.1).

Additive uncertainty. The nominally stable closed loop (positive feedback, by convention) in Figure 8.3 (left), with plant G and controller K, will be stable for a modified plant $G + \Delta$ if the modelling error size, $\|\Delta\|$, satisfies:

$$\|\Delta\|\|K(I - GK)^{-1}\| < 1 \tag{8.11}$$

So, robust stability is ensured if nominal stability holds and the above modelling error bound is satisfied. The condition of nominal stability is obvious, as control design must provide a stable loop at least for the model at hand. In the sequel, it will be implicitly assumed.

The previous condition is conservative in the sense that there are uncertainty sources that do not destabilise the feedback loop with a particular plant even if its norm is arbitrarily greater than $1/\|K(I - GK)^{-1}\|$. However, it can be shown [133] that there exists a particular system with $\|\Delta\| = 1/\|K(I - GK)^{-1}\|$ so the closed-loop is unstable, so the condition is necessary and sufficient if *all* possible unstructured uncertainties below a particular size must be withstood as a design requirement.

As Δ is, by definition, unknown, the usual approach in robust stability analysis is to plot the Bode diagram (singular value plot in MIMO systems) of:

$$\bar{\sigma}\left(K(j\omega)(I - G(j\omega)K(j\omega))^{-1}\right) = \bar{\sigma}\left(KS\right) \tag{8.12}$$

so that its peak value (∞-norm) is the *inverse* size limit for any neglected dynamics, non-linearity or time-variation (Theorem C.6), and its frequency-by-frequency value is the bound to be satisfied by any neglected linear time-invariant dynamics, following (C.19).

In this way, alternative designs can be compared in terms of nominal performance and robust stability under modelling errors.

Remark 8.6. Note that from Example 7.12, the ρKS term in the transfer matrix whose norm had to be minimised, (7.30), translates directly on enforcing a greater tolerance to unstructured modelling errors as ρ is increased, at the expense of a decreased capability of disturbance rejection (emphasising KS in the cost index results in a bigger value for W_eS, *i.e.*, worse nominal performance). This is one of the reasons the mixed-sensitivity approach is so popular: it directly presents the performance-robustness trade-off as a "tuning knob" in the design methodology. In the next section, a more detailed discussion and a refined procedure will be presented.

Multiplicative uncertainty. Stability under relative error uncertainty (Figure 8.3 (right)), described by $G_{true} = G(I + \Delta_*)$, is guaranteed if Δ_* satisfies:

$$\|\Delta_*\|\|GK(I + GK)^{-1}\| < 1 \tag{8.13}$$

As usual, as Δ_* is unknown, sensitivity to relative error is analysed by plotting:

$$\bar{\sigma}\left(G(j\omega)K(j\omega)(I - G(j\omega)K(j\omega))^{-1}\right) = \bar{\sigma}\left(T\right) \tag{8.14}$$

i.e., the complementary sensitivity function, (4.6), is the inverse of the modelling error limit size. The expression can be evaluated on a frequency-by-frequency basis for neglected LTI dynamics.

From an engineering aspect, relative error descriptions are "reasonable" for low frequencies and smaller additive ones are suited for broadband uncertainty (see Example 8.9 on page 239).

Remark 8.7. The uncertain block Δ needs to be *stable*. This poses fundamental problems for robust control of uncertain unstable processes: note that for a nominal plant $G = 1/(s-1)$, and a real one $G_{true} = 1/(s - 1.001)$, $\Delta = G_{true} - G$ is an unstable function so the discussion above does not allow for analyzing such a simple case (see Example F.2 on page 324). There is the need of more sophisticated uncertainty descriptions in unstable plants or for plants whose uncertain models should allow for poles and zeros crossing the stability boundaries. The required refinements are discussed before the referred example.

Nominal Performance and Robust Stability Analysis

Whatever the controller design method used, the achieved performance and robustness margins may be analysed with the techniques just discussed:

- to analyse the disturbance-rejection performance of a feedback control system (that is what closed-loop is mainly for, anyway) a singular value plot of the sensitivity, (4.5), must be lower than a target diagram,
- to analyse the robustness to unstructured relative error at the output, a sigma-plot of T (of its inverse) should be drawn,
- to analyse the robustness to additive error, the relevant plot is that of KS,
- to analyse the sensitivity to modelling errors at other locations the block-diagram procedures in Appendix F.1 can be easily adapted, as long as there is only *one* uncertainty block.

Example 8.8. A SISO system can be described as:

$$G(s) = \frac{Ke^{-ds}}{(\tau s+1)(\tau_2 s+1)} \quad \begin{array}{l} K \in [1.9, 2.1] \quad d \in [0, 0.2] \\ \tau \in [0.9, 1.1] \quad \tau_2 \in [0, 0.05] \end{array}$$

with the "nominal plant" being $2/(s+1)$.

Plotting the sigma plot of several random plants (using third-order Padé for the delays), the relative modelling error as a function of frequency can be bounded by $W(s) = 0.06(5s+1)/(0.1s+1)$, drawn in a thick line in Figure 8.4 below.

```
sigma((pade(grandom,3)-gn)/gn,logspace(-1,3))
```

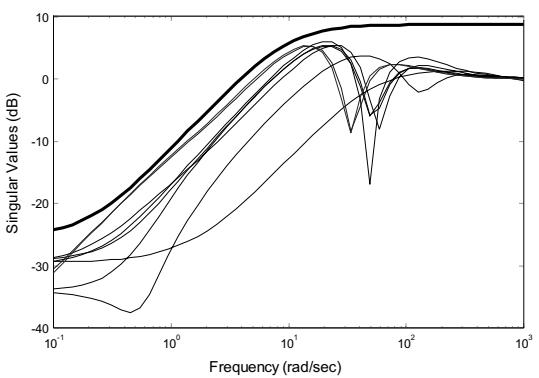

Figure 8.4. Bounding parametric and neglected dynamics errors

So, if a controller is designed, it will be later shown that (F.2) can be applied to ensure that robust stability for these plants will be achieved if $|KG/(1+GK) * W| < 1$, where G is the nominal plant. For instance, using a proportional controller,

$K = 2.75$ is the maximum tolerable gain to stabilise all possible plants, as the graph resulting from `sigma(k*gn/(1+gn*k)*w)` confirms[6].

Example 8.9. Recalling Section 8.3.2, the reader is left to check that, using the techniques above, when designing a second-order pole-placement controller + observer for the process $G(s) = \frac{s-4}{(s-4.5)(s+10)}$ so that poles are placed in -4:

```
k=place(gs.a,gs.b,[-4 -4.1]); l=place(gs.a',gs.c',[-4 -4.1]);
rg=reg(gs,k,l'); ks=feedback(-rg,gs);
T=feedback(series(rg,gs),-1); sens=1-T;
```

it will be guaranteed to be successful if the relative modelling error (Bode diagram of `T`) is, at low frequencies, less than 0.9%. This is extremely difficult to achieve in practice so the design will not stabilise the real plant and, even if it did, it would *heavily* amplify disturbances, if any (Bode diagram of the sensitivity, `sens`). Plotting the Bode diagram of `ks` will show the sensitivity to additive uncertainty. These diagrams show that process with close RHP poles and zeros are difficult to control, as discussed in Section 8.3.2: in this case, process redesign might be the wisest decision.

Note that the regions where the peaks of T and KS do occur are the ones in which the modelling effort should be concentrated. In this way, larger errors can be tolerated at other frequencies (see Section 8.4.2).

8.5.4 Structured Uncertainty

As previously commented, structured uncertainty arises when more than one uncertain block is present in the system or some of them are not full-complex uncertain blocks[7]. The robustness analysis with structured uncertainty is based on "pulling up" all uncertainty sources on a control system so a block-diagram such as the one in Figure 8.5 can be drawn and Δ can be considered to be block-diagonal.

If M and Δ_i are linear operators, the characteristic closed-loop equation is:

$$\det(I - M\Delta) = 0$$

Usually, the size of $\|\Delta\|$ is normalised to 1 including, if necessary, the known scaling matrices in M.

A conservative approach would be to consider all uncertainties being a particular case of a full Δ, and apply unstructured analysis (so stability would be guaranteed if $\overline{\sigma}(M) < 1$). However, less conservative techniques have been developed [133, 43, 97]. These techniques are based on the so-called *structured* singular value, usually denoted by μ.

[6] Note that the uncertainty has been assumed linear time-invariant so (F.2) can be applied on a frequency-by-frequency basis (see (C.19) on page 302).

[7] *i.e.*, apart from knowing its maximum size, some characteristics about its phase lag are known. The most common case is a *real* uncertain parameter: phase lag is equal to zero.

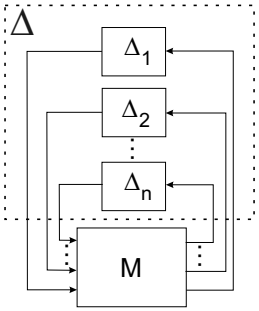

Figure 8.5. Structured uncertainty

The basic idea of the approach is trying to find the maximum scaling factor, α, so that there exists no matrix so that $\det(I - \alpha M\Delta) = 0$; its inverse is defined as the *structured singular value*, μ_Δ. If it is greater than 1, robust stability is ensured. If Δ is not a "full" matrix, perhaps the scaling α may be made substantially bigger than in the unstructured case, where α_{max} can be shown to be $\bar{\sigma}(M)^{-1}$. In Appendix F.2, suitable definitions and one elementary example on the use of μ are given.

Robust performance. Note that, up to now, only nominal and robust stability have been discussed. One important application of the structured stability analysis setup is robust performance analysis, *i.e.*, how modelling errors will affect *performance* objectives, this one being the actual problem the engineer wishes to solve: even if maintaining stability, the resulting loop may be unacceptable in terms of tracking of disturbance-rejection behaviour.

The structured-uncertainty framework enables some robust performance analysis problems to be cast as robust stability ones, in particular those based in norm-optimisation discussed in Section 7.4. Section F.2.1 briefly outlines the procedure.

Other methodologies. Structured uncertainty can be studied under other frameworks. In particular, interval-uncertain systems and the implications in control design are analysed in [25, 65]. Basic techniques are based in the Kharitonov theorem which asserts that robust stability of a full family of interval-uncertain polynomials can be determined by checking stability of a *finite* number of them (see the above references for details).

8.6 Controller Synthesis

As mentioned in Section 8.1.2, one of the main objectives of robust synthesis is obtaining the "best" regulators according to some optimality criteria regarding maximising performance (given an uncertainty structure and size), or maximising tolerance to modelling errors (given a target performance level).

There are various synthesis methodologies, of different degrees of complexity. Detailed descriptions of them are out of the intended scope of this book. However, an outline and usage guidelines for some of them will be given.

In the remaining of this section, the mixed-sensitivity optimisation introduced in Section 7.4 will be adapted for robust control synthesis under unstructured additive and multiplicative uncertainty bounds. Due to the requirement of stability of Δ (Remark 8.7), the techniques have limited application in directly guaranteeing robustness of unstable systems. Other widely used techniques, based on coprime-factor uncertainty, without this restriction, are briefly outlined in Appendix F.3.

The synthesis methodologies for the general case of RS and RP constraints with structured uncertainty is termed μ-synthesis, with the aim of minimising the structured singular value, μ_Δ, of a particular closed-loop transfer function. The problem is not yet fully solved and approximate methodologies are used, with very high-order final controllers. The interested reader can consult details in [119, 133].

8.6.1 Mixed Sensitivity

In previous sections and Appendix F, it has been shown that achieving RS amounts to having a reduced gain (norm) of a particular closed-loop transfer function. Note also that in Section 7.4, optimisation-based controllers were also cast in a norm-minimisation framework to solve an optimal *nominal performance* problem. Thus, both problems are quite related and, in many cases, $\mathcal{H}_\infty$ norm-optimisation can be used to pose robust control problems with engineering significance towards the objective of designing the best performing controller for a particular uncertainty bound.

Indeed, at this point, we are in conditions of a deeper analysis of Example 7.12 on page 211 and Figure 7.3. The conclusion of the example was that control problems with practical significance can be formulated as the minimisation of the norm of a transfer function matrix such as:

$$\| \begin{pmatrix} \rho KS \\ W_e S \end{pmatrix} \| \tag{8.15}$$

where ρ is a weight on the control action size needed to carry out disturbance rejection, and W_e is a frequency weight on the loop error.

As commented in Remark 8.6, on one hand, large values for W_e will drive the optimiser towards minimising the size of the error (at the expense of higher control activity, of course). On the other hand, KS is exactly the expression derived in (8.11) for RS to additive model uncertainty. Hence, a large value of ρ will force the optimiser to minimise the size of KS, achieving a better robustness margin, formalising the intuition, already discussed, asserting that less performing regulators must be designed to improve robustness. The key conclusion is that *weight selection* in (8.15) enables the designer to explicitly select its desired position in the performance-robustness tradeoff.

For stable plants, maximum robustness ($W_e \to 0$) will yield $K = 0$ and, on the contrary, maximum disturbance rejection performance ($\rho \to 0$) would yield an ill-posed infinite-gain controller tolerating only extremely small amounts of modelling error.

Note, however, that we are trading off robust stability against *nominal* performance. Robust performance (how the index to be optimised degrades with modelling error) is not being considered for the moment. The problem of robust performance will imply considering the allowed modelling error so that the worst-case disturbance gain is below an acceptable (finite) value for *any* possible plant, and not only considering S with the nominal model.

Although RP is not guaranteed by this design, RS + NP usually yields "acceptable" controllers from an engineering point of view in many plants (in particular, all SISO ones and well-conditioned MIMO ones as well). Further discussion on robust performance may be found in Section F.2.1.

The usefulness in practice of the scheme proposed in Figure 7.3 admits an easy generalisation to also include robustness to multiplicative error by using (8.13). Figure 8.6 presents the *augmented* generalised plant.

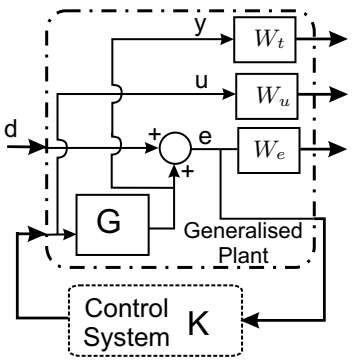

Figure 8.6. Mixed sensitivity problem setup

It can be easily shown with block-diagram operations that the transfer function matrix from d to the three fictitious outputs in Figure 8.6 is:

$$\begin{pmatrix} W_t T \\ W_u K S \\ W_e S \end{pmatrix} = \begin{pmatrix} W_t GK(I-GK)^{-1} \\ W_u K(I-GK)^{-1} \\ W_e (I-GK)^{-1} \end{pmatrix} \tag{8.16}$$

so that increasing weight W_t increases robustness to multiplicative error, increasing W_u increases robustness to additive error and decreases control activity, and increasing W_e reduces the loop error. These three weights are the tuning knobs of the control engineer to balance disturbance rejection and tolerance to low-frequency and high-frequency modelling errors.

Weight selection. A widely used, convenient weight selection methodology involves setting the weights to the *inverse* of a desired limit for a particular transfer function (matrix):

- **Performance.** If it is desired $\|S\| < \|S_{\max}\|$, then $\|S_{\max}^{-1}S\| < 1$. $S_{\max}$ is thus a performance limit, and considering $W_e = S_{\max}^{-1}$, performance goals are satisfied if $\|W_e S\| < 1$. As control implies reducing sensitivity in one frequency range to increase them in others (waterbed effect), $\|S_{\max}\|$ must be above 1 at high frequencies. A usual choice is a high-pass filter:

$$S_{\max} = \frac{\gamma(\alpha s + 1)^n}{(\beta s + 1)^n} \qquad \gamma\frac{\alpha^n}{\beta^n} > 1, \quad \alpha > \beta, \quad \gamma \in [0.01, 0.1]$$

- **Robustness.** Modelling error bounds determine the maximum sizes for T and KS (denoted as $T_{\max}$ and $(KS)_{\max}$ respectively) and the equations to be satisfied are $\|T_{\max}^{-1}T\| < 1$ and $\|(KS)_{\max}^{-1}KS\| < 1$ for each of the following types of uncertainty:
  - *relative error.* User must specify $T_{\max}$ based on relative-error bounds. As $S + T = 1$, to achieve small sensitivities, T must be around 1. As T is the usual "reference-to-output" transfer function, generalising the SISO concepts on the closed-loop resonance peak, bounds in T with engineering sense have a peak in the range 6–10 dB. Also, from SISO insights, T should have a reasonable roll-off at high frequencies to avoid measurement noise amplification and increase robustness. A common choice for usual plants is a bound in the form of a low-pass filter:

$$T_{\max} = \frac{\alpha}{(\frac{1}{\omega_t}s + 1)^n} \qquad \alpha \in [1.5, 7]$$

    or a diagonal matrix with these bounds in the MIMO case.ω_t is a guess of the frequency beyond which relative modelling errors increase above 100%, or the approximate upper frequency up to which the model is thought to be roughly correct,
  - *additive error.* To set up $(KS)_{\max}$, a constant diagonal matrix may be reasonable, expressing the maximum high-frequency gain of the resulting controller, as S should tend to I for high frequencies.

Once the limits for S, KS and T have been set up, the corresponding weights are the *inverses* of $S_{\max}$, $T_{\max}$ and $(KS)_{\max}$, and the norm-minimisation routines are called. If a solution is found so that the norm of the overall transfer function is less than 1, as the maximum gain of a matrix is greater than or equal to that of any of its submatrices, all closed-loop functions are below the desired bounds. So, bounds on performance and robustness can, in the above context, be cast as obtaining a regulator K so that $\|\mathcal{F}_l(P, K)\| < 1$. If no solution with a norm less than 1 is found, one (or several) of the design bounds have been exceeded: the design must be carried out again with more relaxed performance or robustness specifications.

8.7 Conclusions and Key Issues

Tolerance to modelling error (robustness) is mandatory in industrial control implementations, as large errors are present in many applications.

Some intuitive design guidelines can be thought of, mainly limiting performance objectives to limit input usage, in amplitude and frequency.

The importance of the issues led to significant research effort in the 1970s and 1980s. Robustness analysis can be carried out for an already-designed regulator (singular value plots) or some $\mathcal{H}_\infty$ optimality problems can be posed to maximise the allowed modelling error. The theory is complex but software exists for advanced applications.

8.8 Case Studies

8.8.1 Cascade Control

In this example, a two-sensor/one-actuator cascade-control configuration will be compared to an equivalent SISO one in terms of tolerance to modelling error in the plant being controlled.

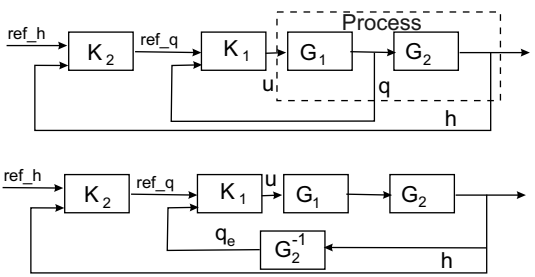

Figure 8.7. Cascade control (top) *vs.* equivalent SISO control (bottom)

In the top diagram in Figure 8.7, a cascade-control structure with two sensors is used. G_1 is controlled via a fast slave loop and output of G_2 is the main variable to control. In the bottom diagram, the missing sensor is estimated via inversion of the main plant, G_2. Straightforward block-diagram operations show that the transfer function from the main set-point to output h is identical in both cases so there would be no difference in nominal tracking performance.

To check for robustness to mismodelling, it is left to the reader as an exercise to check that if an unstructured additive-uncertainty block is added in parallel to G_2 the transfer functions needed to apply the small-gain theorem for robust stability (note that set-point inputs are not needed for RS analysis), with negative feedback, are:

$$W_2 = \frac{G_1K_1K_2}{1 + G_1K_1 + G_2G_1K_1K_2}; \quad W_1 = \frac{G_1K_1(K_2 + G_2^{-1})}{1 + G_1K_1 + G_2G_1K_1K_2} \tag{8.17}$$

for the two-sensor and one-sensor set-ups respectively, where K_1 and K_2 are usual error-based (1-DoF) regulators. For a particular simple case:

$$G_1 = \frac{9}{s+1}; \quad K_1 = 0.25; \quad G_2 = \frac{4}{(10s+1)(0.4s+1)}; \quad K_2 = 5 + 0.6s + \frac{0.6}{s}$$

where the P and PID regulators have been hand-tuned for a target settling time of 3 s and 15% overshoot, the resulting amplitude Bode diagrams for W_1 and W_2 are plotted in Figure 8.8.

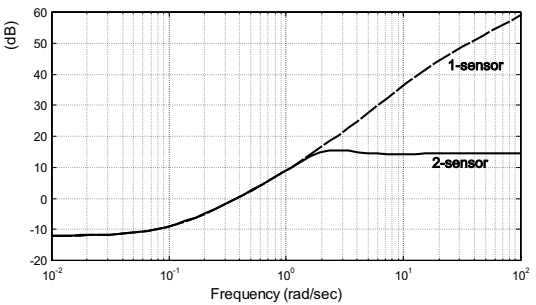

Figure 8.8. Inverse model error bounds

The diagrams show that the one-sensor set-up is more sensitive to additive uncertainty than the two-sensor one, as intuitively expected[8].

8.8.2 Distillation Column

Let us analyse the sensitivity to additive modelling error of the distillation control designs in the case study in the previous chapter.

The lower feedback loop in Figure 8.3 (left) can be built with the code:

```
ks=feedback(k2ss,gpss); sigma(ks)
```

The singular value plot (maximum gain) depicts the *inverse* worst-case unstructured uncertainty that the small-gain theorem guarantees for stability. It will be compared with an alternative "aggressive" design, with 40 times less weighting on the control-action magnitude: `wu=[0.2 0;0 0.2];`.

Regarding nominal performance, the second design achieves a significantly faster loop, with 40% less peak effect under a step disturbance (Figure 8.9).

The sigma plots of `ks` in both cases are depicted in Figure 8.10 (minimum gains are irrelevant). The more aggressive design tolerates three times lower worst-case error bounds (-17 dB (0.14), -6.9 dB (0.45) in the original design).

[8] Further advantages of the cascade loop can be derived if measurement noise amplification is considered (G_2^{-1} is a high-pass filter), not being discussed here.

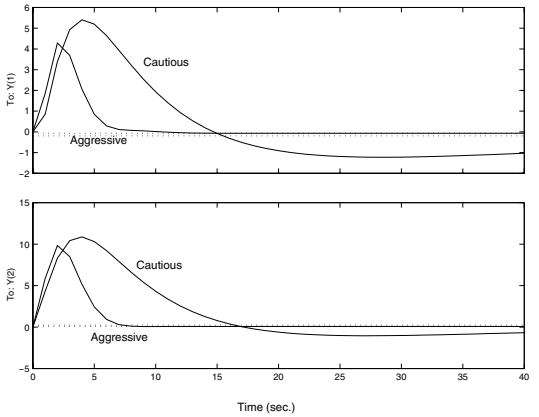

Figure 8.9. A design with higher nominal performance

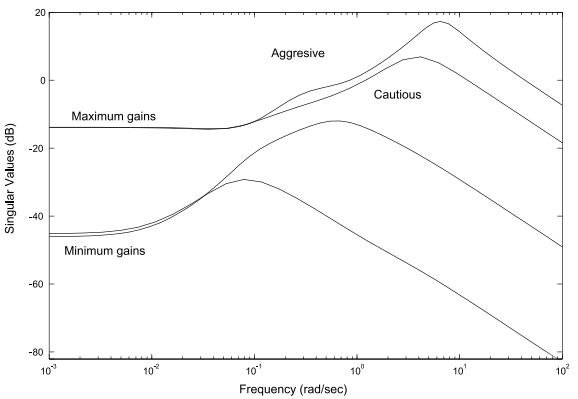

Figure 8.10. Inverse model error bounds (additive uncertainty)

To determine realistic uncertainty bounds, a methodology such as the one in Example 8.8 can be applied. In this case, the additive error is $G^* - G$ where G^* refers to the perturbed plant and G is the nominal one.

Indeed, by setting `gpss2` to be a fifth-order Padé approximation of the delayed plant and `gpss` the nominal one, plotting `sigma((gpss2-gpss)*ks)` for the second aggressive design pinpoints possible stability problems, as at frequencies around 5 rad/s the $\bar{\sigma}$-plot is *above* 0 dB. The conservative design is 5 dB below the limit, leaving room for additional uncertainty before instability onset.

In fact, simulation on the higher-order Padé plant approximation renders an *unstable* loop with the aggressive controller: the new design is useless.

Conservative design. As the peak singular value of `ks` is 6.94 dB = 2.22 (by inspection on Figure 8.10, or evaluating `norm(ks,inf)`), if the differ-

ence between the real plant and the nominal one is any (even dynamic) nonlinearity with gain less than $1/2.22 \approx 0.45$, stability is guaranteed. Indeed, very small differences appear between the higher-order plant approximation and the nominal responses.

However, if parameter changes were also put in place, margins would have been reduced. Depending on the expected modelling errors, perhaps even this design would be risky to implement in practice in a real column, and further increase of W_u might be advisable. If the modelling error bound is not known *a priori*, the usual methodology is to start with *very* conservative controllers and progressively decrease W_u until a terminal controller is found.

Standard Mixed-sensitivity Design

As disturbance models may not be readily available, the mixed-sensitivity design considers deviations caused by a "unit-size" disturbance, $y = Gu + d$ (Figure 4.3 on page 112), leading to the generalised weighted plant in (7.28):

$$\begin{pmatrix} u_w \\ y_w \\ y \end{pmatrix} = \begin{pmatrix} 0 & \rho \\ W_e & W_eG \\ I & G \end{pmatrix} \begin{pmatrix} d \\ u \end{pmatrix} \tag{8.18}$$

The objective is to make sensitivity lower at frequencies where disturbances act with more intensity. Lower frequencies are usually the ones where tight control is required, aiming, for example, to reduce the disturbance effect by 20 times. So, $W_e = diag(20/(\lambda s{+}1))$. The objective is to determine the maximum performance attainable (in the sense that $SW_e < 1$) tolerating an additive modelling error of, say, 0.8 units (to account for Padé approximation, *etc.*). In this case, the weight to the transfer function from d to u will account for robustness constraints (with $\rho = 0.8I$, if $\|KS\rho\| < 1$ then (8.11) is satisfied). If the closed-loop transfer matrix between d and (u_w, y_w) has a norm less than 1, the design will be accomplished (as its two sub-components will themselves be less than 1). The maximum performance will be carried out iterating on λ (progressively decreasing values, indicating higher desired bandwidth).

The design now is:

```
rho=[0.8 0;0 0.8]; we=[20/(lambda*s+1) 0;0 20/(lambda*s+1)];
Pw=[zeros(2,2) rho; we we*gptf; eye(2) gptf];
P2ss=minreal(ss(Pw)); Psm=pck(P2ss.a,P2ss.b,P2ss.c,P2ss.d);
[k cl gamf]=hinfsyn(Psm,2,2,0.05,150,0.005);
```

and the `hinfsyn` command informs us about the achieved norm value. Trying with several values of λ ($\lambda = 1000$, norm $= 0.933$; $\lambda = 500$, norm $= 0.965$; ...) the maximum performance achieved is $\lambda \approx 350$, *i.e.*, strongly attenuating slow disturbances, and progressively decreasing control effectiveness up to 0.06 rad/s where the weight reaches 0 dB. Figures are not drawn for brevity, but this design responds to a disturbance step allowing a maximum transient deviation of 15 units and with a settling time of 45 minutes. That would be the best performance attainable for 0.8 units estimated model error bound.

Alternative design. If the modelling error bound is lowered so comparable performance to design (7.32) is achieved (regarding peak value of closed-loop disturbance step):

```
l=250; rho=[0.35 0;0 0.35]; we=[80/(l*s+1) 0;0 70/(l*s+1)];
```

the resulting regulator has similar step response, but *lower* tolerance to modelling error. If the only significant disturbance source is flow F in model (7.22), a *specialised* design such as (7.32) will beat (regarding robustness) designs trying to minimise disturbances in *all* channels (for the same nominal performance levels). Figure 8.11 depicts the comparison results.

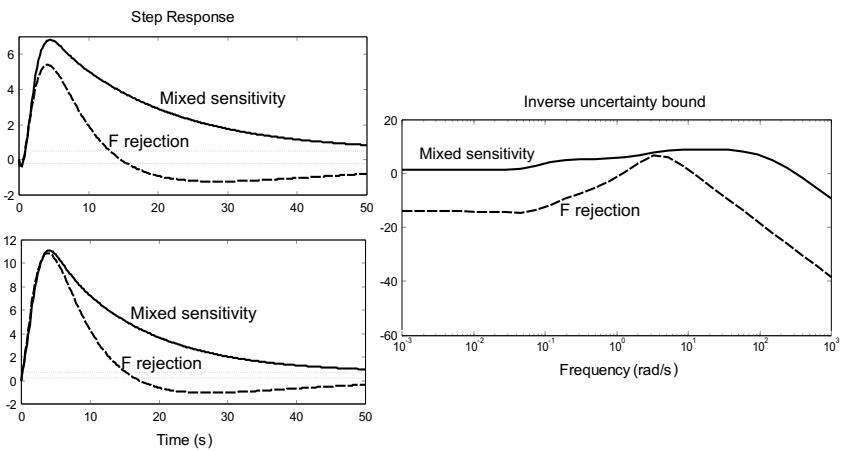

Figure 8.11. Comparison mixed-sensitivity / specialised rejection.

The reason for this result is, as commented on Remark 7.14, that the specialised design only minimises sensitivity in one direction, squeezing out additional robustness at the expense of no rejection performance at all in the other direction.

9

Implementation and Other Issues

This chapter discusses implementation of multivariable controllers, basically on computer platforms as well as some practical issues such as non-liniearities, non-conventional sampling, fault handling and changes in operating modes. As most techniques are model-based, a model of the plant should be available. Estimation and refinements of the model must be carried out in the implementation stage. Also, limitations on the instrumentation should be analysed to assure it is working in the nominal conditions. Otherwise, supplementary control activities should be included.

9.1 Control Implementation: Centralised vs. Decentralised

The previous chapters discussed several options in control design, grouped into two broad categories: on one hand, decentralised, decoupled, cascade, *etc.* strategies and, on the other hand, centralised, optimisation-based control.

The issue here is to discuss critically the advantages and drawbacks of the implementation in practice of these structures, and the technologies involved.

The information flow when implementing the first alternative ("decomposed" control structures) in a complex plant easily becomes complex and difficult to maintain and understand, with lots of nested loops, block-diagrams, *etc.* and dozens of individual regulators to tune. In fact, in many cases, 80% of the PIDs in a large facility are left to factory defaults or with the result of the "autotune" command if available or, in any case, they are never re-tuned when plant characteristics change over time.

So, it appears that, in terms of simplicity and performance, setting up the control problems as optimisations and letting computers obtain the "optimal" solution seems the wisest strategy. However, the results of the optimisation in practice depend on the quality of the model used, and accurate models of large plants are not readily available in many cases.

A fundamental reason to use cascade and decentralised control in most practical applications is because they require less modelling effort.

Other advantages of cascade and decentralised control are:

- its behaviour can be understood by low-qualification operators and technicians, with obvious advantages in maintenance and repairs,
- standard equipment (such as PIDs) is used, so, jointly with the previous consideration, it often results in cost-effective solutions,
- their "localised" and "decoupled" behaviour enables easier tuning, often on-line, with very few parameters to be tuned (*e.g.*, a gain and an integral action), with model-free strategies (such as PID *tuning charts*),
- the extra sensors and actuators in cascade configurations yield less sensitivity to uncertainty, in particular, actuator uncertainty[1].
- decentralised implementation tends to be more fault-tolerant, as individual loops will try to keep their set-points even in the case some other components have failed (if coupling does not destabilise the overall combination). Fault tolerance further increases when implementing override selectors (Section 5.5.2). On the other hand, if a "centralised" controller fails, this may often result in a catastrophic fault and significant downtime.

Depending on the design philosophy, two main implementation variants can be conceived:

- decentralised implementation (of decentralised control designs),
- centralised implementation (all operations carried out on a single location). Centralised designs obviously need a centralised implementation, but decentralised ones can also be implemented on only one processing unit (computer).

By far, in present-day industrial processes, the most popular choice is decentralised PID implementation, with industrial PIDs, sensors and actuators connected to a communication network based on an industry-wide standard. Sometimes PLCs act as middleware between PIDs and the network, and also provide the necessary set-point scheduling. Some PLCs do incorporate AD, DA and PID calculations, sparing the need for some local regulator hardware.

The advice is to use centralised control as an intermediate-hierarchy part of a larger cascade and decoupled structure for subsystems where:

- a stronger degree of coupling and the lack of additional sensors and actuators hinder the use of non-centralised structures, and
- an accurate enough model is available.

[1] Note that, however, a centralised approach using the extra instrumentation can also be designed for enhanced robustness, but in many cases, with a significant modelling cost.

Many applications of centralised control, such as popular predictive controller implementations, interact in this way with the overall plant-wide control structure.

9.2 Implementation Technologies

9.2.1 Analog Implementation

Centralised multivariable controllers are usually implemented on computer frameworks. However, decentralised implementations (multi-loop, cascade, *etc.*) can be realised by means of analog electronics. Based on operational amplifiers, basic circuits can be cascade-connected to form more complex transfer functions.

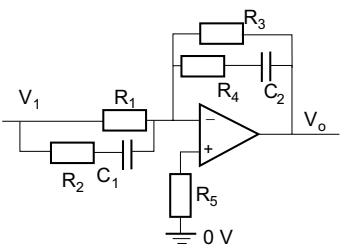

Figure 9.1. operational amplifier basic stage

For example, the circuit in Figure 9.1, with transfer function

$$\frac{V_o(s)}{e(s)} = -\frac{(R_1+R_2)Cs+1}{R_1(R_2C_2s+1)}\frac{R_3(R_4C_2s+1)}{(R_3+R_4)C_2s+1} \tag{9.1}$$

can act as a low-pass filter, high-pass filter (approximate derivative), multiplication by a constant (no capacitors), P, PI, PD regulator, *etc.* depending on the values of its components. The reader is referred to references such as [99] for additional circuits and implementation-related information.

9.2.2 Digital Implementation

In present-day multivariable plants, the control is carried out by means of computer implementation of the controller equations. In fact, the computer is a key element in the overall information flow in plant-wide control, as discussed in Section 9.7. In this section, the basic skills for implementing a control loop routine will be outlined.

The control loop consists of a phase of data acquisition (sensor readings), data processing (control algorithm) and another data output stage (sending

commands to manipulated actuators). These phases should be carried out at the appropriate sampling latency. Let us discuss in more detail some of these tasks.

Hardware

Multivariable controllers can be implemented either in a PC-based framework, in a DSP or micro-controller board, or in specially-made chips. Many of the current platforms have C or C++ compilers available, with 32- or 64-bit IEEE floating-point support. However, in embedded systems, there might be occasions where assembly language and fixed-point math must be used.

In industrial control a PC-based solution has the advantage of being a standard hardware platform, with multiple operating systems and compilers to choose from. However, with tighter space requirements and/or the need for high sampling frequencies, an embedded micro-controller system can be the choice in stand-alone 3-phase inverters, robots, *etc.* Some programmable logic controllers (PLC) include PID regulation modules to implement multi-loop and cascade control, but usually at sampling periods greater than 0.2 s.

Scheduling

The control loop is called from a upper-level routine in charge of scheduling its execution at regular time intervals. This can be done in various ways:

- inside a real-time multitasking operating system. The control loop is set-up as a stand-alone task. After execution, the task is suspended until the next sampling instant. Lower-priority tasks would execute routines related to the user interface (set-points, graphs) and data transmission (in a networked environmment) or logging. Pseudocode might look as follows:

```
task body Controller is
...
begin loop
now=clock;
ControlLoop();
delay until now+period;
exit when EndCondition();
end loop
end Controller
```

  Depending on the operating system and programming language, the `delay until` construction might not be available and then a `delay(period);` should be inserted instead,
- in a single-tasking environment. A delay loop is needed and also incorporating all additional data-logging and communications into the main control loop,
- using an external timer in the controller hardware. In this case, the controller code would either continuously read (*polling*) a particular timer register to determine if it is time to apply a new control action, or a processor *interrupt* could be set up to be triggered by the timer event. The

second option is best, and it can be considered as an elementary form of multitasking, available in many hardware configurations.

Data acquisition

Data acquisition is done via specialised I/O cards in PC frameworks, or with AD converters on a micro-controller board. From the software point of view, the data can be acquired by reading a particular *port* number. I/O ports are used to send and receive information from peripherals.

As data conversion may take a few microseconds to complete, some cards need to first *write* to a port an *AD conversion triggering* command, and then *read* another port until a particular bit (flag) is set to signal that conversion is completed. Then, reading a third port will yield the desired data. In *multiplexed* AD converters (using one physical converter for several input channels), the first operation to be made should be *writing* to a multiplexer register which channel should be read. For example, the C code that reads a 12-bit AD channel on a PCL–8112PG data acquisition card is:

```
double readAD(int ch) {
  word counts;
  outportb(REGMUX,ch);
  outportb(REGTRIGGER,1);
  while ((0x10 & inportb(ADHIGH)));
  counts = (((word)(inportb(ADHIGH)& 0x0f)<<8) +
(word)(inportb(ADLOW)) & 0x0FFF);
  return ((double) counts *(upperlimAD-lowerlimAD)/4095.0) +
lowerlimAD;
}
```

where `REGMUX`, `REGTRIGGER`, `ADHIGH`, `ADLOW` refer to I/O ports and have been initialised to the appropriate addresses from the card manual, and `upperlimAD`, `lowerlimAD` denote the equivalence in true physical units of $2^{12} - 1 = 4095$ and 0 (the range of the 12-bit converter) respectively.

Some cards incorporate a specialised framework for counting encoder pulses to acquire rotational axis positions. Card drivers might provide a higher-level interface to the user, encapsulating the low-level I/O port programming. In modern multitasking operating systems, drivers are mandatory because in most processes the users have not direct access to hardware ports. This would also be the case in most of the situations where data come from a remote component in a networked environment.

The reader is referred to the user manuals of the various cards and to bibliographies such as [64] for details on data acquisition.

Aliasing. In sampled-data control systems, an *analog* low-pass filter (Section A.2.2) is sometimes required to act as an *anti-aliasing* filter, as high-frequency components (higher than the Nyquist frequency $w_s/2$, where w_s is

the sampling frequency) can manifest themselves as low-frequency artifacts[2]. The anti-aliasing filter must be placed in the signal path before the AD converter. Low-pass filters can be built using the circuits in Figure 9.1. For details, see [95].

Control algorithm

The basic implementation of a control system has the following control loop core:

```
//control loop
y=ReadSensors(); r=ReadSetpoint();
CalculateControlAction();
WriteCards(u);
```

DT SISO (for example, PID) controllers are usually implemented by evaluating a difference equation. For example, to implement a regulator with transfer function:

$$G(s) = \frac{u(s)}{e(s)} = \frac{0.3 - 0.4z^{-1} + 0.2z^{-2}}{1 - z^{-1}}$$

the needed code for `CalculateControlAction()` is:

```
ek=y–r;
uk=uk1+0.3*ek-0.4*ek1+0.2*ek2;
uk1=uk;ek2=ek1;ek1=ek;
```

However, in multivariable loops, it is better to implement the state space equations as the state representation, is the one with the minimum number of "memory" variables needed. In fact, implementing transfer function matrices as difference equations may lead to non-minimal realisations. This can become a problem with unstable regulator poles if a "copy" of an unstable mode becomes uncontrollable or unobservable.

So, in this way, a possible implementation of a controller (control action calculation) with a DT realisation (A_r, B_r, C_r, D_r) would be:

```
GetOperatingPoint(u0,y0,r)
dy=y–y0;
du=Cr*xr+Dr*dy;
xr=Ar*xr+Br*dy;
u=du+u0;
```

In the sensor vector appear all possible sensors: tracked outputs, intermediate variables, measurable disturbances. If a matrix library is not available, the involved equations should be fully written as a set of scalar equations.

The feedforward operating point calculation can range from:

- a simple $y0 = r$, $u0 = Fr$ where F is a constant inverse DC gain matrix, for a situation where the number of actuators equals the number of sensors and references,

[2] For instance, sampling $\cos(100\pi t)$ every 0.04 s yields $\cos(100\pi \times 0.04k) = \cos(4\pi k) = 1$, *i.e.*, an "aliased" DC signal.

- more complex steady-state calculations (determining operating point for intermediate variables, for extra actuators using pseudo-inverses, *etc.*),
- full 2-DoF implementation, where `GetOperatingPoint()` itself incorporates state space equations (Section 8.4.3). For example, if all sensors have a reference and the plant is square, as in the first case considered above, equations $u0 = G^{-1}(z)M(z)r$ and $y0 = M(z)r$ should be implemented via suitable realisations,
- inversion of a steady-state non-linear model (or steady-state optimisation of a particular cost index),
- inversion or optimisation of a non-linear dynamic model.

When implementing 2-DoF multivariable controllers with varying set-points for high-order plants, matrix and/or system simulation libraries are indeed helpful.

Continuous-time designs

If a continuous-time realisation (A_r, B_r, C_r, D_r) of a regulator has been calculated, then the core control code would be:

```
GetOperatingPoint(u0,y0,r);dy=y–y0;
xr=xr+T*(Ar*xr+Br*dy);
du=Cr*xr+Dr*dy; u=du+u0;
```

and its evaluation must be carried out fast enough : for reasonable approximation, sampling period T should be at most one tenth of the *fastest* regulator time constant. Longer sampling times could strongly affect the system dynamics. The most elementary rectangle (Euler) numerical integration procedure, see Section 3.4, is also the most common, being the one used in the code above. Other methodologies could be implemented [104]. However, apart from the issues related to the integration method used, the presence of the zero-order hold is an unavoidable source of deviations from the nominal CT calculations. This is why sophistication of the integration methodology is, in principle, not recommended and instead the advice is to use CPU power to increase the sampling frequency. Euler or bilinear $x_k = (I - A_rT/2)^{-1}\left((I + A_rT/2)x_{k-1} + B_rT/2(y_k + y_{k-1})\right)$ integration will suffice for most applications if sampling is reasonably fast.

Depending on particular circumstances regarding operating system behaviour, continuous implementation can be advantageous if a wide variation of the actually achieved sampling time appears due to multitasking, hardware changes, *etc.* If a `clock()` function is available, the controller can determine the time elapsed between two successive control evaluations and apply the suitable T in the integration. For example, Pentium ™ processors have an `rdtsc` assembler opcode that returns the number of clock cycles since the machine start-up.

Non-conventional sampling patterns and controller design techniques, introduced in Section 9.4, including time delays, multi-rate sampling or the

treatment of missing data situations, is a practical matter for reducing the degradation of performance in real-time control implementation (see, for instance [2]).

Numerical precision

High-order regulators need a significant precision in the coefficients in their DT implementations. Otherwise, position of the controller poles may vary significantly (perhaps catastrophically) due to round-off.

Example 9.1. The polynomial $(z - 0.7)^9$ evaluated to four-digit precision (approximately 14 bits) has unstable roots! Check the following MATLAB® output, and notice the unstable root at 1.07:

```
d=poly(0.7*ones(1,9)); d2=0.0001*round(10^4*d); roots(d2)'
    1.0686 !!!     0.9761-0.2419i  0.9761+0.2419i  0.7483-0.3563i
   0.7483+0.3563i  0.5148-0.2955i  0.5148+0.2955i  0.3764-0.1129i
   0.3764+0.1129i
```

As another example, `[b a]=butter(6,20/22050,'high')` designs a sixth-order high-pass Butterworth filter for audio signals, but it also yields[3] an unstable, useless result.

In particular, in the generalised-plant framework optimisation problems (Sections 7.4 and F.3) the resulting controller order is that of the generalised plant, *i.e.*, including *frequency weights*. As there are usually specifications (weights) for each sensor and actuator, the final order can be high. In μ-synthesis there is no *a priori* bound on the resulting controller complexity. For success in practice, implementation of fast-sampled high-order DT controllers should be carried out with double-precision calculations and using an order reduction procedure (see Section 3.10) and, of course, using numerically reliable routines in the controller synthesis steps.

As a (very coarse) rule of thumb, to avoid adding significant error, the number of mantissa bits (significant binary digits) needed in the regulator parameters for reliable implementation of a high-order controller should be at least five times the regulator order (the required digits for a particular pole error spread grow at least linearly with order [87]). Roughly speaking, for regulators up to order five, 32-bit IEEE 754 `float` data representation will suffice, `double`-precision being recommended for higher orders.

For further information on these issues, see [87, 73, 82].

Anti-windup. Note that practical implementations must incorporate in many cases the anti-windup and bumpless-transfer mechanisms discussed in Section 9.3.

[3] MATLAB® Signal-processing toolbox, version 4.2, Pentium ™ processor.

Sending actuator commands

Writing to a DA converter is usually straightforward, carried out by writing to an output port on the I/O address space in the computer. In multitasking operating systems, however, the need for a *device driver* might be mandatory. The same would usually be required to interact with a device on a network.

9.2.3 User Interface

Designing consistent user interfaces for multivariable controllers is a time-consuming task. Although direct coding of an application in charge of control action calculation, card I/O and user interface could be conceivable, and it could even be a reasonable solution at a prototype scale, most large-scale industrial control software is designed to interact with a cascade-like configuration. At the lowest level, servoactuators and decoupled loops receive commands from an intermediate-level layer composed of PLCs and/or industrial PCs in charge of sequencing tasks and control algorithm implementation. The intermediate layer interacts in a networked environment with an upper-level supervision and operator interface software, in charge of data-logging, event reporting (such as alarms) and graphical representation of the state of the process. Operators change set-points and trigger tasks at this level but that user interface application is not usually in charge of real-time control, unless the required sampling period is large enough.

A multitude of commercial applications for plant-wide control implementation are available, such as the SCADA/HMI (supervisory control and data acquisition, human/machine interface) tools or CIM (computer-integrated manufacturing) implementation. Many obstacles in these kind of projects usually arise from conflicting communication standards from different manufacturers. Current trends aim towards integration of SCADA/HMI with the global enterprise information management system.

The interested reader is referred to books such as [31], or commercial software information such as LabView DSC (National Instruments), Delta V (from Emerson Process), FIX (CIM Intellution), Intouch (Wonderware), Plantscape (Honeywell), Scada-VS (Foxboro), Simatic IT (Siemens), Xfactory (Usdata) and coordination and management tools offered from PLC, sensor and actuator manufacturers as well as applications tailored to specific sectors (oil industry, aerospace, *etc.*).

9.3 Bumpless Transfer and Anti-windup

Let us analyse two common related implementation issues, occurring when actual input to the process differs from the one calculated by the controller, either because of saturation non-linearities or because the controller is disconnected from the actuator and manual operation is in place.

Anti-windup (Saturation)

In practice, all actuators saturate sooner or later. Furthermore, a sensible design must expect actuator saturation occurring with non-negligible frequency: if no actuator saturation whatsoever occurs then it is quite likely that the actuator is oversized for the particular application.

Saturation implies that the feedback path is broken. This fact has several implications:

- unstable processes: the process output (related to a constant step response) might go out-of-control to an *unrecoverable zone*. For instance, a commercial jet cannot recover from a nose-down dive, as aerodynamic surfaces cannot exert the required forces. Hence, actuator saturation is a major concern with unstable plants and clear specification of the safe operating regimes and maximum allowed disturbances must be pointed out. In general, in addition to saturation, any significative non-linearity requires the designer to specify to the end-user the disturbance sizes and set-point regions for operation within acceptable performance bounds,
- multi-loop and centralised control: even with stable plants, opening a feedback path may cause the overall loop to become unstable if integrity conditions are not satisfied.
- unstable controllers: it is essentially the same phenomenon as above; however, the instability may affect *internal* regulator variables so they can be modified appropriately. These schemes are termed *anti-windup* configurations. Note that unstable controllers are frequently put in place: *integral actionintegral action* is a paradigmatic example.

Let us discuss anti-windup in more detail. The wind-up problem appeared with integral-action regulators: during significative step changes in the set-point, the integral of the error keeps accumulating (winding up) and when reaching the desired set-point the accumulated integral action produces a significant overshoot increment. If the set-point change is small, the transient time is also small and linear simulations provide a good approximation to the expected response; however, for large set-point changes, continuously-saturating control takes a significantly longer time to reach the desired set-point... and it will force the process beyond it due to the higher-than-expected accumulated wind-up! In SISO PID regulators [20], anti-windup schemes are implemented by either stopping integration if the actuator is saturated or by implementing equations such as:

$$u = K(r-y) - KT_D\frac{dy}{dt} + \int KT_i^{-1}(r-y) + T_t^{-1}(u_m - u)\,dt \qquad (9.2)$$

where u is the calculated control action and u_m is the actual control action applied to the plant (obtained by directly measuring it or by a non-linear actuator model such as a static saturation). In non-saturated behaviour, $u = u_m$ and the equation is the ordinary PID. In saturation, u_m is a "constant"

and the resulting equations drive u down towards u_m dynamically, with time constant T_t. For further detail, see [20, 100].

The situation becomes simpler in "controller + observer" implementations: in this case, the observer must be fed with the *actual* control action, u_m (either measured or from an actuator model). Taking care of this, the observer state will reflect the actual state of the plant being controlled.

Implementation then cannot be a controller realisation such as (6.36), but a direct implementation of the DT observer concept:

```
y=ReadSensors()-y0;
yest=C*x;
x=x+L(y-yest); // do not apply with saturating sensors, see below.
u=-K*x+u0;
WriteActuators(u);
um=ActuatorModel(u)-u0;// or measuring it, and transforming to incremental
variables.
x=Ax+B*um;
```

With saturated sensors or clear outliers, the updating of the state with $L(y - yest)$ should not be executed: saturating sensors would also imply "opening" the loop.

In the previous schemes, the actuator model should include *rate saturation* if it is known to be significant, and the same applies to other non-invertible actuator non-linearities, such as quantisation (multi-level actuators). Invertible non-linearities are discussed in Section 9.5.3.

Note also that anti-windup in cascade control must take into account the limitations on the actual slave actuator to stop integration if the inner loop is saturating.

If the controller is not implemented in "observer + state feedback" form but in a normalised state space realisation (A, B, C, D), anti-windup may take the form:

$$\begin{aligned} \dot{x} &= Ax + By + G(u - u_m) \\ u &= -Cx + Dy \end{aligned} \tag{9.3}$$

so with $u = u_m$ ordinary regulator equations arise and, when u_m is constant (saturation), the system has stable (non-integrating) dynamics given by $A - GC$. Also, the "larger" G is, the more the steady-state gain from u_m to u (equal to $-C(A - GC)^{-1}G$) can be approached to unity. Generalising (9.2), anti-windup implementations in transfer matrix form implement equations such as:

$$u = K_1(s) \begin{pmatrix} r \\ y \end{pmatrix} + K_2(s)(u - u_m) \tag{9.4}$$

One additional issue arising in multivariable anti-windup design is directionality effects (in significantly non-diagonal-dominant systems). For tracking tasks, some anti-windup designs are based on generating a *realisable reference*

[57] by solving a constrained optimisation problem when one actuator saturates. The cost index is the (weighted) difference between the current command, r, and the achievable one, r^*, *i.e.*, $J = \|W(r - r^*)\|$. By doing the calculation every sampling period (or an approximation of it) an "artificial non-linearity" is inserted in the loop prior to the actual saturation.

Systems with redundant actuators can also implement split-range control (see Section 5.5.3).

Bumpless Transfer

A related issue is trying to achieve bumpless transfer when switching from manual to automatic operation mode or under operating mode changes.

On one hand, if the regulation is off when in manual mode, when switching to automatic, as the state of the controller (or the integrators in PID regulators) is usually initialised to zero, a transient appears. It is highly undesirable to be near the desired steady-state set-point by manual driving and then switching to automatic to see a significant transient deviation.

On the other hand, if regulation is on when in manual mode, a situation similar to the wind-up previously discussed appears: integrators (or unstable regulators) accumulate and internal variables go unstable.

The solution is, then, similar to that of the wind-up phenomenon: the regulator should be always on, carrying out calculations by using expression (9.2) with u_m indicating the actually applied (manual) control. Likewise, in multivariable state feedback plus observer set-ups, the observer input should be the actually applied control action.

Controller switching. Similar situations arise when switching between different regulation strategies [96]. This is a frequent case in control structures such as override and selector ones (see Section 5.5.2) where commutation between various controllers is put into practice: unstable (integrating) regulators will wind up unless taken into account in implementation. Some specific issues may also arise in cascade control.

A related issue is the possibility of controller *parameter changes* made, on the fly, by the end-user. In particular, in implementation of (9.2), integration must be carried out *after* multiplication by KT_i^{-1} and T_t^{-1}. Otherwise (*i.e.*, implementing $KT_i^{-1}\int(r-y)\,dt$), changing K or T_i would result in a discontinuous control action that will drive the process away from its operating point, producing an undesirable windup-related transient. A similar situation may be considered with integral gains in MIMO set-ups.

9.4 Non-conventional Sampling

The more samples available in a particular time interval the better the state estimates can be obtained, in the sense of better filtering of measurement

noise (less expected variance of the prediction error). On the other hand, frequent actuator activity can produce vibration in mechanical components, which causes faults because of fatigue and, anyway, actuator bandwidth is usually limited. These considerations, jointly with communication latency, *etc.* may justify in some cases the need for different sampling periods for sensors and actuators (non-conventional sampling). This situation is called "*multi-rate* output control". In some other cases, the opposite framework is forced by the instrumentation: the output sampling latency is slow (image processing sensors or chemical analysers, for instance) but the control updating can be done faster to avoid strong variations. The "multirate input control", or any combination of situations, claim a flexible sampling/updating pattern.

In fact, some non-conventionally-sampled control loops are transparent to the user because some so-called *smart sensors* carry out internal oversampling and filtering. In addition to that, some actuators use cascade control at a fixed rate independent of the latency of commands from a master regulator.

In centralised control, the separation principle discussed in Section 6.2.4 can be extended so that sensing (state estimation) and state feedback control can be carried out at different rates provided that the L and K matrices are calculated with a common CT model discretised at the two different sampling periods. Implementation is easily carried out in the case that one of the sampling periods is a multiple of the other, and it is used in practice in some cases, either implicitly (as in the case of smart sensors) or explicitly in the computer code.

More general cases with different periods for each sensor and actuator can be thought of. The reader is referred to [7] for further details.

Example 9.2 (Dual-rate control). The following example illustrates a possible procedure for implementing a controller actuating with a period $T_c = 1$ s, with sensing every $T_s = 5$ s. The system to be controlled is:

$$\dot{x} = \begin{pmatrix} -0.1 & 0.06 \\ 0.09 & -0.3 \end{pmatrix} x + \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} u$$
$$y = \begin{pmatrix} 1 & 1 \end{pmatrix} x$$

First, the system is discretised at the lowest period (1 s). Let us denote by A_1 and B_1 the state equation matrices for this discretised system. Then, for the sensing period, the state equation is:

$$x((k+1)T_s) = A_1^5 x(kT_s) + A_1^4 B_1 u(kT_s) + A_1^3 B_1 u(kT_s + T_c) + A_1^2 B_1 u(kT_s + 2T_c) + A_1 B_1 u(kT_s + 3T_c) + B u(kT_s + 4T_c) \quad (9.5)$$

The output equation is unchanged with sampling. So the higher-period system can be thought as a multi-input one, with $A_5 = A_1^5$ and the same C matrix. Only A_5 and C will be needed for observer design at the slow rate.

The system's open-loop settling time is around 50 s. Pole-placement techniques will be used to achieve a settling time for both controller and observer around 10 s. This amounts to a desired pole position of $s = -0.4$ in the continuous-time complex

plane. In the discrete plane, for controller design, the pole position is $z_c = e^{Ts} = e^{1*(-0.4)} = 0.67$. For observer design[4], it is $z_o = e^{5*(-0.4)} = 0.135$. As both process poles are slower than the desired ones, the calculation is solved as:

```
K=place(A1,B1,[0.67 0.66]); L=place(A5',A5'*C',[0.135 0.13]);
```

yielding $K = \begin{pmatrix} 0.5773 & 0.0831 \end{pmatrix}$ and $L = \begin{pmatrix} 0.66897 & 0.2016 \end{pmatrix}^T$. Simulation of the dual-rate controller for a starting state $(10, 10)^T$ can be carried out by the following MATLAB® code:

```
contr=[];listx=[];listy=[]; x=[10;10];listx=[listx x];
xm=[0;0];
for n=1:10
  y=C*x;
  ym=C*xm;
  xm=xm+L'*(y-ym);
  for l=1:5
    u=-K*xm; contr=[contr u];
    x=Ad1*x+Bd1*u;
    listax=[listax x]; listay=[listay C*x];
    xm=Ad1*xm+Bd1*u;
  end
end
```

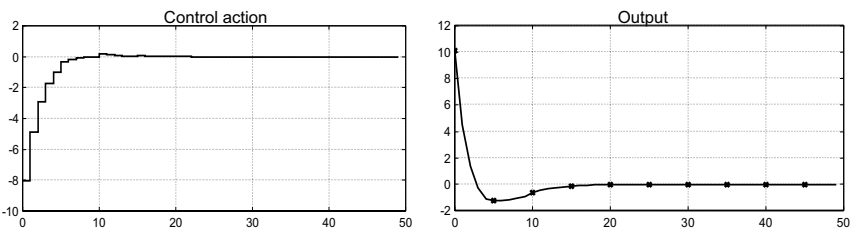

Figure 9.2. Control action and output (output samples marked x)

Figure 9.2 plots the control action and output achieved.

The idea of assimilating multi-rate systems to MIMO ones with suitable multiplication of the number of inputs or outputs as in (9.5) is called *lifting*.

It is left to the reader to solve and simulate the previous problem with $T_s = 1$ and $T_c = 5$. In this case, the lifted system would have as equations:

[4] Usually, observer poles are placed at a faster location than regulator ones. In this set-up, where few samples are available, if observer settling time were to be made faster, it would approach the minimum-time observer (poles at the origin, increasing its sensitivity to noise): note that the 10 s target settling time corresponds to only two measurements. Note also that the same time constant is mapped onto different $\mathcal{Z}$-plane poles.

$$x((k+1)T_c) = A_1^5 x(kT_c) + + \left(A_1^4 B_1 + A_1^3 B_1 + A_1^2 B_1 + A_1 B_1 + B\right) u(kT_c)$$

$$\begin{pmatrix} y(kT_c) \\ y(kT_c + T_s) \\ y(kT_c + 2T_s) \\ y(kT_c + 3T_s) \\ y(kT_c + 4T_s) \end{pmatrix} = \begin{pmatrix} C \\ CA_1 \\ CA_1^2 \\ CA_1^3 \\ CA_1^4 \end{pmatrix} x(kT_c) + \begin{pmatrix} 0 \\ CB_1 \\ CA_1 B_1 \\ CA_1^2 B_1 \\ CA_1^3 B_1 \end{pmatrix} u(kT_c)$$

Figure 9.3 depicts the dual-rate structure.

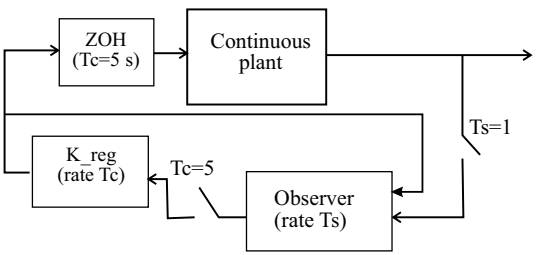

Figure 9.3. Dual-rate control system.

The use of other techniques (such as the ones in Chapter 6) poses no fundamental problems for these simple cases. It is just a question of designing a controller for the lifted system. In some cases, a ripple may appear in the control action commands. However, note that the "true" frequency response and the lifted one are not comparable so consideration of modelling errors and high-frequency response (around and above the lowest sampling frequency – largest period – in the system) becomes cumbersome: minimisation of a norm on the lifted system does not entail the same on the original system. This is a matter of current research. In a more general situation (for example, when the greatest common divisor of T_s and T_c is not one of them, *i.e.*, where lifting is applied to input and output vectors) direct application of the techniques can lead to *causality* issues such as generating controllers whose action depends on future measurements. The reader is referred to [23, 1] for further details on the issues involved.

9.5 Coping with Non-linearity

Real plants always have non-linearities. So, taking into account its presence, the likelihood of a simulated design to work in practice will increase. Controllers based on linearised models are guaranteed to work only in some neighbourhood of a single equilibrium point. However, non-linear systems exhibit a plethora of interesting phenomena without a parallel or approximation in linear systems [72]. Those phenomena are sub-harmonic oscillations, chaos, bifurcations, *etc.* An example of this is the chaotic behaviour of a PI-controlled

CSRT, Example 2.1, in [101]. Non-linear control is thus a challenging subject. Some possibilities and illustrative examples will be briefly discussed in the remaining of this section. The interested reader may consult [120, 74] for ample details on the topic.

9.5.1 Basic Techniques

***Ad hoc* strategies.** By far, the most frequent non-linearity is actuator saturation. Apart from specific design techniques [61], some considerations about input amplitude constraints have been discussed in Sections 7.3.3 and 8.3.1, but usually it is disregarded in the design phase or, in the case that unstable regulators are in place, using any anti-windup mechanism (Section 9.3). Indeed, if a model is available, anti-windup techniques may mitigate other actuator non-linearity effects.

Other frequent non-linearities are stiction, dead-zone or hysteresis in mechanical actuators. A usual, simple approach to deal with these is *dithering* (addition of low-amplitude, medium-to-high frequency zero-mean signals to actuator commands). With it, small pulses above the dead-zone amplitude avoid some of the effects (in particular, the ineffectiveness of small control command increments) and diminish steady-state errors.

Robust linear control. An LTI control system, based on a linearised model, must be robust to slight non-linearities in a certain range of operation. Essentially, the wider the desired range of operation the lower the nominal performance that should be required to increase robustness margins. Note that the considerations in Section 8.5.3 still apply if Δ is a smooth non-linearity, but (8.11) and (8.13) cannot be interpreted on a frequency-by-frequency basis as we are not under the assumption of LTI uncertainty, (C.19).

Note also that linear integral control can compensate for significant non-linearities at steady-state. This would not be the case if the sign of the gain or the determinant of $G(0)$ in the MIMO case change at different operating points, following (5.2).

Additional sensors and actuators. Cascade control (Section 5.4) may also be used to compensate for non-linearities if subsystems exist with fast enough dynamics compared to the overall desired performance level. Note that, however, cascade control requires extra instrumentation.

In general, adding sensors and actuators usually improves both nominal performance *and* robustness margins, so the valid operation region of a linearised controller may be extended. In complex systems, this is a convenient, widely-used approach if the cost of the additional instrumentation is reasonable.

Feedforward non-linear control. If available, sometimes a non-linear model can be used to generate off-line the sequence of control actions to be fed in open loop to the plant to achieve particular objectives. This can be

part of a 2-DoF structure where a (possibly linear) robust enough feedback will try to minimise the effect of disturbances and modelling error. Of course, the idea only applies to reference tracking problems. For repetitive tasks, the feedforward sequence can be iteratively learnt [111].

9.5.2 Gain-scheduling

Gain-scheduling encompasses a family of methodologies to interpolate linear controllers. The basic philosophy involves four steps:

- with a non-linear model, a linearised model can be obtained for different desired points of operation. These models can even be identified by experiments with reasonably small control action amplitudes,
- a finite set of key operation points are usually selected. Afterwards, a linear controller is designed for each of these points[5],
- the controllers are interpolated using as interpolation variable a set of measurable *scheduling variables*, σ. These scheduling variables are assumed to be related to the main cause of non-linearity and, hopefully, being slowly varying and not too dependent on the plant state,
- the performance of the resulting controller must be evaluated. Usually, the performance assessment is carried out by extensive simulations and prototype tests, as controller switching, non-linearities and modelling errors make theoretical stability proofs a hard task in many cases [83].

Example 9.3. In aeronautics, the forces exerted by deflection of aerodynamic surfaces depends on the air atmospheric pressure. Autopilots use this measurement as scheduling variable to calculate and compensate for this effect. As the non-linearity mainly affects actuator gain, this was the first "gain-scheduling" (literally) application from which other refinements took the name. Current designs use as a scheduling variable a vector containing Mach number, altitude, pressure and angle of attack.

In this approach, a non-linear model:

$$\begin{aligned} \dot{x} &= f(x, \sigma, u) \\ y &= h(x, \sigma, u) \end{aligned} \tag{9.6}$$

is transformed into a linearised one (now x, u refer to incremental variables):

$$\begin{aligned} \dot{x} &= A(\sigma)x + B(\sigma)u \\ y &= C(\sigma)x + D(\sigma)u \end{aligned} \tag{9.7}$$

so that a controller with transfer matrix $K(\sigma, s)$ is designed. If the operating space is divided in disjoint regions centred on a set of selected values σ_i, the easiest way to carry out scheduling is switching to controller $K(\sigma_i, s)$ if the distance between σ_i and the measured σ is the lowest. However, generic

[5] Perhaps there exists a common linear controller fulfilling the design requirements. However, the operating point needs, anyway, to be scheduled.

controller-switching may be subject to problems regarding bump transfer and windup-like phenomena and it may yield an overall *unstable* closed loop even if each controller K_i stabilised the model with σ_i.

If the controller is in observer + state feedback form, gain-scheduling gets easier as all controllers share the same state so bump transfers are mitigated. An application of gain-scheduled $\mathcal{H}_\infty$ loop-shaping is reported in [62].

Progressive interpolation between controllers and matrices can be carried out based, for example, on linear interpolation for a single scheduling variable or on weighted averages of inverse distance to points σ_i for multiple scheduling variables (*i.e.*, σ_i is itself a vector).

Note that, as linearised models are expressed in incremental variables, scheduling on the operating points must be also set up. This scheduling can be actually implemented or taken care of with integral action. Some gain-scheduling techniques implement this integration by linearising a model on the derivatives of state variables, under the name of velocity algorithms [71, 80].

Gain-scheduled control is usually intended for slow, smooth transitions between different operation points (with dynamics of σ and switching significantly slower than that of the desired "local" closed loops). Simulations may work for these kind of changes, but become unstable for sudden jumps in set-points. So, 2-DoF implementations with set-point rate saturation (see Section 8.4.3) are indeed advisable.

The reader is referred to [109, 74, 62, 124] for further details.

Example 9.4. An elementary example of gain-scheduling in decoupling is described in the case study in Section 5.8.2.

9.5.3 Global Linearisation

Sometimes, in control system design, invertible non-linearities are found, such as valve characteristics, dead-zones or Coulomb friction. Including the "inverse" of the non-linearity (or an approximation of it) in the control action can substantially widen the acceptable operating range of a controller designed by linear methods. These methodologies are denoted as *global* linearisation.

The following example illustrates feedforward linearisation (no sensor needed) inverting a static non-linearity. Another simple example was given in the footnote on page 164.

Example 9.5 (Actuator non-linearity). Let us have a control system with actuators with known invertible non-linear characteristics, $u = f(v)$, where u is the actual input received by the plant, v is the input command from the controller and f is a (usually diagonal) non-linearity.

If the plant is modelled by a linear model and a linear controller is designed for it, producing a desired control action, u, the controller must output $v = f^{-1}(u)$.

For instance, an actuator dead-zone or Coulomb friction can be modelled as:

$$u = \begin{cases} v - \delta & v > \delta \\ 0 & -\delta \leq v \leq \delta \\ -(v - \delta) v < -\delta \end{cases}$$

Its inversion amounts to inserting a line of code such as:

```
WriteActuators(u+sign(u)*delta);
```

Feedback Linearisation

Sensor readings can take part in the non-linearity inversion. The methodology is then denoted as *feedback* linearisation.

Example 9.6 (Invertible plant non-linearity). Let us have a tank system with equation:

$$A\frac{dh}{dt} = u - b\sqrt{2gh}$$

where u is the inflow command achieved by opening a servovalve.

By defining an auxiliary variable $v = u - b\sqrt{2gh}$, the system equation in this variable is:

$$A\frac{dh}{dt} = v$$

so a linear, proportional controller can be designed in terms of the auxiliary variable if v is assumed to act as an input: $v = -kh$, to fulfill the desired specifications. So, the control input to the tank system would be, reversing the change of variable:

$$u = v + b\sqrt{2gh} = -kh + b\sqrt{2gh}$$

In coupled MIMO systems, auxiliary variables usually need to be defined in a more complex way, generalising the methodology in Section 5.3.

Let us have a non-linear system which can be expressed in the form:

$$\dot{x} = f(x) + g(x)u; \quad y = h(x) + q(x)u \tag{9.8}$$

Considering output y_i, perhaps it does not depend directly on u. Then, its derivative can be calculated in a similar way as in the linear case in (5.15) or in Example 5.10 on page 140, yielding:

$$\dot{y} = h_i^1(x) + q_i^1(x)u \tag{9.9}$$

where h^1 and q^1 are new functions. If it does not depend on u ($q_i^1(x) = 0$), further derivatives are taken until u appears. This derivative index is, as usual, the relative degree, r_i. Carrying out the same operation for all outputs, the result, generalising (5.17), is:

$$\tilde{y} = \begin{pmatrix} \frac{d^{r_1} y_1}{dt^{r_1}} \\ \vdots \\ \frac{d^{r_m} y_m}{dt^{r_m}} \end{pmatrix} = \begin{pmatrix} h_1^{r_1}(x) \\ \vdots \\ h_m^{r_m}(x) \end{pmatrix} + \begin{pmatrix} q_1^{r_1}(x) \\ \vdots \\ q_m^{r_m}(x) \end{pmatrix} u = \tilde{H}(x) + \tilde{Q}(x)u \tag{9.10}$$

So, if x is measurable, $\tilde{H}$ and $\tilde{Q}$ can be calculated and so the non-linear state feedback:

$$u = \tilde{Q}^{-1}(x)\left(v - \tilde{H}(x)\right) \tag{9.11}$$

transforms the process equations, as in (5.18), into linear, decoupled chains of integrators[6]:

$$\tilde{y} = v$$

Example 9.7 (Robot control). Robot arms, with a torque or force exerted by a motor on each joint (rotational or linear) can be modelled [115] by equations such as:

$$I(q)\frac{d^2q}{dt^2} = \tau + G(q) + H(q, \frac{dq}{dt}) \tag{9.12}$$

where q is a vector formed by suitable angular θ_i and linear z_j coordinates describing robot position, $I(q)$ is an *inertia matrix* (depending on the joint positions), τ is a vector with the joint inputs (torques and forces), $G(q)$ is a vector of gravitational effects (depending on the joint position) and $H(q, \dot{q})$ encompasses friction models and Coriolis-effect terms. Trigonometric expressions form the elements of I, G and H. The above reference gives full details on robot-arm modelling.

In this class of systems, rewriting equation[7] (9.12) as:

$$\frac{d^2q}{dt^2} = I(q)^{-1}\left(\tau + G(q) + H(q, \frac{dq}{dt})\right)$$

and defining the auxiliary variable as:

$$v = I(q)^{-1}\left(\tau + G(q) + H(q, \frac{dq}{dt})\right) \tag{9.13}$$

the system has a decoupled, diagonal, double-integrator transfer function matrix:

$$\frac{d^2q}{dt^2} = v$$

so a set of linear controllers for each component of q can be easily designed (in fact decentralised PD-like ones, $v_i = K_P^i q_i + K_D^i \dot{q}_i$, would suffice). The control forces and torques, τ, to be applied to the actual plant are, reversing the change of variable:

$$\tau = I(q)v - G(q) - H(q, \frac{dq}{dt}) \tag{9.14}$$

The final control law is centralised, non-linear and requires real-time computation of I, G and H with measurements of all positions and speeds and actuators at all joints. This robot control strategy is called *computed torque* control.

[6] These operations may require significant computational load on MIMO non-linear systems as fast sampling is required (most of the remarks on page 141 extend to the non-linear case).

[7] The inertia matrix is always invertible (and positive definite) for any realistic mechanical system.

Notice that this technique achieves, in non-linear MIMO systems, both *linearisation* and *decoupling*. Of course, some technical conditions need to apply in the case of general non-linear systems for the existence, smoothness, finite relative degree and invertibility of the derivatives involved in the calculations. The reader is referred to [120, 74] for further details. In some cases, insight into the physics of the system (such as in the previous examples) can help the control engineer in finding a suitable linearising transformation. In other cases, differential geometry tools are needed.

The importance of these results is that linearisation is achieved not only "around" a "small" neighbourhood of an operating point, as in a Taylor-series based one (Section 2.5), but in a much wider region of the state space (necessary conditions are that inverse matrices do exist and the relative degrees do not change). For example, in many fully-actuated mechanical systems (one drive for each degree of freedom) such as the previous example, this feedback linearisation is *global*.

9.5.4 Other Approaches

Non-linear control is a vast topic and a wealth of techniques are being developed, aiming for non-linear objectives without any conversion or use of linear techniques. Some of the techniques address adaptation and robustness as well.

Adaptive control. Based on error signals, the controller itself can modify its underlying process model and/or its parameters reacting to operating point changes. It is the subject of intense theoretical research but with limited practical applications. The reader is referred to [27, 22, 45, 79] for details.

Non-linear optimal control. The optimisation cost indexes and procedures in Chapter 7 can be generalised to some non-linear models using dynamic programming and calculus of variations [122, 81]. The robustness-related norm-optimisation is also being studied [58]. As optimisation yields stable controllers in most cases (under full-information state feedback), in the chemical industry some packages incorporate non-linear optimisation of PID parameters or non-linear models in predictive control (Section 7.3).

Energy-based control. With some models, stabilisation with non-linear controllers can be carried out if a controller is designed so that the total energy is decreasing (dissipative systems). Physical insight into mechanical systems, electrical power grids, *etc.* can help in finding a globally stabilising controller. For arbitrary non-linear systems, advanced geometrical methods may be needed. The reader is referred to [74, 13] for details.

9.6 Reliability and Fault Detection

The main objective of control is reducing the effect of uncertainty on the process outcome. Apart from modelling errors and disturbances, faults are

sources of significant deviations of the controlled magnitudes, so reliability can be considered as one of the objectives of control [108].

A reliable control system implementation must:

- detect faults as soon as possible to minimise the loss due to off-specification product,
- compensate for the fault effects changing the control configuration, if possible, to avoid plant down-time,
- avoid damaging key process components.

High reliability is the main goal of automatic (or human) supervision systems. Reliability requirements are very strict in human life-related situations or in those activities where failure can lead to important economic loss (many aerospace, automotive, nuclear and chemical industries).

Instrumentation faults. In closed-loop control, the possibility of sensor faults must be considered. A typical scenario involves integral controllers: a sensor fault will drive the corresponding actuators to saturation, so a loose wire may produce significant product loss or plant damage!

- unstable regulators (such as integral-action ones) may destabilise the process due to sensor faults. In MIMO multi-loop control, integrity-related issues arise (see Section 5.2.2),
- unstable processes may become out of control due to any actuator or sensor fault, or under the presence of large disturbances,
- stable processes with high-gain stable regulators may become unstable on the loss of a particular sensor or actuator.

Instrumentation design should aim to minimise the effect of its own faults on the process. For example, in integrating regulators, anti-windup schemes are implemented and, in some cases, integration is carried out only if the error is below a certain value, protecting from the above sensor fault.

Redundancy. The key to enhancing reliability is the addition of redundant elements. Redundancy may be set up by a combination of:

- physical redundancy, with several identical sensors or actuators,
- analytical redundancy, using mathematical models to reconfigure the system. For example, if multiple (not necessarily identical) sensors and actuators are available, a sensor loss can be handled by recalculating the observer gain, an actuator loss can be handled by recalculating the state feedback gain (performance decrease can be expected).

Fault Detection

Faults are usually detected when significant deviations of particular variables occur. Fault detection has a significant relationship and parallelism in some methodologies from industrial quality control [92]. Fault detection is usually based on setting *thresholds* over particular magnitudes:

- thresholds over *directly measured* variables, for example, pressure or temperature alarms. In many industrial plants, a significant percentage of the available sensors are devoted to this mission,
- Thresholds over *calculated* quantities, on the basis of measurements:
  - evaluating mean, variance or correlations on a batch of samples. Most applications compare deviations with respect to a nominal mean. However, in some cases, faults may be detected because the variability (variance) of a measurement significantly changes,
  - setting thresholds on a linear combination of measured variables (calculating variance ellipsoids),
  - monitoring model-based static algebraic relations between variables. For example, detecting pipe obstruction by comparing flow-meter readings with pressure-drop sensor readings,
  - evaluating relations on variables after a dynamic processing. Typical examples are:
    - Model-free processing, evaluating low-pass filtered variables, "tendencies" (filtered derivatives $s/(\lambda s + 1)$), to remove measurement noise and irrelevant short-term dynamics,
    - evaluating Fourier transform (frequency response) of signals, looking for presence of information on a particular frequency band. This is a widely-used tool in diagnosing rotating machinery and combustion engines,
    - using dynamic plant models. For example, by comparing the measured output with a model's output $G(s)u(s)$,
- Parameter *estimation*. Using identification algorithms to monitor the values of a set of physical parameters related to faults being identified.

Threshold setting must balance a trade-off between *false positives* generated by transient higher-than-expected disturbances and *false negatives* caused by a too-high threshold. High thresholds diminish sensitivity to disturbances but also faults take longer to be detected or they can even remain undetected.

For a better compromise, sometimes additional processing on thresholds is carried out (for instance, triggering a fault if a threshold has been reached for a particular number of times over a set of recent past samples), or even on-line modifications of them based on past data records.

Fault Isolation

Once a fault has been detected by a particular threshold being triggered, the cause of the fault must be isolated. The first issue is to distinguish a process fault from an instrumentation one. This can be done, in some cases, by means of voting:

- voting between several identical sensors, using physical redundancy. If all or most of them yield abnormal measurement, the fault is on the process. Otherwise, the fault is probably in the discrepant sensor,

- voting between related variables, using analytical redundancy. If all are out of bounds, an actuator or process fault can be detected, otherwise a sensor fault must be suspected.

If the fault is suspected to lie in the process itself, isolating which particular component is responsible for the deviation may not be an easy task in some circumstances, due to:

- similar fault modes (fault in component A yields signal variations that are very similar to fault in component B),
- cascade faults: fault in one component entails defective operation on others (or even overload and subsequent failure),
- effect of closed-loop control: regulators, trying to keep variables around their set-points, often hide the fault effects or they appear in secondary, less tightly controlled, variables. So, closed-loop operation may modify the *failure mode*.

Fault isolation may require, on many occasions, plant down-time to carry out special experiments and inspections.

Fault Quantification and Reconfiguration

After isolating the fault cause and determining its severity, one of these three actions must be decided on:

- maintenance, with likely need of plant close-down,
- continue operation with degraded specifications (for example, switching to open-loop control after a confirmed sensor failure), until a scheduled maintenance stop,
- continue operation with no degrading in specifications due to redundance in sensors, actuators or power sources, updating some parameters.

The concepts of controllability and observability (and the SVD decomposition of the related matrices) are quite related to the ability of reconfiguration after actuator and sensor faults respectively.

Not all of these fault-management tasks can be carried out automatically in a general situation. Some of these need human intervention. There are some diagnostic tools based on rule-based expert systems [10, 132].

As a conclusion, the topic of reliability and fault detection needs considerable attention in practical engineering process design. The reader is referred to [37, 17] and references therein for in-depth treatment.

9.7 Supervision, Integrated Automation, Plant-wide Control

Modern control tasks in plant-wide control designs involve many objectives, and not only set-point tracking and regulation. So, a control engineer must integrate all designs into a unified framework that is able to provide:

- process control (set-point tracking, regulation against disturbances),
- start-up and close-down phases,
- report plant performance to the plant information systems,
- report possible fault sources,
- provide a user interface for these tasks.

So, over a basic layer of process control interface (sensor and actuator management), several coordinated tasks must be set up in a working control system. The coordination of these tasks is in charge of the *supervision* system. Some of these are:

- basic regulation (the main topic of this book),
- scheduling of procedures if the task comprises several stages, establishing a suitable sequence of set-point settings for all variables,
- coordination in distributed decentralised and cascade strategies,
- management of shared resources,
- handling of unexpected alarms and exceptions (scheduling of recovery procedures when an abnormal situation is met, informing the operator about the events),
- recording the different signals, events, performance measures, *etc.* onto a database, for further analysis to determine the need of readjustment or improvement based on long-term histories.

To achieve these roles, the control system cannot be one all-powerful routine, but it should be made up of components operating in a coordinated fashion. These components not only refer to physical ones (valves, connections, PLCs) but also to *software components*, *i.e.*, modules and tasks acting on the same or different computers coordinated via a communication network. In fact, it should be noted that in a distributed process control computer package, the code involving the control or regulatory activities is a minor part (sometimes just a few lines of code) of the global software. Nevertheless, we can say that the rest of the code makes the system work and the control code determines a "good" or "optimal" behaviour.

In this setting, issues relating to real-time operating systems, standard object architectures (CORBA, ...) should also be taken into account. So, in this way, the need for integrated information systems for plant-wide automation quickly arises: the computer has become the main technology in modern industrial process control, and all control devices, operator stations, remote access points, *etc.* are part of a networked environment.

These issues are out of the intended scope of this work, but are critical in the implementation of large-scale control systems. The reader is referred to [114, 46, 31] and the references therein for an overview of these issues.

This page intentionally left blank

A

Summary of SISO System Analysis

In this brief appendix, some basic results on SISO control are summarised to ease reading of some of the main chapters without resorting to other sources.

A.1 Signals

A *continuous-time* signal is expressed as a function of a real (positive) time variable $y(t) : \mathbb{R}^+ \to \mathbb{R}$. It can be also represented by its Laplace transform

$$y(s) = \mathcal{L}[y(t)] = \int_0^\infty e^{-st} y(t) dt \tag{A.1}$$

the symbol s being an undeterminate complex variable. Under some constraints, it can be also represented by its frequency content as expressed by its Fourier transform and, by extension, computing the Laplace transform for $s = j\omega$:

$$y(j\omega) = y(s)|_{s=j\omega}$$

A *discrete-time* signal is expressed by a sequence of data, usually representing the signal value at discrete instants of time:

$$\{y_k\} = \{y_0, y_1, y_2, \ldots, y_n, \ldots\}$$

If the data are taken at regularly spaced instants of time, using the unit delay operator, z^{-1}, the DT signal can be represented by its $\mathcal{Z}$-transform:

$$y(z) = \mathcal{Z}[y_k] = \sum_{k=0}^{\infty} y_k z^{-k} \tag{A.2}$$

A.2 Continuous Systems

Linear time-invariant continuous SISO systems are described by an ordinary differential equation in the form:

$$a_n y^{(n)}(t) + \cdots + a_0 y(t) = b_m u^{(m)}(t) + \cdots + b_0 u(t) \tag{A.3}$$

With $u \equiv 0$, the solution to these equations is denoted as *free response* and has the form:

$$y(t) = \sum_{i=1}^{n} P_i(t) e^{\lambda_i t} \tag{A.4}$$

$$P_i(t) = m_{i0} + m_{i1} t + \ldots \tag{A.5}$$

where λ_i are the zeros of the *characteristic polynomial*:

$$a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0 \tag{A.6}$$

For a particular set of initial conditions $y(0)$, $y'(0)$, ..., $y^{(n-1)}(0)$, the solution is unique (initial conditions determine the coefficients m_{ij}).

With $u \neq 0$, the solution can be decomposed into two components, $y(t) = y_h(t) + y_p(t)$, where $y_h(t)$ has the form (A.4) and corresponds to the *transient response* and $y_p(t)$ is denoted as the *particular*, or input-related, solution. If $u(t)$ is a constant, there exists a constant particular solution, called *steady-state* or *equilibrium* point.

Laplace transform. The Laplace transform, (A.1), can be used to solve ordinary linear differential equations. The inverse Laplace transform is:

$$f(t) = \frac{1}{2\pi j}\, p.v. \int_{c-j\infty}^{c+j\infty} F(s) e^{st} ds$$

where $c \in \mathbb{R}$ has a real part bigger than that of any singularity in $F(s)$. The inverse Laplace transform can also be evaluated by partial fraction expansion and table look-up in many cases. Tables for the most common cases appear in Section A.5.

Among the essential properties of the transform are:

- linearity: $\mathcal{L}(\alpha f + \beta g) = \alpha \mathcal{L}(f) + \beta \mathcal{L}(g)$,
- derivative: $f(0)$ being the initial condition,

$$\mathcal{L}\left(\frac{df}{dt}\right) = s\mathcal{L}(f) - f(0)$$

- successive derivative:

$$\mathcal{L}\left(\frac{d^n}{dt^n} f(t)\right) = s^n \mathcal{L}(f) - \left(s^{n-1} f(0) + \cdots + f^{(n-1)}(0)\right) \tag{A.7}$$

- time delay: given $f(t)$, defining:

$$f^*(t) = \begin{cases} 0 & t < T \\ f(t-T) & t \geq T \end{cases}$$

then:

$$\mathcal{L}(f^*) = \mathcal{L}(f)e^{-Ts} \tag{A.8}$$

- final value theorem. If the left limit in the expression below exists, then:

$$\lim_{t\to\infty} f(t) = \lim_{s\to 0} sf(s) \tag{A.9}$$

- initial value theorem. If the left limit in the expression below exists, then:

$$\lim_{0\leftarrow t} f(t) = \lim_{s\to\infty} sf(s) \tag{A.10}$$

Using the above properties, the Laplace transform of the differential equation (A.3) is:

$$\begin{aligned} &a_n\left(s^n y(s) - (s^{n-1}y(0) + \cdots + y^{(n-1)}(0))\right) + \cdots + a_0 y(s) \\ &\quad = b_m\left(s^m u(s) - (s^{m-1}u(0) + \cdots + u^{(m-1)}(0))\right) + \cdots + b_0 u(s) \end{aligned} \tag{A.11}$$

so the output is given by:

$$y(s) = \frac{b_m s^m + b_{m-1}s^{m-1} + \cdots + b_0}{a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0}u(s) + \frac{\Psi(y(0), \ldots, y^{(n-1)}(0))}{a_n s^n + a_{n-1}s^{n-1} + \cdots + a_0} \tag{A.12}$$

where the rational function multiplying $u(s)$ is denoted as the *transfer function*, $G(s)$. Note that its denominator is the characteristic polynomial (A.6). In Laplace transform, for zero initial conditions, the output of a SISO linear system can be expressed as:

$$y(s) = G(s)u(s) = \frac{n(s)}{d(s)}u(s) \tag{A.13}$$

Once $y(s)$ is found by replacing the actual value of the input $u(s)$, the inverse Laplace transform obtains the solution in the time domain. For details, see classical textbooks such as [94, 78].

A.2.1 System Analysis

Poles and zeros. The roots z_i of $n(s) = 0$ are named *zeros* of the system, so $G(z_i) = 0$. Under some conditions, the system's response to $e^{z_i t}$ is null.

The roots of $d(s) = 0$ are the *poles*, p_i, of the system, and they are the coefficients of the exponentials of the free response, and $G(p_i) = \infty$. There are a finite number of poles and zeros.

Complex poles yield the time-response $e^{(a\pm bj)t}$, that can be cast in the form $e^{at}(M\sin bt + N\cos bt)$, $M, N \in \mathbb{R}$. A pair of complex poles are the roots of the factor $(s^2 + 2\xi\omega_n s + \omega_n^2)$, where $-1 < \xi < 1$ is denoted as damping ratio, and ω_n is the natural frequency.

Stability. If *all* the poles have a strictly negative real part, the system is said to be stable. Otherwise, it is said to be unstable. Unstable systems with no poles in the open positive half-plane are said to be marginally unstable.

Gain. When a *stable* system with transfer function $G(s)$ is subject to a constant input, U, the final value achieved is $Y = G(0)U$.

Settling time. The exponential $e^{-\alpha t}$ diminishes to 5% of its initial value in $\frac{3}{\alpha}$ time units. It diminishes to 2% in $\frac{4}{\alpha}$ units. These figures are denoted as the *settling time* of a first-order system. The value $\tau = 1/\alpha$ is usually denoted as the *time constant*.

The settling time of a higher-order system can be approximated by the settling time associated with the slower pole (dominant pole) in some cases. Multiple poles have a settling time coarsely given by $(1 + m)/2$ times the settling time of the single pole, where m is the root multiplicity.

Impulse response. If a system with transfer function $G(s)$ is subject to an impulse input, $u(t) = \delta(t)$, where $\delta(t)$ is Dirac's delta function (*i.e.*, $\delta(t) = 0$ for all $t \neq 0$, and $\int_{-\infty}^{\infty} \delta(t)\, dt = 1$, $\delta(s) = 1$), its output is given by:

$$y(s) = G(s); \qquad y(t) = \mathcal{L}^{-1}[G(s)] = g(t) \tag{A.14}$$

Convolution. A system with impulse response $g(t)$, with null initial conditions and input $u(t)$, has a time-response given by:

$$y(t) = \int_0^t g(t - \tau)u(\tau)\, d\tau \tag{A.15}$$

A.2.2 Frequency response

The stationary output of a linear system, with transfer function $G(s)$, subject to a sinusoidal input, $u(t) = \sin \omega t$, is given by:

$$y(t) = |G(j\omega)| \sin\left(\omega t + \arg G(j\omega)\right) \tag{A.16}$$

i.e., the formal argument in $G(s)$ is substituted by the complex frequency $j\omega$, and its modulus and argument are evaluated. The complex number $G(j\omega)$ is denoted as the *frequency response*.

Frequency response is usually plotted in a Bode diagram. The amplitude diagram represents $y = 20 \log_{10} |G(j\omega)|$ as a function of $x = \log_{10} \omega$. The phase diagram represents $\arg G(j\omega)$ as a function of $\log_{10} \omega$. There are techniques for fast drawing of the amplitude diagram. In particular, a pole in $\pm a$ "bends" downwards the diagram at $\omega = a$, denoted as the *break-down* frequency, with a slope of -20 units. A zero bends the diagram upwards.

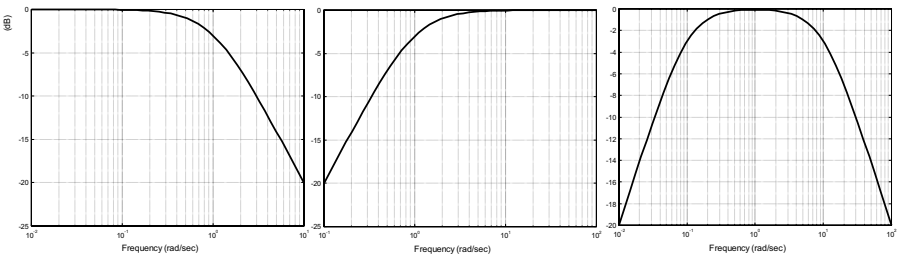

Figure A.1. Typical frequency response shapes (low-, high- and band-pass)

Filters. In filtering and frequency weighting, some standard shapes in frequency are widely used. These are the following:

Low-pass filter	High-pass filter	Band-pass filter
$\frac{A}{(\frac{1}{\omega_c}s+1)^n}$	$\frac{As^n}{(s+\omega_c)^n}$	$\frac{As^n}{(\frac{1}{\omega_l}s+1)^n(s+\omega_u)^n}$

where ω_c, ω_l and ω_u are the *cut-off frequencies*.
In Figure A.1, a frequency-response diagram of the above filters is plotted. The cut-off frequencies are 10^0 rad/s for the low- and high-pass filters, and 10^{-1} and 10^1 for the band-pass one. There are many other filter design alternatives. The reader is referred to basic references for details.

Usually, closed-loop transfer functions (sensitivity, (4.5), and complementary sensitivity, (4.6)) have approximate high-pass and low-pass behaviour respectively. The cut-off frequencies of those functions is denoted as the *closed-loop bandwidth*.

A.3 Discrete Systems

Linear time-invariant discrete-time systems are defined by difference equations of the form:

$$a_n y_{k+n} + a_{n-1} y_{k+n-1} + \cdots + a_1 y_{k+1} + a_0 y_k = b_0 u_k + \cdots + b_m u_{k+q} \quad \text{(A.17)}$$

If the sequence of inputs and outputs is obtained by sampling the actual sequence fed to a CT system, the system is said to be a *sampled-data* system, and each k corresponds to time KT_s, T_s being the *sampling period*.

With $u = 0$ and non-zero initial conditions, the free response is a set of *geometrical progressions*:

$$y(k) = \sum_{i=1}^{n} P_i(k) {\lambda_i}^k \quad \text{(A.18)}$$

$$P_i(k) = m_{i0} + m_{i1}k + \ldots \quad \text{(A.19)}$$

where λ_i are the roots of the *characteristic polynomial*:

$$a_n z^n + a_{n-1} z^{n-1} + \cdots + a_1 z + a_0 \quad \text{(A.20)}$$

$\mathcal{Z}$-transform. The $\mathcal{Z}$-transform can be used to solve linear difference equations. The inverse transform can be evaluated as an integral formula or, equivalently, by partial fraction expansion and table look-up.

Among its essential properties are:

- linearity: $\mathcal{Z}(f_k + g_k) = \mathcal{Z}(f_k) + \mathcal{Z}(g_k)$
- advance: f_0, f_1, ... being initial conditions,

$$\mathcal{Z}(f_{k+1}) = z(\mathcal{Z}(f_k) - f_0) \tag{A.21}$$

$$\mathcal{Z}(f_{k+n}) = z^n(\mathcal{Z}(f_k) - \sum_{i=0}^{n-1} z^{n-1-i} f_i) \tag{A.22}$$

- final value theorem. If the left limit in the expression below exists, then:

$$\lim_{k\to\infty} f_k = \lim_{z\to 1}(z-1)f(z) \tag{A.23}$$

The $\mathcal{Z}$-transform of the difference equation (A.17) can be shown to be:

$$y(z) = \frac{b_m z^m + b_{m-1}z^{m-1} + \cdots + b_0}{a_n z^n + a_{n-1}z^{n-1} + \cdots + a_0} u(z) + M(y_0, y_1, \ldots, y_{n-1}) \tag{A.24}$$

where M is an initial condition term. Applying partial fraction expansion, a solution in the discrete time domain is obtained. Analogous to the CT case, the expression multiplying $u(z)$ is denoted as the *discrete transfer function*, $G(z) = n(z)/d(z)$.

A.3.1 System Analysis

Similar to the CT case, an analysis of the discrete transfer function properties can be carried out.

Poles and zeros. The roots, z_i, of $n(z) = 0$ are named the *zeros* of the system. Under some conditions, the system's response to ${z_i}^k$ is null.

The roots of $d(z) = 0$ are the *poles* of the system, and they are the coefficients λ_i of the progressions of the free response. Complex poles of module α and argument ϕ yield a time-response that can be cast in the form $\alpha^k(M \sin \phi k + N \cos \phi k)$.

Stability. If *all* the poles have a module strictly less than 1, the system is said to be stable. Otherwise it is said to be unstable. Unstable systems with all poles in the closed unity circle are said to be marginally unstable.

Gain. When a stable system with discrete transfer function $G(z)$ is subject to a constant input, U, the final value achieved is $Y = G(1)U$.

Settling time. To define the settling time, again, we realise that the sequence α^k diminishes to 5% of its initial value in $\frac{-3}{\log \alpha}$ time units, and it diminishes to 2% in $\frac{-4}{\log \alpha}$ units. If the system is a sampled-data one, the figures should be multiplied by the sampling period to obtain a settling time in the appropriate units.

Impulse response. The impulse response sequence of a DT system, $\{g_k\}$, is the output to sequence $\delta_k = \{1, 0, 0, \dots\}$. As in the CT case, as the sequence's $\mathcal{Z}$-transform is 1, the impulse response is the inverse transform of the transfer function. There is also a DT convolution formula:

$$y_k = \sum_{i=0}^{k} g_{k-i} u_i \tag{A.25}$$

Frequency response. In sampled-data systems, frequency response is evaluated as $G(e^{j\omega T_s})$, *i.e.*, replacing the formal argument in the transfer function by $e^{j\omega T_s}$. Similar conclusions to CT systems can be made. For details, see [95].

A.4 Experimental Modelling

Once we know how to characterise the behaviour of a model, we are in a position to estimate approximate models of our process by comparing its behaviour with that of prototype models. Experimental modelling relies on the data gathered from the process. Thus, if the experiments are not run adequately the data will be useless and the models will not represent the process behaviour.

Experimental modelling (system identification and parameter estimation) is a crucial part of control system design and there is a natural interplay between the modelling and the control design stages. There is a lot of literature (and many good books and references) on this wide subject. The basic approaches are just mentioned, their detailed treatment being out of the scope of this book.

Matching of the Temporal Response

In process control, most processes behave in an overdamped way, and their step response (reaction curve) has the shape shown in Figure A.2. First proposed by Strej and later on with many variations (see, for instance, [89]), this kind of response can be approximated by a four-parameters model such as:

$$G(s) = k \frac{e^{-Ts}}{(1 + \tau s)^n}$$

where:

– k is the static gain,

– T is the time delay,

– n and τ being the order and time constant of the undelayed part respectively, selected from the following table:

n	T_a/τ	T_u/τ	T_u/T_a
1	1	0	0
2	12.7	0.28	0.104
3	3.7	0.8	0.22
4	4.46	1.42	0.32
5	5.12	2.1	0.41
6	5.7	2.8	0.49

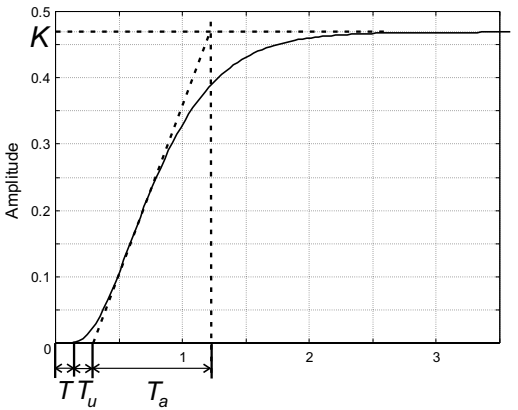

Figure A.2. Overdamped unit-step response (reaction curve)

Based on the measurement of T_u/T_a, the time constant order is approximated and the delay is tuned to match the second and third columns.

In a MIMO system, this experiment can be done for each input, keeping the rest constant. Small step increments are used to guarantee a linear behaviour, however, too-small ones will have a reduced signal-to-noise ratio.

If the process is underdamped, other higher-order models should be considered [89].

Matching the Frequency Response

Data gathered from a frequency analysis, carried out by entering at each input a sinusoidal signal, will allow us to plot the frequency response (Bode diagram) of each element of the frequency response matrix. By approximation of the cut-off frequencies and the time delay, a model similar to the previous one can be estimated. This technique is suitable for faster processes (electromechanical systems, for instance).

Parameter Estimation

From input/output data taken under arbitrary (in many cases, purposely random) inputs, least square parameter estimation approaches are quite popular.

An academic example will illustrate some ideas in one of the simplest cases. The reader is referred to [84] for in-depth coverage of this important topic.

Example A.1. A record of open-loop operation of a particular plant showed the following input/output sequences ($u_k = y_k = 0$ for $k < 0$):

$$\{u_k\} = \{1, -1, 0, 2, -2, -1, -1, 0, \dots\}$$
$$\{y_k\} = \{0, 0.05, 0.13, -0.037, -0.19, -0.51, \dots\}$$

A so-called ARX process model, in the form

$$y_k = a_1 y_{k-1} + a_2 y_{k-1} + b_1 u_{k-1} + b_2 u_{k-2} + \varepsilon_k \tag{A.26}$$

where ε is assumed to be zero-mean white noise, will be fitted to the data. The criteria would be minimising the average squared difference between the model prediction and the actual output measurement y_k. So, the following equations need to be solved in a least-squares sense:

$$\begin{aligned} 0.05 &= 0a_1 + 0a_2 + 1b_1 + 0b_2 \\ 0.13 &= 0.05a_1 + 0a_2 - 1b_1 + 1b_2 \\ -0.037 &= 0.13a_1 + 0.05a_2 - 0b_1 - 1b_2 \\ -0.19 &= -0.037a_1 + 0.13a_2 - 2b_1 - 0b_2 \\ &\vdots \end{aligned}$$

Expressing the equations in matrix form and using the pseudoinverse, the procedure is straightforward. For higher quantities of data, MATLAB® *System Identification* toolbox provides the commands to automatically carry out the above operations. In this case, being u and y column vectors with the data sequences, the code returning the model would be:

```
th=arx([y u],[2 2 1]); present(th);
```

where two denominator parameters (a_i), two numerator ones (b_i) are to be estimated, and one sample pure delay is assumed in the model. The result is: $a_1 = -0.925$, $a_2 = 0.289$, $b_1 = 0.0591$ and $b_2 = 0.143$. Other MATLAB® commands carry out the identification of the input-output models discussed in page 46, in particular: `armax, oe, bj`.

An important issue is, of course, how to determine the distubance model (model class), order and delay of the underlying process, *a priori*. Of course, the more parameters the model has the better the fit to the training data is. However, too many parameters will also fit spurious noise, yielding an useless model. With real data, several combinations are tried and the number of parameters involved is traded off against the accuracy achieved on a second, independent data set, denoted as the *test set*, striking a suitable compromise.

Data preparation [26, 84] is also an important issue for applicability of these algorithms in practice: using suitable band-pass filtered input signals (and/or also pre-filtering the input and output sequences prior to application of the ID algorithms) maximises signal-to-noise ratio at the key frequencies regarding robustness [112], ensuring a better fit there and thus improving the likelihood of success of the final controller on the real plant.

A.5 Tables of Transforms

$f(t)$	Laplace Transf.	$f(t)$	Laplace Transf.
$\delta(t)$	1	$\sin \omega t$	$\frac{\omega}{s^2+\omega^2}$
1	$\frac{1}{s}$	$\cos \omega t$	$\frac{s}{s^2+\omega^2}$
t	$\frac{1}{s^2}$	$e^{-at} \sin \omega t$	$\frac{\omega}{(s+a)^2+\omega^2}$
e^{-at}	$\frac{1}{s+a}$	$e^{-at} \cos \omega t$	$\frac{s+a}{(s+a)^2+\omega^2}$
$\frac{1}{(n-1)!} t^{n-1} e^{-at}$	$\frac{1}{(s+a)^n}$		

Sequence f_n	$\mathcal{Z}$-Transform	f_n	$\mathcal{Z}$-Transform
$\{1, 0, 0, \ldots\}$	1	$\cos(Bn)$	$\frac{1-z^{-1}\cos(B)}{1-2z^{-1}\cos(B)+z^{-2}}$
$\{0, \ldots, 0, 1, 0, \ldots\}$	z^{-k}	$\sin(Bn)$	$\frac{z^{-1}\sin(B)}{1-2z^{-1}\cos(B)+z^{-2}}$
$\{1, 1, 1, 1, \ldots\}$	$\frac{1}{1-z^{-1}}$	$a^n \cos(Bn)$	$\frac{1-z^{-1}a\cos(B)}{1-2az^{-1}\cos(B)+a^2z^{-2}}$
$\{0, T, 2T, 3T, 4T, \ldots\}$	$\frac{Tz^{-1}}{(1-z^{-1})^2}$	$a^n \sin(Bn)$	$\frac{az^{-1}\sin(B)}{1-2az^{-1}\cos(B)+a^2z^{-2}}$
$\{1, a, a^2, a^3, \ldots\}$	$\frac{1}{1-az^{-1}}$		

B

Matrices

In this appendix, some definitions and properties regarding matrices are outlined. For in-depth analysis, the reader is referred to textbook algebra sources such as [91, 125].

B.1 Column, Row and Null Spaces

Definition B.1 (Column space $C(A)$). *The column space of a matrix $A_{m\times n}$ is the set of all linear combinations of its columns,* i.e.*, the set of all $y \in \mathbb{R}^m$ expressed as $y = Ax$ where x is a vector of n arbitrary coefficients. The column space is a subspace of $\mathbb{R}^m$.*

Definition B.2 (Row Space). *The set of linear combinations of the rows of $A_{m\times n}$ is a subspace of $\mathbb{R}^n$, and it is isomorph to the column space of A^T,* i.e.*, $C(A^T)$.*

Definition B.3 (Null Space $N(A)$). *It is defined as the set of y such that $(A_{m\times n})y_{n\times 1} = 0$.*

Note that the null space is the set of vectors orthogonal to all rows of A (columns of A^T): $N(A) \perp C(A^T)$. Also $C(A) \cup N(A^T) = \mathbb{R}^m$.

Systems of equations. Solutions to $Ax = b$, for $b \in C(A)$, are in the form $x_0 + x_H$, where x_0 is a *particular* solution and x_H is any vector in $N(A)$ (solution of $Ax = 0$). If $b \notin C(A)$ the system of equations has no solution.

Definition B.4 (Rank). *The dimension of the subspace $C(A)$ is denoted as* rank *of matrix A. It is equal to the dimension of the row space and to the dimension of the biggest nonzero*[1] *minor (determinant of square submatrix).*

[1] This definition is not useful for numeric calculations. See Section B.4.1 for a computational definition.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `null,orth,rank,det`.

B.2 Matrix Inversion

Inverse. A square matrix A with nonzero determinant allows the calculation of its inverse, A^{-1}:

$$A^{-1} = \frac{1}{\det A}\text{adj}A \qquad (\text{adj}A)_{ij} = (-1)^{i+j}\det A^{ji} \tag{B.1}$$

where A^{ji} is a submatrix formed by deleting the row j and the column i from A. The determinant of A^{-1} is $1/\det(A)$. These matrices are called *invertible* or *regular*, and $AA^{-1} = A^{-1}A = I$.

Unitary (orthogonal) matrices. A matrix, U, is unitary if it verifies:

$$U^H = U^{-1}$$

where superscript ^{H} stands for complex conjugate transpose (transpose in the real case). In that case, all rows have norm 1 and are orthogonal to the rest[2].

Pseudoinverse

Left pseudoinverse. If a matrix $A_{m\times n}$ with $m \geq n$ has rank n (full column rank), the matrix:

$$^\dagger A = (A^T A)^{-1} A^T \tag{B.2}$$

fulfills:

$$^\dagger A A = I_{n\times n}$$

Matrix $^\dagger A$, of dimensions $n \times m$, is denoted as the left pseudoinverse of A.

Given a system of equations, $Ax = y$, under full-column rank assumptions, the value of x minimising the least squares error criteria $J = \|Ax - y\|_2$ (see norm definitions in Appendix C) with more equations than unknowns is:

$$x_{opt} = {}^\dagger A y$$

as shown in Example D.2 on page 304.

[2] In the following, mostly matrices with real elements will be considered. To extend the results to the complex case, any transposition must be replaced by conjugate-transpose.

Right pseudoinverse. If a matrix $A_{m\times n}$ with $m \leq n$ has rank m (full row rank), matrix:

$$A^{\dagger} = A^T(AA^T)^{-1} \tag{B.3}$$

fulfills:

$$AA^{\dagger} = I_{m\times m}$$

So matrix $A^{\dagger}$, with dimensions $n \times m$, is denoted as right pseudoinverse.

In this case, a system, $Ax = y$, has more unknowns than equations thus existing infinite solutions. The solution with minimum norm is:

$$x_{opt} = A^{\dagger}y$$

If $A_{m\times n}$ has rank less than $\min(m, n)$, no pseudoinverse exists.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `inv`,`pinv`.

Schur formula. The following formula is useful in calculating determinants of partitioned matrices:

$$\det\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix} = \det(A_1)\det(A_4 - A_3A_1^{-1}A_2) = \det(A_4)\det(A_1 - A_2A_4^{-1}A_3) \tag{B.4}$$

Push-through rule. For matrices of appropriate dimensions:

$$G_1(I - G_2G_1)^{-1} = (I - G_1G_2)^{-1}G_1 \tag{B.5}$$

B.3 Eigenvalues and Eigenvectors

The eigenvalues of a square matrix $A_{n\times n}$ are the solutions of the equation:

$$\det(A - \lambda I) = 0$$

where the left-hand side is a polynomial in λ, denoted as the characteristic polynomial of A, and the equation above is named *characteristic equation* of A. There exist n (possibly repeated) eigenvalues. Diagonal and triangular matrices have the diagonal entries as eigenvalues.

Eigenvectors, v_i, corresponding to eigenvalue λ_i verify $Av_i = \lambda_i v_i$. Eigenvectors for different eigenvalues are linearly independent. That may not be the case for repeated eigenvalues. If there are n linearly independent eigenvectors, a matrix, T, can be formed by juxtaposing them in column form so that:

$$AT = T\Lambda$$

where Λ is a diagonal matrix, so $\Lambda = T^{-1}AT$. In that case, it is said that A is diagonalisable. If matrix T cannot be formed with independent eigenvalues, the so-called Jordan upper triangular canonical form can be obtained.

A matrix A is *positive-definite* if all its eigenvalues are real and positive (then $x^T Ax > 0$ for all $x \neq 0$), and *positive semi-definite* if they are ≥ 0.

Similarity transform. $B = T^{-1}AT$ has the same eigenvalues as A, for any invertible T. Indeed:

$$\det(\lambda I - T^{-1}AT) = \det(T^{-1}(\lambda I - A)T) = \det(\lambda I - A)$$

Cayley-Hamilton theorem. A square $n \times n$ matrix satisfies its own characteristic equation, replacing powers of λ by powers of A, *i.e.*, if $\det(sI - A) = \sum_{i=0}^{n} \alpha_i s^i$, then $\sum_{i=0}^{n} \alpha_i A^i = 0$.

From this expression, $A^n = \alpha_n^{-1} \sum_{i=1}^{n-1} \alpha_i A^i$. So, by induction, A^p for any $p \geq n$ can be expressed as a linear combination of I, A, A^2, A^{n-1}.

Other Properties

- the product of the eigenvalues is equal to the determinant,
- the sum of the eigenvalues is the trace of A (sum of diagonal elements),
- the matrix $A + cI$, c being a real or complex scalar, has as eigenvalues $\lambda_i + c$, and the same eigenvectors.

Gershgorin's theorem. The eigenvalues of an $n \times n$ matrix lie, in the complex plane, in the union of the closed disks with centre a_{ii} (diagonal element) and radius $\sum_{j \neq i} |a_{ij}|$ (absolute row sum). Also, they lie in the union of closed disks with same centre and radius $\sum_{j \neq i} |a_{ji}|$ (absolute column sum). The Gershgorin theorem gives numerical indexes of *diagonal dominance* (3.37) (see also Section 5.2.3).

Eigenvalues and differential equations. Let us consider a system of ordinary linear differential equations:

$$\frac{dx}{dt} = Ax \tag{B.6}$$

If we look for an exponential component in the response, as in scalar ordinary equations, $x(t) = e^{\lambda t}v$, then $\dot{x} = e^{\lambda t}\lambda v$, hence, as $e^{\lambda t}$ is a non-zero scalar for every t:

$$e^{\lambda t}\lambda v = Ae^{\lambda t}v \quad \Rightarrow \quad \lambda v = Av$$

so λ must be an eigenvalue of A and v must be an eigenvector[3] : *the exponential terms in solutions of* (B.6) *are the eigenvalues of A.*

[3] The same happens for a DT system, $x_{k+1} = Ax_k$, and a solution in the form $x_k = \lambda^k v$.

B.4 Singular Values and Matrix Gains

The problem of determining the maximum and minimum "gain" of a matrix is related to an eigenvalue problem, when working with Euclidean vector norms (see Appendix C).

Given $y = Ax$, to determine the maximum and minimum gains, and the directions in which they occur, $\|x\| = 1$ will be assumed (as the effect of the matrix for any x will be proportional to that on its unit vector $x/\|x\|$). So, the problem is stated as follows:

- maximise $\|y\|^2 = \|Ax\|^2 = x^T A^T Ax$, for $x \in \mathbb{R}^n$,
- subject to the restriction $x^T x = 1$

The constrained optimisation problem (see Section D.1) can be transformed into an unconstrained one by using one Lagrange multiplier λ:

$$J = x^T A^T Ax + \lambda(1 - x^T x) \tag{B.7}$$

So, differentiating with respect to x (using the notation in (D.1)), the result is the eigenvalue problem:

$$A^T Ax = \lambda x \tag{B.8}$$

So the maximum and minimum gains occur at some of the *eigenvectors* of $A^T A$, and the eigenvalues are the associated Lagrange multipliers. As $A^T A$ is symmetric, positive semi-definite ($\|Ax\| = x^T A^T Ax \geq 0$), all its eigenvalues are *real*, non-negative and the eigenvectors can be chosen to be an *orthonormal* basis[4].

It can be shown that for any eigenvector x_i, $i = 1, \ldots, n$ of $(A^T A)$, the output norm $\|y_i\| = \|Ax_i\|$ attained is $\sqrt{x_i^T \lambda_i x_i} = \sqrt{\lambda_i}\|x_i\|$. So, the maximum gain corresponds to the eigenvector associated to the maximum eigenvalue, and the analogue for the minimum one. Let us denote as $\lambda_1 \geq \lambda_2 \geq \lambda_n$ the eigenvalues in decreasing sequence, so x_1 is the most amplified input direction and x_n is the least.

The intermediate vectors are *saddle* points on the optimisation problem. Furthermore, λ_2 maximises the gain of A subject to $x^T x = 1$ and $x^T x_1 = 0$ (the maximum gain on directions orthogonal to the x_1). Carrying on, λ_i maximises the gain of the matrix in directions orthogonal to all x_j for $j < i$.

As the eigenvectors form an orthonormal basis, also the outputs $y_i = Ax_i$ and $y_j = Ax_j$ are orthogonal as $y_i^T y_j = x_i^T A^T Ax_j = x_i \lambda_j x_j = 0$.

So, the action of A on an arbitrary x can be decomposed as:

1. Decompose (project) x on the basis formed by the eigenvectors x_i. Let us denote as V the matrix whose rows are x_i.
2. Scale each component by $\sqrt{\lambda_i}$. If $\lambda_i = 0$, the component is irrelevant.

[4] Multiplication by an orthonormal matrix can be considered as a change of basis by rotation.

3. Rotate the result to be aligned with y_i and add up all components.

The result is that matrix A can be written as:

$$A = U\Sigma V^T \tag{B.9}$$

where U and V are orthonormal rotation matrices and Σ is diagonal. Furthermore, if A is $m \times n$, the columns of V $(n \times n)$ are the eigenvectors of $A^T A$, Σ $(m \times n)$ contains the (positive) square root of the eigenvalues ($\sigma_i = \sqrt{\lambda_i}$ are denoted as *singular values*) in the diagonal, and columns of U $(m \times m)$ are the eigenvectors of AA^T (by a similar argument with $A^T = V\Sigma U^T$). As the determinants of U and V are 1, $\det(A) = \det(\Sigma)$.

Equation (B.9) is denoted as the singular value decomposition (SVD). Usually, the maximum and minimum singular values are written as:

$$\overline{\sigma}(A) = \lambda_1; \qquad \underline{\sigma}(A) = \lambda_n$$

so the following inequalities hold:

$$\underline{\sigma}(A)\|x\| \leq \|Ax\| \leq \overline{\sigma}(A)\|x\| \tag{B.10}$$

As an orthogonal (rotation) matrix has its transpose as the inverse, the inverse of a matrix A can be expressed as:

$$A^{-1} = V\Sigma^{-1}U^T \tag{B.11}$$

so its singular values are the inverse of those of A. In particular:

$$\overline{\sigma}(A^{-1}) = 1/\underline{\sigma}(A) \tag{B.12}$$

so a matrix is invertible if all its singular values are non-zero. Left and right pseudoinverses can also be expressed in terms of the SVD of A.

The maximum singular value is a *matrix norm*, in the sense of the definitions of Section C.2. It satisfies some norm properties, in particular:

$$\overline{\sigma}(AB) \leq \overline{\sigma}(A)\overline{\sigma}(B) \tag{B.13}$$

so, unless otherwise stated, the following notation will be assumed:

$$\|A\| = \overline{\sigma}(A)$$

The minimum gain, $\underline{\sigma}()$, satisfies $\underline{\sigma}(A)\underline{\sigma}(B) \leq \underline{\sigma}(AB)$.

B.4.1 Condition number

Based on the results above, the *condition number* of a matrix A is defined as:

$$\gamma(A) = \frac{\overline{\sigma}(A)}{\underline{\sigma}(A)} \geq 1$$

From (B.12), it is straightforward to prove that $\gamma(A^{-1}) = \gamma(A)$.

Let us now analyse some important singular value applications.

Noise amplification. In a system $Ax = y$, where A is a regular matrix and y are noise-corrupted experimental data, an estimate on the relative accuracy of $x = A^{-1}y$ can be obtained.

Let us denote as δy the "error" (noise) present on the experimental data. Then, $\delta x = A^{-1}\delta y$ and, based on (B.10):

$$\|\delta x\| \leq \bar{\sigma}(A^{-1})\|\delta y\|; \qquad \|x\| \geq \underline{\sigma}(A^{-1})\|y\|$$

so dividing the first inequality by $\|x\|$:

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\bar{\sigma}(A^{-1})\|\delta y\|}{\|x\|} \leq \frac{\overline{\sigma}(A^{-1})}{\underline{\sigma}(A^{-1})}\frac{\|\delta y\|}{\|y\|} = \gamma(A^{-1})\frac{\|\delta y\|}{\|y\|} \tag{B.14}$$

The interpretation of this formula is that the relative error on the experimental data may have a worst-case amplification given by the condition number. A similar argument would prove that relative error in x may appear in y amplified by the factor $\gamma(A)$.

Modelling error. In this case, the issue to be considered is the difference in solutions for u from a nominal model, $y = Gu$, and a perturbed one, $y = (G+\delta G)u^*$, denoting as δu the difference between solutions, $u^* = (u+\delta u)$. Subtracting these expressions, $0 = Gu - (G + \delta G)(u + \delta u)$, results in:

$$\delta u = -G^{-1}\delta G(u + \delta u)$$

so, using (B.13) and (B.12):

$$\|\delta u\| \leq \overline{\sigma}(G^{-1})\overline{\sigma}(\delta G)\|u + \delta u\| = \frac{\overline{\sigma}(\delta G)}{\underline{\sigma}(G)}\|u + \delta u\| = \frac{\overline{\sigma}(G)}{\underline{\sigma}(G)}\frac{\overline{\sigma}(\delta G)}{\overline{\sigma}(G)}\|u + \delta u\|$$

so relative modelling error also may get amplified in the results by $\gamma(G)$:

$$\frac{\delta u}{\|u + \delta u\|} \leq \gamma(G)\frac{\|\delta G\|}{\|G\|} \tag{B.15}$$

Rank determination. One application of the above result is that, in matrices obtained from noisy or unaccurate experimental data (even round-off errors), no matrix has "exactly" zero determinant or zero singular values.

To determine in practice the reliability of a matrix inversion, the condition number is calculated. If it is very large, it implies that the matrix inversion is unreliable (very high error figures may be expected, from (B.14) and (B.15)).

It can be easily shown that the rank of a matrix is the number of non-zero singular values. This is an operational definition where a *tolerance* can be specified, on the minimum singular value or on the condition number.

Example B.5. This is an example to show that the value of the determinant, by itself, is not a good indicative of how close the matrix is to being singular:

```
A = -38.8843   44.4703   92.1813
      1.8504   61.5432   73.8207
     82.1407   79.1937   17.6266
 det(A) = 857.5811
 svd(A) =  153.2032  105.2887  0.0532
```

Condition number is around 3000, *i.e.*, in some directions, the gain is almost three thousand times larger than in other ones so, depending on the reliability of the data, the "practical" rank might be 2. Any control design philosophy relying on that matrix having rank 3 will be quite sensitive to modelling errors. If element $A(1,1)$ were -38.7042, the matrix would become singular: the allowable relative error for singularity is 0.003 in some elements. This matrix is "singular" (rank 2) if data do not have more than three precise significant figures.

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `svd,cond,condeig`.

For further detail on these topics, the reader is referred to [125, 53, 91].

Minimised condition number. In MIMO transfer function matrices, as diagonal scalings represent changes of units in physical magnitudes, unless all variables are suitably scaled to have the same practical meaning, the condition number by itself may be a misleading indicator of the "difficulty" of the matrix inversion problem. This is the reason why it is important in some applications to calculate the minimised condition number:

$$\gamma^*(G) = \min_{D_I, D_O} \gamma(D_O G D_I) \tag{B.16}$$

where D_O and D_I are output and input diagonal scalings respectively. It indicates the practical conditioning for a particular optimal set of measurement units. Its determination requires convex optimisation routines, related to μ-analysis (structured uncertainty robust control) [32]. In a practical case, evaluation of the condition number of a plant must be carried out once it has been suitably scaled (see Section 3.5).

Example B.6. A plant with a diagonal transfer matrix is indeed easy to control, as there is no coupling. However, the condition number of `G=diag([1,1000])`, equal to 1000, unacceptably overestimates the sensitivity to uncertainty in (B.14) and (B.15): a trivial change of units yields a plant with $\gamma = 1$, *i.e.*, $\gamma^*(G) = 1$.

Column and Null spaces. If all singular values below a tolerance limit are assumed to be zero, then the matrix SVD can be understood as:

$$[U_1 U_2] \begin{pmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{pmatrix} [V_1 V_2]^T$$

where Σ_1 contains the non-zero singular values. In this case, V_2 is a basis of the *null* space, and U_1 is a basis of the *column* space. This is a numerically reliable way of calculating such spaces, used in MATLAB®.

B.5 Matrix Exponential

The definition of a matrix exponential is the same as in the exponential of an scalar:

$$\exp(A) = e^A \stackrel{\text{def}}{=} \sum_{n=0}^{\infty} \frac{A^n}{n!} = I + A + \frac{A^2}{2} + \frac{A^3}{6} + \cdots \tag{B.17}$$

In solving linear differential equations, matrices are multiplied by a time scalar:

$$\exp(At) = e^{At} = \sum_{n=0}^{\infty} \frac{A^n t^n}{n!} = I + At + \frac{A^2 t^2}{2} + \frac{A^3 t^3}{6} + \cdots \tag{B.18}$$

Properties. The matrix exponential has the following properties:

- $e^{0_{n\times n}} = I_{n\times n}$
- $e^{At_1} e^{At_2} = e^{A(t_1+t_2)}$
- $\left(e^{At}\right)^{-1} = e^{-At}$
- $\frac{de^{At}}{dt} = Ae^{At} = e^{At}A$
  Indeed,

$$\frac{de^{At}}{dt} = \sum_{n=1}^{\infty} \frac{A^n\, nt^{n-1}}{n!} = A + \frac{A^2\, 2t}{2} + \cdots = A(I + \frac{A^2\, t}{1} + \frac{A^3\, t^2}{2} + \cdots)$$
$$= \sum_{n=1}^{\infty} \frac{A^n\, t^{n-1}}{(n-1)!} = A \sum_{n=0}^{\infty} \frac{A^n t^n}{n!} = (\sum_{n=0}^{\infty} \frac{A^n t^n}{n!})A = Ae^{At} = e^{At}A \tag{B.19}$$

- based on the previous property, if A is non-singular, $\int_0^t e^{A\tau}\, d\tau = A^{-1}(e^{At} - I)$. The integral also exists for singular A, but it must be carried out with the power series.

Differential equations. The solution of the equation:

$$\dot{x} = Ax + Bu \tag{B.20}$$

can be expressed in terms of a matrix exponential, as:

$$\frac{d}{dt}(e^{-At}x) = e^{-At}\dot{x} - e^{-At}Ax = e^{-At}Bu$$
$$e^{-At}x(t) - e^{-A0}x(0) = \int_0^t e^{-A\tau}Bu\, d\tau$$

so, as $e^0 = I$, multiplying by e^{At} the result is:

$$x(t) = e^{At}x(0) + e^{At}\int_0^t e^{-A\tau}Bu\, d\tau = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu\, d\tau \tag{B.21}$$

Calculation. If the matrix is diagonal, the exponential is just the exponential of the diagonal elements:

$$\exp\left[\begin{pmatrix} c_1 & 0 \\ 0 & c_2 \end{pmatrix}\right] = \begin{pmatrix} e^{c_1} & 0 \\ 0 & e^{c2} \end{pmatrix} \tag{B.22}$$

If the matrix is diagonalisable, $A = V^{-1}\Lambda V$, as:

$$A^2 = V^{-1}\Lambda V V^{-1}\Lambda V = V^{-1}\Lambda^2 V; \quad \ldots \quad A^n = V^{-1}\Lambda^n V \tag{B.23}$$

it can be shown that:

$$e^{At} = V^{-1}e^{\Lambda t}V \tag{B.24}$$

If the matrix is not diagonalisable, it admits a canonical Jordan form. It can be shown that a Jordan block of dimension m has a matrix exponential given by:

$$exp(\begin{pmatrix} \lambda & 1 & 0 \ldots \\ 0 & \lambda & 1 \ldots \\ 0 \ldots & \lambda & 1 \\ 0 \ldots & 0 & \lambda \end{pmatrix}) = e^{\lambda t} \times \begin{pmatrix} 1 & t & \ldots & \frac{t^{m-1}}{(m-1)!} \\ 0 & 1 & \ldots & \frac{t^{m-2}}{(m-2)!} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \tag{B.25}$$

Computer calculation. Calculation of matrix exponentials via a computer must be made by taking a few terms of the power series defining the matrix exponential. However, the *Padé* approximation, $e^{At} = e^{At/2}\left(e^{-At/2}\right)^{-1}$, from (2.42), gives better results, evaluating $e^{At/2}$ and $e^{-At/2}$ by a truncated Taylor series expansion. Furthermore, as the series accuracy decreases for large values of t, the matrices are scaled down by a power of 2 until its maximum singular value is less than a prefixed amount. After calculation, repeated multiplication renders the final result (for example, $e^{At} = (e^{At/8})^8$).

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `expm,expm1,expm2,expm3,funm`.

B.6 Polynomial Fraction Matrices

Rational transfer function matrices have as elements quotients of polynomials $g_{ij} = \frac{n_{ij}(s)}{d_{ij}(s)}$ (polynomial fractions), with $d_{ij}(s) \neq 0$, where equality refers to the identically zero polynomial.

As polynomial fractions form a *field* (+ and $\cdot$ are commutative, associative, have neutral element and inverse), many results extend directly from those from the real matrices (as real numbers are also a field), understanding zero as the zero polynomial. *Polynomial matrices* (those with denominator equal to 1) have polynomials as elements (polynomials are a *ring*, subset of the polynomial fractions).

Inversion. A square matrix, $G(s)$, is invertible if $det(G(s)) \neq 0$, and its inverse has the same formula as in the real number case.

Equivalently, a polynomial fraction matrix is invertible if its determinant *evaluation* is non-zero except at a finite number of points (the roots of a non-zero polynomial).

Note that s is considered an arbitrary symbolic variable, and not a complex number where the matrix is evaluated. For example:

$$G(s) = \begin{pmatrix} \frac{1}{s+1} & \frac{s}{s+4} \\ \frac{2}{s+1} & 0.4 \end{pmatrix} \quad det(G) = \frac{1.6(1-s)}{(s+1)(s+4)}$$

$$G^{-1}(s) = \frac{(s+1)(s+4)}{1.6(1-s)} \begin{pmatrix} 0.4 & -\frac{s}{s+4} \\ -\frac{2}{s+1} & \frac{1}{s+1} \end{pmatrix}$$

are invertible as polynomial fraction matrices. However, interpreted as a function, $G(s) : \mathbb{C} \to \mathbb{C}$ is not defined for $s = -4$ or $s = -1$.

The *rank* of a polynomial fraction matrix is defined, as usual, as the size of the largest non-zero minor (as a polynomial). In this sense, the rank of $G(s)$ above is 2. Note that, however, the rank of the scalar matrix $G(1)$ is 1, as $\det(G(1)) = 0$. In this case, it is said that 1 is a *zero* of $G(s)$.

Unimodular matrices. The inverse of a polynomial matrix is, in general, a polynomial *fraction* matrix. From (B.1), only if its determinant is a *constant* polynomial (the only polynomials that have inverse) the inverse will also be a polynomial matrix. In this case, the matrix is said to be unimodular.

Coprime Factorisation

In order to analyse the stability, as well as for dealing with uncertainties and model reduction, it is interesting to express a transfer matrix by the product of two stable transfer matrices.

We denote by a *right coprime factorisation*, the product:

$$G(s) = N_r(s)\, M_r^{-1}(s) \tag{B.26}$$

where $N_r(s)$ and $M_r(s)$ are two coprime stable matrices, that is, there is no pole-zero cancellation between them, and both are stable, *i.e.*, it is required that all the RHP zeros of $G(s)$ should be in $N(s)$ and all its RHP poles in $M(s)$.

It can be proved that two matrices are *right coprime* if they satisfy the Bezout identity, *i.e.*, there exist stable $X(s)$ and $Y(s)$ such that:

$$X(s)N(s) + Y(s)M(s) = I$$

Similarly, a *left coprime factorisation* is expressed by the product:

$$G(s) = M_l^{-1}(s)\, N_l(s) \tag{B.27}$$

Please, note that, instead of the product of two polynomial matrices $(N(s), D(s))$, as used in the polynomial representation (2.35), here the factors are polynomial fraction matrices (transfer matrices).

Note that given any stable and minimum-phase transfer matrix, Q, a nominal plant $G = M^{-1}N$ can also be expressed as $G = (QM)^{-1}(QN)$, so the factorisation is not unique. The factorisation is called *normalised* (NCF) if:

$$M(s)M^T(-s) + N(s)N^T(s) = 1 \tag{B.28}$$

In the SISO case, this amounts to forcing the sum of squares of the modulus of "numerator" N and "denominator" M to 1, and it is a convenient choice for many problems. The quotes remark that M and N in an NCF are not polynomials.

C

Signal and System Norms

C.1 Normed Spaces

Definition C.1. *A norm is a real function $\|\cdot\| : \mathcal{V} \to \mathbb{R}_+$ where $\mathcal{V}$ is a vector space, satisfying the following conditions:*

1. $\|x\| \geq 0 \quad \forall x \in \mathcal{V}$.
2. $\|x\| = 0 \Leftrightarrow x$ *is the null vector* ($x = 0$).
3. $\|\alpha x\| = |\alpha| \cdot \|x\| \quad \forall x \in \mathcal{V}$ *and for any scalar* α *(real or complex,* $|\alpha|$ *being the absolute value or modulus).*
4. $\|x + y\| \leq \|x\| + \|y\| \quad \forall x, y \in \mathcal{V}$ *(triangle inequality).*

Examples of those norms are, in $\mathbb{R}^n$,

$$\|(a_1, \ldots, a_n)\|_p = \left(\sum_{i=1}^{n} |a_i|^p \right)^{\frac{1}{p}} \qquad p \geq 1$$

The norm $\|a\|_2 = a^T a$ is usually named the *Euclidean* norm[1].

Another example is the *maximum* norm:

$$\|(a_1, \ldots, a_n)\|_{max} = \max_i a_i$$

This norm is also usually denoted as $\|a\|_\infty$.

C.2 Function Spaces

Considering two normed vector spaces $(\mathcal{V}_a, \|\cdot\|_a)$ and $(\mathcal{V}_b, \|\cdot\|_b)$, the set of applications between V_1 and V_2 is also a vector space, defining $(f_1 + f_2)(x) = f_1(x) + f_2(x)$ and $(\alpha f)(x) = \alpha f(x)$.

[1] Vector a is assumed to be in column form.

Definition C.2. *In a function vector space, the norm of a continuous function $f : \mathcal{V}_a \to \mathcal{V}_b$ such that $f(0) = 0$ can be defined as:*

$$\|f\|_{ab} = \sup_{\|v\|_a \neq 0} \frac{\|f(v)\|_b}{\|v\|_a} \tag{C.1}$$

Remark C.3. The norm of a function is a measure of its maximum amplification (worst-case gain), as:

$$\|f(v)\|_b \leq \|f\|_{ab} \, \|v\|_a \tag{C.2}$$

and the norm can be interpreted as the size of the smallest circular "cone" containing $f(v)$ for all possible v. Non-linearities with finite norm are called *sector-bounded* non-linearities.

The above defined norms are also denoted as *induced norms*.

If f is *linear*, the norm definition is equivalent to:

$$\|f\|_{ab} = \sup_{\|v\|_a = 1} \|f(v)\|_b \tag{C.3}$$

Example C.4. The norm of function $y = u + \sin u$ is 2, as its graph is tangent to the cone $y = \pm 2u$. $\sqrt{|u|}$ does not have a finite norm.

Example C.5. Consider the set of linear functions between $\mathbb{R}^m$ and $\mathbb{R}^n$, both with the Euclidean norm, $\|\cdot\|_2$. Note that each linear function can be uniquely represented by a matrix, A, of dimensions $m \times n$. The norm of the function $y = Ax$ (termed the norm of matrix A, with a slight abuse of notation) is the maximum singular value defined in Section B.4:

$$\|A\| = \sup_{\|x\|=1} \|Ax\| = \overline{\sigma}(A) \tag{C.4}$$

Submultiplicative property. One important property of these norms in function spaces is the *submultiplicative* property regarding *composition*, $\circ$, of functions $f_1 : \mathcal{V}_b \to \mathcal{V}_c$, $f_2 : \mathcal{V}_a \to \mathcal{V}_b$:

$$\|f_1 \circ f_2\|_{ac} = \sup_{\|v\| \neq 0} \frac{\|f_1(f_2(v))\|_c}{\|v\|_a} \leq \sup_{\|v\| \neq 0} \frac{\|f_1\|_{bc} \|f_2(v)\|_b}{\|v\|_a} = \|f_1\|_{bc} \cdot \|f_2\|_{ab} \tag{C.5}$$

where the inequality comes from (C.2).

The composition of linear functions is equivalent to *matrix multiplication*. The above property is useful for evaluating bounds on the maximum gain of series-connected systems (Figure 2.3).

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `norm,lti/norm`.

C.3 Signals and Systems Norms

A particular application of some of the above definitions to signal spaces (functions of time) and dynamical systems (interpreted as functions between two signal spaces) allows the following definitions.

C.3.1 Signal Norms

Signals are defined as functions of time $f : \mathbb{R} \to R^n$. If a certain norm $\| \, \|$ on $\mathbb{R}^n$ is assumed, it can be proved that the following expression constitutes a norm in the space of signals[2]:

$$\|f\|_p = \left(\int_{-\infty}^{\infty} \|f(t)\|^p \, dt \right)^{\frac{1}{p}} \quad p \geq 1 \tag{C.6}$$

For instance, usual signal norms, applied to an error signal vector, $e(t)$, with components $e_i(t)$, are:

1-norm (integral of absolute error IAE):

$$\|e\|_1 = \int_{-\infty}^{\infty} \sum_{i=1}^{n} |e_i(t)| \, dt \tag{C.7}$$

2-norm (integral of squared error ISE):

$$\|e\|_1 = \sqrt{\int_{-\infty}^{\infty} \sum_{i=1}^{n} e_i(t)^2 \, dt} \tag{C.8}$$

Maximum-norm (∞-norm) :

$$\|e\|_\infty = \sup_t \max_i |e_i(t)| \tag{C.9}$$

Unless otherwise stated, it will be understood that *bounded* signals are those with finite maximum norm and *bounded-energy* ones are those with finite 2-norm.

In most cases, signals are defined only for positive t and the lower time limits in the above expressions may be changed to zero.

[2] To verify property 2 in definition C.1, some technical conditions defining two functions as equal if they differ on a zero-measure set are needed, and also regarding existence of the defining integral.

C.3.2 System Norms

Systems are functions from one signal space (input space) to another one (output space). So, norms of these can be defined according to Definition C.2.

In the particular case of norms of *linear* systems, these systems can be characterised by the impulse response, $G(t)$, or frequency response, $G(j\omega)$, matrices. Defining:

$$\|G(s)\|_2 = \sqrt{\frac{1}{2\pi}\int_{-\infty}^{+\infty}\sum_{ij}|G_{ij}(j\omega)|^2\,d\omega} \tag{C.10}$$

$$\|G(s)\|_\infty = max_\omega \bar{\sigma} G(j\omega) \tag{C.11}$$

$$\|G(t)\|_2 = \sqrt{\int_{0}^{+\infty}\sum_{ij}|G_{ij}(t)|^2\,dt} \tag{C.12}$$

$$\|G(t)\|_1 = \int_{0}^{+\infty} max_i \sum_{j}|G_{ij}(t)|\,dt \tag{C.13}$$

Parseval theorem proves that $\|G(t)\|_2 = \|G(s)\|_2$. The 2-norm of a system can also be interpreted as the variance to a white noise input.

It can be shown that the worst-case gain (*i.e.*, the system norm in definition C.2) using specific input and output norms are those in Table C.1.

Table C.1. Induced system norms for specific input and output norms

	$\|u\|_2$	$\|u\|_\infty$
$\|y\|_2$	$\|G(s)\|_\infty$	∞
$\|y\|_\infty$	$\|G(s)\|_2$	$\|G(t)\|_1$

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `norm (lti/norm)`.

C.4 BIBO Stability and the Small-gain Theorem

Given two signal spaces, $\mathcal{U}$ and $\mathcal{Y}$, with a particular norm on them, and a dynamical system, Σ, relating signals in those spaces by $y = \Sigma(u)$, the system is said to be BIBO (bounded input bounded output) stable if Σ has a finite induced norm, *i.e.*, if a constant C exists so that $\|y\| \leq C\|u\|$, for zero initial conditions. It can be shown that the infimum of those constants is the norm of $\|\Sigma\|$ defined in the sense of Definition C.2.

As systems are interconnected to conform more complex ones, it is interesting to investigate the stability properties of a system from the stability properties of its subsystems and its structure.

Series and Parallel Connection

Two systems Σ_1 and Σ_2 are series-connected if:

$$y_2 = \Sigma_2(y_1); \qquad y_1 = \Sigma_1(u)$$

Equation (C.5) shows that if two systems are BIBO-stable, its series connection also has that property. The result is also straightforward in parallel connections, $y = \Sigma_1(u) + \Sigma_2(u)$.

Feedback Connection (Small-gain Theorem)

Let us analyse the feedback loop in Figure C.1 in order to derive a sufficient stability condition.

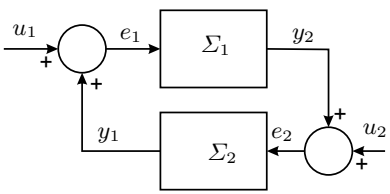

Figure C.1. Feedback interconnection

Let us assume that Σ_1 and Σ_2 are BIBO-stable (*i.e.*, that a finite norm can be calculated so that $\|y_2\| \leq \|\Sigma_1\| \cdot \|e_1\|$ and $\|y_1\| \leq \|\Sigma_2\| \cdot \|e_2\|$). By successively applying the definition of a norm (maximum gain) and the triangle inequality:

$$\begin{aligned}\|y_2\| \leq \|\Sigma_1\| \cdot \|e_1\| \leq \|\Sigma_1\|(\|u_1\| + \|y_1\|) \leq \\ \|\Sigma_1\|(\|u_1\| + \|\Sigma_2\| \cdot \|e_2\|) \leq \|\Sigma_1\|(\|u_1\| + \|\Sigma_2\|(\|u_2\| + \|y_2\|))\end{aligned} \quad \text{(C.14)}$$

solving for $\|y_2\|$ gives:

$$(1 - \|\Sigma_1\| \cdot \|\Sigma_2\|)\|y_2\| \leq \|\Sigma_1\| \cdot \|u_1\| + \|\Sigma_1\| \cdot \|\Sigma_2\| \cdot \|u_2\| \quad \text{(C.15)}$$

so, if $\|\Sigma_1\| \cdot \|\Sigma_2\| < 1$, any solution y_2 verifies that gains from inputs to y_2 are bounded (BIBO stability). Similar expressions can be obtained analogously for the rest of signals, so the following result can be stated:

Theorem C.6 (Small-gain Theorem). *For bounded inputs, all signals in the closed-loop in Figure C.1 are bounded if Σ_1 and Σ_2 are stable and:*

$$\|\Sigma_1\| \cdot \|\Sigma_2\| \leq 1$$

The results holds for Σ_1, Σ_2 being any dynamical system (linear or non-linear) as long as stability and the system norm can be suitably defined. In the application of this theorem to robust control, one of the blocks will be the linear closed-loop system and the other will be the unknown, possibly non-linear, modelling errors.

Example C.7. In the case of one of the systems (say, Σ_2) being a stable LTI one, and that the signal norm $\|y\|$, $\|u\|$ to be considered is the Euclidean 2-norm (inducing the ∞-norm for systems, see Table C.1) a sufficient condition for stability of the closed-loop system is:

$$\|\Sigma_1\| \sup_{w\in\mathbb{R}} \bar{\sigma}(\Sigma_2(j\omega)) \leq 1 \tag{C.16}$$

If there is knowledge that Σ_1 and Σ_2 are both linear time-invariant (that will apply to neglected linear dynamics in robust control), then the previous calculations can be carried out exactly in Laplace transform, operating with block-diagram transformations. The result is:

$$\begin{aligned}(I - \Sigma_1(s)\Sigma_2(s))y_2(s) &= \Sigma_1(s)u_1(s) + \Sigma_1(s)\Sigma_2(s)u_2(s)\\ y_2(s) &= (I - \Sigma_1(s)\Sigma_2(s))^{-1}(\Sigma_1(s)u_1(s) + \Sigma_1(s)B(s)u_2(s))\end{aligned} \tag{C.17}$$

so that if Σ_1 and Σ_2 are stable, $det(I-\Sigma_1(s)\Sigma_2(s))$ evaluated at $s = j\omega$ should *not* encircle the origin to apply the Nyquist criterion (see Section 4.5.1), as a necessary and sufficient condition.

It can be shown that a (less conservative than (C.16)) sufficient condition for this particular case is:

$$\sup_{w\in\mathbb{R}} \bar{\sigma}(\Sigma_1(j\omega)\Sigma_2(j\omega)) < 1 \tag{C.18}$$

and another one, using the submultiplicative property (C.5), is:

$$\sup_{w\in\mathbb{R}} \bar{\sigma}(\Sigma_1(j\omega))\bar{\sigma}(\Sigma_2(j\omega)) < 1 \tag{C.19}$$

In the SISO case, the last conditions amount to:

$$|\Sigma_1(j\omega)\Sigma_2(j\omega)| < 1$$

i.e., if the Nyquist diagram of the "open-loop" transfer function is *strictly inside* the unit circle, it will never touch the point -1.

D

Optimisation

In this appendix, the fundamentals on optimisation of multivariable static and dynamic models are outlined.

D.1 Static Optimisation

First, the problem of obtaining the points where a function attains a maximum or minimum will be addressed. Argument x can be a single variable or a vector.

Optimisation on one variable. A function $y = f(x)$, $f : \mathbb{R} \to \mathbb{R}$ with derivative at x_0 achieves a local maximum (minimum) at $x = x^0$ if $\frac{df}{dx}(x^0) = 0$ and the order of the first non-zero derivative at x^0 is even, and its value is negative (positive).

Optimisation on multiple variables. Now the case $x \in \mathbb{R}^N$ will be discussed, *i.e.*, $y = F(x_1, x_2, \ldots x_n)$, $y \in \mathbb{R}$, $f : \mathbb{R}^N \to \mathbb{R}$. Local maxima (minima) x^0 have zero partial derivatives (Jacobian):

$$\frac{\partial F}{\partial x_i}(x^0) = 0 \ \forall i$$

Points with null partial derivatives are denoted as *critical points*. To verify if the point is a maximum, minimum or saddle point, the matrix of the second derivatives at x^0 should be calculated:

$$H_{ij} = \frac{\partial^2 F}{\partial x_i \partial x_j}$$

so if the matrix is positive (negative) definite, x^0 is a local minimum (maximum), and if it has positive and negative eigenvalues (*i.e.*, if it is indefinite), x^0 is a saddle point. If the matrix is semidefinite, further derivatives would need to be taken.

Quadratic Indexes

Quadratic cost indexes are quite common in optimisation problems. To maximise or minimise the square of linear matrix equations, the derivatives of the following factors:

$$J = x^T P x = \sum x_i p_{ij} x_j = p_{11}x_1^2 + p_{22}x_2^2 + \cdots + (p_{12} + p_{21})x_1 x_2 \ + \ldots$$
$$\eta = a^T x = \sum a_i x_i = a_1 x_1 + \ldots \ + a_n x_n$$

have to be calculated. These derivatives are easy to calculate:

$$\frac{\partial J}{\partial x_i} = 2p_{ii}x_i + (p_{i1} + p_{1i})x_1 + (p_{i2} + p_{2i})x_2 + \cdots = \sum_{j=1}^{n}(p_{ij} + p_{ji})x_j$$
$$\frac{\partial \eta}{\partial x_i} = a_i$$

Arranging these derivatives as a column vector, there is a convenient matrix notation for them, allowing easier bookkeeping in many linear least squares problems:

$$\frac{\partial J(x_1, x_2, \ldots, x_n)}{\partial x} \stackrel{\text{def}}{=} \begin{pmatrix} \frac{\partial J}{\partial x_1} \\ \frac{\partial J}{\partial x_2} \\ \vdots \\ \frac{\partial J}{\partial x_n} \end{pmatrix} \tag{D.1}$$

$$\frac{\partial x^T P x}{\partial x} = (P + P^T)x; \qquad \frac{\partial a^T x}{\partial x} = \frac{\partial x^T a}{\partial x} = a \tag{D.2}$$

Example D.1. Let us calculate the derivative of:

$$M = x_1^2 + 3x_1x_2 + 2x_2^2 + 5x_1 + 2x_2 = x^T \begin{pmatrix} 1 & 1.5 \\ 1.5 & 2 \end{pmatrix} x + \begin{pmatrix} 5 & 2 \end{pmatrix} x$$

$$\frac{\partial M}{\partial x_1} = 2x_1 + 3x_2 + 5 \qquad \frac{\partial M}{\partial x_2} = 3x_1 + 4x_2 + 2$$
$$\frac{\partial M}{\partial x} = \begin{pmatrix} 2x_1 + 3x_2 + 5 \\ 3x_1 + 4x_2 + 2 \end{pmatrix} = 2 \begin{pmatrix} 1 & 1.5 \\ 1.5 & 2 \end{pmatrix} x + \begin{pmatrix} 5 \\ 2 \end{pmatrix}$$

Example D.2. The value of x minimising the squared error in a system of linear equations:

$$y_{p\times 1} = A_{p\times n} x_{n\times 1} \qquad E_{p\times 1} = y - Ax$$
$$J_{1\times 1} = E^T E = (y - Ax)^T (y - Ax) = y^T y - y^T Ax - x^T A^T y + x^T A^T Ax =$$
$$= y^T y - 2y^T Ax + x^T A^T Ax$$

can be obtained in a straightforward way in matrix notation:

$$0 = \frac{dJ}{dx} = -2A^T y + 2(A^T A)x; \qquad x^* = (A^T A)^{-1} A^T y$$

Matrix $(A^T A)^{-1} A^T$ is the *left pseudoinverse* ${}^{\dagger}A$ in (B.2).

Constrained Optimisation (equality)

If a cost index $J(x_1, \ldots, x_n)$ must be optimised subject to a set of differentiable restrictions:

$$f_i(x_1, \ldots, x_n) = 0, \quad i = 1, \ldots m, \quad m < n$$

it has a local minimum or maximum at a point $(x_1^0, \ldots, x_n^0)$ if the unconstrained problem consisting in the minimisation of:

$$H(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_m) = J(x_1, \ldots, x_n) + \sum_{i=1}^{m} \lambda_i f_i(x_1, \ldots, x_n) \qquad \text{(D.3)}$$

(the scalars λ_i, being denoted as *Lagrange multipliers*) has a critical point at the referred point, *i.e.*, if there exist λ_i^0 such that:

$$\frac{\partial H}{\partial x_i}(x_i^0) = 0; \qquad \frac{\partial H}{\partial \lambda_i}(\lambda_i^0) = 0$$

D.2 Discrete Linear Quadratic Regulator

The discrete linear quadratic regulator is defined as the one yielding control actions, minimising the following cost index:

$$J = \frac{1}{2} \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k) + \frac{1}{2} x_N^T S_N x_N \qquad \text{(D.4)}$$

starting from a non-zero[1] initial condition, x_0. Q and S_N are assumed to be symmetric, positive semidefinite matrices, and R is assumed to be symmetric and positive definite.

In Section 7.1, motivation and engineering insight into the above cost index were discussed. Now, the procedure for solving the problem will be detailed.

For small N and small process order, the optimisation can be carried out by brute-force derivative calculation, as in the example below.

Example D.3. For a first-order plant, $x_{k+1} = 0.8x_k + u_k$, and a cost index:

$$J = x_1^2 + x_2^2 + x_3^2 + 0.1u_0^2 + 0.1u_1^2 + 0.1u_2^2$$

writing down the future states using the model:

$$x_1 = 0.8x_0 + u_0; \quad x_2 = 0.8^2 x_0 + 0.8u_0 + u_1; \quad x_3 = 0.8^3 x_0 + 0.8^2 u_0 + 0.8u_1 + u_2$$

the cost, J, is:

$$(0.8x_0+u_0)^2+(0.8^2x_0+0.8u_0+u_1)^2+(0.8^3x_0+0.8^2u_0+0.8u_1+u_2)^2+0.1(u_0^2+u_1^2+u_2^2)$$

[1] Obviously, "zero" refers to any desired operating point once linearisation and a change of variables to incremental ones is made (Section 2.5).

So by equating to zero the derivatives of J with respect to u_0, u_1 and u_2, the optimal values for them will be obtained (as a function of x_0). Indeed:

$$\begin{aligned}
\frac{\partial J}{\partial u_2} &= 0.2u_2 + 2(0.64u_0 + 0.8u_1 + u_2 + 0.512x_0) \\
\frac{\partial J}{\partial u_1} &= 0.2u_1 + 1.6(0.64u_0 + 0.8u_1 + u_2 + 0.512x_0) + 2(0.8u_0 + u_1 + 0.64x_0) \\
\frac{\partial J}{\partial u_0} &= 0.2u_0 + 1.28(0.64u_0 + 0.8u_1 + u_2 + 0.512x_0) + 1.6(0.8u_0 + u_1 + 0.64x_0) \\
&\quad + 2(u_0 + 0.8x_0)
\end{aligned}$$

Equating to zero and solving the resulting linear system of equations, the optimal control actions are:

$$u_0 = -0.7309x_0; \quad u_1 = -0.0505x_0; \quad u_2 = -0.00347x_0$$

and the optimal state trajectory would be:

$$x_1 = 0.0691x_0; \quad x_2 = 0.00478x_0; \quad x_3 = 0.00035x_0$$

However, the procedure gets cumbersome for high N and more complex plants. Introducing matrix notation for the operations and the derivatives will allow us to find a practical solution for a general case, with the so-called dynamic programming approach [24], based on an *induction* argument.

One-step-ahead optimisation

Let us first solve the optimisation problem for one time step, *i.e.*, the sum below containing one term:

$$J_{N-1} = \frac{1}{2} \sum_{k=N-1}^{N-1} (x_k^T Q x_k + u_k^T R u_k) + \frac{1}{2} x_N^T S_N x_N \tag{D.5}$$

In this case, from a starting ("initial condition") state x_{N-1}, the only manipulated variable needing to be calculated is u_{N-1}, and the term $x_{N-1}^T Q x_{N-1}$ is irrelevant as it cannot be modified by the control action.

Replacing x_N by the model output, $x_N = Ax_{N-1} + Bu_{N-1}$, the last term in the index is $x_N^T S_N x_N = \frac{1}{2}(Ax_{N-1} + Bu_{N-1})^T S_N (Ax_{N-1} + Bu_{N-1})$ so after straightforward manipulations, the cost index can be written as:

$$\begin{aligned}
J_{N-1} = \frac{1}{2} \big(& u_{N-1}^T R u_{N-1} + x_{N-1}^T (A^T S_N A + Q) x_{N-1} \\
& + 2x_{N-1}^T A^T S_N B u_{N-1} + u_{N-1}^T B^T S_N B u_{N-1} \big)
\end{aligned} \tag{D.6}$$

a summation consisting, of course, of scalar (1×1) terms.

To compute the optimal, and using the notation defined in (D.1):

$$\frac{\partial J_{N-1}}{\partial u_{N-1}} = R u_{N-1} + B^T S_N A x_{N-1} + B^T S_N B u_{N-1} = 0$$

Solving for u, the result is:

$$u_{N-1} = -(B^T S_N B + R)^{-1} B^T S_N A x_{N-1} \stackrel{\text{def}}{=} -K_{N-1} x_{N-1} \tag{D.7}$$

The minimum value of J can be obtained by replacing the optimal control action in (D.6):

$$J_{N-1}^{opt} = \frac{1}{2} x_{N-1}^T \left(K_{N-1}^T R K_{N-1} + Q + A^T S_N A \right. \\ \left. -2A^T S_N B K_{N-1} + K_{N-1}^T B^T S_N B K_{N-1} \right) x_{N-1} \tag{D.8}$$

so, replacing K by the expression in (D.7), after some operations, the optimal index is:

$$J_{N-1}^{opt} = \frac{1}{2} x_{N-1}^T (Q + A^T S_N A - A^T S_N B (B^T S_N B + R)^{-1} B^T S_N A) x_{N-1} \tag{D.9}$$

For convenience, let us denote as S_{N-1} the matrix:

$$S_{N-1} = Q + A^T S_N A - A^T S_N B (B^T S_N B + R)^{-1} B^T S_N A \tag{D.10}$$

so the achieved optimal cost is a quadratic function $J_{N-1}^{opt} = \frac{1}{2} x_{N-1}^T S_{N-1} x_{N-1}$.

D.2.1 Multi-step Optimisation (Dynamic Programming)

Richard Bellmann [24] stated a *Principle of optimality*, for sum-over-time indices, as:

> *An optimal policy has the property that, whatever the initial state and past decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decisions*[2].

In plain terms, once a particular state is reached, the optimal policy is optimising the "cost-to-go" function (this cost is being a function of the state in which the system is left).

Remark D.4. An important engineering consequence is that this dependence on the state justifies the existence of a *state feedback control* strategy that solves the optimisation problem. Note that, in the brute-force solution in Example D.3, all control actions depended on x_0, so there was no simple way of presenting the problem in a feedback form (u_1 depending on x_1 and so on).

[2] Proof by contradiction: if $u^*(t)$ is the optimal policy for time $[0, t_f]$, let us apply it on an interval $[0, t_m]$, $t_m < t_f$, with end system state $x(t_m)$, and now calculate the optimal policy $u^m(t)$ from that starting state in the interval $[t_m, t_f]$. If it achieved a different J in that interval to that of $u^*(t)$, applying u^* on the first interval and u^m on the second would achieve, by definition of u^m, a total cost:

$$J_{[0,t_m]}(u^*) + J_{[t_m,t_f]}(u^m) < J_{[0,t_m]}(u^*) + J_{[t_m,t_f]}(u^*)$$

i.e., lower than that of the optimal, contradicting the assumption.

In our case, at this moment, the cost to go is J_{N-1} and the optimality principle implies that the optimal policy for J will have the state feedback in (D.7) applied at time $N-1$. The optimal J^{opt}_{N-1} in (D.9) is a function of x_{N-1}.

Stating the problem now in a two-step horizon:

$$J_{N-2} = \frac{1}{2}\sum_{k=N-2}^{N-1}(x_k^T Q x_k + u_k^T R u_k) + \frac{1}{2}x_N^T S_N x_N$$
$$= (x_{N-1}^T Q x_{N-2} + u_{N-2}^T R u_{N-2}) + J_{N-1} \quad \text{(D.11)}$$

the optimality principle amounts to using the previous solution, $u_{N-1} = -K_{N-1}x_{N-1}$, as the last control action, and determining u_{N-2} so that:

$$J^*_{N-2} = \frac{1}{2}(x_{N-2}^T Q x_{N-2} + u_{N-2}^T R u_{N-2}) + J^{opt}_{N-1}(x_{N-1})$$
$$= \frac{1}{2}(x_{N-2}^T Q x_{N-2} + u_{N-2}^T R u_{N-2}) + \frac{1}{2}x_{N-1}^T S_{N-1} x_{N-1} \quad \text{(D.12)}$$

is minimised. Note that the above equation has exactly the same form as (D.4), but has "advanced" one time step.

So, based on Bellman's principle, the solution to the two-step optimisation will be, rewriting the steps in (D.6), (D.7), (D.9):

$$u_{N-2} = -(B^T S_{N-1} B + R)^{-1} B^T S_{N-1} A x_{N-2} \stackrel{\text{def}}{=} -K_{N-2}x_{N-2}$$
$$S_{N-2} = Q + A^T S_{N-1} A - A^T S_{N-1} B (B^T S_{N-1} B + R)^{-1} B^T S_{N-1} A$$
$$J^{opt}_{N-2} = x_{N-2}^T S_{N-2} x_{N-2}$$

By reiterated application of the optimality principle, the three-step problem would imply solving for u_{N-3} so that:

$$J^*_{N-3} = \frac{1}{2}(x_{N-3}^T Q x_{N-3} + u_{N-3}^T R u_{N-3}) + \frac{1}{2}x_{N-2}^T S_{N-2} x_{N-2} \quad \text{(D.13)}$$

and, so on, iterating backwards in time for $l = N-1, \ldots, 0$ the calculation of the first optimal control action, u_0, is achieved.

$$u_l = -(B^T S_{l+1} B + R)^{-1} B^T S_{l+1} A x_l \stackrel{\text{def}}{=} -K_l x_l \quad \text{(D.14)}$$
$$S_l = Q + A^T S_{l+1} A - A^T S_{l+1} B (B^T S_{l+1} B + R)^{-1} B^T S_{l+1} A \quad \text{(D.15)}$$

Example D.5. Let us apply the obtained solution to Example D.3. In this case, $Q = 1$, $R = 0.1$, $S_3 = 1$, $A = 0.8$, $B = 1$.

The backwards iterations result is:

$$
\begin{aligned}
u_2 &= -(B^T S_3 B + R)^{-1} B^T S_3 A x_2 = -0.7273 x_2 \\
S_2 &= Q + A^T S_3 A - A^T S_3 B (B^T S_3 B + R)^{-1} B^T S_3 A = 1.0582 \\
u_1 &= -(B^T S_2 B + R)^{-1} B^T S_2 A x_1 = -0.7309 x_1 \\
S_1 &= Q + A^T S_2 A - A^T S_2 B (B^T S_2 B + R)^{-1} B^T S_2 A = 1.0585 \\
u_0 &= -(B^T S_1 B + R)^{-1} B^T S_1 A x_0 = -0.7309 x_0
\end{aligned}
$$

So, from a starting state $x_0 = 10$, $u_0 = -7.309$ is obtained, and then $x_1 = 0.6910$, yielding $u_1 = -0.5051$, then $x_2 = 0.0478$ yielding $u_2 = -0.0348$. The results are the same (except round-off) but now they are expressed in *feedback form*.

Note that the obtained feedback regulator is linear but *time-varying*. The formulae are very similar to those from the Kalman filter ((E.42) and (E.43) on page 320). They are, in fact, *dual* results pinpointing hidden common structures in optimal control and optimal estimation [9].

D.2.2 Stationary Regulator

Time-varying regulators are difficult to implement. Furthermore, many problems of practical significance can be stated in terms of *infinite-horizon* cost indexes (see Section 7.1 for details).

In particular, the solution to:

$$
J = \frac{1}{2} \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \tag{D.16}
$$

can be obtained, under some sensible assumptions, taking the limit $N \to \infty$ in the backward-iteration-based procedure previously presented, *i.e.*, executing enough iterations of:

$$
S_l = Q + A^T S_{l+1} A - A^T S_{l+1} B (B^T S_{l+1} B + R)^{-1} B^T S_{l+1} A \tag{D.17}
$$

until convergence is achieved, starting from an arbitrary, positive definite S_N. The state feedback, (D.14), then becomes a constant, time-invariant controller.

The final result is independent of the chosen S_N seed. It is intuitively expected as, if Q is positive definite, for long horizons, the optimal policy is trying to drive the state to zero fairly soon as, in other cases, $x^T Q x$ will accumulate significant penalty during a long time interval. If the state, after a long horizon, is near zero, the value of $x_N^T S_N x_N$ will be very small (compared to the accumulated $\Sigma x^T Q x$), provided S_N is not "too large".

For the iterations to converge and guarantee a *stable* closed loop, the system (A, B) must be stabilisable and all states should be "seen" by the cost function, *i.e.*, $(A, \sqrt{Q})$ must be detectable and R positive definite.

The final value is the positive definite solution, S, to the so-called discrete-time algebraic Riccati equation:

$$S = Q + A^T SA - A^T SB(B^T SB + R)^{-1} B^T SA \tag{D.18}$$

This is a system of $n \times n = n^2$ non-linear equations with n^2 unknowns (the elements of S). As the equations are non-linear, the solution is not unique but it can be shown that, under general assumptions, there is only one positive definite solution (the sought one).

The solution to infinite-horizon optimisation problems is a time-invariant state-feedback linear regulator, in the same form as pole-placement ones.

Continuous cost index

The solution to the CT optimal control problem, (7.2), requires posing a dynamic programming problem in infinitesimal time steps. The reader is referred to [81, 122, 134]. The result is that the matrix differential equation:

$$-\dot{S} = A^T S + SA - SBR^{-1}S + Q \tag{D.19}$$

must be evaluated backwards in time for an initial condition, $S(t_f) = S_N$, and the control action is linear time-variant:

$$u(t) = -R^{-1}B^T S(t)x(t) \tag{D.20}$$

A stationary state-feedback linear time-invariant controller for infinite horizon problems is obtained by solving the Riccati equation:

$$0 = A^T S + SA - SBR^{-1}S + Q \tag{D.21}$$

MATLAB®: Some commands implementing algorithms related to the contents of this section are: `are,dare,lqr,dlqr`.

Mixed Indices

The indexes can be generalised to include cross-terms. For example, in the discrete case:

$$J = \frac{1}{2}\sum_{k=0}^{\infty}(x_k^T Q x_k + u_k^T R u_k + x_k^T 2N u_k) \tag{D.22}$$

a solution for it can be calculated by transforming the index to the form (D.4) via a change of variable $\bar{A} = A - BR^{-1}N^T$, $\bar{Q} = Q - NR^{-1}N^T$. An application of mixed indexes is continuous optimal control with a discrete (ZOH) regulator (Section 7.1).

E

Multivariable Statistics

E.1 Random Variables

The concept of a *random variable* formalises situations with uncertain outcome. It takes values in a sample space, Ω, and a probability measure is defined over the measurable subsets of the sample space, fulfilling particular axioms. The reader is referred to standard statistics textbooks [98, 14] for details.

For example, tossing a coin is a random variable with sample space {Heads, Tails} with the probability {p(Heads) = 0.5, p(Tails) = 0.5, p($\emptyset$) = 0, p(Heads $\cup$ Tails) = 1}. If the sample space is a finite or countable set, the variables are called *discrete* random variables. On the other hand, if the sample space is uncountable (such as real numbers) they are denoted as *continuous* random variables.

The basics of probability are intuitively modelled as a limit of the fraction of occurrence of an event, after infinite repetitions of an experiment. However, that concept does not apply to all situations. The world population in 01/01/2020 is also a random variable, but there will be only one realisation. Hence, assigning probabilities to the variable is either subjective or based on models establishing a relationship between the variable and other ones with known probability. Note that operating in this general set-up is much more complex than "coin" problems and the like.

In the following sections, the variables to be considered will be the outputs from discrete-time linear dynamic systems. These variables take values of the real line, and probability distributions will be defined by density functions (under some smoothness requirements).

In most cases, "randomness" is originated by non-linear deterministic systems exhibiting "chaotic" behaviour. The interested reader may consult [72] for an introductory overview.

In a real-valued space with a probability defined on it by means of a density function, $f(x)$, the *mathematical expectation*, $E(\cdot)$, of a particular expression $h(x)$ is defined as:

$$E[h(x)] = \int h(x) f(x) dx \tag{E.1}$$

This calculates the "weighted average" of $h(x)$ multiplying each possible outcome by its probability (an infinitesimal $f(x)dx$). In discrete variables, the expectation is actually a weighted sum.

The *mean* and *variance* of a random variable v is:

$$\bar{v} = E(v) = \int_{-\infty}^{+\infty} \tau f_v(\tau)\, d\tau \tag{E.2}$$

$$\sigma_v^2 = E((v - \bar{v})^2) = \int (\tau - \bar{v})^2 f_v(\tau)\, d\tau \tag{E.3}$$

The square root of the variance is termed the *typical deviation*, σ_v.

For a high number of realisations, v^j, these figures can be obtained from sample averages:

$$\lim_{N\to\infty} \frac{\sum_{j=1}^{N} v^j}{N} = \bar{v} \tag{E.4}$$

$$\lim_{N\to\infty} \frac{\sum_{j=1}^{N} (v^j - \bar{v})^2}{N-1} = \sigma_v^2 \tag{E.5}$$

Covariance between two variables a and b is defined as:

$$\sigma_{ab}^2 = E((a - \bar{a})(b - \bar{b})) \tag{E.6}$$

Obviously, $\sigma_{ab}^2 = \sigma_{ba}^2$. If a joint two-dimensional density function is available (Section E.2), covariance can be expressed as:

$$\sigma_{ab}^2 = \int (\tau - \bar{a})(\psi - \bar{b}) f_{ab}(\tau, \psi)\, d\tau\, d\psi \tag{E.7}$$

Alternatively, from experimental data (sample covariance):

$$\lim_{N\to\infty} \frac{\sum_{j=1}^{N} (a^j - \bar{a})(b^j - \bar{b})}{N-1} = \sigma_{ab}^2 \tag{E.8}$$

E.1.1 Linear Operations with Random Variables

The addition of two random variables, x and y, is defined as the random variable representing the uncertain outcome of the sum of a realisation of x and another of y. For example, the addition of two dice throws has a probability for each outcome:

$$\{\, p(2) = 1/36,\ p(3) = 2/36,\ p(4) = 3/36,\ p(5) = 4/36,\ p(6) = 5/36,\ p(7) = 6/36,$$
$$p(8) = 5/36,\ p(9) = 4/36,\ p(10) = 3/36,\ p(11) = 2/36,\ p(12) = 1/36\}$$

The product by a scalar, λ, is defined as the result of multiplying the outcome by λ. For example, multiplying the dice throw by 2 yields: $\{p(2) = 1/6, p(4) = 1/6, p(6) = 1/6, p(8) = 1/6, p(10) = 1/6, p(12) = 1/6\}$. Note that "dice + dice $\neq 2 \times$ dice".

Addition and multiplication by scalar verifies that:

$$E(a+b) = E(a) + E(b) \tag{E.9}$$
$$E(\alpha v) = \alpha E(v) \tag{E.10}$$

and, regarding variance:

$$E((a+b)^2) = E(a^2) + E(b^2) + 2E(ab) \tag{E.11}$$
$$\sigma^2_{\alpha v} = E((\alpha(v - \bar{v}))^2) = \alpha^2 \sigma^2_v \tag{E.12}$$

If variables are independent (Section E.2), the variance of the sum is the sum of variances as $E(ab) = 0$.

The addition of a constant, non-random, offset, k, yields:

$$E(a+k) = E(a) + k; \qquad \sigma^2_{(a+k)} = \sigma^2_k \tag{E.13}$$

Example E.1 (Numerical derivative). Two close measurements of a variable $x = x^{clean} + n$, where n is a measurement noise, with zero-mean and variance σ_n, can be used to estimate its derivative, $\dot{x}_k \approx (x_k - x_{k-1})/T$, where T is the sampling period. In this case:

$$E(\dot{x}_k) = (x^{clean}_k - x^{clean}_{k-1})/T; \quad \sigma^2(\dot{x}_k) = \frac{2}{T^2}\sigma^2_n \tag{E.14}$$

So the typical deviation (square root of variance) of the estimate is $\sqrt{2}/T$ that of the sensor noise. For small sampling periods, where the derivative approximation is valid, this may imply significant amplification.

E.2 Multi-dimensional Random Variables

Several random variables can be stacked as a *random vector*:

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}; \quad y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \tag{E.15}$$

Its outcomes will be vectors in $\mathbb{R}^n$:

$$x^0 = \begin{pmatrix} x^0_1 \\ x^0_2 \end{pmatrix}; \quad y^0 = \begin{pmatrix} y^0_1 \\ \vdots \\ y^0_n \end{pmatrix} \tag{E.16}$$

This type of variable will represent the signals to be analysed in the multivariable control context.

The *mean* of a random vector is the vector of means from each component variable:

$$\bar{x} = \begin{pmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_n \end{pmatrix} \tag{E.17}$$

and its *variance* is defined as a matrix with all pairwise combinations of variances and covariances arranged as:

$$\Sigma_{xx} = \begin{pmatrix} \sigma^2_{x_1} & \sigma^2_{x_1x_2} & \sigma^2_{x_1x_3} & \cdots & \sigma^2_{x_1x_n} \\ \sigma^2_{x_2x_1} & \sigma^2_{x_2} & \sigma^2_{x_2x_3} & \cdots & \sigma^2_{x_2x_n} \\ \vdots & & \ddots & & \vdots \\ \sigma^2_{x_nx_1} & & \cdots & & \sigma^2_{x_n} \end{pmatrix} \tag{E.18}$$

This matrix is also usually denoted as a variance-covariance matrix (VC), and it is always symmetric and positive semidefinite, always diagonalisable. Shorthand for the VC matrix using vector notation is:

$$\Sigma_{xx} = E((x - \bar{x})_{n\times 1}(x - \bar{x})^{\mathrm{T}}_{1\times n}) \tag{E.19}$$

generalising (E.3). The covariance between two variables, v and z, can be expressed as a matrix with dimensions $n_v \times n_z$:

$$\Sigma_{vz} = E((v - \bar{v})_{n_v\times 1}(z - \bar{z})^{\mathrm{T}}_{1\times n_z}); \qquad [\Sigma_{vz}]_{ij} = \sigma^2_{v_iz_j} \tag{E.20}$$

In the sequel, Δx will denote $x - E(x)$.

A multi-dimensional density function, $f : \mathbb{R}^n \to [0, 1]$, defines the probability distribution, and it can be interpreted as the probability per unit "volume", and its integral over its definition domain should be equal to 1.

As an example, in the two-dimensional case, the probability of a particular rectangle is defined as:

$$\begin{aligned} &P((x_1, x_2) \in [a_1, b_1] \times [a_2, b_2]) \\ &\quad = P(x_1 \in [a_1, b_1] \text{ and } x_2 \in [a_2, b_2]) = \int_{a_1}^{b_1}\int_{a_2}^{b_2} f_x(v_1, v_2)\, dv_1 dv_2 \end{aligned} \tag{E.21}$$

The expectation of a function, $h(x_1, \ldots, x_n)$, can be calculated as:

$$E(g) = \int \ldots \int h(v_1, \ldots, v_n) f_x(v_1, \ldots, v_n) dv_1 \ldots dv_n \tag{E.22}$$

where integration is carried out on all of the domain of definition of f. For example, the mean of x_1 is:

$$E(x_1) = \int \int v_1 f_x(v_1, v_2) dv_1 dv_2 \tag{E.23}$$

Linear Operations

Linear operations with random variables in $\mathbb{R}^n$ are vector addition and product of a vector by a constant matrix.

If v is a random vector obtained from another random vector, w, by $Aw+B$ where A and B are constant matrices, its mean vector and variance matrix can be obtained as:

$$\begin{aligned}\bar{v} &= E(v) = E(Aw+B) = AE(w) + B = A\bar{w} + B \qquad \text{(E.24)}\\ \Sigma_{vv} &= E((v-\bar{v})(v-\bar{v})^T) = E((Aw+B-A\bar{w}-B)(Aw+B-A\bar{w}-B)^T)\\ &= E(A(w-\bar{w})(w-\bar{w})^T A^T) = AE((w-\bar{w})(w-\bar{w})^T)A^T = A\Sigma_{ww}A^T\end{aligned}$$

Independent Variables

Two random variables are independent if their joint density function, $f(x_1, x_2)$, can be expressed as a product of two single-variable densities:

$$f_x(x_1, x_2) = f_{x_1}(x_1) * f_{x_2}(x_2) \qquad \text{(E.25)}$$

In this way,

$$P((x_1, x_2) \in [a_1, b_1] \times [a_2, b_2]) = P(x_1 \in [a_1, b_1]) \times P(x_2 \in [a_2, b_2]) \qquad \text{(E.26)}$$

and also, the expectation of any expression, $g(x_1)$, is unrelated to x_2 as:

$$\begin{aligned}E(x_1) &= \int\int g(v_1) f_x(v_1, v_2) dv_1 dv_2 = \int\int g(v_1) f_{x_1}(v_1) f_{x_2}(v_2) dv_1 dv_2\\ &= \int g(v_1) f_{x_1}(v_1) dv_1 \int f_{x_2}(v_2) dv_2 = \int g(v_1) f_{x_1}(v_1) dv_1 \qquad \text{(E.27)}\end{aligned}$$

Independent variables have null covariance:

$$\begin{aligned}E((a-\bar{a})(b-\bar{b})) &= \int\int (a-\bar{a})(b-\bar{b}) f(a) f(b)\, da\, db\\ &= \int (a-\bar{a}) f(a)\, da \int (b-\bar{b}) f(b)\, db = 0 \times 0 = 0 \qquad \text{(E.28)}\end{aligned}$$

However, the reciprocal is *not* true (see Example E.2). Variables with null covariance are said to be *uncorrelated*. In this way, a random vector composed of independent variables has a *diagonal* covariance matrix.

The variance of the sum of uncorrelated variables is the sum of variances:

$$E[(a+b)(a+b)^T] = E[aa^T] + E[bb^T] + E[ab^T] + E[bA^T] = E[aa^T] + E[bb^T]$$

Example E.2. A random variable, x, is uniformly distributed in the interval $[-1, 1]$ ($f(x) = 0.5$). Another random variable, y, is obtained from x by means of $y = \frac{1}{3} - x^2$. Note that y is a random variable as indeed its outcome, and without knowledge of

x is uncertain. However, if x is known, the model of y is *deterministic*, *i.e.*, the opposite situation to statistical independence[1]. However, the mean of x is 0 and that of y is:

$$\int_{-1}^{1} y(x)f(x)\,dx = \int_{-1}^{1} (\frac{1}{3} - x^2)\ 0.5\,dx = 0$$

Covariance is:

$$E(xy) = \int_{-1}^{1} 0.5x(\frac{1}{3} - x^2)dx = 0$$

The following relations hold:

- if two uncorrelated variables are known not to be related by a non-linear expression they are independent, *i.e.*, two uncorrelated variables may be non-independent (necessarily related by a non-linear model),
- correlated variables are non-independent.

E.3 Linear Predictors (Regression)

There can be several meanings to the concept "prediction". Let us explicitly define the one used in this book.

Definition E.3 (Optimal prediction.). *The optimal prediction, p, of a random variable, v, given a known value, z^0, of another random variable, z, is a function, $p(z)$, such that $v - p(z)$ is a random variable independent of z, with zero mean.*

Example E.4. If z itself is independent of v, a constant $p(z) = \bar{v}$ is the best prediction in the above sense as $v - \bar{v}$ is zero-mean.

Definition E.5 (Optimal linear prediction.). *The optimal linear prediction p of a random variable v given a known value z^0 of another random variable z is a* linear *function $p(z)$ such that $v - p(z)$ is a random variable* uncorrelated *with z, with zero mean.*

In a general multivariable case, assuming the variance matrices for v and z are known, as well as the covariance (for example by calculating them through a long data record), the following result holds:

Proposition E.6. *The optimal linear prediction of v, given z^0, is:*

$$P = \bar{v} + \Sigma_{vz}\Sigma_{zz}^{-1}(z^0 - \bar{z}) \tag{E.29}$$

where Σ_{vv}, Σ_{zz} are the VC matrices of variables v and z and Σ_{vz} is the covariance between v and z.

[1] Independence would not entail being able to improve predictions of y given x (see Section E.3). Determinism can be represented by a density function equal to Dirac $\delta(t)$.

Indeed, the prediction error is:

$$E_p = v - \left[\bar{v} + \Sigma_{vz}\Sigma_{zz}^{-1}(z - \bar{z})\right]$$

so its mean amounts to:

$$E(E_p) = \bar{v} - \bar{v} - \Sigma_{vz}\Sigma_{zz}^{-1}(\bar{z} - \bar{z}) = 0 \tag{E.30}$$

and the covariance with the known variable is:

$$E((E_p - 0)(z - \bar{z})^T) = \Sigma_{vz} - \Sigma_{vz}\Sigma_{zz}^{-1}\Sigma_{zz} = 0 \tag{E.31}$$

In this way, no further linear expression can be used to approximate v from z. This does not preclude, however, the possibility of the existence of additional non-linear relations between v and z (even to the point of being deterministic as in Example E.2).

The quality of the linear prediction (expected squared error) is given by:

$$E(E_p E_p^T) = \Sigma_{vv} - \Sigma_{vz}\Sigma_{zz}^{-1}\Sigma_{vz}^T \tag{E.32}$$

The above expression can be interpreted in the sense that the knowledge of variables with non-zero correlation Σ_{vz} with the one trying to be predicted decreases the uncertainty of its estimation. If any known variable z has $\Sigma_{vz} = 0$ then (E.29) reduces to $P = \bar{v}$, with error variance Σ_{vv}.

Example E.7. A temperature sensor, z_1, reads the true temperature, T, plus an independent zero-mean noise, n_1. Another sensor, z_2, has similar characteristics, so:

$$z = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} T + \begin{pmatrix} \eta_1 \\ \eta_2 \end{pmatrix} = CT + \eta \tag{E.33}$$

The true temperature for a particular date is known to have an average value of 20^oC and a variance of $9(^0\text{C})^2$, by historic records. Sensor noise has variance $\sigma_n^2 = 1$ and each of the channels is independent: $E(\eta\eta^T) = \sigma_n^2 I_{2\times 2}$.

The mean of z is $\bar{z} = C\bar{T}$ equal to the measured temperature as noise means are zero. The variance matrix of z is $E((z - \bar{z})(z - \bar{z})^T)$ and:

$$\begin{aligned}(z - \bar{z})(z - \bar{z})^T &= (C(T - \bar{T}) + \eta)(C(T - \bar{T}) + \eta)^T \\ &= C(T - \bar{T})^2 C^T + C(T - \bar{T})\eta + \eta(T - \bar{T})C + \eta\eta^T \end{aligned} \tag{E.34}$$

so, as T and η are, by assumption, independent, $E((T - \bar{T})\eta) = E(\eta(T - \bar{T})) = 0$ and the variance results:

$$\Sigma_{zz} = C\sigma_T^2 C^T + \Sigma_{\eta\eta} = \sigma_T^2 \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} + \sigma_n^2 I$$

The covariance between sensors and real temperature is:

$$\Sigma_{Tz} = E((T - \bar{T})(z - \bar{z})^T) = E((T - \bar{T})(C(T - \bar{T}) + \eta)^T) = \sigma_T^2 C^T$$

also by using the independence hypothesis.

So, the best prediction of the temperature is:

$$P = 20 + \begin{pmatrix} 1 & 1 \end{pmatrix} \sigma_T^2 \begin{pmatrix} \sigma_T^2 + \sigma_n^2 & \sigma_T^2 \\ \sigma_T^2 & \sigma_T^2 + \sigma_n^2 \end{pmatrix}^{-1} \left(\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} - \begin{pmatrix} 20 \\ 20 \end{pmatrix} \right)$$

Replacing the actual estimated temperature variance, and carrying out operations:

$$P = 20 + 0.4737(z_1 - 20) + 0.4737(z_2 - 20)$$

and the estimated squared prediction error is:

$$E(E_p^2) = 9 - \begin{pmatrix} 1 & 1 \end{pmatrix} \sigma_T^2 \cdot \begin{pmatrix} \sigma_T^2 + 1 & \sigma_T^2 \\ \sigma_T^2 & \sigma_T^2 + 1 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \sigma_T^2 = 0.4737$$

A quite common situation is sensor fusion without any historic record to extract estimated variance: the initial assumption on T is that it has mean 20 and infinite variance. Evaluating the previous expressions, letting $\sigma_T^2 \to \infty$, they result in:

$$P = 0.5z_1 + 0.5z_2; \quad E(E_p^2) = 0.5$$

as intuitively expected: the best prediction is the average of the readings. The expected error is about 0.71 times the error using only one sensor ($0.71 = \sqrt{0.5}$).

E.4 Linear Systems

The previous example shows how a static model with disturbances can be used to carry out sensor fusion to obtain an improved estimate. This section will extend the result to dynamical models in order to use the model of the system jointly with sensors in different locations and in different time samples to obtain the best linear prediction of the internal system state.

E.4.1 Simulation

The expressions (E.24) directly apply to a randomly-perturbed linear system with equations:

$$x_{k+1} = Ax_k + Bu_k + v_k \tag{E.35}$$

$$y_k = Cx_k + w_k \tag{E.36}$$

Indeed, all the past history of deterministic and random inputs is summarised in a state vector with a particular probability distribution, having a mean in sample k given by $\bar{x}_k$ and a variance matrix Σ_{x_k}. If the process noise variance is V and the measurement noise one is W, and they are independent of the state, then the linear process equations allow us the determination of the situation in the current output sample as:

$$\bar{y}_k = C\bar{x}_k \quad \Sigma_{y_k} = C\Sigma_{x_k}C^T + W \tag{E.37}$$

and the probable situation of the next state:

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k; \quad \Sigma_{x_{k+1}} = A\Sigma_{x_k}A^T + V \tag{E.38}$$

These expressions constitute the *simulation* of a linear stochastic process. Let us deduce, for instance, the second one.

As the state x_{k+1} is related to the past state by (E.35), and v is zero-mean, its expected value will be:

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k$$

but the actual value will differ from the expected one. The variance of the prediction error is given by:

$$\begin{aligned}\Sigma_{x_{k+1}} &= E(\Delta x_{k+1}\Delta x_{k+1}^T) \\ &= E((A\Delta x_k + B(u_k - \bar{u}_k) + v_k)(A\Delta x_k + B(u_k - \bar{u}_k) + v_k)^T \\ &= AE(\Delta x_k \Delta x_k^T)A^T + E(v_k v_k^T) = A\Sigma_{x_k}A^T + V \end{aligned} \tag{E.39}$$

as $u_k = \bar{u}_k$ because u is not a random variable, and x and v are independent so the expectation of its products is zero.

E.4.2 Prediction: The Kalman Filter

The dynamic models to be considered are those in Section 2.10.3, in particular the general linear stochastic process with manipulated input referred above, (E.35).

Let us consider now the optimal prediction problem. In this set-up, the optimal predictor for x_k is a linear time-varying recurrent system. Let us detail how this predictor can be built.

Let us assume that we know data from the best prediction of system state at time $k-1$ (x_{k-1}), in particular, its expected value, $\bar{x}_{k-1}$, and the prediction error variance, $\Lambda_{k-1} = E((x_{k-1} - \bar{x}_{k-1})(x_{k-1} - \bar{x}_{k-1})^T)$. This will be the core of the recurrence equations, as our predictor will output the best prediction (mean) and its expected squared error (variance) for time k, and that will be used for the next sample, *etc.*

Note that, to start the procedure from a null-information situation, the initial conditions need to be stated as $\Lambda_{k-1} \to \infty$, with arbitrary $\bar{x}_{k-1}$.

In this case, as the state x_k is related to the past state by $x_k = Ax_{k-1} + Bu_{k-1} + v_{k-1}$, its expected value and variance will be, applying (E.38):

$$\bar{x}_k = A\bar{x}_{k-1} + Bu_{k-1}; \quad \Sigma_{x_k} = A\Lambda_{k-1}A^T + V$$

The above formulae refer to the estimate with information from sample $k-1$ ($\hat{x}_{k|k-1}$, with the notation in Chapter 6). Incorporating information from sample y_k will modify both $\bar{x}_k$ and Σ. So, these $\bar{x}_k$ and Σ_{x_k} are denoted as the *a priori* state estimation.

Furthermore, as output y_k is related to x_k by $Cx_k + w_k$, it can be easily shown that $\bar{y}_k = C\bar{x}_k$ and:

$$\Sigma_{y_k} = E(\Delta y_k \Delta y_k^T) = C\Sigma_{x_k} C^T + W \tag{E.40}$$

Additionally, the sensor entails x_k and y_k being non-independent and, as the equation is linear, with non-zero correlation:

$$\Sigma_{xy} = E(\Delta x_k \Delta y_k^T) = E(\Delta x_k (C\Delta x_k + w_k)^T = \Sigma_{x_k} C^T \tag{E.41}$$

where the assumption of w_k being independent of x_k has been used.

Now we are in the situation analogous to example E.7, ready to incorporate measurement y_k, as we know the expected value and variance of x_k, the expected value and variance of output y_k and the actual measurement (with a slight abuse of notation, denoted equally as y_k).

Applying the equation of the best linear prediction (E.29), the best *a posteriori* prediction of x_k ($\hat{x}_{k|k}$) after incorporating a known output measurement, y_k, is:

$$\begin{aligned} P &= \bar{x}_k + \Sigma_{xy}\Sigma_{y_k}^{-1}(y_k - \bar{y_k}) \qquad\qquad (E.42)\\ &= A\,\bar{x}_{k-1} + Bu_{k-1} + \Sigma_{x_k} C^T (C\Sigma_{x_k} C^T + W)^{-1}(y_k - C(A\bar{x}_{k-1} + Bu_{k-1})) \end{aligned}$$

So the best prediction is given by a discrete-time current observer with gain:

$$L = (A\Lambda_{k-1}A^T + V)C^T(C(A\Lambda_{k-1}A^T + V)C^T + W)^{-1}$$

The prediction error will have a variance given by (E.32):

$$\Lambda_k = \Sigma_{xx} - \Sigma_{xy}\Sigma_{y_k}^{-1}\Sigma_{xy}^T = (I - LC)(A\Lambda_{k-1}A^T + V) \tag{E.43}$$

Note that expressions (E.42) and (E.43) provide the expected value and the prediction error variance for x_k so they can be used as the basis for determining x_{k+1} from measurement y_{k+1} by applying the procedure above to increasing the time indexes.

Repeated evaluation of (E.42) and (E.43) constitutes the so-called *Kalman filter*, this filter being a time-varying one as the observer gain, L, changes at each sample. Note that, however, Λ_k does *not* depend on y_k but only on the assumed model characteristics so it can be calculated off-line.

To summarise the Kalman filter equations in a compact form:

$$\Sigma_{x_k} = A\Lambda_{k-1}A^T + V; \qquad L_k = \Sigma_{x_k} C^T (C\Sigma_{x_k} C^T + W)^{-1} \tag{E.44}$$

$$\bar{x}_k = A\bar{x}_{k-1} + Bu_{k-1} + L_k(y_k - C(A\bar{x}_{k-1} + Bu_{k-1})) \tag{E.45}$$

$$\Lambda_k = (I - L_k C)\Sigma_{x_k} \tag{E.46}$$

Note E.8. If the procedure starts with no information at all, then $\Lambda_{-1} \to \infty$. However, numerical round-off and conditioning problems may arise unless the first iteration is carried out by symbolic computing.

Note E.9. If significant correlation between v_k and w_k exists, the above formulae need to be modified. This is the case when the same physical phenomenon influences both plant and sensor. Details are available in [9].

Example E.10. Let us apply the Kalman filtering to the elementary case of trying to estimate a constant variable from a series of noisy measurements of it. This can be understood as the design of an observer for the process:

$$x_{k+1} = x_k; \qquad y_k = x_k + w_k \tag{E.47}$$

so, $A = 1$, $C = 1$, $V = 0$ and, without loss of generality, the measurement noise variance, W, will be assumed to be 1. Starting with no information, from (E.44) and (E.46):

$$L_0 = \lim_{\Sigma_{x_0}\to\infty} \frac{\Sigma_{x_0}}{\Sigma_{x_0}+1} = 1; \quad \Lambda_0 = \lim_{\Sigma_{x_0}\to\infty} (1 - \frac{\Sigma_{x_0}}{\Sigma_{x_0}+1})\Sigma_{x_0} = 1$$

the best estimation, (E.45), is $\hat{x}_0 = 1 \times y_0$. For the next sample:

$$\Sigma_{x_1} = 1; \quad L_1 = \frac{1}{1+1} = 1/2; \quad \Lambda_1 = (1 - 1/2) * 1 = 1/2$$

so: $\hat{x}_1 = y_0 + 0.5 \times (y_1 - y_0) = (y_0 + y_1)/2$. It can be easily proved that:

$$\Sigma_{x_k} = \frac{1}{k}; \quad L_k = \frac{1/k}{(1/k)+1} = \frac{1}{k+1}; \quad \Lambda_k = (1 - \frac{1}{k+1})\frac{1}{k} = \frac{1}{k+1}$$

so: $\hat{x}_2 = (y_0 + y_1)/2 + \frac{1}{3}(y_2 - (y_0 + y_1)/2) = (y_0 + y_1 + y_2)/3$ and, indeed, for any k, the optimal prediction, $\hat{x}_k$, is the *sample average*. Of course, the Kalman filter can be applied to processes much more complex than the constant one. If a process noise $x_{k+1} = x_k + v_k$ were added, the Kalman filter would carry out a *weighted average*, giving more importance to recent samples. The calculations with $V = 0.05$ are left to the reader.

Stationary filter

It can be shown that the sequence Λ_k converges to Λ_∞ for positive-definite V and W, under general technical conditions. So, in most applications, at the expense of a larger error in the initial samples, only Λ_∞ is calculated, and the L_∞ associated with it. Afterwards, the Kalman filter is implemented as a time-invariant observer.

See Section 7.2 for further comments regarding practical implementation and engineering insight.

Example E.11. For a first-order marginally stable plant:

$$\begin{aligned} x_{k+1} &= x_k + u_k + v_k; \quad \sigma_v^2 = 1 \\ y_k &= x_k + w_k; \quad \sigma_w^2 = 0.3 \end{aligned}$$

the time-varying Kalman observer is given by:

$$\hat{x}_{k+1} = \hat{x}_k + u_k + L_k(y_{k+1} - (\hat{x}_k + u_k))$$

where the estimated variance (a scalar, as it is a simple one-dimensional system) starting with no information at all is:

$$\Lambda_{-1} = 10^6, \Lambda_0 = 0.3, \Lambda_1 = 0.2437, \Lambda_2 = 0.2417, \Lambda_3 = 0.2416, \Lambda_4 = 0.2416, \ldots$$

and the observer gains to correct the estimated stated once measurements $y_0, y_1, \ldots$ are taken are:

$$L_0 = 1, L_1 = 0.8125, L_2 = 0.8057, L_3 = 0.8054, L_4 = 0.8054, \ldots$$

The optimal strategy is believing the first measurement (with $L_0 = 1$, $\hat{x}_0 = y_0$), and, as more information is gathered, rely more on past information via the model. At stationary state, the optimal strategy drifts towards combining simulation (20%) and measurements (80%) for a variance 0.2416, better than that from the sensor alone (*i.e.*, 0.3). The first estimate has the same variance as the sensor (for $C = 1$).

Example E.12. Let us assume the same process and two measurement devices with independent measurement noise. The model will be:

$$\begin{aligned} x_{k+1} &= x_k + u_k + v_k; & \sigma_v^2 &= 1 \\ y_{1k} &= x_k + w_{1k}; & \sigma_{w1}^2 &= 0.3 \\ y_{2k} &= x_k + w_{2k}; & \sigma_{w2}^2 &= 0.3 \end{aligned}$$

The output matrix is $C = (1\ 1)^{\mathrm{T}}$, and the VC measurement matrix will be diagonal with entries 0.3.

Redoing the iterations, it yields:

$$\Lambda_{-1} = 10^4, \Lambda_0 = 0.15, \Lambda_1 = 0.1327, \Lambda_2 = 0.1325, \Lambda_3 = 0.1325, \ldots$$

the filter gain being:

$$L_0 = (0.5\ 0.5), L_1 = (0.442\ 0.442), L_2 = (0.4415\ 0.4415), L_3 = (0.4415\ 0.4415), \ldots$$

That is, the variance in the state estimation using two identical sensors is reduced to 0.1325 (once in steady-state).

Note that the first optimal prediction is to evaluate the mean of both readings, reducing to a half the confidence interval: $\hat{x}_0 = 0.5y_{10} + 0.5y_{20}$.

F

Robust Control Analysis and Synthesis

In Chapter 8, the description of the sources of uncertainty and some intuition about how to obtain robust designs were detailed. This appendix summarily details some robust-control related results outlined in the referred chapter. The interested reader, however, should seek further information in the bibliography [133, 134, 119, 55].

F.1 Small-gain Stability Analysis

In this section, analysis of the allowed "size" of *unstructured* modelling error for a particular closed-loop controller will be carried out based on the small-gain theorem.

The main idea is first to draw the block-diagram of a closed-loop system to be evaluated, including in it an uncertain transfer function matrix Δ. Then, it is "pulled-out" from its location to establish a structure to apply the small-gain theorem.

Additive uncertainty. Considering the feedback loop in Figure 8.3 on page 236 (left), it depicts a nominal closed loop with plant G and controller K, but an uncertain block, Δ, is added to the nominal plant to conform the modified plant, $G + \Delta$. Interpreting uncertainty Δ as block Σ_1 in Figure C.1 and the nominal loop as block Σ_2, calculating with the block-diagram the transfer function matrix between y_δ and u_δ, the result is:

$$u_\delta = K(Gu_\delta + y_\delta) \rightarrow u_\delta = (I - KG)^{-1}Ky_\delta$$

and, applying the push-through rule (B.5), $u_\delta = K(I - GK)^{-1}y_\delta$ so, by the small-gain theorem, if the loop is stable for $\Delta = 0$, Expression (8.11) is proved.

Remark F.1. According to Table C.1, if the same norm is to be used for all signals in the M–Δ structure, only the ∞-norm and the 1-norm may be used for robustness analysis, *i.e.*, minimisation of the ∞-norm of $K(I - GK)^{-1}$

maximises a modelling error bound. However, minimisation of the 2-norm of the same expression is *not* directly related to maximising modelling error bounds.

Multiplicative uncertainty. Under relative error uncertainty:

$$\Delta_* \stackrel{\text{def}}{=} G^{-1}\Delta \Rightarrow G_{true} = G(I + \Delta_*) \tag{F.1}$$

by carrying out similar operations to the previous case, the feedback loop in Figure 8.3 (right) can be shown to be stable for any Δ_* satisfying:

$$\|\Delta_*\|\|GK(I + GK)^{-1}\| < 1 \tag{F.2}$$

Input-multiplicative uncertainty. Input-multiplicative uncertainty, Δ_{*i}, is defined as:

$$G_{true} = (I + \Delta_{*i})G$$

Input and output uncertainties are equivalent in SISO systems, but this is not the case in MIMO ones as arbitrary matrices (even if unknown) do not commute. The size of the relative unstructured error at "input" should be multiplied by the plant's condition number (Appendix B.4.1) to pose an equivalent worst-case problem at the "output" [119]. This is an important issue in ill-conditioned plants such as distillation columns, regarding unavoidable amplification of actuator uncertainty (particularly relevant with valves, for instance).

Coprime factor uncertainty. The so-called coprime factor (CF) (see Appendix B.6) unstructured-uncertainty description [133, 85] is expressed as:

$$G = (M + \Delta_M)^{-1}(N + \Delta_N) \tag{F.3}$$

where Δ_M and Δ_N are "denominator" and "numerator" uncertainties. For example, as pointed out in Remark 8.7 on page 237, with the CF representation $\Delta_M = 0.01$ is a stable transfer function. Note that in MIMO systems, "denominator" and "numerator" are transfer matrices.

Denoting as $\Delta = [\Delta_N \ \Delta_M]$ a stable matrix modelling error, after some operations [133], the condition for robust stability is:

$$\| \begin{pmatrix} K \\ I \end{pmatrix} (I - GK)^{-1} M^{-1} \| \times \|\Delta\| \leq 1 \tag{F.4}$$

CF uncertainty allows suitable handling of problems regarding uncertainty in unstable plants, as the following example shows.

Example F.2. Let us analyse the robust stability of a control loop with a nominal design for a plant model $G_{model} = 1/(s - 1)$, trying to guarantee stability with a real plant $G_{true} = 1/(s - 1.01)$. Note that, with an additive uncertainty model,

$$\Delta = G_{true} - G_{model} = \frac{0.01}{(s - 1.01)(s - 1)}$$

is unstable so conditions for applying (8.11) do *not* hold. However, in this case the normalised coprime factorisation of G_{model} is:

$$G_{model} = \left(\frac{1-s}{1.414+s}\right)^{-1} \frac{-1}{1.414+s}$$

so the modelling error is $\Delta_M = 0.01/(s+1.414)$, stable and (F.4) can be applied if a nominally stabilising regulator is available. The size of the error (peak of frequency response) is 0.0071. The command `sncfbal` renders the above factorisation:

```
[sysnlcf,sig,sysnrcf] = sncfbal(grsys);
[an,bn,cn,dn]=unpck(sysnlcf);nocf=ss(an,bn,cn,dn);
zpk(nocf);n=nocf(1,1); m=nocf(1,2);
```

For example, a proportional controller, $K = -2$, allows an unstructured CF uncertainty margin of 0.3162, thus guaranteeing stability with G_{true} in this trivial case. The margin can be calculated with the MATLAB® command `emargin`:

```
grss=ss(gm); kss=ss(k); grsys=pck(grss.a,grss.b,grss.c,grss.d);
sysk2=pck(kss.a,kss.b,kss.c,kss.d); emargin(grsys,sysk2)
```

F.2 Structured Uncertainty

As commented in the main text, *structured* uncertainty arises in cases such as:

- uncertain real parameters,
- diagonal actuator and sensor uncertainties,
- a general configuration with uncertain models in various independent subsystems.

Figure 8.5 on page 240 depicted a system with multiple sources of uncertainty, "pulled up" from its original position in a block-diagram. In this way, a general configuration for robust stability analysis is built, where M represents a linear (feedback) system and Δ_i are uncertainty blocks. Each of them can be:

- **scalars:** (1×1 matrices), representing either uncertain parameters (denoted as *real* uncertainty) or uncertain SISO (1×1) subsystems and undermodelling (denoted as *complex* uncertainty). For example, an uncertain mass varying between 5 and 7 Kg could be represented by a nominal parameter 6 plus a real uncertainty block with a known bound in norm $\|\Delta\| < 1$ (Δ pulled away to the structure in Figure 8.5). Neglected actuator dynamics will be represented by an uncertain complex number with known bound.
- **full complex transfer matrices:** representing full complex unstructured uncertainty of a particular subsystem, of generic dimension $m \times n$. The unstructured uncertainty case is then a particular case of only *one* full block.

Uncertainty blocks can be *repeated* if, for example, the same physical parameter appears in several transfer function coefficients. This issue cannot, in general, be represented in block-diagram form but some software tools allow the user to specify that option.

It is easy to see that all uncertainty sources can be thought of forming part of a bigger block-diagonal Δ matrix (Figure 8.5). It will be assumed that a size bound for each of the blocks, $\|\Delta_i\| < d_i$, is known. In fact, to simplify developments, without loss of generality, it will be assumed that the d_i will be included as error-free scaling factors in M, and Δ is a block-triangular matrix with normalised unity maximum size: $\bar{\sigma}(\Delta) < 1$.

If M and all Δ_i are linear operators, the characteristic closed-loop equation (C.17) is:

$$\det(I - M\Delta) = 0$$

so, from MIMO Nyquist criterion (Section 4.5.1), in a structured uncertainty setup (M, Δ),we are interested in finding the maximum scaling factor, α, so that there exists no matrix with the Δ structure so that $\det(I - \alpha M\Delta) = 0$.

This comes from expression (C.17): as with $\alpha = 0$, $\det(I - 0 \cdot M\Delta) = 1$ evidently does *not* encircle the origin, instability will not arise until the possibility of touching the origin is allowed.

Definition F.3 (Structured singular value). *In a structured uncertainty setup (M, Δ), assuming $\bar{\sigma}(\Delta) < 1$, denoting as α_m the minimum scaling factor $\alpha > 0$ so that there exists a matrix with the block-diagonal structure specified by Δ and $\bar{\sigma}(\Delta) = \alpha$ so $\det(I - M\Delta) = 0$, the structured singular value μ_Δ is defined as*

$$\mu_\Delta = \frac{1}{\alpha_m}$$

The notation μ_Δ stresses the fact that for the same system, the value of μ is different for different uncertainty structures. In fact, as M depends on frequency, μ_Δ is itself a function of frequency, $\mu_\Delta(\omega)$. The concept was historically introduced in [43, 110].

The overall system will be robustly stable if $\mu_\Delta(\omega) < 1$ for all ω, as it means that uncertainty must be scaled by a factor α greater than 100%. If δ is a full uncertain matrix, it can be shown that $\mu_\Delta = \bar{\sigma}(\Delta)$.

Obviously, the interest of this definition is that there are tools for calculating μ_Δ. The available tools are approximate in a general situation, and calculate a lower and upper bound for μ_Δ, except for some cases where exact solutions can be found. There are several algorithms for computing these bounds, with different computational complexity. Further details on properties and computation of the structured singular value can be found in [133]. The MATLAB® μ-toolbox implements some of them in its command `mu`, called with two arguments: the plant, M, and an uncertainty structure description matrix (see the toolbox documentation for details).

Example F.4. This is an example of how unstructured analysis can be overly conservative in some cases. A plant and a controller given by:

$$G = \begin{pmatrix} \frac{1}{s+1} & \frac{15}{s+1} \\ 0 & \frac{4}{(s+2)^2} \end{pmatrix}; \qquad K = \begin{pmatrix} 10 & 0 \\ 0 & 10 \end{pmatrix}$$

must be analysed against robustness to diagonal multiplicative input uncertainty. The transfer function to be analysed in the $M - \Delta$ structure in Figure 8.5 is:

```
g2=minreal(feedback(series(gpss,k2ss),eye(2)));
```

and the μ-toolbox in MATLAB® yields a comparison between its norm (unstructured uncertainty) and the structured singular value for diagonal complex uncertainty (unmodelled actuator dynamics).

```
g2p=pck(g2.a,g2.b,g2.c,g2.d); w=logspace(-1,2); gpf=frsp(g2p,w);
[bd,dv,sss,pv]=mu(gpf,[1 0;1 0]); vplot('liv,lm',vnorm(gpf),bd);
```

where `bd` is an approximate bound for μ, and `vnorm(gpf)` is the usual maximum singular value norm. The result appears in Figure F.1. The plant is triangular, so actuator uncertainty only affects each of the loops. However, the full uncertainty description wrongly considers the possibility of a non-existent actuator cross-coupling exciting the off-diagonal dynamics, giving a much more conservative bound.

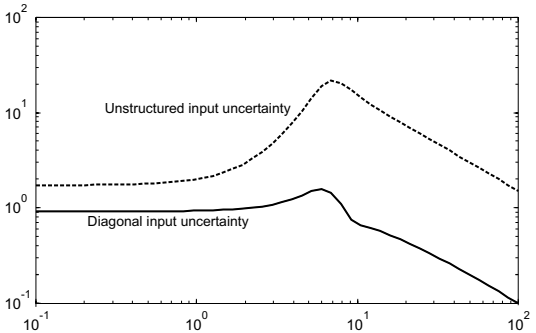

Figure F.1. Comparison between inverse uncertainty bounds with unstructured and structured uncertainty analysis

F.2.1 Robust Performance

As discussed in Section 8.6.1, a nominal-performance problem can be stated, with suitable weights incorporated into the generalised plant, as achieving a particular transfer function norm $W_e S$ below 1. The same idea, applied to

modelling error sizes, can be used to pose RS problems as achieving a norm less than 1 between output and input of the uncertainty block Δ.

In Section 8.5.4, it was argued that the robust performance (RP) problem is the actual problem of interest in feedback control design.

Mixed sensitivity combines nominal performance and robust stability in a single index. However, the design does not guarantee RP in a general case[1]. The structured uncertainty framework enables some robust performance analysis problems to be cast as robust stability ones.

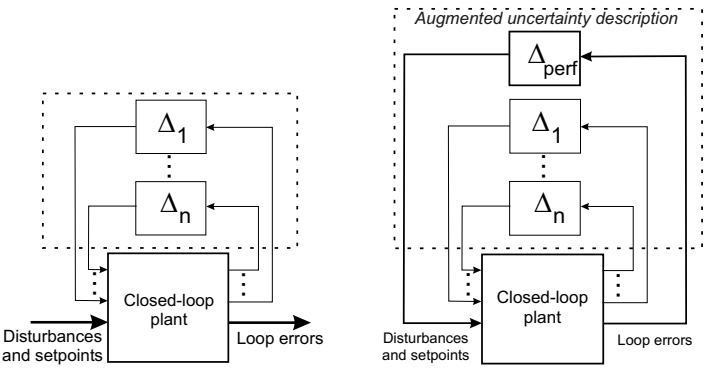

Figure F.2. Robust performance (norm-based criterion) recast as (structured) robust stability

Robust performance analysis implies, from the block-diagram of Figure F.2 (left), that for any Δ verifying $\bar{\sigma}(\Delta) < 1$, the norm from disturbances to errors should be less than 1. Hence, it is equivalent to inserting an additional fictitious Δ_{perf} with norm 1 and checking the robust stability with the augmented uncertainty structure to the right of the referred figure [133].

F.3 Additional Design Techniques

A detailed description of the available techniques for robust control synthesis is out of the scope of this book. Mixed sensitivity has been discussed in Section 8.6.1. In this section, an exposition of a widespread loop-shaping procedure, interesting for both stable and unstable plants, will be now outlined based on an academic example. The interested reader should consult the references for further details.

[1] Although, fortunately, it is approximately verified in SISO systems and well-conditioned MIMO ones.

F.3.1 Robust Stabilisation

Direct norm-optimisation can be used to design the controller that stabilises a particular plant with maximum uncertainty bounds. An example of this, with an unstable plant, tolerating the maximum amount of coprime factor uncertainty is detailed below.

Example F.5 (Robust stabilisation). Maximising the robustness margin to uncertainty in normalised coprime factor form, for a system such as:

$$\begin{pmatrix} \frac{1}{(s-1)(s+2)} \\ \frac{s}{(s-1)(s+2)} \end{pmatrix}$$

modelling an unstable mechanical system with position and speed measurements, can be carried out with the command `ncfsyn`, from the MATLAB® μ-toolbox:

```
g1=1/(s-1)/(s+2); g2=s/(s-1)/(s+2);
g=[g1; g2]; g=ss(g);
sys=pck(g.a,g.b,g.c,g.d);
[sysk, emax]=ncfsyn(sys,1.1);
[ka,kb,kc,kd]=unpck(sysk);
k=ss(ka,kb,kc,kd);
```

In this case, the resulting controller, `k`, is designed to allow for 1.1 times less error size than the "optimal" one, for numeric precision reasons. Of course, maximising robust stability margins does not imply any further performance requirements. An input disturbance step yields a behaviour depicted in Figure F.3 on the following page (two-sensor maximum-margin label). The maximum allowed uncertainty size is 0.197 units.

F.3.2 McFarlane-Glover Loop Shaping

A refinement of the previous methodology [85, 86] to add performance objectives is also a widely used choice. Another example describes the basic steps.

Example F.6. To add performance requirements, the above procedure may be applied to a *shaped* plant, $G_s = W_2 G W_1$, where weights W_1 and W_2 are designed so that the loop has high-gain at frequencies where tight control is needed (input and output weights allow for specific design on a particular actuator or sensor), and low-gain at frequencies where no control activity is wished. After accomplishing the design, the final regulator must include the weights (as they are not part of the real plant), contrary to the mixed-sensitivity design in Section 8.6.1. In our case, adding a weight for integral action $W_1 = (s+6)/s$, $W_2 = I_{2\times 2}$, the code that implements this design is:

```
gr=[g1; g2]; g=w2*gr*w1;g=ss(g);
sys=pck(g.a,g.b,g.c,g.d);
[sysk,emax]=ncfsyn(sys,1.1);
[ka,kb,kc,kd]=unpck(sysk);
k=ss(ka,kb,kc,kd);
k=w1*minreal(k)*w2;
```

Note that the synthesis designs the optimal regulator assuming uncertainty on the weighted plant. This is not true in practical terms, so the resulting regulator tolerates less error than the "optimal" one in the previous example. In this case,the final allowed unstructured coprime factor error on the unweighted plant, G, is 0.17 (slightly less than 0.19 on the previous design, intuitively expected, as slow integral action does not usually involve severe additional stability problems). The step responses for output 1 (position) to a unit input disturbance of both examples are depicted in Figure F.3. The response of a SISO unweighted maximum stability margin design using only the first sensor is also depicted for the sake of comparison (the stability margin is 0.14). Further increases on the weights will improve performance at the expense, as usual, of reducing stability margins. As usual, adding extra sensors improved the achievable performance and/or robustness margins.

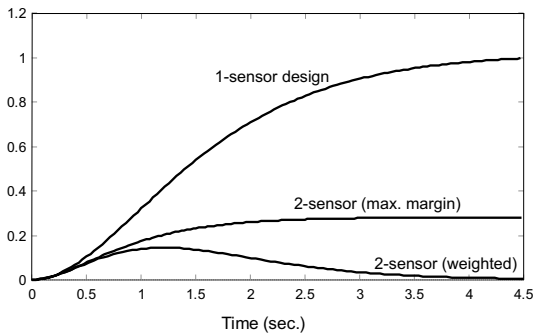

Figure F.3. Robust stabilisation of a second-order unstable plant (Examples F.5 and F.6, nominal responses to a step input disturbance)

Interestingly, the resulting regulator with the presented procedure can be realised as an observer plus state feedback control law [117] (on the weighted plant). The observer structure is advantageous towards understanding the configuration in a "classical" language and its ease of implementation in gain-scheduled configurations (discussed in Section 9.5.2) [62].

References

1. P. Albertos. Block multirate input-output model for sampled-data control systems. *IEEE Trans. Automatic Control*, AC-35:1085–1088, 1990.
2. P. Albertos and A. Crespo. Real-time control of non-uniformly sampled data systems. *Control Engineering Practice*, 7:445–458, 1999.
3. P. Albertos and M. Olivares. The control effort and its implications. In *IFAC Workshop in Advances in Control Education*, Gold Coast, Australia, 2000. Elsevier.
4. P. Albertos and J.V. Roig. *Problemas de Ingeniería de Control (in Spanish).* Publicaciones UPV, Valencia, 2000.
5. P. Albertos and A. Sala, editors. *Iterative Identification and Control (Advances in Theory and Applications).* Springer-Verlag, London, 2002.
6. P. Albertos, M. Salgado and M. Olivares. Are delays in digital control implementation always bad? In *Proc. Asian Control Conference*, volume 1, pages 634–639, Shanghai, China, 2000.
7. P. Albertos, J. Salt and J. Tornero. Dual-rate adaptive control. *Automatica*, 32(7):1027–1030, 1996.
8. F. Allgower and A. Zheng, editors. *Nonlinear Model Predictive Control*, volume 26 of *Progr. in Syst. Contr. Theory.* Birkhauser, Berlin, 2000.
9. B.D.O. Anderson and J.B. Moore. *Optimal Filtering.* Prentice Hall, Englewood Cliffs, NJ, 1979.
10. C. Angeli. Online expert system for fault diagnosis in hydraulic systems. *Expert Systems*, 16:115–120, 1999.
11. P.J: Antsaklis and A.N. Michel. *Linear Systems.* McGraw-Hill, NY, 1997.
12. A. Arbel, I. Rinard and R. Shinnar. Dynamics and control of fluidized catalytic crackers, 3. designing the control system: Choice of manipulated and controlled variables. *Ind. Eng. Chem. Res.*, 34:3014–3026, 1996.
13. A. Astolfi, R. Ortega and R. Sepulchre. Passivity–based control of nonlinear systems. *In [17]*, pages 39–76, 2001.
14. K.J. Åström. *Introduction to Stochastic Control Theory.* Academic Press, NY, 1970.
15. K.J. Åström. Limitations on control system performance. *European Journal of Control*, 6:1–19, 2000.
16. K.J. Åström. Model uncertainty and feedback. *In [5]*, pages 63–98, 2002.

17. K.J. Åström, P. Albertos, M. Blanke, A. Isidori, W. Schaufelberger and R. Sanz, editors. *Control of Complex Systems*. Springer-Verlag, London, 2001.
18. K.J. Åström and R.D. Bell. Drum-boiler dynamics. *Automatica*, 36(3):363–378, 2000.
19. K.J. Åström and K. Eklund. A simple non-linear drum-boiler model. *International Journal of Control*, 22:739–740, 1975.
20. K.J. Åström and T. Hägglund. *PID Controllers: Theory, Design and Tuning*. Instrument Society of America, North Carolina, USA, second edition, 1995.
21. K.J. Åström and B. Wittenmark. *Computer Controlled Systems*. Prentice Hall, Englewood Cliffs, NJ, 1984.
22. K.J. Åström and B. Wittenmark. *Adaptive Control*. Addison-Wesley, second edition, 1995.
23. B. Bamieh, B. Francis and A. Tannenbaum. A lifting technique for linear periodic systems with aplications to sampled-data control. *Systems and Control Letters*, 17:79–88, 1991.
24. R. Bellmann. *Dynamic Programming*. Princeton University Press, 1957.
25. S.P. Bhattacharyya, H. Chapellat and L.H. Keel. *Robust Control – The Parametric Approach*. Prentice Hall, Upper Saddle River, NJ, 1995.
26. R. Bitmead and A. Sala. Data preparation for identification. *In [5]*, pages 41–62, 2002.
27. R.R. Bitmead, M. Gevers and V. Wertz. *Adaptive Optimal Control – The Thinking Man's GPC*. Prentice Hall, NY, 1990.
28. H.W. Bode. Relations between attenuation and phase in feedback amplifier design. *Bell System Technical Journal*, 19:421–454, 1940.
29. H.W. Bode. *Network analysis and feedback amplifier design*. D. Van Nostrand Co., NY, 1945.
30. G.E.P Box and G.M. Jenkins. *Time Series Analysis, forecasting and control*. Holden-Day, Oakland, California, 1976.
31. S.A. Boyer. *SCADA: Supervisory Control and Data Acquisition*. ISA Society, NC, second edition, 1999.
32. R.D. Braatz and M. Morari. Minimizing the Euclidean condition number. *SIAM J. on Control and Optimization*, 32(6):1763–1768, 1994.
33. E.H. Bristol. on a new measure of interactions for multivariable process control. *IEEE Trans. Automatic Control*, 11:133–134, 1966.
34. E.F. Camacho and C. Bourdons. *Model Predictive Control in the Process Industry*. Springer-Verlag, 1995.
35. P.J. Campo and M. Morari. Achievable closed-loop properties of systems under decentralized control: Conditions involving the steady-state gain. *IEEE Trans. Automatic Control*, 39(5):932–943, 1994.
36. J. Chen and G. Gu. *Control-Oriented System Identification, an $\mathcal{H}_\infty$ Approach*. Wiley, NY, 2000.
37. L.H. Chiang, E.L. Russell and R.D. Braatz. *Fault Detection and Diagnosis in Industrial Systems*. Springer-Verlag, 2001.
38. I.L. Chien, C.J. Wang, D.S.H. Wong, C.H. Lee *et al.* Experimental investigation of conventional control strategies for a heterogeneous azeotropic distillation column. *J. of Process Control*, 10:333–340, 2000.
39. M.S. Chiu and Y. Arkun. A methodology for sequential design of robust decentralised control systems. *Automatica*, 28(5):997–1001, 1992.
40. D.W. Clarke and C. Mohtadi. Properties of generalized predictive control. *Automatica*, 25(6):859–875, 1989.

41. B. Coleman and J. Babu. *Techniques of Model Based Control*. Prentice Hall, Englewood Cliffs, NJ, 2002.
42. J.C. Doyle. Guaranteed margins for LQG regulators. *IEEE Trans. Automatic Control*, AC-23:756–757, 1978.
43. J.C. Doyle. Analysis of feedback systems with structured uncertainties. *IEE Proc.*, 129-D(6):242–250, November 1982.
44. J.C. Doyle, K. Glover, P.P. Khargonekar and B.A. Francis. State space solutions to standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems. *IEEE Trans. Automatic Control*, 34:831–846, 1989.
45. P.V. Kokotović (Editor). *Foundations of Adaptive Control*, volume 160 of *Lecture Notes in Control and Information Sciences*. Springer-Verlag, 1991.
46. K.T. Erickson and J.L. Hedrick. *Plantwide Process Control*. Wiley, NY, 1999.
47. W.R. Evans. *Control-System Dynamics*. McGraw-Hill, NY, 1954.
48. G.F. Franklin, D.J. Powell and M.L. Workman. *Digital Control of Dynamic Systems*. Addison-Wesley, third edition, 1997.
49. G.F. Franklin, J.D. Powell and A. Emami-Naeini. *Feedback Control of Dynamic Systems (third edition)*. Addison-Wesley, 1994.
50. T. Fuji, T. Tsujino, K.Suematu, K.Sasaki and Y.Murata. Multivariable control of a magnetic levitation system with an Y-shape iron plate. In K. Furuta, editor, *Proc. of IFAC Symp. on Advances in Control Education*, pages 25–28. Elsevier, 1994.
51. T. Fujii. Design of tracking systems with LQ optimality and quadratic stability. In G. Goodwin and R. Evans, editors, *Proc. 12th IFAC World Congress*, volume 2, pages 479–486. Elsevier, 1993.
52. M. Gevers. Identification for control. *Annual Reviews in Control*, 20:95–106, 1997.
53. G.H. Golub and C.F. van Loan. *Matrix Computations*. John Hopkins Univ. Press, third edition, 1996.
54. G.C. Goodwin, S.F. Graebe and M.E. Salgado. *Control System Design*. Prentice Hall, Englewood Cliffs, NJ, 2001.
55. M. Green and D.J.N. Limebeer. *Linear Robust Control*. Prentice Hall, Englewood Cliffs, 1995.
56. G. Gu and P.P. Khargonekar. A class of algorithms for identification in $\mathcal{H}_\infty$. *Automatica*, 28:299–312, 1992.
57. R. Hanus, M. Kinnaert and J.L. Henrotte. Conditioning technique, a general anti-windup and bumpless transfer method. *Automatica*, 23(6):729–739, 1987.
58. J.W. Helton and M.R. James. *Extending $\mathcal{H}_\infty$ Control to Nonlinear Systems*. SIAM Publications, 1999.
59. M. Hovd, R.D. Braatz and S. Skogestad. SVD controllers for $\mathcal{H}_2$, $\mathcal{H}_\infty$, and μ-optimal control. *Automatica*, 33:433–439, 1996.
60. M. Hovd and S. Skogestad. Sequential design of decentralised controllers. *Automatica*, 30:1601–1607, 1994.
61. T. Hu and Z. Lin. *Control Systems With Actuator Saturation: Analysis and Design*. Birkhäuser, 2001.
62. R.A. Hyde and K. Glover. The application of scheduled $\mathcal{H}_\infty$ controllers to a VSTOL aircraft. *IEEE Trans. Automatic Control*, 38(7):1021–1039, 1993.
63. R. Isermann. *Digital Control Systems*. Springer-Verlag, second edition, 1989.
64. K. James. *PC Interfacing and Data Acquisition: Techniques for Measurement, Instrumentation and Control*. Newnes, 2000.

65. L. Jaulin, M. Kieffer, O. Didrit and E. Walter. *Applied Interval Analysis*. Springer-Verlag, 2001.
66. C.D. Johnson. *Process Control Instrumentation Technology*. Prentice Hall, Englewood Cliffs, NJ, 2002.
67. T. Kailath. *Linear Systems*. Prentice Hall, Englewood Cliffs, NJ, 1980.
68. R.E. Kalman. A new approach to linear filtering and prediction problems. *Transactions ASME, Series D, J. Basic Eng.*, 82:34–45, 1960.
69. R.E. Kalman. On the general theory of control systems. In *Proc. 1st IFAC Congress*, pages 481–492, Moscow, 1961.
70. R.E. Kalman and R.S. Bucy. New results in linear filtering and prediction theory. *Trans. ASME Journal of Basic Engineering*, 83:95–108, 1961.
71. I. Kaminer, A.M. Pascoal, P.P. Khargonekar and E.E. Coleman. A velocity algorithm for the implementation of gain-scheduled controllers. *Automatica*, 31:1185–1191, 1995.
72. D. Kaplan and L. Glass. *Understanding Nonlinear Dynamics*. Springer-Verlag, 1995.
73. L.H. Keel and S.P. Bhattacharyya. Robust, fragile or optimal? *IEEE Trans. Automatic Control*, 42:1098–1105, 1997.
74. H.K. Khalil. *Nonlinear Systems*. Prentice Hall, third edition, 2002.
75. R.E. Klein. Using bicycles to teach system dynamics. *IEEE Control Systems Magazine*, 9(3):4–9, 1989.
76. P.V. Kokotović. The joy of feedback: Nonlinear and adaptive. *IEEE Control Systems Magazine*, June:7–17, 1992.
77. R.L. Kosut, M. Lau and S. Boyd. Set-membership identification of systems with parametric and nonparametric uncertainty. *IEEE Trans. Automatic Control*, 37:929–941, July 1992.
78. B. Kuo and F. Golnaraghi. *Automatic Control Systems*. John Wiley and Sons, NY, 8th edition, 2002.
79. I.D. Landau. From robust control to adaptive control. *Control Engineering Practice*, 7(9):1113–1124, 1999.
80. D.J. Leith and W.E. Leithead. Gain-scheduled and nonlinear systems: dynamic analysis by velocity-based linearisation families. *International Journal of Control*, 70:289–317, 1998.
81. F.L. Lewis. *Optimal Control*. Wiley-Interscience, NY, 1986.
82. G. Li. On the structure of digital controllers with finite word length considerations. *IEEE Trans. Automatic Control*, 43:689–693, 1998.
83. D. Liberzon and A.S. Morse. Basic problems in stability and design of switched systems. *IEEE Control Systems*, 19(5):59–70, 1999.
84. L. Ljung. *System Identification: Theory for the User*. Prentice Hall, Englewood Cliffs, NJ, second edition, 1999.
85. D.C. MacFarlane and K. Glover. *Robust controller design using normalized coprime factor plant descriptions*. Springer, New York, 1990.
86. D.C. MacFarlane and K. Glover. A loop shaping design procedure using $\mathcal{H}_\infty$-synthesis. *IEEE Trans. Automatic Control*, AC-37:759–769, 1992.
87. P.M. Mäkilä. Puzzles in systems and control. In A. Garulli, A. Tesi and A. Vicino, editors, *Robustness in Identification and Control*, LNCS, pages 242–257. Springer-Verlag, London, 1999.
88. V. Manousiouthakis, R. Savage and Y. Arkun. Synthesis of decentralized process control structures using the concept of block relative gain. *AIChE Journal*, 32:991–1003, 1986.

89. T.E. Marlin. *Process Control*. Chem. Eng. McGraw-Hill, Singapore, second edition, 2000.
90. D.Q. Mayne, J.B. Rawlings, C.V. Rao and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.
91. C.D. Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial & Applied Mathematics (SIAM), 2001.
92. D.C. Montgomery. *Introduction to Statistical Quality Control*. Wiley, NY, 1985.
93. A. Niederlinski. Heuristic approach to the design of linear multivariable control systems. *Automatica*, 7:691–701, 1971.
94. K. Ogata. *Modern Control Engineering*. Prentice Hall, fourth edition, 2002.
95. A.V. Oppenheim, R.W. Schafer and J.R. Buck. *Discrete-Time Signal Processing*. Prentice Hall, Englewood Cliffs, NJ, second edition, 1999.
96. M. Valles P. Albertos and A. Valera. Controller updating under operational logic changes. In *Proc. of IFAC Workshop on Modelling and Analysis of Logic Controlled Dynamic Systems*. Elsevier.
97. A. Packard and J.C. Doyle. The complex structured singular value. *Automatica*, 29(1):71–109, 1993.
98. A. Papoulis and S. Unnikrishna. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, fourth edition, 2001.
99. E.A. Parr. *Industrial Control Handbook*. Industrial Press (Reed-Elsevier), third edition, 1998.
100. Y. Peng, D. Vrancić and R. Hanus. Anti-windup, bumpless and conditioned transfer techniques for PID controllers. *IEEE Control Systems*, 16(4):48–57, 1996.
101. M. Perez and P. Albertos. Self oscillating and chaotic behaviour of a PI controlled CSTR with control valve saturation. *Journal of Process Control*, page In print, 2003.
102. J.W. Polderman and J.C. Willems. *Introduction to Mathematical Systems Theory: A Behavioral Approach*. Springer-Verlag, 1998.
103. L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze and E.F. Mischenko. *The Mathematical Theory of Optimal Processes*. Pergamon Press, London, 1964.
104. W.H. Press, S.A. Teukolsky, W.T. Vetterling and B.P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, second edition, 2002.
105. Q. F. Qiu, G. P. Rangaiah and P. R. Krishnaswamy. Application of a plant-wide control design to the HDA process. *Computers & Chemical Engineering*, 27(1):73–94, 2003.
106. S.J. Quin and T.A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–754, 2003.
107. J.B. Rawlings. Tutorial overview of model predictive control. *IEEE Control Systems Magazine*, 20(3):38–52, 2000.
108. Paul M. Frank Ron J. Patton and Robert N. Clark (Eds). *Issues of Fault Diagnosis for Dynamic Systems*. Springer Verlag, London, 2002.
109. W.J. Rugh and J.S. Shamma. Research on gain scheduling. *Automatica*, 36:1401–1425, 2000.
110. M.G. Safonov. Stability margins of diagonally perturbed multivariable feedback systems. *IEE Proc.*, 129-D(6):251–256, November 1982.
111. A. Sala and P. Albertos. Open-loop iterative learning control. *In [5]*, pages 99–119, 2002.

112. A. Sala and A. Esparza. Reduced-order controller design via iterative identification and control. *European Journal of Control*, 9(1):103–115, April 2003.
113. R.S. Sánchez-Peña and M. Sznaier. *Robust Systems, Theory and Applications*. Wiley, NY, 1998.
114. R. Sanz, C. Pfister, W. Schaufelberger and A. de Antonio. Software for complex controllers. *In [17]*, pages 143–164, 2001.
115. L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators*. Adv. Textbooks Control Signal Proc. Springer-Verlag, second edition, 2000.
116. D. E. Seborg, T. F. Edgar and D.A. Mellichamp. *Process dynamics and control*. Wiley, NY, 1989.
117. J. Sefton and K. Glover. Pole-zero cancellations in the general $\mathcal{H}_\infty$ problem with reference to a two-block design. *Syst. Control Letters*, 14:295–306, 1990.
118. R.E. Skelton. Model error concepts in control design. *Int. Journal of Control*, 49(5):1725–1753, 1989.
119. S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control*. Wiley, NY, 1996.
120. J.J.E. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, 1991.
121. G. Stanley, M. Marino-Galarraga and T.J. McAvoy. Shortcut operability analysis. 1. the relative disturbance gain. *Industr. and Eng. Chemistry Proc. Design and Development*, 24:1181–1188, 1985.
122. R.F. Stengel. *Optimal Control and Estimation*. Dover, New York, second edition, 1994.
123. G. Stephanopoulos. *Chemical Process Control*. Prentice Hall, New Jersey, 1989.
124. D.J. Stilwell and W.J. Rugh. Stability-preserving interpolation methods for gain scheduled controllers. *Automatica*, 36(5):665–671, 2000.
125. G. Strang. *Linear Algebra and Its Applications*. Thomson Publishing, third edition, 1988.
126. J.O. Trierweiler and S. Engell. A case study for control structure selection: air separation plant. *J. of Process Control*, 10:237–243, 2000.
127. A. Tuladhar. *Dynamic Modelling and Control of a Paper Machine Headbox-Ms. Thesis*. University of British Columbia, Vancouver, Canada, 1995.
128. R.K. Wood and M.W. Berry. Terminal composition control of a binary distillation column. *Chem. Eng. Sci.*, 28:1707–1717, 1973.
129. K.-L. Wu, C.-C. Yu, W.L. Luyben and S. Skogestad. Reactor/separator processes with recycles-2. design for composition control. *Computers & Chemical Engineering*, 27(3):401–421, 2003.
130. E. Zafiriou. Robust model predictive control of processes with hard constraints. *Computers in Chem. Eng.*, 14:359–371, 1990.
131. G. Zames. Feedback and optimal sensitivity: Model reference transformations, multiplicative seminorms, and approximative inverses. *IEEE Trans. Automatic Control*, AC-26:301–320, 1981.
132. G. Zhang, M. Xu and S. Xu. Design of the expert system for fault diagnosis of 200 MW turbine generator set. *Advances in Modelling and Analysis (B)*, 40:51–59, 1998.
133. K. Zhou and J.C. Doyle. *Essentials of Robust Control*. Prentice Hall, Englewood Cliffs, NJ, 1998.
134. K. Zhou, J.C. Doyle and K. Glover. *Robust and Optimal Control*. Prentice Hall, Englewood Cliffs, NJ, 1995.

Index