Lecture notes for the course Advanced Control of Industrial Processes

Morten Hovd
Institutt for Teknisk Kybernetikk, NTNU

November 3, 2009

Contents

Chapter 1

Introduction

1.1 Scope of note

This note originates from course notes for the course 'Design og vedlikehold av reguleringsfunksjoner',given in cooperation between Cyberlab.Org AS and the Engineering Cybernetics Department of the Norwegian University of Science and Technology (NTNU). Parts of this note has later been used in the course Advanced Process Control, which has been offered by the Engineering Cybernetics Department in cooperation with the Chemical Engineering Department at NTNU. The most recent version is further adapted for the course Advanced Control of Industrial Processes, offered by the Engineering Cybernetics Department.

The target audience is students in the fourth year of the 5-year MSc programme in Engineering Cybernetics. Thus, the note is written for people with a relatively broad background in control engineering, who are familiar with both frequency response and time domain analysis. Whereas frequency response (or Laplace domain) analysis is used predominantly for single-loop control, time domain description (in discrete time) is used extensively in the description of multivariable Model Predictive Control.

Concepts from systems theory such as (state) controllability and (state) observability are also used without introduction[1].

It is this authors intent to keep the focus on issues of importance for industrial applications. Frequently, results are presented and discussed, without presenting formal proofs. Readers interested in mathematical proofs will have to consult the references.

Readers are also assumed to be familiar with finite dimensional linear algebra, i.e., have a working knowledge of matrices and vectors. Although the subject matter is by necessity of a mathematical nature, mathematical elegance is often sacrificed for clarity. In addition to students of control engineering, students with a Process Systems Engineering specialization within Chemical Engineering should also be able to read and benefit from this note.

[1] Although the importance of these concepts are not exaggerated in this work.

1.2 Why is process control needed?

Many texts on process control implicitly assume that it is obvious when and why control is needed. It seems obvious that even a moderately complex process plant will be very difficult to operate without the aid of process control. Nevertheless, it can be worthwhile to spend a few minutes thought on why process control is needed. In the following, a short and probably incomplete list of reasons for the need of process control is provided, but the list should illustrate the importance of process control in a process plant.

1. *Stabilizing the process.* Many processes have integrating or unstable modes. These have to be stabilized by feedback control, otherwise the plant will (sooner or later) drift into unacceptable operating conditions. In the vast majority of cases, this stabilization is provided by automatic feedback control[2]. Note that in practice, "feedback stabilization" of some process variable may be necessary even though the variable in question is asymptotically stable according to the control engineering definition of stability. This happens whenever disturbances have sufficiently large effect on a process variable to cause unacceptably large variations in the process variable value. Plant operators therefore often use the term "stability" in a much less exact way than how the term is defined in control engineering. A control engineer may very well be told that e.g., "this temperature is not sufficiently stable", even though the temperature in question is asymptotically stable.

2. *Regularity.* Even if a process is stable, control is needed to avoid shutdowns due to unacceptable operating conditions. Such shutdowns may be initiated automatically by a shutdown system, but may also be caused by outright equipment failure.

3. *Minimizing effects on the environment.* In addition to maintaining safe and stable production, the control system should also ensure that any harmful effects on the environment are minimized. This is done by optimizing the conversion of raw materials[3], and by maintaining conditions which minimize the production of any harmful by-products.

4. *Obtaining the right product quality.* Control is often needed both for achieving the right product quality, and for reducing quality variations.

5. *Achieving the right production rate.* Control is used for achieving the right production rate in a plant. Ideally, it should be possible to adjust the production rate at one point in the process, and the control system should automatically adjust the throughput of up- or downstream units accordingly.

[2]However, some industries still use very large buffer tanks between different sections in the process. For such tanks it may be sufficient with infrequent operator intervention to stop the buffer tank from overfilling or emptying.

[3]Optimizing the conversion of raw materials usually means maximizing the conversion, unless this causes unacceptably high production of undesired by-products, or requires large energy inputs.

6. *Optimize process operation.* When a process achieves safe and stable operation, with little down-time, and produces the right quality of product at the desired production rate, the next task is to optimize the production. The objective of the optimization is normally to achieve the most cost-effective production. This involves identifying, tracking and maintaining the optimal operating conditions in the face of disturbances in production rate, raw material composition and ambient conditions(e.g., atmospheric temperature). Process optimization often involves close coordination of several process units, and operation close to process constraints.

The list above should illustrate that process control is vital for the operation of process plants. Even plants of quite moderate complexity would be virtually impossible to operate without process control. Even where totally manual operation is physically feasible, it is unlikely to be economically feasible due to product quality variations and high personnel costs, since a high number of operators will be required to perform the many (often tedious) tasks that the process control system normally handles.

Usually many more variables are controlled than what is directly implied by the list above, there are often control loops for variables which have no specification associated with them. There are often good reasons for such control loops - two possible reasons are

1. *To stop disturbances from propagating downstream.* Even when there are no direct specification on a process variable, variations in the process variable may cause variations in more important variables downstream. In such cases, it makes sense to remove the disturbance at its source.

2. *Local removal of uncertainty.* By measuring and controlling a process variable, it may be possible to reduce the effect of uncertainty with respect to equipment behaviour or disturbances. Examples of such control loops are valve positioners used to minimize the effect of valve stiction, or local flow control loops which may be used to counteract the effects of pressure disturbances up- or downstream of a valve, changes in fluid properties, or inaccuracies in the valve characteristics.

1.2.1 What knowledge does a process control engineer need?

The list on page 10 also indicates what kind of knowledge is required for a process control engineer. The process control engineer needs to have a thorough understanding of the process. Most stabilizing control loops involve only one process unit (e.g., a tank or a reactor), and most equipment limitations are also determined by the individual units. Process understanding on the scale of the individual units is therefore required. Understanding what phenomena affect product quality also require an understanding of the individual process units. On the other hand, ensuring that the specified production rate propagates throughout the plant, how the effect of disturbances propagate, and optimizing the process operation, require an understanding of how the different process units interact, i.e., an understanding of the process on a larger scale.

Most basic control functions are performed by single loops, i.e., control loops with one controlled variable and one manipulated variable. Thus, when it is understood *why* a particular process variable needs to be controlled, and what manipulated variable should be used to control it[4], the controller design itself can be performed using traditional single-loop control theory (if any theoretical considerations are made at all). Often a standard type of controller, such as a PID controller, is tuned on-line, and there is little need for a process model. Other control tasks are multivariable in nature, either because it is necessary to resolve interactions between different control loops, or because the control task requires coordination between different process units. Process models are often very useful for these types of control problem. The models may either be linear models obtained from experiments on the plant, or possibly non-linear models derived from physical and chemical principles. Some understanding of mathematical modelling and system identification techniques are then required. Non-linear system identification from plant experiments are not in standard use in the process industries.

Optimizing process operation requires some understanding of plant economics, involving the costs of raw materials and utilities, the effect of product quality on product price, the cost of reprocessing off-spec product, etc. Although it is rare that economics is optimized by feedback controllers[5], an understanding of plant economics will help understanding where efforts to improve control should be focused, and will help when discussing the need for improved control with plant management.

A process control engineer must thus have knowledge both of process and control engineering. However, it is not reasonable to expect the same level of expertise in either of these disciplines from the process control engineer as for "specialist" process

[4]Determining what variables are to be controlled, what manipulated variables should be used for control, and the structure of interconnections between manipulated and controlled variables, are quite critical tasks in the design of a process control system. This part of the controller design is often not described in textbooks on "pure" control engineering, but will be covered in some detail in later sections.

[5]It is more common that economic criteria are used in the problem formulation for socalled Real Time Optimization (RTO) problems, or for plant production planning and scheduling.

or control engineers. There appears to be a "cultural gap" between process and control engineers, and the process control engineer should attempt to bridge this gap. This means that the process control engineer should be able to communicate meaningfully with both process and control engineers, and thereby also be able to obtain any missing knowledge by discussing with the "specialists". However, at a production plant there will seldom be specialists in control theory, but there will always be process engineers. At best, large companies may have control theory specialists at some central research or engineering division. This indicates that a process control engineer should have a fairly comprehensive background in control engineering, while the process engineering background should at least be sufficient to communicate effectively with the process engineers.

In the same way as for other branches of engineering, success at work will not come from technological competence alone. A successful engineer will need the ability to work effectively in multi-disciplinary project teams, as well skills in communicating with management and operators. Such non-technical issues will not be discussed further here.

1.3 The structure of control systems in the process industries.

1.3.1 Overall structure

When studying control systems in the process industries, one may observe that they often share a common structure. This structure is illustrated in Fig. 1.1.

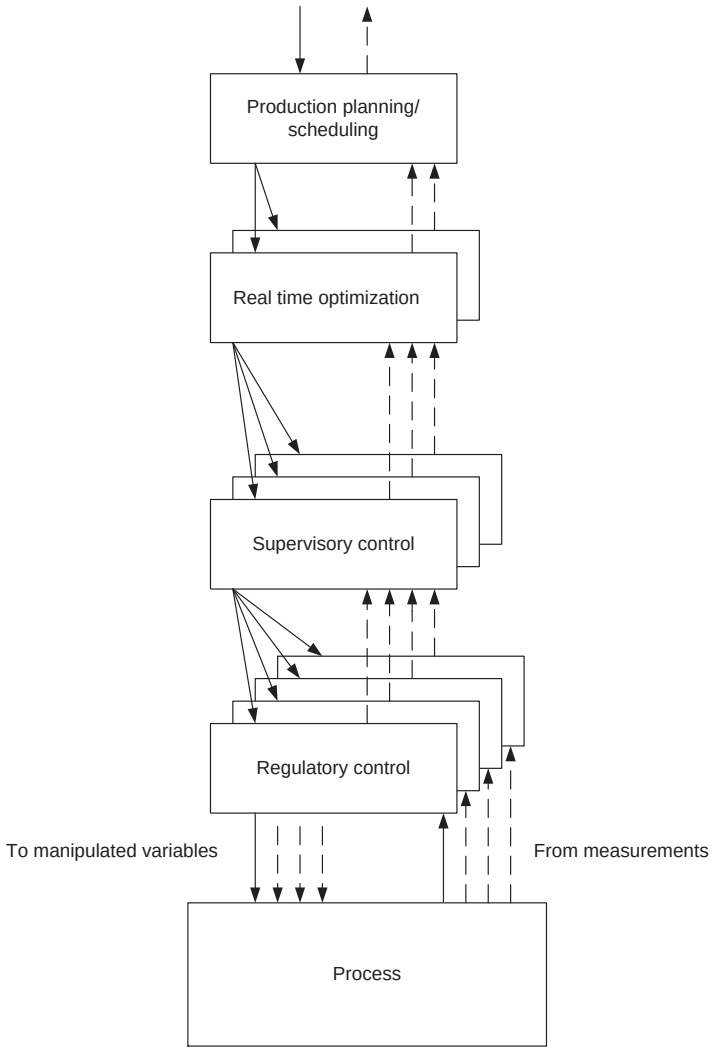

Figure 1.1: Typical structure of the control system for a large plant in the process industries.

The lower level in the control system is the *Regulatory control* layer. The structure of the individual controllers in the regulatory control layer is normally very simple. Standard single-loop controllers, typically of PI/PID type are the most common, but other simple control functions like feed forward control, ratio control, or cascaded

control loops may also be found. Truly multivariable controllers are rate at this level. The regulatory control system typically controls basic process variables such as temperatures, pressures, flowrates, speeds or concentrations, but in some cases the controlled variable may be calculated based on several measurements, e.g., a component flowrate based on measurements of both concentration and overall flowrate or a ratio of two flowrates. Usually a controller in the regulatory control layer manipulates a process variable directly (e.g., a valve opening), but in some cases the manipulated variable may be a setpoint of a cascaded control loop. Most control functions that are essential to the stability and integrity of the process are executed in this layer, such as stabilizing the process and maintaining acceptable equipment operating conditions.

The *Supervisory control* layer coordinates the control of a process unit or a few closely connected process units. It coordinates the action of several control loops, and tries to maintain the process conditions close to the optimal while ensuring that operating constraints are not violated. The variables that are controlled by supervisory controllers may be process measurements, variables calculated or estimated from process measurements, or the output from a regulatory controller. The manipulated variables are often setpoints to regulatory controllers, but process variables may also be manipulated directly. Whereas regulatory controllers are often designed and implemented without ever formulating any process model explicitly, supervisory controllers usually contain an explicitly formulated process model. The model is dynamic and often linear, and obtained from experiments on the plant. Typically, supervisory controllers use some variant of Model Predictive Control (MPC).

The optimal conditions that the supervisory controllers try to maintain, may be calculated by a *Real Time Optimization* (RTO) control layer. The RTO layer identifies the optimal conditions by solving an optimization problem involving models of the production cost, value of product (possibly dependent on quality), and the process itself. The process model is often non-linear and derived from fundamental physical and chemical relationships, but they are usually static.

The higher control level shown in Fig. 1.1 is the *Production planning and scheduling* layer. This layer determines what products should be produced and when they should be produced. This layer requires information from the sales department about the quantities of the different products that should be produced, the deadlines for delivery, and possibly product prices. From the purchasing department information about the availability and price of raw materials are obtained. Information from the plant describes what products can be made in the different operating modes, and what production rates can be achieved.

In addition to the layers in Fig. 1.1, there should also be a separate safety system that will shut the process down in a safe and controlled manner when potentially dangerous conditions occur. There are also higher levels of decision making which are not shown, such as sales and purchasing, construction of new plants, etc. These levels are considered to be of little relevance to process control, and will not be discussed further.

Note that there is a difference in time scale of execution for the different layers. The regulatory control system typically have sampling intervals on the scale of

one second (or faster for some types of equipment), supervisory controllers usually operate on the time scale of minutes, the RTO layer on a scale of hours, and the planning/scheduling layer on a scale of days (or weeks). The control bandwidths achieved by the different layers differ in the same way as sampling intervals differ. This difference in control bandwidths can simplify the required modelling in the higher levels; if a variable is controlled by the regulatory control layer, and the bandwidth for the control loop is well beyond what is achieved in the supervisory control layer, a static model for this variable (usually the model would simply be *variable value = setpoint*) will often suffice for the supervisory control.

It is not meaningful to say that one layer is more important than another, since they are interdependent. The objective of the lower layers are not well defined without information from the higher layers (e.g., the regulatory control layer needs to know the setpoints that are determined by the supervisory control layer), whereas the higher layers need the lower layers to implement the control actions. However, in many plants human operators perform the tasks of some the layers shown in Fig. 1.1, it is only the regulatory control layer that is present (and highly automated) in virtually all industrial plants.

Why has this multi-layered structure for industrial control systems evolved? It is clear that this structure imposes limitations in achievable control performance compared to a hypothetical optimal centralized controller which perfectly coordinates all available manipulated variables in order to achieve the control objectives. In the past, the lack of computing power would have made such a centralized controller virtually impossible to implement, but the continued increase in available computing power could make such a controller feasible in the not too distant future. Is this the direction industrial control systems are heading? This appears not to be the case. In the last two of decades development has instead moved in the opposite direction, as increased availability of computing power has made the Supervisory control and Real Time Optimization layers much more common. Some reasons for using such a multi-layered structure are:

- *Economics.* Optimal control performance - defined in normal control engineering terms (using e.g., the H_2- or H_∞norm) - does not necessarily imply optimal economic performance. To be more specific, an optimal controller synthesis problem does not take into account the cost of developing and maintaining the required process (or possibly plant economic) models. An optimal centralized controller would require a dynamic model of most aspects of the process behaviour. The required model would therefore be quite complex, and difficult to develop and maintain. In contrast, the higher layers in a structured control system can take advantage of the model simplifications made possible by the presence of the lower layers. The regulatory control level needs little model information to operate, since it derives most process information from feedback from process measurements[6].

[6]A good process model may be of good use when *designing control structures* for regulatory

- *Redesign and retuning.* The behaviour of a process plant changes with time, for a number of reasons such as equipment wear, changes in raw materials, changes in operating conditions in order to change product qualities or what products are produced, and plant modifications. Due to the sheer complexity of a centralized controller, it would be difficult and time-consuming to update the controller to account for all such changes. With a structured control system, it is easier to see what modifications need to be made, and the modifications themselves will normally be less involved.

- *Start-up and shutdown.* Common operating practice during start-up is that many of the controls are put in manual. Parts of the regulatory control layer may be in automatic, but rarely will any higher layer controls be in operation. The loops of the regulatory control layer that are initially in manual are put in automatic when the equipment that they control are approaching normal operating conditions. When the regulatory control layer for a process section is in service, the supervisory control system may be put in operation, and so on. Shutdown is performed in the reverse sequence. Thus, there may be scope for significant improvement of the start-up and shutdown procedures of a plant, as quicker start-up and shutdown can reduce plant downtime. However, a model which in addition to normal operating conditions also is able to describe start-up and shutdown, is necessarily much more complex than a model which covers only the range of conditions that are encountered in normal operation. Building such a model would be difficult and costly. Start-up and shutdown of a plant with an optimal centralized control system which does not cover start-up and shutdown, may well be more difficult than with a traditional control system, because it may not be difficult to put an optimal control system gradually into or out of service.

- *Operator acceptance and understanding.* Control systems that are not accepted by the operators are likely to be taken out of service. An optimal centralized control system will often be complex and difficult to understand. Operator understanding obviously makes acceptance easier, and a traditional control system, being easier to understand, often has an advantage in this respect. Plant shutdowns may be caused by operators with insufficient understanding of the control system. Such shutdowns should actually be blamed on the control system (or the people who designed and installed the control system), since operators are an integral part of the plant operation, and their understanding of the control system must therefore be ensured.

- *Failure of computer hardware and software.* In traditional control systems the operators retain the help of the regulatory control system in keeping the process in operation if a hardware or software failure occurs in higher levels of the control system. A hardware backup for the regulatory control system is

control. However, after the regulatory controllers are implemented, they normally do not make any explicit use of a process model.

much cheaper than for the higher levels in the control system, as the regulatory control system can be decomposed into simple control tasks (mainly single loops). In contrast, an optimal centralized controller would require a powerful computer and it is therefore more costly to provide a backup system. However, with the continued decrease in computer cost this argument may weaken.

- *Robustness.* The complexity of an optimal centralized control system will make it difficult to analyze whether the system is robust with respect to model uncertainty and numerical inaccuracies. Analyzing robustness need not be trivial even for traditional control systems. The ultimate test of robustness will be in the operation of the plant. A traditional control system may be applied gradually, first the regulatory control system, then section by section of the supervisory control system, etc. When problem arise, it will therefore be easier to analyze the cause of the problem with a traditional control system than with a centralized control system.

- *Local removal of uncertainty.* It has been noted earlier that one effect of the lower layer control functions is to remove model uncertainty as seen from the higher layers. Thus, the existence of the lower layers allow for simpler models in the higher layers, and make the models more accurate. The more complex computations in the higher layers are therefore performed by simpler, yet more accurate models. A centralized control system will not have this advantage.

- *Existing traditional control systems.* Where existing control systems perform reasonably well, it makes sense to put effort into improving the existing system rather than to take the risky decision to design a new control system. This argument applies also to many new plants, as many chemical processes are not well understood. For such processes it will therefore be necessary to carry out model identification and validation on the actual process. During this period some minimum amount of control will be needed. The regulatory control layer of a traditional control system requires little information about the process, and can therefore be in operation in this period.

It should be clear from the above that this author believes that control systems in the future will continue to have a number of distinct layers. Two prerequisites appear to be necessary for a traditional control system to be replaced with a centralized one:

1. The traditional control system must give unacceptable performance.

2. The process must be sufficiently well understood to be able to develop a process model which describes all relevant process behaviour.

Since it is quite rare that a traditional control system is unable to control a process for which detailed process understanding is available (provided sufficient effort and expertise have been put into the design of the control system), it should follow that majority of control systems will continue to be of the traditional structured type.

Chapter 2

Mathematical and control theory basics

2.1 Introduction

This section will review some mathematical and control theory basics, that in actual fact is assumed covered by previous control courses. Both the coverage of topics and their presentation will therefore be sketchy and incomplete, aimed at

- correcting what is this author's impression of what are the most common misconceptions among students who follow this course, as well as
- to establish some basic concepts and introduce some notation.

2.2 Models for dynamical systems

Many different model representations are used for dynamical systems, and a few of the more common ones will be introduced here.

2.2.1 Dynamical systems in continuous time

A rather general way of representing a dynamical system in continuous time is via a set of ordinary differential equations:

$$\dot{x} = f(x, u, d) \tag{2.1}$$

where the variables x are termed the *system states* and $\dot{x} = \frac{dx}{dt}$ is the time derivative of the state. The variables u and d are both external variables that affect the system. In the context of control, it is common to distinguish between the *manipulated variables* or *(control) inputs* u that can be manipulated by a controller, and the *disturbances* d that are external variables that affect the system but which cannot be set by the controller.

The system states x are generally only a set of variables that are used to describe the system's behaviour over time. Whether the individual components of the state vector can be assigned any particular physical interpretation will depend on how the model is derived. For models derived from fundamental physical and chemical relationships (often termed 'rigorous models'), the states will often be quantities like temperatures, concentrations, velocities, etc. If, on the other hand, the model is an empirical model identified from observed data, it will often not be possible to assign any particular interpretation to the states.

Along with the state equation (2.1), one typically also needs a *measurement equation* such as

$$y = g(x, u, d) \tag{2.2}$$

where the vector y is a vector of *system outputs*, which often correspond to available *physical measurements* from the systems. Control design is usually at its most simple when all states can be measured, i.e., when $y = x$.

Disturbances need not be included in all control problems. If no disturbances are included in the problem formulation, equations (2.1) and (2.2) trivially simplify to $\dot{x} = f(x, u)$ and $y = g(x, u)$, respectively.

Since we are dealing with *dynamical* systems, it is hopefully obvious that the variables x, y, u, d may all vary with time t. In this section time is considered as a continuous variable - in accordance with our usual notion of time.

Together, equations (2.1) and (2.2) define a system model in continuous time. This type of model is rather general, and can deal with any system where it suffices to consider system properties at specific points in space, or where it is acceptable to average/lump system properties over space. Such models where properties are averaged over space are often called *lumped models*.

For some applications, it may be necessary to consider also *spatial distribution* of properties. Rigorous modelling of such systems typically result with a set of partial differential equations (instead of the ordinary differential equations of (2.1)). In addition to derivatives with respect to time, such models also contain derivatives with respect to one or more spatial dimensions. Models described by partial differential equations will not be considered any further in these notes. Although control design based on partial differential equations is an active research area (in the area of *flow control*, in particular), the more common industrial practice is to convert the set of partial differential equations to a (larger) set of ordinary differential equations through some sort of spatial *discretization*.

2.2.2 Dynamical systems in discrete time

Although time in the 'real world' as we know it is a continuous variable, control systems are typically implemented in computer systems, which cyclically execute a set of instructions. Measurements and control actions are therefore executed at discrete points in time, and to describe system progression from one time instant to subsequent instants we will need a discrete time model. Such models may be

represented as

$$x_{k+1} = f(x_k, u_k, dk) \tag{2.3}$$
$$y_k = g(x_k, u_k, d_k) \tag{2.4}$$

where x_k, y_k, u_k and d_k are the discrete-time counterparts to the system states, outputs, inputs and disturbances introduced above for continuous-time systems. Note that although the same letter f is used to represent the system dynamics for both continuous- and discrete-time systems, these functions will be different for the two different model types. The measurement equation, on the other hand, will often be identical for the two model types.

2.2.3 Linear models and linearization

Many control design methods are based on *linear* models. It is therefore necessary to be able to convert from a nonlinear model to a linear model which is (hopefully) a close approximation to the nonlinear model. This is called linearization of the nonlinear model.

A systems is linear if to functions f and g (in (2.1) and (2.2) for the case of continuous time models, or in (2.3) and (2.4) for the case of discrete time models) are linear in all the variables x, u and d. Thus, a linear continuous-time model may be expressed as

$$\dot{x} = Ax + Bu + Ed \tag{2.5}$$
$$y = Cx + Du + Fd \tag{2.6}$$

where A, B, C, D, E, F are matrices of appropriate dimensions, and the matrix elements are independent of the values of x, u, d. Linear models for discrete-time systems follow similarly.

Most systems are to some extent non-linear. However, controller design and verification is usually much simpler for linear than for non-linear system. It is therefore important to be able to convert from a non-linear model to a 'approximately equal' linear model. This process is called *linearization*.

Linearization is based on the Taylor series expansion of a function. Consider a function $h(a)$. We want to approximate the value of $h(a)$ in the vicinity of $a = a^*$. The Taylor series expansion then provides the approximation

$$h(a) = h(a^* + \delta a) \approx h(a^*) + \frac{\partial h}{\partial a}|_{a^*}\delta a + \frac{1}{2}\delta a^T \frac{\partial^2 h}{\partial a^2}|_{a=a^*}\delta a + ... \tag{2.7}$$

where the notation $|_{a=a^*}$ indicates that the value $a = a^*$ is used when evaluating the derivatives.

Linearization at a given point

When linearizing a dynamical system model we terminate the Taylor series expansion after the first order term. The underlying non-linear system is therefore naturally

assumed to be continuous and have continuous first order derivatives. Assume that the linearization is performed at the point

$$a = \begin{bmatrix} x \\ u \\ d \end{bmatrix} = \begin{bmatrix} x^* \\ u^* \\ d^* \end{bmatrix} = a^* \tag{2.8}$$

The terminated Taylor series expansion of (2.1) then becomes

$$\frac{dx}{dt} = \frac{d\delta x}{dt} \approx f(a^*) + \left.\frac{\partial f}{\partial x}\right|_{a=a^*} \delta x + \left.\frac{\partial f}{\partial u}\right|_{a=a^*} \delta u + \left.\frac{\partial f}{\partial d}\right|_{a=a^*} \delta d \tag{2.9}$$

Similarly, we get for (2.2)

$$y = y^* + \delta y \approx g(a^*) + \left.\frac{\partial g}{\partial x}\right|_{a=a^*} \delta x + \left.\frac{\partial g}{\partial u}\right|_{a=a^*} \delta u \left.\frac{\partial g}{\partial d}\right|_{a=a^*} \delta d \tag{2.10}$$

where it is understood that $y^* = g(a^*)$.

Next, define $A = \frac{\partial f}{\partial x}|_{a=a^*}$, $B = \frac{\partial f}{\partial u}|_{a=a^*}$, $E = \frac{\partial f}{\partial d}|_{a=a^*}$, $C = \frac{\partial g}{\partial x}|_{a=a^*}$, $D = \frac{\partial g}{\partial u}|_{a=a^*}$, $F = \frac{\partial d}{\partial x}|_{a=a^*}$

Linearizing at an equilibrium point The point a^* used in the linearization is usually an equilibrium point. This means that

$$f(a^*) = 0 \tag{2.11}$$
$$g(a^*) = y^* \tag{2.12}$$

Thus, we get

$$\frac{dx}{dt} = A\delta x + B\delta u + E\delta d \tag{2.13}$$
$$\delta y = C\delta x + D\delta u + F\delta d \tag{2.14}$$

Linearizing a discrete-time model is done in the same way as for continuous-time models. The only slight difference to keep in mind is that for a discrete-time model at steady state $x_{k+1} = x_k$, and therefore $f(a^*) = x_k$ when linearizing at a steady state.

Deviation variables It is common to express the system variables (x, u, d and y) in terms of their *deviation* from the linearization point a^*. When doing so the δ's are typically suppressed for ease of notation - *as will be done in the remainder of this note*. It is, however, important to beware that when converting from deviation variables to 'real' variables, the linearization point has to be accounted for.

To illustrate: A model for a chemical reactor is linearized at steady state conditions corresponding to a reactor temperature of $435K$. If the linearized model, expressed in deviation variables, indicates a temperature of -1, the corresponding 'real' temperature would be $434K$.

It appears that many students, even after introductory control courses, do not appreciate that *our socalled 'linear' controllers are only linear when expressed in deviation variables*. This can lead to many frustrations, until the misunderstanding has been clarified - which might actually take some time, because the importance of this issue will depend on both controller structure and controller type. Consider a simple feedback loop, with a (linear) controller K controlling a system G, as illustrated in Fig. 2.1.

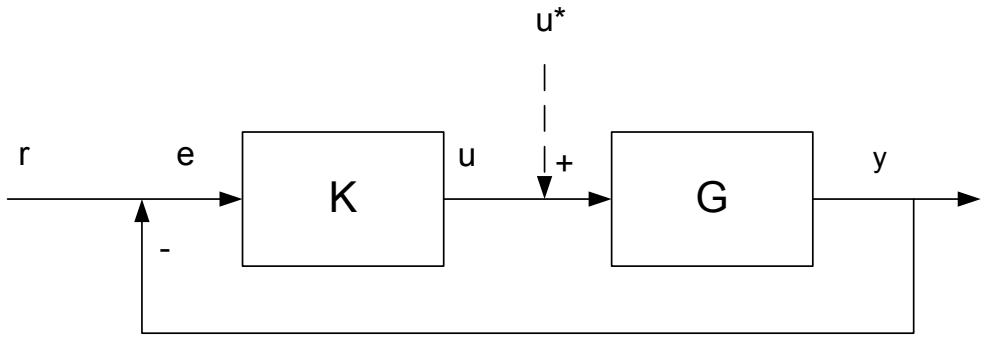

Figure 2.1: A simple feedback loop with a one degree of freedom controller and possible 'output bias'.

This type of controller is called a 'one degree of freedom controller', since it has only one input, the control offset $e = r - y$. We can make the following observations:

- Clearly, it does not matter whether the reference r and measurement y are expressed in 'physical' variables or deviation variables, as long as the same scale is used for both. This is because the controller input is the difference between these to variables.

- Consider the case when the controller K is a pure proportional controller, i.e., $u = K(r - y)$ with K constant. It is then necessary to add u^* as an 'output bias'[1] to the controller output, as indicated by the dashed arrow in the figure.

- Consider next the case when the controller K contains integral action. In this case the 'output bias' is not strictly necessary, since the value of the integrating state will adjust for this when the system reaches steady state. However, an output bias may improve transient response significantly when putting the controller into operation.[2]

Consider next a loop where the controller has separate entry port for the reference and the measurement, as shown in Fig. 2.2. This type of controller is used when one wants to treat the measurement and reference signals differently in the controller. We note that

[1] Some system vendors may use different terminology.
[2] See also the chapter on Bumpless Transfer.

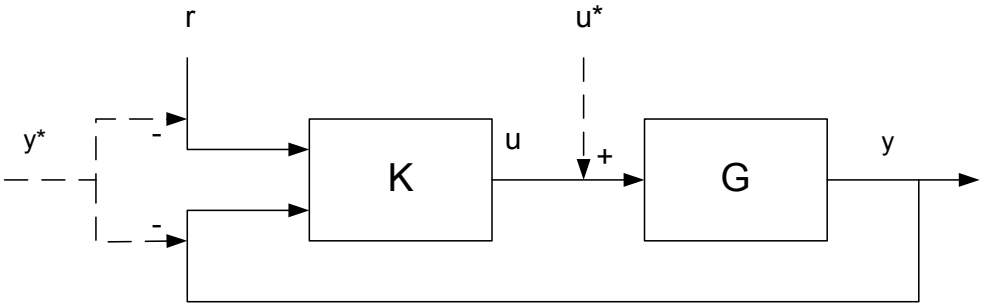

Figure 2.2: A simple feedback loop with a two degree of freedom controller and possible 'bias' on both controller inputs and controller output.

- In this case we need to subtract the value of the measurement at the linearization point, y^*, from both the reference and the measurement.

- Whether to add u^* to the controller output is determined by the same considerations as for the one degree of freedom controller.

Linearizing around a trajectory. It was noted above that it is most common to linearize around a steady state. However, in some cases, one may want to linearize around a trajectory, i.e., around a series of consistent future values of x, u and d. This most commonly occurs in non-linear model predictive control (MPC). Each time an MPC controller executes, it solves an optimization problem that optimizes system behaviour over a 'prediction horizon'. However, for some strongly non-linear problems, using the same linearized model for the entire prediction horizon may not give sufficient accuracy. In such cases, one may choose to linearize around a trajectory instead.

Given the present state, a prediction of the future manipulated variables (typically obtained from the previous execution of the MPC), and predicted values for future disturbances, the nonlinear model can be used to simulate the system in the future. This gives predicted future states that are consistent with the present state and the predicted future manipulated variables and disturbances.

For each timestep in the future, the linearization is performed around the predicted state, manipulated variable and disturbance values. This will give different matrices A, B, CD, E, F for each timestep. In this way, a non-linear system is approximated by a linear, *time-varying* model.

Linearizing around a trajectory clearly complicates the model. In addition to the added complexity of having to ensure that the right model matrices are used at the right timestep in the future, one also has to remember that the linearization point varies from timestep to timestep. This adds additional complexity when converting between physical variables and deviation variables.

2.2.4 Converting between continuous- and discrete-time models

It will often be necessary to convert from continuous- to discrete-time models (and less frequently necessary to convert the other way). Process models based on first principles modelling will typically result in continuous-time models. Often, control design is performed with a continuous-time model. The continuous-time controller is thereafter converted to a discrete-time controller for implementation in a computer. There are also controller types that are more conveniently designed using discrete-time models. The most notable example of such controllers are the socalled Model Predictive Control (MPC) controllers, which will be described in some detail later in these notes.

To convert from continuous to discrete time, we need to

- choose a numerical integration method for the system dynamics, and
- determine (assume) how the external variables (u and d) change *between* the time instants for the discrete-time model.

It is common to assume socalled 'zero order hold'[3], i.e., that the external variables are constant at the value of the previous time instant until the next time instant is reached. This agrees with what is common practice for control inputs in control systems.

Most control design software will have functions for converting between continuous- and discrete-time linear models. It is also included in most basic control textbooks. We will nevertheless give a short introduction here, primarily in order to discuss the handling of time delay when converting from a continuous to a discrete time model. The presentation is inspired by that of Åström and Wittenmark [7].

Consider a continuous-time linear model

$$\dot{x} = A_c x(t) + B_c u(t) \tag{2.15}$$

Assuming zero order hold and a timestep of length h, integration over one timestep (from $t = kh$ to $t = kh + h$) gives

$$x(kh + h) = e^{A_c h} x(kh) + \int_{kh}^{kh+h} e^{A_c(kh+h-r)} B_c u(r) dr \tag{2.16}$$

This is commonly expressed as the discrete-time model

$$x_{k+1} = A_d x_k + B_d u_k \tag{2.17}$$

where the sampling interval h is assumed known and therefore not explicitly stated[4].

[3] An n^{th} *order hold* means that the nth time derivative is held constant between the sample instants of the discrete time model

[4] Note also that the subscript d refers to *discrete time* rather than 'disturbance'. Elsewhere in this note B_d is sometimes used as 'the B-matrix for the disturbance'.

The matrices A_d and B_d are given by

$$\begin{aligned} A_d &= e^{A_c h} \\ B_d &= \int_{kh}^{kh+h} e^{A_c(kh+h-r)} B_c u(r) dr = A_c^{-1}\left(e^{A_c h} - I\right) B_c \end{aligned}$$

Time delay in the manipulated variables

Consider next the case when the manipulated variables u do not affect the state derivative $\dot{x}$ directly, but only after a time delay τ. The model (2.15) thus becomes

$$\dot{x} = A_c x(t) + B_c u(t-\tau) \tag{2.18}$$

Note that there is no exact representation of a pure time delay using ordinary differential equations - this would require an infinite number of states. Therefore, the time delay is instead introduced explicitly in the argument when representing the manipulated variable u as a function of time.

Multiple timestep time delays If the time delay is an integer number of sampling intervals, this is easily captured in a discrete-time model. Let $u_\Delta(k) = u(k-nh)$. This can be expressed as

$$\begin{aligned} x_\Delta(k+1) &= A_\Delta x_\Delta(k) + B_\Delta u_\Delta(k) \\ &= \begin{bmatrix} 0 & I & 0 & \cdots & 0 \\ 0 & 0 & I & \vdots & 0 \\ 0 & \vdots & \vdots & \vdots & 0 \\ 0 & \vdots & \vdots & 0 & I \\ 0 & \dots & \dots & \dots & 0 \end{bmatrix} x_\Delta(k) + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ I \end{bmatrix} u(k) \\ u_\Delta(k) &= C_\Delta x_\Delta(k) = \left[\underbrace{I \quad 0 \quad 0 \quad \cdots \quad 0}_{n} \right] x_\Delta(k) \end{aligned} \tag{2.19}$$

Fractional timestep time delays If the time delay $\tau$is only a fraction of the sampling interval h, we must account for the fact that the value of the manipulated variable which affects $\dot{x}$ in (2.15) from time kh to time $kh+\tau$ is actually $u(kh-h)$. Thus, the integral in (2.16) must be split in two, and we get

$$\begin{aligned} x(kh+h) &= e^{A_c h} x(kh) + \int_{kh}^{kh+\tau} e^{A_c(kh+h-r)} B_c dr u(kh-h) + \int_{kh+\tau}^{kh+h} e^{A_c(kh+h-r)} B_c dr u(kh) \\ &= A_d x(kh) + B_{d0} u(kh) + B_{d1} u(kh-h) \\ B_{d1} &= e^{A_c(h-\tau)} A_c^{-1}\left[e^{A_c \tau} - I\right] B_c = e^{A_c(h-\tau)} \int_0^\tau e^{A_c r} dr B_c \\ B_{d0} &= A_c^{-1}\left[e^{A_c(h-\tau)} - I\right] B_c = \int_0^{h-\tau} e^{A_c r} dr B_c \end{aligned} \tag{2.20}$$

This can be expressed in state space form as

$$\begin{bmatrix} x(kh+h) \\ u(kh) \end{bmatrix} = \begin{bmatrix} A_d & B_{d1} \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} B_{d0} \\ I \end{bmatrix} u(kh) \tag{2.21}$$

For time delays lasting more than one timestep, but a non-integer number of timesteps, the overall model is found by the series interconnection of the multiple timestep delay model in (2.19) and the system dynamics + fractional timestep delay model in (2.21).

Some modern control techniques like MPC are computationally intensive, and may induce a computational time delay. If the computational time is significant compared to the sampling interval, it may be necessary to include a fractional time delay in the model even for plants that by itself have no time delay.

Time delay in the measurement. Time delays in measurements may occur both due to the characteristics of the sensor equipment (e.g., delays in analyzers such as on-line gas cromatographs) or due to transportation delays (long pipes or conveyor belts from the plant to the sensor).

For linear, time invariant systems, it does not matter whether the time delay is modelled at the input or the output of the plant. However, for multivariable systems, the time delay may be different for different measurements. In such cases, the time delay must be modelled at the output, since it cannot be moved to the input.

Also, a measurement is often dependent on multiple states. The number of discrete-time states used to model the time delay can then be reduced by delaying the measurement in the model instead of delaying the states and calculating the measurement from the delayed states [11].

Time delays in the measurements can be handled in much the same way as that explained above for time delay in the manipulated variables. The details are therefore left to the reader.

2.2.5 Laplace transform

The Laplace transform should be familiar to all readers from introductory control courses, and no attempt is made here at providing a complete or self-contained introduction to the topic. It is merely introduced here as a minimal introduction to its use later in this note.

Restating first the linear(ized) ordinary differential equation model, we have

$$\dot{x} = Ax + Bu + Ed \tag{2.22}$$

$$y = Cx + Du + Fd \tag{2.23}$$

where the δ's are suppressed for notational simplicity. We should nevertheless keep in mind that the linear model is expressed in deviation variables. The model described by (2.22) and (2.23) is called a (linear) *state space model* of a system.

Using standard rules for the Laplace transformation (available in standard undergraduate mathematics textbooks), we have

$$sx(s) + x(t=0) = Ax(s) + Bu(s) + Ed(s) \tag{2.24}$$

$$y(s) = Cx(s) + Du(s) + Fd(s) \tag{2.25}$$

where s is a complex-valued scalar. The effect of the initial conditions (the term $x(t = 0)$ above) is usually ignored, since stability and common measures of performance do not depend on initial conditions. Nevertheless, one should be aware that the initial response will depend on initial conditions. If the closed loop system contain modes that are poorly damped, the effects of the initial conditions may be felt for a significant time.

Ignoring the term involving the initial conditions (or assuming the initial conditions equal to zero in deviation variables) we obtain by simple manipulations

$$y(s) = \left[C(sI - A)^{-1}B + D\right] u(s) + \left[C(sI - A)^{-1}E + F\right] d(s) = G(s)u(s) + G_d(s)d(s) \tag{2.26}$$

where $G(s)$ and $G_d(s)$ are the (monovariable or multivariable) transfer functions from the manipulated variable and the disturbance, respectively, to the system output.

2.2.6 Similarity transformations

Whereas the transfer function is unique for a given input-output behaviour, there is an infinite number of different state space models that describe the same dynamics.

Given a state space model such as (2.22) - (2.23), and consider the case where we instead of the original states x want to use the alternative states $\tilde{x}$. The state vectors x and $\tilde{x}$ must be related through

$$x = T\tilde{x} \tag{2.27}$$

where T is an invertible matrix. This ensures that when specifying the state in one set of state variables, we also uniquely specify the states in the other set of state variables. Trivial manipulations then yield

$$\dot{\tilde{x}} = T^{-1}AT\tilde{x} + T^{-1}Bu + T^{-1}Ed \tag{2.28}$$

$$y = CT\tilde{x} + Du + Fd \tag{2.29}$$

from which the state space matrices for the transformed state space model are easily identifiable. This reveals the fact that the state space representation of a dynamical system is not unique - via similarity transforms the exact same dynamics can be represented by 'different' state space models. In addition, a state space model may contain 'redundant' states, as discussed next. In contrast, the frequency response of a model in the Laplace domain (such as (2.26)) is unique. Furthermore, the transfer function model $G(s)$ itself is unique provided any redundant states have been removed, i.e., provided the it is obtained from the Laplace transformation of a *minimal* model.

2.2.7 Minimal representation

A state space model may contain states that either cannot be affected by the inputs (an uncontrollable state) or cannot affect any of the outputs of the system (an unobservable state). Such states do not contribute to the input-output behaviour of the

system. The model then contains more states than the minimal number of states required to represent the input-output behaviour of the system. Therefore, such models are called non-minimal.

Many control calculations assume that the model supplied is minimal, and numerical problems may occur if this is not the case. It is therefore common practice to remove uncontrollable or unobservable states, and standard control software have functions for doing this (such as *minreal* in Matlab).

However, one should bear in mind that the uncontrollable or unobservable system states may represent important quantities for the overall system. Whether it is advisable to remove uncontrollable or unobservable states can depend on several factors:

- How was the model obtained? If the model is the result of rigorous modelling based on physical and chemical principles, the states will typically represent physical/chemical quantities in the system.

- Empirical models identified from experiments will typically result in models containing only observable and controllable states - although not all states need to represent real phenomena in the system.

- When assembling a system model from models of parts of the system, states representing the same physical quantity may be represented in several of the smaller models. This can easily lead to a non-minimal model when assembling the overall system model. Such 'duplicate states' can safely be removed.

- It is usually considered safe to delete *stable* uncontrollable and unobservable modes.

    1. If a stable mode is uncontrollable, its effect on the output will die out over time - unless it is excited by some disturbance. A state may be 'controllable' from a disturbance even if it is uncontrollable from the manipulated variables. This is the situation in many disturbance attenuation problems. Although such states may be removed from the plant model (from manipulated to controlled variables), it cannot be removed from the disturbance model (from disturbances to controlled variables).
    2. A controllable but unobservable mode will be excited by the manipulated variables, and even if it is stable will not necessarily decay to zero if the state is continuously excited by the manipulated variables or disturbances. If the state represents some quantity of little importance, this situation would appear acceptable. It may, however, be the case that the state represents some important quantity, and the fact that it is unobservable merely reflects an inappropriate set of measurements.

When discovering unobservable or uncontrollable states, the engineer should therefore reflect on how and why these states are introduced in the model. It may be that such states can safely be removed from the model. It may also be the case that one

should install new measurements or new actuators to make the states observable and controllable.

For *diagonalizable* systems, i.e., systems with a full rank eigenvector matrix, it is straight forward to perform a similarity transform to identify the uncontrollable or unobservable states. Let M be the eigenvector matrix of the matrix A in (2.22), and Λ the corresponding (diagonal) eigenvalue matrix. Choosing $T = M^{-1}$ in (2.27) then yields

$$\dot{\tilde{x}} = \Lambda\tilde{x} + MBu + MEd \tag{2.30}$$
$$y = CM^{-1}\tilde{x} + Du + Fd \tag{2.31}$$

Uncontrollable states (in terms of the states $\tilde{x}$) can then be identified from rows that are equal to zero in MB, whereas unobservable states are identified from columns in CM^{-1} equal to zero.

2.3 Analyzing linear dynamical systems

2.3.1 Poles and zeros of transfer functions

Consider a scalar transfer function, that can be factored as

$$G(s) = k\frac{(s+z_1)(s+z_2)\cdots(s+z_n)e^{-Ts}}{(s+p_1)(s+p_2)\cdots(s+p_m)} \tag{2.32}$$

where $m \geq n$, as otherwise there would be no state space model that represent the transfer function dynamics. The parameters z_i are known as the *zeros* of the transfer function, whereas the p_i are termed *poles*. The term e^{-Ts} represents a pure time delay (transportation delay) of T time units. Zeros and poles can be either strictly real or complex valued. However, complex-valued zeros or poles always appear in complex conjugate pairs, since both the numerator and denominator of the transfer function has only real-valued coefficients (for transfer functions corresponding to a model described by ordinary differential equations). Note that the time delay cannot term e^{-Ts} does not correspond to a model that can be described (exactly) by ordinary differential equations.

Zeros and poles are often classified according to whether their real parts are positive or negative. Poles and zeros whose real part are strictly negative are called *left half plane* (LHP) poles and zeros, respectively. Similarly, poles and zeros whose real parts are positive are called *right half plane* (RHP) poles and zeros.

Poles and zeros of multivariable systems can be similarly defined, but need to be defined with a little more care. In particular, one should be aware that there need to be no relationship between the zeros of individual transfer function (matrix) elements and the zeros of the overall system. We will refer to Skogestad and Postlethwaite [93] for more precise definitions of multivariable poles and zeros.

For a minimal representation of a system, the poles may also be defined as the roots of the characteristic polynomial

$$\phi(s) = det(sI - A) \tag{2.33}$$

2.3.2 Stability

Assuming that we have a minimal representation of a linear system. The system is then stable if

$$Re(\lambda_i(A)) < 0 \forall i \tag{2.34}$$

where $\lambda_i(A)$ denotes an eigenvalue of the matrix A in the state space model. It follows from (2.26) that the eigenvalues of the A matrix also appear as poles of the transfer function. Stable systems thus have their poles strictly in the left half plane.

Control textbooks may differ somewhat on whether systems with poles on the imaginary axis are considered stable. In some cases (as a result of a strict mathematical definition of stability), systems with *single* poles on the imaginary axis are classified as stable or 'marginally stable', whereas systems with two or more poles in the same place on the imaginary axis are called unstable.

In most practical situations systems with poles on the imaginary axis will need to be 'stabilized' by feedback, irrespective of whether these poles are 'single' or 'multiple' poles. We will therefore classify all systems with poles on the imaginary axis as unstable.

2.3.3 Frequency analysis

In recent years, frequency analysis has been given less room in process control education. This seems to be a particularly prominent trend in Chemical Engineering departments in the USA, where control seems to be squeezed by the wish to include 'newer' topics such as materials/nano-/bio. Although many esteemed colleagues argue that control can be taught just as well entirely with time domain concepts, it is this authors opinion that the same colleagues are making the mistake of elevating a necessity to a virtue.

Despite this worrisome trend, the presentation of frequency analysis in this note will be sketchy, assuming that the reader has had a basic introduction to the topic in other courses.

This author agrees with the arguments expressed by Skogestad and Postlethwaite [93] on the advantages of frequency analysis. While those arguments will not be repeated here, but we will note that many control-relevant insights are easily available with a working understanding of frequency analysis.

In this note, the frequency response will be used to describe a systems response to sinusoidal inputs of varying frequency. Although other interpretations of the frequency response are possible (see, again, [93]), the chosen interpretation has the advantage of providing a clear physical interpretation and a clear link between the frequency and time domain.

The frequency response of a system with transfer function $G(s)$ at the frequency ω is obtained by evaluating $G(s)$ at $s = j\omega$. The result is a complex-valued number (or a complex-valued *matrix*, for multivariable systems). It should be noted that the frequency ω is measured in *radians/time*[5], and thus the oscillation period corresponding to the frequency ω is $t_p = 2\pi/\omega$.

[5]Usually time is measured in seconds, but minutes are also sometimes used for slow process units

The complex-valued frequency response is commonly presented in polar coordinates in the complex plane, with the length being termed the *gain* and the angle being termed the *phase*. Anti-clockwise rotation denotes positive phase.

That is, consider $G(j\omega) = a + jb$. The gain is then $|G(jw)| = \sqrt{a^2 + b^2}$, whereas the phase is given by $\angle G(j\omega) = tan^{-1}(b/a)$. Thus, assume that a sinusoidal input is applied:

$$u(t) = u_0 \sin(\omega t + \alpha) \tag{2.35}$$

Once the effect of any initial conditions have died out (or, we might make the 'technical' assumption that the input has been applied 'forever', since $t = -\infty$), the output will also oscillate sinusoidally at the same frequency:

$$y(t) = y_0 \sin(\omega t + \beta) \tag{2.36}$$

We will then observe that $|G(j\omega)| = y_0/u_0$ and $\angle G(j\omega) = \beta - \alpha$. For multivariable systems, the response of each individual output can be calculated as the sum of the responses to each of the individual inputs. This property holds for all linear systems - both in the time domain and in the frequency domain.

For $G(s)$ in (2.32) we have

$$|G(j\omega)| = |k| \cdot \frac{\prod_{i=1}^{n} |(j\omega + z_i)|}{\prod_{i=1}^{m} |(j\omega + p_i)|} \tag{2.37}$$

$$\angle G(j\omega) = \angle k + \sum_{i=1}^{n} \angle(j\omega + z_i) - \sum_{i=1}^{m} \angle(j\omega + p_i) \tag{2.38}$$

The phase and gain of a single terms $(s + a)$ is illustrated in Fig. 2.3.

The phase and gain of the time delay term can be found from Euler's formula

$$e^{ja} = \cos a + j \sin a \tag{2.39}$$

from which we find that $|e^{-j\omega T}| = 1 \forall \omega$ and $\angle e^{-j\omega T} = -\omega T(\text{rad}) = -\frac{\omega T}{\pi} \cdot 180^\circ$. Clearly, the real constant k will have a phase of zero if $k > 0$ and a phase of $-\pi = -180^\circ$ if $k < 0$.

Steady-state phase adjustment

The steady state value of the transfer function is obtained by evaluating the transfer function at $s = 0$. Clearly, at $s = 0$ the transfer function takes a real value, and thus must have a phase of $n \times 180^\circ$, where n is some integer.

It is customary to adjust or 'correct' the phase such that the phase contribution for the constant k is zero. Similarly, the phase contribution of any RHP zero in (2.32) is adjusted such that its phase at steady state is zero.

This phase adjustment is necessary to be able to assess closed loop stability from the open loop frequency response. For *open loop stable* systems this corresponds to setting the steady state phase to zero, or assuming a positive steady state gain. If the

such as large distillation towers.

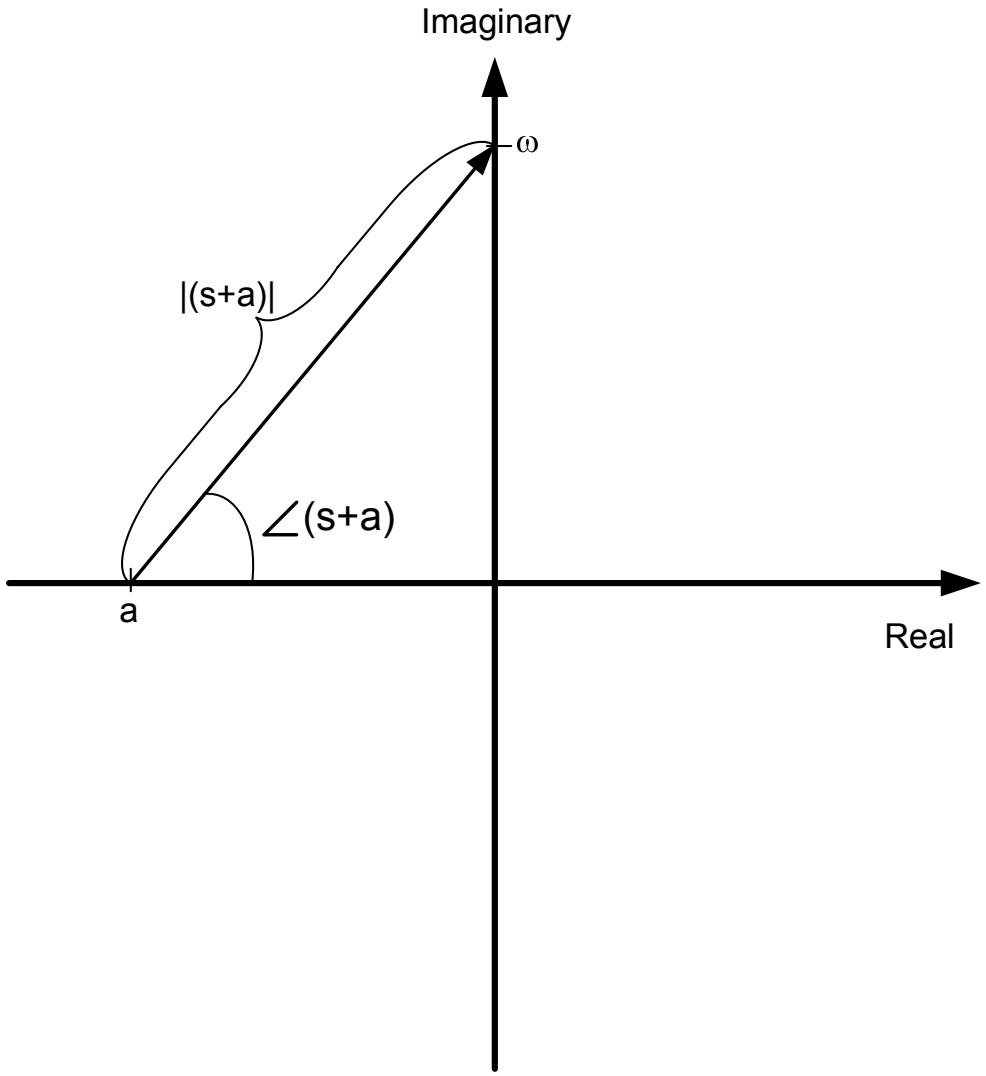

Figure 2.3: The phase and gain of a simple term $(s + a)$ for $a > 0$.

real steady state gain is negative (if the output decreases when the input increases), this in corrected for by simply reversing the sign of the gain of the controller - often this is done by specifying that the controller should be 'reverse acting'.

The phase adjustment described above is done irrespective of whether the system is stable in open loop. Note, however, that the phase of any unstable (RHP) poles are *not* adjusted in this way. This may appear inconsistent, but is possibly most easily understood by noting that one cannot 'normalize the steady state phase' for a RHP pole. An RHP pole represents an instability in the system, the output will grow exponentially without bounds as a response to a change in the input, and thus there *is no steady state* for an RHP pole.

2.3.4 Bode diagrams

The frequency response of a scalar system is often presented in a *Bode diagram* (sometimes also called Amplitude-Phase-Frequency diagram). The Bode diagram consists of two plots, the *magnitude plot* and the *phase plot*.

In the magnitude plot, the transfer function magnitude (or gain) is plotted versus frequency. Both the magnitude and the frequency axes are logarithmic (to the base 10).

Remark. Note that the magnitude scale used for the Bode magnitude plot in this note is the conventional logarithmic scale (to the base 10). In some books, one can still see the decibel (dB) scale used in the Bode magnitude plot, where

$$|G(jw)|(dB) = 20\log_{10}|G(j\omega)| \tag{2.40}$$

We repeat that the decibel scale is *not* used in this note (or in this course).

In the Bode phase plot, the phase is plotted against frequency. The phase is usually plotted in degrees using a linear scale (radians are seldom used), whereas a logarithmic scale is used for the frequency axis. A Bode diagram of the simple system $g(s) = \frac{s+0.1}{(s+0.01)(s+1)}$ is shown in solid lines in Fig. 2.3.4.

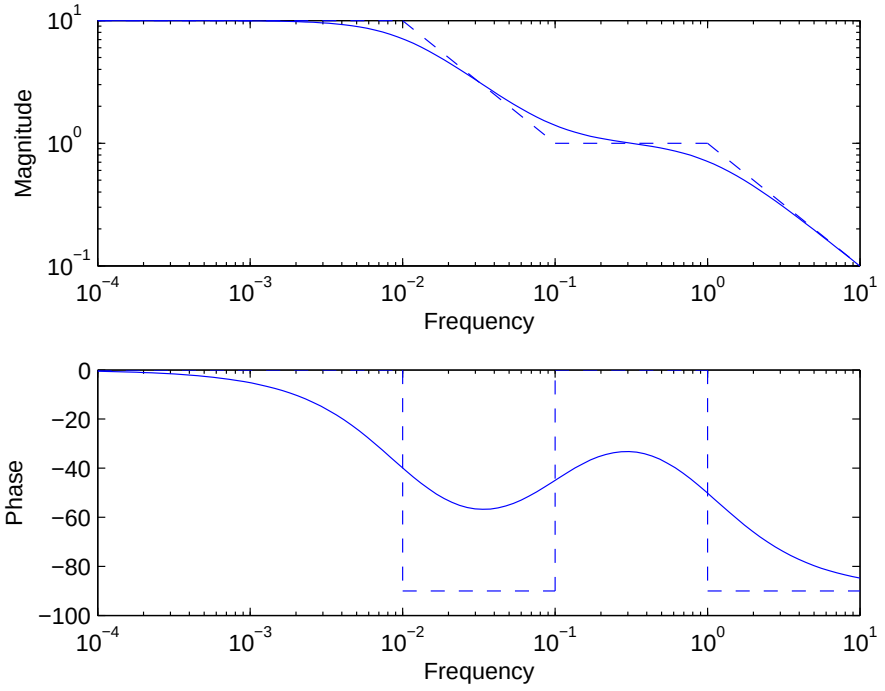

Figure 2.4: The Bode diagram for the simple system $g(s) = 10\frac{(10s+1)}{(100s+1)(s+1)} = \frac{s+0.1}{(s+0.01)(s+1)}$.

Control software that plots Bode diagrams are now easily available, and manual procedures for drawing Bode diagrams are therefore obsolete. One should, however, take a little care to ensure that the steady state phase is correctly adjusted, as outlined above. Otherwise, the steady state phase can easily be off by some multiple of 180°.

Bode diagram asymptotes. Although procedures for manually drawing Bode diagrams are now obsolete, it is useful to be able to quickly visualize the phase-gain relationships of the Bode diagram - possibly without drawing any diagram at all. For this purpose, knowledge about the Bode diagram asymptotes are useful. This is particularly useful when considering changes to controller parameters for PI/PID controllers, since it can give an intuitive understanding of the effects of such changes and thereby simplify the search for appropriate controller parameters. These asymptotes are rather inaccurate approximations to the exact diagram in the frequency range near a pole or zero, but good approximations at frequencies removed from poles and zeros.

To obtain the asymptotes for the Bode magnitude plot,

- Start from the steady state gain of the system, $|G(0)|$. If the system has 'pure integrators' (poles at $s = 0$), evaluate the transfer function instead at some very low frequency, several decades below any other pole or zero.

- The gradient of the magnitude asymptote (in the loglog scale used in the magnitude plot) at low frequencies is $-n$, where n is the number of pure integrators. In the less common case that $G(s)$ contains m pure differentiators (m zeros at $s = 0$), the steady state gain is zero, and the initial gradient is m in the magnitude plot.

- Increase frequency ω. Whenever $\omega = z_i$, *increase* the gradient of the asymptote by 1. Whenever $\omega = p_i$, *decrease* the gradient of the asymptote by 1.

The asymptotes for the Bode phase plot are obtained as follows:

- If the transfer function contains n pure integrators, they contribute a total of $-90° \cdot n$ of phase at (very) low frequencies. Similarly, if the transfer function contains m pure differentiators, these contribute a total of $90° \cdot m$ of phase at (very) low frequencies.

- Poles in the left half plane (the closed left half plane except the origin) do not contribute to the phase at steady state. The zeros (anywhere except at $s = 0$) also do not contribute the phase at steady state.

- Poles in the open *right half plane* each contribute $-180°$ to phase at steady state.

- Add the phase contributions at steady state. This gives the value of the low frequency phase asymptote.

- Gradually increase frequency ω. If $\omega = z_i$ (a zero in the left half plane), *increase* the asymptote phase by 90°. If $\omega = -z_i$ (a zero in the right half plane), *decrease* the asymptote phase by 90°. If $\omega = p_i$ (a pole in the left half plane), *decrease* the asymptote phase by 90°. If $\omega = -p_i$ (a pole in the right half plane), *increase* the asymptote phase by 90°.

The phase asymptote thus changes in steps of (multiples of) 90°. Note that this way of finding the phase asymptote does not include the time delay. The phase contribution of any time delay therefore has to be added separately afterwards, as described above. With the logarithmic frequency axis used in the Bode diagram, the time delay contributes little to the phase at $\omega << 1/T$, but adds a lot of negative phase at higher frequencies.

To use the above description to account for the phase and magnitude contributions of complex-valued poles or zeros (which have to appear in complex conjugate pairs), use the absolute value of the poles or zeros instead of the complex-valued p_i or z_i. Note that if the system has complex conjugate poles close to the imaginary axis, the magnitude plot may have a large 'spike' that is not captured by the asymptote.

Note from the above description that the phase contribution at low frequencies of a zero in the right half plane is essentially the same as that of the zero's 'mirror image' in the left half plane, whereas at high frequencies the phase contribution of the two differ by 180°.

In contrast, the phase contribution at low frequencies of a pole in the right half plane is 180° different from that of its 'mirror image' in the left half plane, but at high frequencies the phase contribution of the two are essentially the same.

The asymptotes are shown with dashed lines in Fig. 2.3.4). The system $g(s) = \frac{s+0.1}{(s+0.01)(s+1)}$ has a steady state gain of 10, no pure integrators or differentiators. The magnitude asymptote therefore starts with a gradient of 0, while the phase asymptote starts with a phase of 0°. The first pole is at $p_i = 0.01$. At $\omega = 0.01$, the gradient of the magnitude asymptote therefore changes to -1, whereas the phase asymptote goes to $-90°$. At $\omega = 0.1$ we encounter the (LHP) zero, and thus the gradient of the magnitude asymptote increases to 0, and the phase asymptote goes to 0° again. Finally at $\omega = 1$ we encounter the second pole, changing the gradient of the magnitude asymptote to -1 and the phase asymptote to $-90°$.

Minimum phase systems. It should be clear from the above that whether a pole or a zero is in the right or left half plane does not affect the Bode magnitude plot, whereas it does affect the phase plot. It turns out that for any system with a given magnitude plot[6], there is a minimum possible phase that the system can have. This minimum possible phase can be quantified in terms of the Bode phase-gain relationship, which from which the minimum possible phase can be calculated from an integral over all frequencies of an expression involving the magnitude. The precise form of this expression is of little importance in our context, the interested reader may consult [93] or other textbooks on linear systems theory. One can, however, find

[6]assuming that this magnitude plot makes physical sense, i.e., that it can correspond to a state-space model

from the expression that the local phase depends strongly on the local gradient of the magnitude in the loglog plot (the Bode magnitude plot). Thus, the minimum possible phase is approximately given by

$$\angle G(j\omega)_{min} \approx -90^\circ \cdot \frac{d\log(|G(j\omega)|)}{d\log(\omega)} \tag{2.41}$$

Whereas this approximation is exact at all frequencies only for a series of integrators ($G(s) = s^{-n}$), it can be a reasonable approximation for most minimum phase systems except at frequencies where complex poles or zeros are close to the imaginary axis.

From the brief introduction to frequency analysis presented above, it should be clear that a minimum-phase system has

- no poles or zeros in the right half plane, and
- has no time delay.

Minimum phase systems are generally easy to control, as the system dynamics pose no special limitations or requirements for feedback control. In contrast, as we will see later in this course, RHP poles implies a minimum bandwidth requirement, whereas RHP zeros or time delays implies a bandwidth limitation.

2.3.5 Assessing closed loop stability using the open loop frequency response

Let $L(s)$ be the loop transfer function matrix of a feedback system, as illustrated in Fig.2.3.5. The loop transfer function $L(s)$ may be monovariable and multivariable,

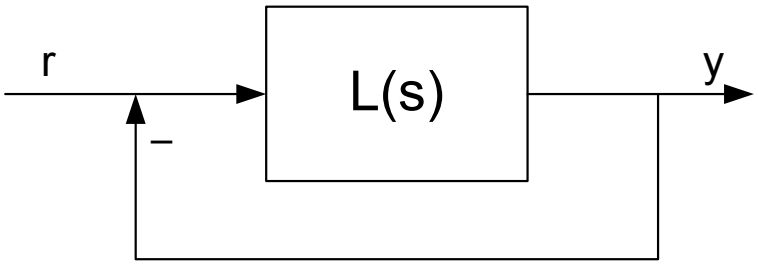

Figure 2.5: A simple feedback loop.

and a feedback control setting typically results from connecting a controller $K(s)$ and a plant $G(s)$ in series, i.e., $L(s) = G(s)K(s)$. We will assume that there are no hidden (unobservable or uncontrollable) unstable modes in $L(s)$, and are interested in determining closed loop stability based on open loop properties of $L(s)$. The Nyquist stability theorem provides such a method for determining closed loop stability, using the socalled *Principle of the Argument*.

The Principle of the Argument and the Nyquist D-contour

The Principle of the Argument is a result from mathematical complex analysis. Let $t(s)$ be a transfer function and C be a closed contour in the complex plane. Assume that the transfer function $t(s)$ has n_Z zeros and n_P poles inside the closed contour C, and that there are no poles on C.

The Principle of the Argument. Let s follow C once in the clockwise direction. Then, $t(s)$ will make $n_Z - n_P$ clockwise encirclements of the origin.

In this context the term 'Argument' refers to the phase of the transfer function.

We are interested in stability of the closed loop, which clearly means that we want to investigate whether the closed loop has any poles in the right half plane. Thus, the contour C will in our case be the 'border' of the entire right half plane, i.e., the entire imaginary axis - turned into a closed loop by connecting the two ends with an 'infinitely large' semi-circle around the right half plane[7]. To fulfill the requirement that there should be no poles on the closed contour, we must make infinitesimal 'detours' into the right half plane to go around any poles on the imaginary axis (most commonly due to pure integrators in the plant $G(s)$ or controller $K(s)$). The closed contour described above is commonly known as the *Nyquist D-contour*.

The Multivariable Nyquist Theorem

It can be shown (e.g., [69]) that the open and closed loop characteristic polynomials are related through

$$\det(I + L(s)) = \frac{\phi_{cl}(s)}{\phi_{ol}(s)} \cdot c \tag{2.42}$$

where c is a constant. The number of open loop poles in the RHP cannot be changed by feedback. However, for closed loop stability we must ensure that there are no closed loop poles in the RHP. Using the principle of the argument, we thus arrive at the General or Multivariable Nyquist Theorem:

Let the number of open loop unstable poles in $L(s)$ be n_{ol}. The closed loop system with negative feedback will then be stable if the plot of $\det(I + L(s))$ does not pass through the origin, but makes $-n_{ol}$ (clockwise) encirclements of the origin as s traverses the Nyquist D-contour.

Note that in practice we only need to plot $det(I + L(s))$ for positive frequencies only, since the plot for negative frequencies can be obtained by mirroring about the real axis.

[7] A brief look at the expression for $G(s)$ in (2.26) - while remembering that the transfer function $t(s)$ above can be expressed similarly - should suffice to convince the reader that the value of $t(s)$ will remain constant as s traveses the 'infinitely large semicircle' around the RHP. For *very* large s, $C(sI - A)^{-1}B \approx 0$ regardless of the direction from the origin to s.

The monovariable Nyquist Theorem

Most readers are probably more familiar with the monovariable Nyquist theorem, which follows from the multivariable version by noting that for a scalar $L(s)$ it is equivalent to count encirclements of $\det(I+L(s))$ around the origin and encirclements of $L(s)$ around -1.

The Bode stability criterion

The Bode stability criterion follows from the monovariable Nyquist Theorem and *thus applies only to monovariable systems*. Let ω_c denote the 'crossover frequency', i.e., $|L(j\omega_c)| = 1$, and assume that $|L(j\omega)| < 1$ for $\omega > \omega_c$. Then the closed loop system is stable provided $\angle L(j\omega_c) > -180°$.

The Bode stability criterion ensures that the Nyquist plot of $L(s)$ passes between the origin and the critical point -1 in the complex plane. For open loop stable systems it is then straight forward to see that there can be no encirclements of the critical point. However, the criterion may also be used for open loop unstable systems provided the Bode phase plot starts from the correct phase of $-180° \cdot n_{ol}$, and the crossover frequency ω_c is unique (i.e., that there is only one frequency ω_c for which $|L(j\omega_c)| = 1$).

If either of the assumptions i) $|L(j\omega)| < 1$ *for* $\omega > \omega_c$ or ii) *uniqueness of* ω_c are violated, the Bode stability criterion is easily misinterpreted, and the use of the Nyquist criterion is recommended instead.

For open loop stable systems the Bode stability criterion may equivalently be stated in terms of ω_{180}, defined such that $\angle L(j\omega_{180}) = -180°$. The closed loop system is then stable if $|L(j\omega)| < 1$ for $\omega \geq \omega_{180}$. For most systems, the magnitude $|L(j\omega)|$ will decrease with increasing frequency, and it will thus suffice to check the criterion only at ω_{180}. However, this version of the criterion cannot be used for open loop unstable systems, since ω_{180} need not be uniquely defined - and the criterion must indeed be violated for one or more of the ω_{180}'s.

Example. Consider the unstable system $g(s) = \frac{1}{10s-1}$, that we want to stabilize with the proportional feedback controller k. The closed loop pole can be found from the closed loop characteristic polynomial, by solving the equation $1 + g(s)k = 0$. We thereby find that the closed loop pole is located at $s = \frac{1-k}{10}$, and the closed loop will be stable for $k > 1$. We note that $\omega_{180} = 0$, and that $\angle L(j\omega) > -180° \forall \omega > 0$. We can easily calculate $\omega_c = \frac{\sqrt{k^2-1}}{10}$. That is, for $k < 1$, $|L(j\omega)| = |g(j\omega)k| < 1 \forall \omega$, and there is thus no crossover frequency ω_c. Thus, we find also from the Bode stability criterion (in terms of ω_c) that we need $k > 1$ for stability. The Bode stability criterion in terms of ω_{180} would fail - but as noted above this is only valid for stable systems.

In Fig. 2.3.5 the Bode diagram for the system in this example is shown for $k = 2$. We find that $\omega_c = \frac{\sqrt{3}}{10}$ and $\angle L(j\omega_c) = -120°$, i.e., the system is stable and we have a phase margin of 60°.

Stability of the closed loop system can also be verified from the monovariable Nyquist theorem. We find that the image of $L(s)$ under the Nyquist D-contour

encircles the critical point $(-1, 0)$ once in the anti-clockwise direction, as shown in Fig. 2.3.5.

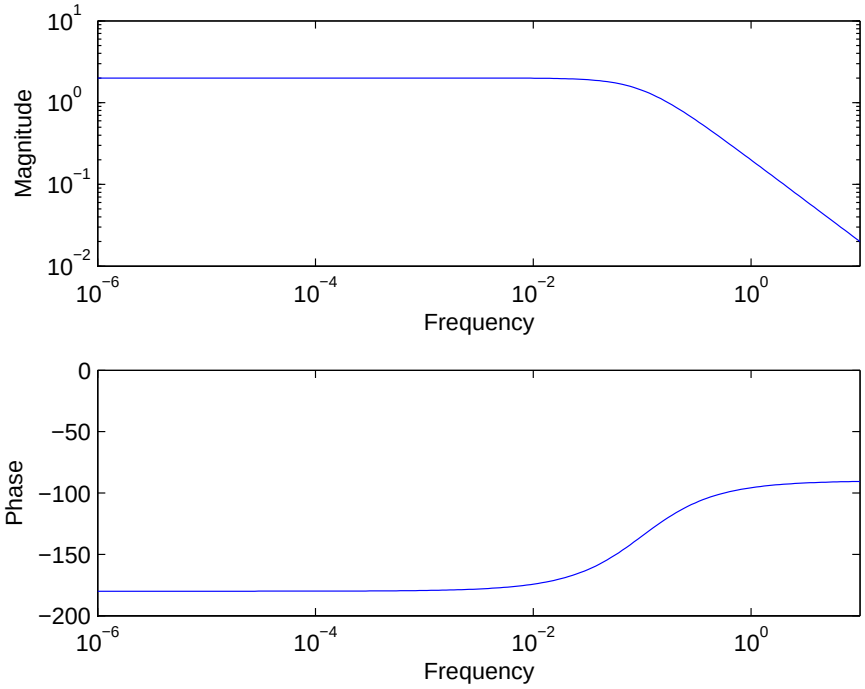

Figure 2.6: Bode diagram for the system $L(s) = \frac{2}{10s-1}$.

Some remarks on stability analysis using the frequency response

Frequency analysis can indeed be very useful. However, some remarks seem to be needed to warn against misuse of frequency analysis for analyzing stability:

- The Nyquist stability theorems and the Bode stability criterion are tools to assess *closed loop stability based on open loop frequency response data.*
- Knowledge of the number of open loop unstable poles is crucial when using Nyquist or Bode.
- Nyquist or Bode should **never** be used to assess open loop stability!
- It is *utterly absurd* to apply the Bode stability criterion to the individual elements of a multivariable system, the Bode stability criterion applies to monovariable systems only. Use the multivariable Nyquist theorem to assess closed loop stability of multivariable systems based on the open loop frequency response.

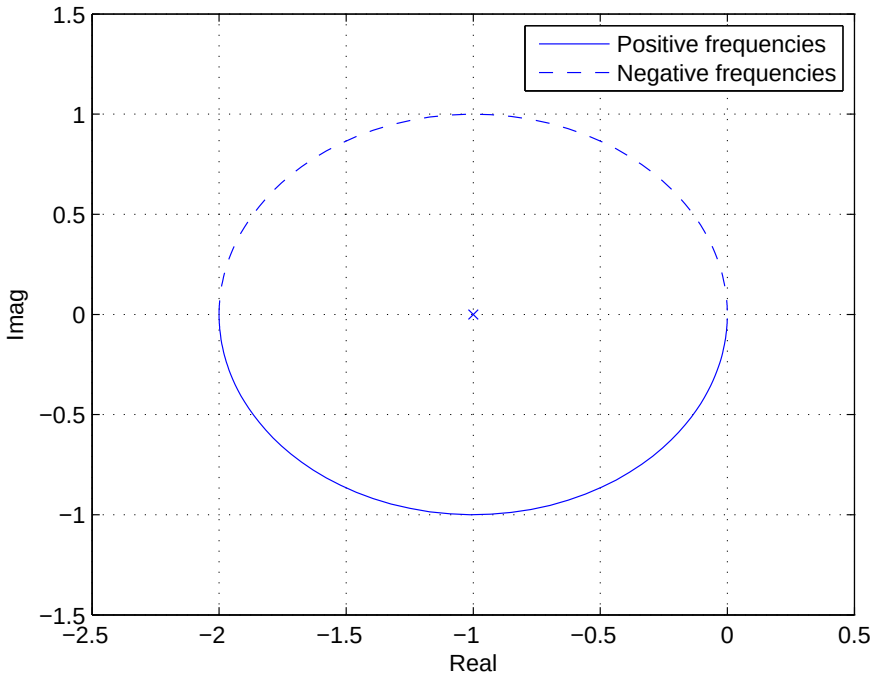

Figure 2.7: The monovariable Nyquist theorem applied to the system $L(s) = \frac{2}{10s-1}$. The curve encircles the critical point $(-1, 0)$ once in the anti-clockwise direction, and the system is hence stable.

Chapter 3

Limitations on achievable control performance

3.1 Introduction

This section will discuss the factors that limit the achievable performance of a control system under feedback control. In order to simplify the discussion, we will first introduce some notation, and discuss how different properties of the closed loop system is related to the different closed loop transfer functions. Thereafter, limitations on achievable performance will be discussed.

3.2 Notation

There is no universally accepted notation in control engineering. However, the notation that is introduced in this section is similar to what is used in much of the more recent control literature. The reader is referred to Fig.3.1.

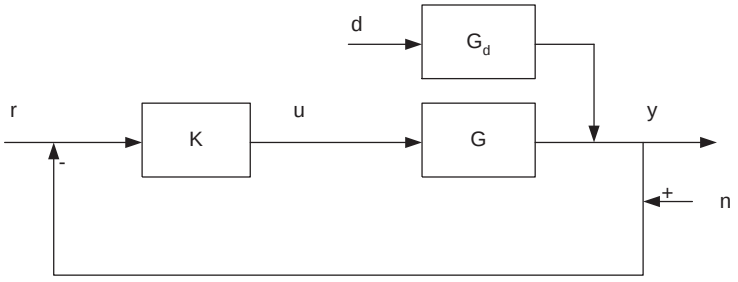

Figure 3.1: An illustration of some of the notation used for describing a feedback control system.

The signals r,u,y,d, and n are the reference signal (setpoint), the manipulated variable, the controlled variable, the disturbance, and the measurement noise, respec-

tively. For multivariable systems, these signals are vectors. The measured value of the controlled variable differs from the true value due to the noise, thus $y_m = y + n$. The transfer functions G, G_d, and K represent the effect of manipulated variables on controlled variables, the effect of the disturbances on the controlled variables, and the controller (how the control offset determines the manipulated variables). For multivariable systems, G, G_d, and K are matrices. Thus, the overall plant is given by

$$y = \begin{bmatrix} G & G_d \end{bmatrix} \begin{bmatrix} u \\ d \end{bmatrix}$$

and has a state space representation given by

$$\begin{aligned} \dot{x} = &\quad Ax + Bu + Ed \\ y = &\quad Cx + Du + Fd \end{aligned}$$

The transfer functions and the state space representation are related through
$\begin{bmatrix} G(s) & G_d(s) \end{bmatrix} = C(sI - A)^{-1} \begin{bmatrix} B & E \end{bmatrix} + \begin{bmatrix} D & F \end{bmatrix}$

The dependency on the complex variable s will frequently be omitted for notational brevity. Unless explicitly stated, we will assume whenever necessary that the state space realisation is minimal, i.e., that all states are observable in y and controllable from $\begin{bmatrix} u & d \end{bmatrix}^T$. Note that all states need not be controllable from u or d alone, thus for numerical computations involving G or G_d alone, one should ensure that a minimal realisation is used.

3.3 Closed loop transfer functions and closed loop system responses

This section will introduce some important closed loop system transfer functions, and discuss their relationship to different closed loop responses relevant to control.

The response of the controlled variable is given by

$$y = (I + GK)^{-1} G_d d + (I + GK)^{-1} GK(r - n) = SG_d d + T(r - n) \tag{3.1}$$

where $S = (I + GK)^{-1}$is known as the (output) sensitivity function, and $T = (I + GK)^{-1}GK = GK(I + GK)^{-1}$ (the latter equality holds also for multivariable systems) is known as the (output) complementary sensitivity function. The transfer function matrices S and T are complementary in the sense that $S + T = I$ by definition.

The response of the manipulated variable is given by

$$\begin{aligned} u &= K(I+GK)^{-1}(r-n) - K(I+GK)^{-1}G_d d \\ &= (I+KG)^{-1}K(r-n) - (I+KG)^{-1}KG_d d \qquad (3.2) \\ &= S_I K(r-n) - S_I K G_d d \qquad (3.3) \end{aligned}$$

where S_I is known as the input sensitivity function. Note that for multivariable systems, S_I is in general different from S.

From Eq.3.1 we see that in order to reject disturbances, we want the sensitivity function S to be small. A small S corresponds to a complementary sensitivity function $T \approx I$.[1] This means that the closed loop system will track setpoint changes very accurately, but unfortunately also means that measurement noise (n) will directly affect the controlled variable. In addition, model uncertaincy will in general imply a need for controller loop gain (exactly what closed loop transfer function needs to be small will depend on the assumed location and structure of the uncertainty, see [92, 69] for details). We thus have four seemingly contradictory requirements to fulfill, disturbance rejection and setpoint following implies high controller gain, whereas the sensitivity to measurement noise and model uncertainty implies low controller gain. Fortunately, these relative importance of these requirements normally depend on frequency. In process control, setpoint tracking and disturbance rejection is often considered more important at low frequencies than at high frequencies, since fast variations in the controlled variables often have little impact on plant profitability. On the other hand, measurement noise is often more severe at high frequencies (steady state bias and slow drift should be counteracted with proper calibration). It is normally simpler to obtain a plant model which is accurate at low frequencies, than to obtain good model accuracy at high frequencies.

3.4 Limitations on achievable performance

3.4.1 Control performance in different frequency ranges

Bode [12] showed for open loop stable, monovariable systems that improved control at some frequencies will necessarily imply poorer control at other frequencies (under mild conditions on the plant transfer function), which can be expressed by the following integral:

$$\int_0^\infty \log|S(j\omega)|\, d\omega = 0 \qquad (3.4)$$

[1] For a multivariable system, S must be small *in all directions* for $T \approx I$ to hold. One may state that the equality $S + T = I$ implies that the gain of $T \approx 1$ *in the directions where* S is small. However, this statement may be criticized for lack of precision, since we have not defined what we mean by directions in a multivariable system.

This implies that a sensitivity reduction in one frequency range must be balanced by a sensitivity increase at other frequency ranges. Typically, the sensitivity function is small at low frequencies and $|S| \approx 1$ (i.e., $\log |S| = 0$) at high frequencies. Therefore, we must have $|S| > 1$ in the bandwidth region, and thus get increased sensitivity to disturbances in that region. It may appear from Eq. (3.4) that one can stretch out the area with increased sensitivity *ad infinitum*, and therefore only get an infinitesimal peak in the sensitivity function. However, this will only be approximately true if a low control bandwidth is used. In most practical cases, the loop gain will have to be reduced with increasing frequency also at frequencies immediately above the loop bandwidth (i.e., the loop gain has to "roll off"). This implies that there will be a definite peak in the magnitude of the sensitivity function, and that also the peak value has to increase if the sensitivity function is reduced in another frequency range. These issues are discussed further by Freudenber and Looze, see [25], who also show that the result in Eq. (3.4) can be generalized to multivariable systems:

$$\int_0^\infty \log |\det(S(j\omega))| \, d\omega = 0 \tag{3.5}$$

For *constant (frequency-independent) directions,* one may transform the inputs and outputs such that the relationship between given input-output directions are described by a scalar sensitivity function $\tilde{S}$, and the relationship in Eq. (3.4) will apply for the transformed sensitivity function $\tilde{S}$. However, if one allow the directions in multivariable space to vary with frequency (like the input and output singular vectors of S will normally do), it is not fully understood whether it is possible to trade off increases in sensitivity in different directions.

3.4.2 Zeros in the right half plane

If the open loop plant has zeros in the right half plane, this will impose a restriction on achievable performance. The implications of RHP zeros are somewhat more complicated for multivariable systems than for monovariable systems. We will therefore discuss monovariable systems first.

Monovariable systems. In order to obtain an internally stable closed loop system, a RHP zero cannot be cancelled by the controller, nor can a zero be moved by feedback. Thus, a RHP zero that occurs in the open loop transfer function G must also appear in the closed loop complementary sensitivity function. That is, consider an open loop transfer function G with a zero at location z in the right half plane. We then have

$$\begin{aligned} G(z) &= 0 \\ T(z) &= G(z)K(z)(I + G(z)K(z))^{-1} = 0 \\ S(z) &= 1 \end{aligned}$$

Freudenberg and Looze [25] have also extended Bode's sensitivity integral to account for RHP zeros, and obtain for an open loop stable plant with a RHP zero at z that:

$$\int_0^\infty \log|S(j\omega)|\, W(z,\omega)d\omega = 0 \tag{3.6}$$

where the wheighting function $W(z,\omega)$ is given by

$$W(z,\omega) = \frac{2z}{z^2+\omega^2} \tag{3.7}$$

For a complex RHP zero at $z = x + jy$, the weight $W(z,\omega)$ becomes

$$W(z,\omega) = \frac{x}{x^2+(y-\omega)^2} + \frac{x}{x^2+(y+\omega)^2} \tag{3.8}$$

This means that most of the sensitivity increase that is needed to fulfill Eq. (3.6) must come at frequencies below the frequency corresponding to the right half plane zero, and that the bandwidth of the closed loop system is effectively constrained to be somewhat below the frequency of the RHP zero. As the bandwidth approaches the frequency of the RHP zero, the peak value of the magnitude of the sensitivity function will increase.

Multivariable systems. Consider the simple 2×2 plant

$$y(s) = G(s)u(s) = \frac{1}{s+1}\begin{bmatrix} 1 & s+1 \\ 2 & s+4 \end{bmatrix} u(s) \tag{3.9}$$

Assume that $a > 0$, i.e., the system is open loop stable. None of the elements of $G(s)$ have zeros in the right half plane. Controlling output y_1 with the controller $u_1(s) = k_1(r_1(s) - y_1(s))$, we get

$$\begin{aligned} y_1 &= \frac{g_{11}k_1}{1+g_{11}k_1}r_1 + \frac{g_{12}}{1+g_{11}k_1}u_2 \\ y_2 &= \frac{g_{21}k_1}{1+g_{11}k_1}r_1 + \left(g_{22} + \frac{g_{21}g_{12}k_1}{1+g_{11}k_1}\right)u_2 \end{aligned}$$

where the term inside the brackets is the transfer function from u_2 to y_2 when y_1 is controlled by u_1, in the following this is denoted $\widetilde{g}_2$. Assume that a simple proportional controller is used, i.e., $k_1(s) = k$ (constant). Some tedious but straight forward algebra then results in

$$\widetilde{g}_2(s) = \frac{1}{(s+1)(s+1+k)}\left[(s+4)(s+1+k) - 2k(s+1)\right]$$

We can then easily see that the system is stable provided $k > -1$ (clearly, a positive value for k would be used). For small values of k, $\widetilde{g}_2$ has two real zeros in the left half plane. For $k = 9 - 3\sqrt{8}$, the zeros become a complex conjugate pair, and the

zeros move into the right half plane for $k > 5$. For $k = 9 + 3\sqrt{8}$, both zeros again become real (but positive), and if k is increased further, one zero approaches $+\infty$ whereas the other zero approaces $+2$. Now, a zero of $\widetilde{g}_2(s)$ far into the right half plane will not significantly affect the achievable bandwidth for loop 2, but the zero which at high values of k approaches $+2$ certainly will.

Note that it will not be possible to avoid the zero in $\widetilde{g}_2(s)$ by using a more complex controller in loop 1. The transfer function $\widetilde{g}_2(s)$ will have a zero in the vicinity of $s = 2$ whenever high bandwidth control is used in loop 1.

If we instead were to close loop 2 first, we would get similar problems with loop 1 as we have just seen with loop 2. That is, if loop 2 were controlled fast, the transfer function from u_1 to y_1 would have a zero in the vicinity of $s = 2$.

We therefore conclude that it is a property of the plant that all directions cannot be controlled fast. Looking at the term inside the square bracket in Eq.3.9, we see that the determinant of $G(s)$ looses rank at $s = 2$ (its normal rank is 2, but at $s = 2$ it has rank 1). In terms of systems theory, the plant $G(s)$ has a *multivariable (transmission) zero at* $s = 2$.

There is no direct relationship between monovariable and multivariable zeros, a zero in an individual transfer function element may be at the same location as a multivariable zero, but often that will not be the case. However, as we have seen above, if a multivariable system with n outputs has a RHPT zero, and $n - 1$ outputs are perfectly controlled using feedback, the RHPT zero will appear in any transfer function from the remaining manipulated variable to the remaining controlled variable (if the transfer function takes account of the fact that the other outputs are controlled).

Right half plane zeros in individual elements of a transfer function matrix need not imply a control performance limitation (they may become serious limitations, however, if parts of the control system is taken out of service, leaving only the loop with the monovariable RHP zero in service). There are several definitions of multivariable zeros, but the multivariable zeros which have serious control implications are the so-called *transmission zeros* in the right half plane (RHPTZ, for short).

A transmission zero is a point in the complex plane where the transfer function matrix looses its normal rank. A multivariable transmission zero is a zero which has the property that it can (if the system initially is at a specific state) fully block the effect on the outputs of an expotentially growing input signal. It is normally computed numerically by starting from a minimal state space realization, and solve a generalized eigenvalue problem

$$\begin{aligned} (zI_g - M)\begin{bmatrix} x_z \\ u_z \end{bmatrix} &= 0 \\ M &= \begin{bmatrix} A & B \\ C & D \end{bmatrix} \\ I_g &= \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{3.10}$$

Multivariable transmission zeros can be computed using a standard Matlab function. The state vector x_z obtained from Eq.(3.10) is the initial condition required for the "transmission blocking" phenomenon, and u_z is known as the zero input vector. Likewise, a zero output vector can be calculated from solving Eq.(3.10) using M^T instead of M.[2] When the magnitudes of the input and output zero vectors are normalized to unity, they are called input and output zero *directions*, respectively. A transmission zero is invariant to feedback, just as monovariable zeros. The location of the zero is invariant to scaling of the inputs and outputs, but the zero directions are not.

The zero directions give information about which inputs and outputs are affected by the RHPT zero, the zero is said to be *pinned* to the inputs/outputs corresponding to non-zero elements in the zero directions. Thus, achievable control performance will be impaired for at least one of the controlled variables corresponding to non-zero elements in the zero output direction. In theory, the RHPT zero can be made to affect only one (and any one) of the controlled variables corresponding to the non-zero elements in the zero output direction. However, large moves in the manipulated variables will normally be required if one wants to steer the effect of an RHPT zero to a controlled variable corresponding to a *small* element of the zero output direction, see Morari and Zafiriou [69, 38] for details. In addition, since $y_z^H G(z) = 0$ the following relationships must hold for the closed loop system to be internally stable:

$$\begin{aligned} y_z^H T(z) &= 0 \\ y_z^H S(z) &= y_z^H \end{aligned}$$

Thus, T must have a zero in the same output direction as G, and S must have a eigenvalue of 1 with corresponding left eigenvector y_z.

Multivariable RHPT zeros affect the sensitivity function in a way similar to the effect of RHP zeros of monovariable systems. However, the available mathematical results are not as tight. Freudenberg and Looze [25] state the following result:

$$\int_0^\infty \log \overline{\sigma}(S(j\omega)) W(z,\omega) d\omega \geq 0 \tag{3.11}$$

where $W(z,\omega)$ is the same as in equation Eq. (3.7) or Eq. (3.8), as appropriate. The implications of multivariable RHP zeros are discussed further in [92].

3.4.3 Unstable systems

It is well known that poles in the right half plane means that the system is unstable in open loop, and needs to be stabilized by feedback. Cancelling the unstable pole with a RHP zero in the controller will result in an internally unstable system. The poles of the system corresponds to the eigenvalues of the autotransition matrix A. In

[2] A numerically less robust way of calculating u_z and y_z is by using the SVD of the transfer function matrix calculated at the zero.

contrast to RHP zeros, RHP poles imposes a bandwidth *requirement* rather than a bandwidth limitation, the control needs to be fast enough to "capture" the instability before the plant runs away.

Let a plant G have n_p poles in the open right half plane (including multiplicities), at locations $\{p_i; i = 1, \cdots, n_p\}$. Then, if the closed loop system is stable, it must satisfy (see [25] for details)

$$\int_0^\infty \log|S(j\omega)|\, d\omega = \pi \sum_{i=1}^{n_p} \mathrm{Re}(p_i) \tag{3.12}$$

For multivariable systems, Eq. (3.12) applies to $|\det(S(j\omega))|$ instead of $|S(j\omega)|$. This means that poles in the right half plane will make the control worse, and the peak value of the sensitivity function will generally increase as the system bandwidth decreases.

For multivariable systems, one can define pole input and output directions, much in the same way as for multivariable zeros. The pole directions can be calculated from the state space realisation. If p is an eigenvalue of A, and t and q are the corresponding right and right eigenvectors, then

$$\begin{aligned} At &= pt, \qquad q^H A = q^H p \\ y_p &= Ct, \qquad u_p = B^H q \end{aligned} \tag{3.13}$$

Here y_p and u_p are the output and input pole directions, respectively (to be strict, they should be normalized to magnitude one). See Havre [36] for full details on the computation of pole and zero directions.

For a monovariable system G with a RHP pole at p (i.e., $G(p) = \infty$), then for internal stability the following relationships must apply:

$$\begin{aligned} T(p) &= 1 \\ S(p) &= 0 \end{aligned} \tag{3.14}$$

The multivariable counterparts to Eq. (3.14) are

$$\begin{aligned} T(p)y_p &= y_p \\ S(p)y_p &= 0 \end{aligned} \tag{3.15}$$

Combined effects of RHP poles and zeros

For monovariable systems with a RHP zero at z and RHP poles at $\{p_i; i = 1, \cdots, n_p\}$ (including multiplicities), Freudenberg and Looze [25] find that

$$\int_0^\infty \log|S(j\omega)|\, d\omega = \pi \prod_{i=1}^{n_p} \left| \frac{p_i + z}{p_i - z} \right| \tag{3.16}$$

How the sensitivity function for a multivariable system is affected in different directions by the presence of poles and zeros is not fully understood, but it is clear that this must depend on pole and zero directions, as illustrated by the following simple example:

$$G_1(s) = \begin{bmatrix} f(s) & 0 \\ 0 & \frac{s-z}{s-p} \end{bmatrix}; \qquad G_2(s) = \begin{bmatrix} \frac{s-z}{\tau s+1} & 0 \\ 0 & \frac{\tau_1 s+1}{s-p} \end{bmatrix}$$

Here $f(s)$ can be an arbitrary minimum phase scalar transfer function. Clearly, for $G_1(s)$ the combined effects of the RHP pole and zero will only affect output 2, whereas for $G_2(s)$ the pole only affects output 2 and the zero only affects output 1.

Another difference between monovariable and multivariable systems is that for multivariable systems there may be a pole and zero at the same location in the complex plane (same value of s), even for a minimal state space realization. This is possible if the pole and zero directions are not parallel.

3.4.4 Time delays

For a monovariable plant with a time delay of θ, a time θ must obviously pass between a step change in the reference value and the response in the controlled variable - not even an ideal controller can do better. The socalled Smith predictor is a controller structure which may allow a closed loop response close to that of the ideal controller. However, achieving anything like such an ideal response will require a very accurate plant model, and the Smith predictor is indeed found to be sensitive to model uncertainty (especially uncertainty in the time delay) unless it is conservatively tuned. Skogestad and Postlethwaite [92] argue that due to robustness considerations the gain crossover frequency (the frequency at which the gain crosses 1) is approximately limited to $\omega_c \leq 1/\theta$.

For multivariable systems, the minimum time that must pass between a step change in the reference value for output i and a response in the output is obviously the same as the smallest time delay in any of the elements in row i of the transfer function matrix. Holt and Morari [37] derive additional bounds, but their usefulness is not clear as they assume decoupled responses in the controlled outputs - which is often not desirable if one wants to optimize control performance.

3.4.5 Limitations due to uncertainty in the plant model

For monovariable systems, model uncertainty will generally have little effect on the control performance at frequencies where the loop gain is high. In the bandwidth region, model uncertainty can cause both poor performance and instability. This can be investigated easily by evaluating the gain and phase margins, or somewhat more rigorously by analyzing the Nichols chart.

The effect of model uncertainty for monovariable feedforward control is discussed by Balchen in [11].

For multivariable systems the situation is much more complex, and small errors in the individual model elements can have disastrous effect on the overall system.

Problems are most likely in the crossover region also for multivariable systems. This can be analyzed rigorously by evaluating the socalled *structured singular value*, see e.g., [92, 69] for details. However, this is a quite complicated concept and advanced mathematics is required in this type of analysis, and this issue will not be explored here. We will only note that problems due to model uncertainty in multivariable systems is particularly likely if the plant is *inherently ill-conditioned*, i.e., the plant is ill-conditioned regardless of what scaling is used for the inputs and outputs. The conventional condition number (ratio of largest to smallest singular value)is thus not a good measure of inherent ill-conditioning, since it is scaling dependent - and obtaining the optimal scaling is non-trivial. A simple alternative measure is the Relative Gain Array (RGA), given by

$$\Lambda(G) = G \times (G^{-1})^T \tag{3.17}$$

where $\times$ denotes the element-by-element (Hadamard or Schur) product. The RGA is independent of scaling, and all rows and columns of Λ sum to 1. Large elements in $\Lambda(G)$ (relative to 1) imply that the plant is inherently ill-conditioned and sensitive to uncertainty. The RGA has a number of other uses which we will return to in later sections.

3.4.6 Limitations due to input constraints

Whenever input constraints are active, control performance will obviously suffer. This likely to occur if $\bar{\sigma}(G^{-1}G_d) > 1$ (provided the disturbances and manipulated variables are appropriately scaled).

Chapter 4

Control structure selection

4.1 Introduction

This section addresses the design of control structures. It starts off by describing several common control loop configurations. Thereafter, more fundamental issues will be discussed, such as

- What variables should be controlled?
- What variables should be manipulated to control the controlled variables?
- What structure should be imposed on the interconnections between the controlled and manipulated variables?

The focus of this note is on the lower layer in the control system, the *regulatory control* layer. The main purpose of of the regulatory control layer is to keep the plant in safe and stable operation, by keeping the controlled variable at or close to their *setpoints*. The actual values of these setpoints will be determined by higher levels in the control hierarchy[1]. Thus, it is the task of the higher levels to identify the optimal operating conditions, whereas the regulatory control layer is important for obtaining and maintaining optimal conditions.

4.2 Common control loop structures for the regulatory control layer

In this section the more common control loop structures for the regulatory control layer are described.

[1]The higher levels in the control system may be automated, but the tasks of the higher levels may also be performed by human operators.

4.2.1 Simple feedback loop

This is by far the more common control loop structure for the regulatory control level, and is illustrated in Fig. 4.1. The controller acts on the difference between

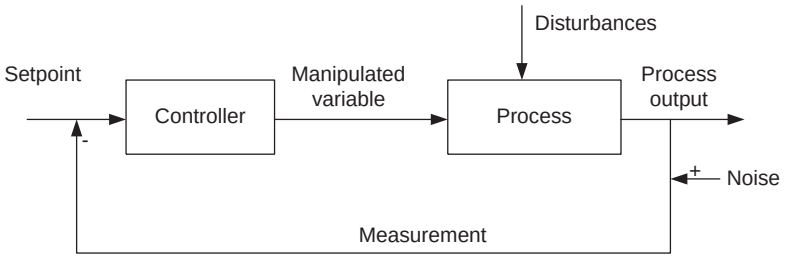

Figure 4.1: Simple feedback control loop.

the desired value for the process output (i.e., the *setpoint* or *reference value*) and the *measurement* of the process output. In order to make the measurement equal the setpoint, a process input is manipulated by the controller, this process input is then known as the *manipulated variable*. Note that the measured value need not equal the actual process output value, due to possible measurement noise or malfunctions. Note also that the manipulated variable is normally one of several process inputs which affects the value of the process output, there are normally additional process inputs which will affect the process output. These additional process inputs which are not manipulated by the controller are termed *disturbances*. The need for feedback of the process measurement arises from uncertainty both with respect to the value of the disturbances, and with respect to the process response. If we could know exactly the value of all disturbances, and the response of the process to both the disturbances and the manipulated value, the measurement would be superfluous, since we would know the exact value of the process output for a specific value of the manipulated variable. In practice such exact process knowledge is unrealistic, and hence feedback of the measurement is needed if accurate control of the process output is required.

4.2.2 Feedforward control

Feedforward control is used to counteract the effect of disturbances without first having to wait for the disturbances to affect the process output. This is illustrated in Fig.4.2

The ideal feedforward signal is the one which exactly cancels the effect of disturbances, i.e.,

$u = u_{ff} + u_{fb}$

where u_{ff} is the output of the feedforward controller. The ideal value of u_{ff} is then given by

$y = Gu_{ff} + G_d d = 0 \Leftrightarrow u_{ff} = -G^{-1}G_d d$

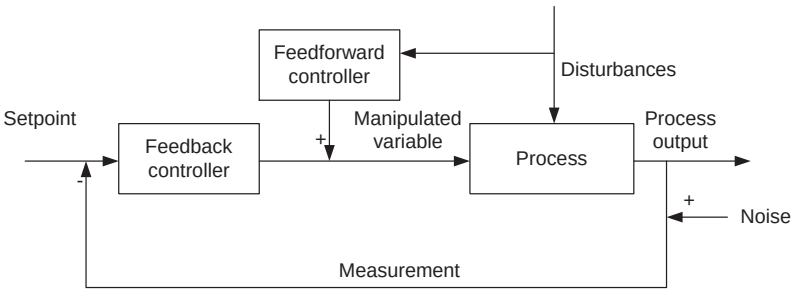

Figure 4.2: Feedforward control from measured disturbances combined with ordinary feedback control.

Clearly, in order to implement the ideal feedforward controller, $G^{-1}G_d$ must be both stable and realizable, and both the process transfer function G and the disturbance transfer function G_d must be known with reasonable accurately. The effect of model inaccuracy on performance of feedforward control is described in e.g. [11] . Since the feedforward control cannot be expected to be perfect, the feedback controller will still be needed if accurate control is required.

A pure feedforward controller cannot by itself cause instability of the closed loop system, and it is therefore very useful whenever there are bandwidth limitations which limit the achieveable performance of a feedback controller, since most such bandwidth limitations (with the exception of input constraints/ input rate of change constraints) will not apply to the feedforward controller. On the other hand, feedforward cannot be used to stabilize an unstable system.

4.2.3 Ratio control

Ratio control may be used whenever the controlled variable is strongly dependent of the ratio between two inputs. Simple examples of control problems where this type of control structure is appropriate are

- Mixing of hot and cold water to get warm water at a specified temperature.
- Mixing of a concentrated chemical solution with a diluent to obtain a dilute chemical solution.

Ratio control may be considered as a special case of feedforward control. It is particularly appropriate when one of the two inputs cannot be controlled, but vary rapidly. Measuring the input that cannot be controlled and applying the other input in a specific ratio to the uncontrolled one, essentially amounts to feedforward control. Figure 4.3 illustrates a typical application of ratio control in mixing two streams.

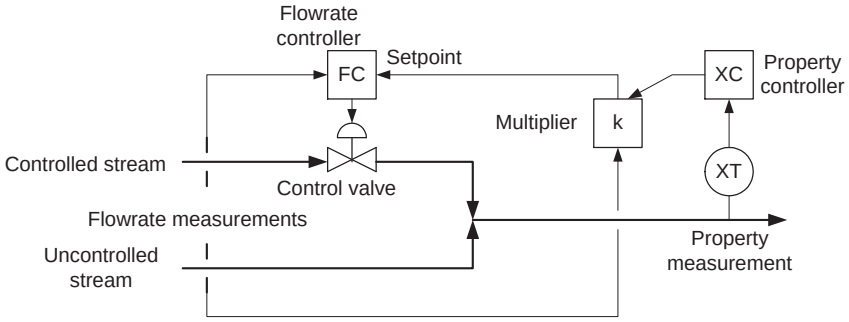

Figure 4.3: Ratio control for mixing two streams to obtain some specific property for the mixed stream. The property controller XC manipulates the multiplication factor, and thereby also the ratio between the two streams.

4.2.4 Cascade control

Cascade control is used in cases where an intermediate measurement can give an indication of what will happen with a more important primary measurement further "downstream". The use of cascaded control loops is illustrated in Fig. 4.4. Note that cascade control is used also in Fig.4.3, since the property controller (via the multiplier) manipulates the setpoint to the flow controller instead of the valve position itself. In Fig.4.3, the flow controller will counteract disturbances in upstream pressure and correct for a possibly nonlinear valve characteristic.

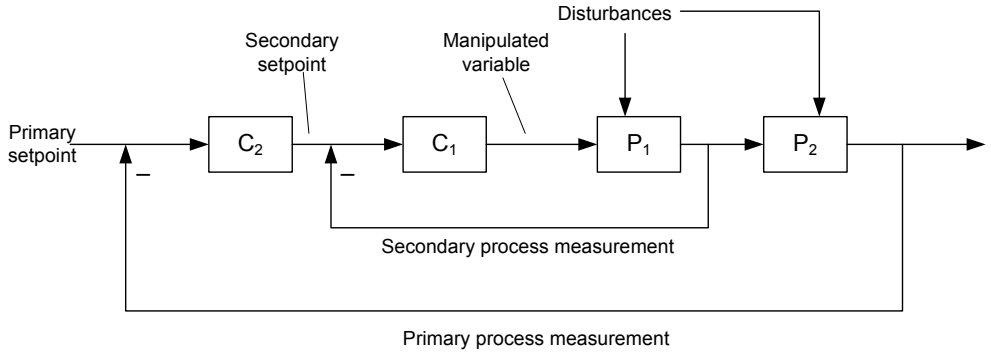

Figure 4.4: Cascaded control loops. Controller C1 controls the output of process section P1, and can counteract disturbances entering P1. The primary process measurement is controlled by controller C2, which uses the setpoint for controller C1 as manipulated variable.

In general, there may be more than two loops in cascade. For instance, a valve positioner can get its setpoint from a flow controller, which in turn gets its setpoint from a level controller (i.e., three loops in cascade).

For cascade control to make sense, the inner loops must be significantly faster than the outer loops - since the intermediate process measurements are of little interest.

If the inner loops do not provide for faster disturbance rejection (of at least some disturbances), they are not very meaningful. Fast inner loops will also make the tuning of the outer loops simpler, since one then can assume that the inner loops are able to follow their setpoints.

4.2.5 Auctioneering control

Auctioneering control is a control structure where the "worst" of a set of measurements is selected for active control, i.e. "the measurement that places the highest bid gets the control". This type of control structure is particularly common in some chemical reactors with exothermal reactions, where the process fluid flows through tubes filled with solid catalyst. If the temperature becomes too high, the catalyst will be damaged or destroyed, therefore the tubes are cooled by a cooling medium on the outside. On the other hand, if the temperature is too low, the reactions will be too slow. Thus temperature control is very important. However, the temperature will vary along the length of the reactor tubes, and the position with the highest temperature will vary with operating conditions. Therefore several temperature measurements along the reactor length are used, and the value of the highest temperature is chosen as the controlled variable. This arrangement is illustrated in Fig. 4.5.

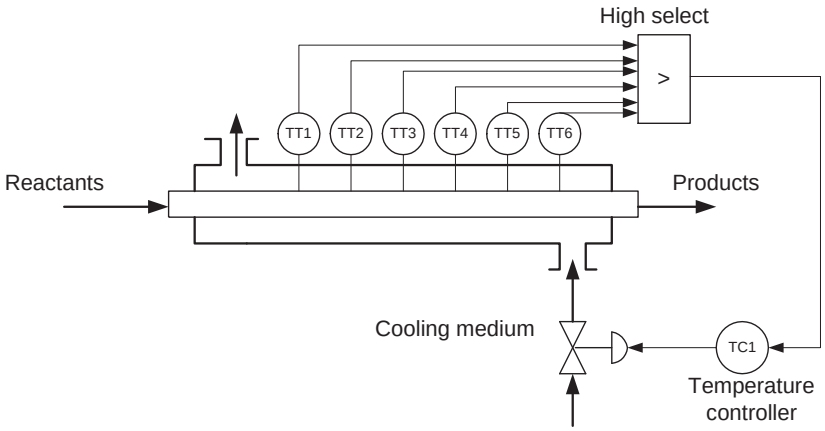

Figure 4.5: A chemical reactor with auctioneering temperature control.

For a control engineer, it might appear to be a better idea to use the temperature measurements as inputs to an estimator which estimates the maximum temperature. Such an estimator could estimate the maximum temperature when the maximum does not occur at the position of a temperature measurement, and could also be made more robust to measurement malfunction (if properly designed).However, this type of chemical reactor is normally strongly nonlinear, and the estimator would therefore need a nonlinear model, probably based on physical and chemical relationships. The modelling work needed could be time consuming, and it could also be difficult to

ascertain that the estimator performs well in all operating regions. Thus, it need not be obvious which approach to temperature control is to be preferred.

4.2.6 Split range control

In split range control, several manipulated variables are used to control one controlled variable, in such a way that when one manipulated variable saturates, the next manipulated variable takes over. In order to obtain smooth control, there is often overlap between the operating ranges of the different manipulated variables. For example, manipulated variable 1 may take value 0% at a controller output of 0%, and value 100% at controller output 60%. Similarly, manipulated variable 2 takes value 0% for controller outputs below 40%, and value 100% for controller output 100%.

It should be clear that there can be a lot of freedom in how to design the split range arrangement. It will normally be advantageous to use this freedom to make the response in the controlled variable to changes in the controller output as linear as possible.

4.2.7 Parallel control

Parallel control is similar to split range control in the sense that more than one physical input is used to control a single controlled variable. However, with parallel control, the operating ranges for the manipulated variables are divided in the frequency domain rather than being based on the magnitude of the controller output. A typical motivation for using parallel control may be that fast control is required for the controlled variable. There are several possible manipulated variables, but all the manipulated variable for which fast control is possible are expensive to use. Thus, the fast control must be performed with a manipulated variable that is expensive to use, whereas the slow control can be performed with a cheaper manipulated variable - thus allowing the expensive manipulated variable to be reset to its optimal value. Three different ways of implementing parallel control are shown in Fig. 4.6.

In Fig.4.6a), the overall effect of the controller is that of a PI controller, with integral action only for the slow, cheap manipulated variable. In Fig.4.6b), there is no integral action in the controller for the fast manipulated variable, and the fast controller will therefore leave an offset which is removed due to the integral action in the controller for the slow variable. In Fig.4.6c), the slow manipulated variable is not used to control the primary process measurement, but rather to control the value of the fast manipulated variable. Whichever of these methods for implementing parallel control are used, one should ensure that the number of pure integrators in parallel in the controller(s) do not exceed the number of feedback paths. Thus, both in Fig.4.6a) and b) the integral action acts only on the slow manipulated variable. If there are more integrators in parallel in the controllers than the number of feedback paths, all the integrators cannot be stabilized by feedback. The result can be that the

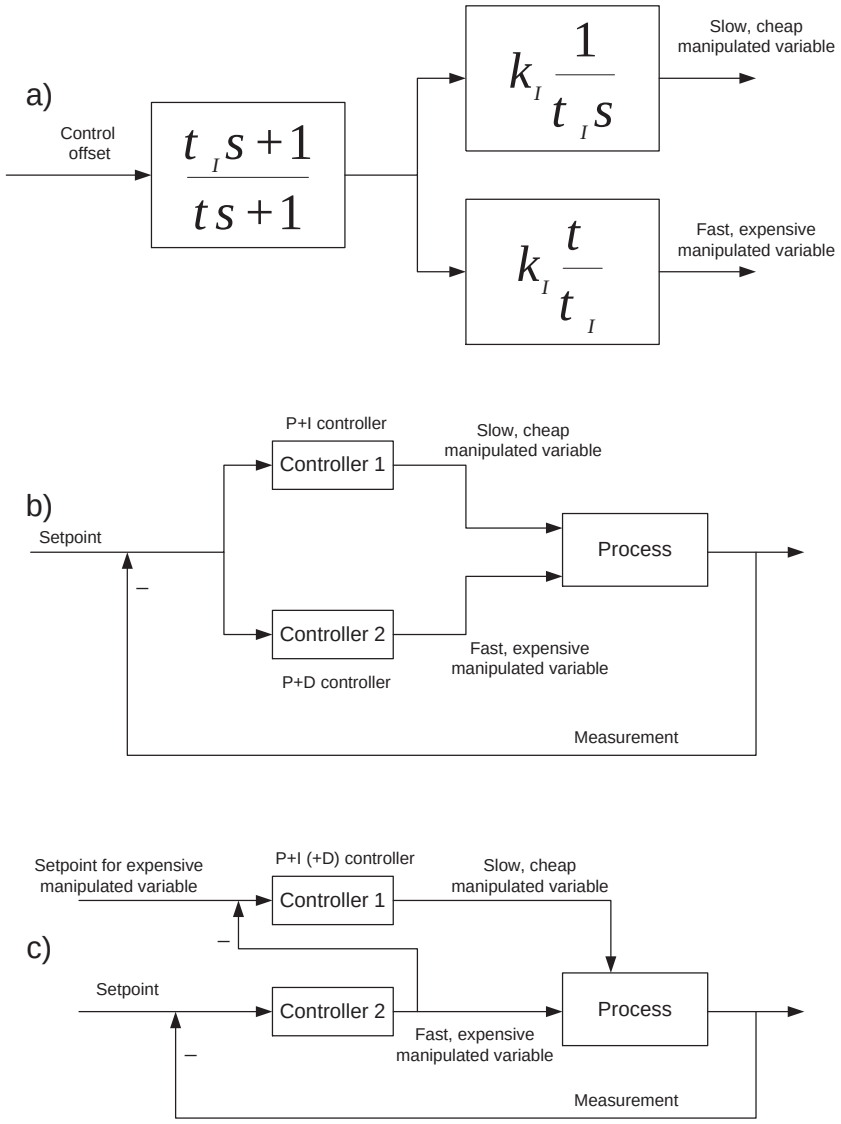

Figure 4.6: Three different ways of implementing parallel control.

manipulated variables start to drift, until the controller output saturates. In Fig.4.6c) there are two independent feedback paths, and both controllers may therefore contain integral action.

4.2.8 Selective control

To this author's knowledge, there is no commonly accepted name for this control structure, yet it is a structure that is seen in many plants. The term "selective control" is coined by the author, who would welcome suggestions for a more illuminating term for this type of control structure. Selective control is sometimes used when

there are more than one candidate controlled variables for a manipulated variable. For each of the candidate controlled variable there is then a separate controller, and the value of the manipulated variable that is implemented is selected among the controller outputs. A simple example of selective control with pressure control on both sides of a valve is shown in Fig. 4.7. Normally one selects simply the highest or lowest value. A few points should be made about this control structure:

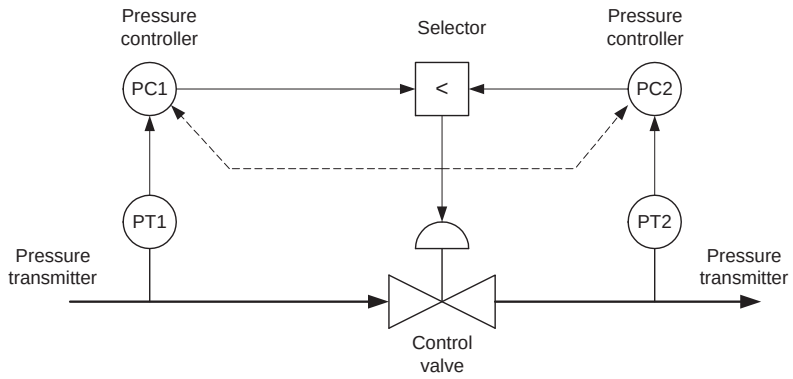

Figure 4.7: Selective control of pressure of both sided of a control valve. Note that the applied control signal is fed back to the controllers.

- Clearly, a single manipulated variable can control only one controlled variable at the time, i.e., the only variable that is controlled at any instant is the variable for which the corresponding controller output is implemented. It might appear strange to point out such a triviality, but this author has been in discussions with several otherwise sensible engineers who have difficulty comprehending this. Thus, one should consider with some care how such a control structure will work.

- The selection of the active controller is usually based on the controller outputs, not the controller inputs. Nevertheless the local operators and engineers often believe that the selection is based on the controller inputs, or that "the control switches when the a measurement passes its setpoint". In principle, the selection of the active controller may also be based on the controller inputs[2]. Some type of scaling will then often be necessary, in order to compare different types of physical quantities (e.g., comparing flowrates and pressures).

- If the controllers contain integral action, a severe problem that is similar to "reset windup" can occur unless special precautions are taken. The controllers that are *not* selected, should be reset (for normal PID controller this is done by adjusting the value of the controller integral) such that for the present controller measurement, the presently selected manipulated variable value is obtained.

[2]Provided appropriate scaling of variables is used, the auctioneering control structure may be a better alternative to using selective control with the selection based on controller inputs.

Commonly used terms for this type of functionality are "putting the inactive controllers in tracking mode" or "using a feedback relay". This functionality should be implemented with some care, this author has seen faulty implementations which permanently lock the inactive controllers. On a digital control system, the controllers should do the following for each sample interval:

1. Read in the process measurement.
2. Calculate new controller output.
3. The selector now selects the controller output to be implemented on the manipulated variable.
4. The controllers read in the implemented manipulated variable value.
5. If the implemented manipulated variable value is different from the controller output, the internal variables in the controller (typically the integral value) should be adjusted to obtain the currently implemented manipulated variable value as controller output, for the current process measurement.

- Some thought should be spent on the function that selects the controller output. If a simple high or low select is used, there is a possibility that measurement noise may temporarily drive the manipulated variable the wrong way. The problem arises due to digital implementation of the controllers, and the need for the tracking function that is explained above. It is more likely to happen if derivative action is used, or the proportional action "dominates" the integral action in a PI controller. To overcome this problem, some logic may be added to the selector function, such that a controller output can only be selected if the corresponding measurement is on the right side of the setpoint. To illustrate, consider a controller for which there is positive (steady state) gain from manipulated variable to measurement. Then, if the selector is of a low select type, this controller should only be selected if the process measurement is above the setpoint.

4.2.9 Combining basic single-loop control structures

Most of the simple control structures shown above may be combined with each other. With the exception of the feedforward control, all the control structures shown are variants of feedback control. Feedforward control is normally combined with some form of feedback control, but it may be somewhat complicated to combine feedforward with auctioneering or selective control.

Note that selective control *should not* be used for one of the manipulated variables of a split range controller. This is because the tracking function will then

constrain the output of the split range controller to be in the range where this manipulated variable is manipulated, and the other manipulated variables in the split range arrangement will not be used (or they may be used over a minor fraction of their operating range). On the other hand, there is nothing conceptually wrong with the *output of the selector* in selective control acting on a set of manipulated variables which operate in a split range arrangement.

4.2.10 Decoupling

The use of decouplers have long been a popular way of converting multivariable control problems into (what appears to be) a number of monovariable control problems. This popularity of decoupling seem to continue, despite the more general and easily applicable multivariable control design methods that have been developed over the last several decades.

The basic idea behind the use of a decoupler can be illustrated in Fig. 4.8. A

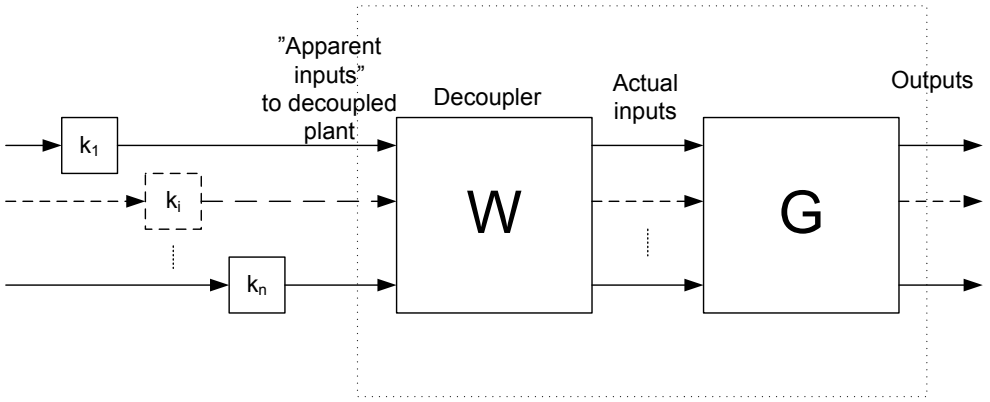

Figure 4.8: The basic idea behind decoupling: A precompensator (W) is used to make the 'decoupled plant' inside the dotted box diagonal, allowing for simple design of monovariable controllers k_i.

precompensator $W(s)$ is used, in order to make the precompensated plant GW diagonal, thus allowing for simple monovariable control design of the individual loop controllers k_i. Assume that the desired precompensated plant is given by $G_{des}(s)$. It is then simple to find the corresponding precompensator, by solving the equation

$$G(s)W(s) = G_{des}(s). \tag{4.1}$$

Note that

- Typically $G_{des}(s)$ is diagonal (which will be assumed henceforth), but occasionally 'one way decouplers' are used, corresponding to $G_{des}(s)$ being upper or lower triangular.
- $G_{des}(s)$ must contain all RHP poles and (multivariable) RHP zeros of $G(s)$ - otherwise the system will be internally unstable.

- The precompensator obviously cannot remove time delays.

- A popular choice is $G_{des}(s) = g_{des}(s) \cdot I$, with $g_{des}(s)$ scalar. Any multivariable RHP zeros in $G(s)$ must then also be present in $g_{des}(s)$. This means that all loops for the precompensated system will be affected by the RHP zero, even if only a few inputs or outputs in $G(s)$ are affected by the multivariable zero.

Decouplers are prone to robustness problems, especially for highly interactive and ill-conditioned plants - which is exactly the type of plant for which one would like to use decoupling. This is discussed in more detail in [93]. The robustness problems can be exasperated by input saturation. Anti-windup for decoupling controllers will therefore be addressed in a subsequent section.

4.3 Control configuration elements and decentralized controller tuning

In this section we will consider how to determine the control configuration, i.e., the structure of the interconnections between the controlled and manipulated variables via the control system. Most of this section will address the pairing of manipulated and controlled variables to arrive at a decentralized control structure (i.e., multiple single control loops) for regulatory control, but the analysis tool (RGA) that is introduced in this section can also be used to assess the appliccability of decoupling control.

4.3.1 The relative gain array

The relative gain array (RGA) was first introduced by Bristol[14] in 1966, and has since proved to be a useful tool both for control structure design and analysis of proposed control structures, as well as an interaction measure and an indicator of robustness problems. Although extensions of the RGA to non-square systems have been proposed, we will here focus on the use of the RGA for square plants, i.e., plant for which the number of controlled variables equal the number of manipulated variables. Consider a $n \times n$ plant $G(s)$

$$y(s) = G(s)u(s) \tag{4.2}$$

The open loop gain from $u_j(s)$ to $y_i(s)$ is $g_{ij}(s)$. Writing Eq.(4.2) as

$$u(s) = G^{-1}(s)y(s) \tag{4.3}$$

it can be seen that the gain from $u_j(s)$ to $y_i(s)$ is $1/[G^{-1}(s)]_{ji}$ when all other y's are perfectly controlled. The relative gain matrix consists of the ratios of these open and closed loop gains. Thus, a matrix of relative gains can be computed from the formula

$$\Lambda(s) = G(s) \times (G^{-1}(s))^T \tag{4.4}$$

Here the symbol '$\times$' denotes the element-by-element product (Schur or Hadamard product).

Note that although the assumption that "all other y's are perfectly controlled" can only be fulfilled at steady state, provided integral action is used in the control of all the "other y's", the RGA can be computed also at non-zero values of s (as implied by Eq.(4.4)) except at the zeros of $G(s)$. Although the interpretation of the RGA as a ratio of open to closed loop gains gradually becomes less accurate as frequency increases, and fails totally in the bandwidth region and beyond, the RGA has repeatedly proven to be a useful analysis tool also at higher frequencies. We will therefore consider the RGA as a function of frequency ($s = j\omega$). The RGA as defined above has some interesting algebraic properties (see e.g. [28]):

- It is scaling independent (independent of the units of measure used for u and y). Mathematically, $\Lambda(D_1GD_2) = \Lambda(G)$ where D_1 and D_2 are diagonal matrices.
- All row and column sums equal one.
- Any permutation of rows or columns of G result in the same permutation in $\Lambda(G)$.
- If G is triangular (and hence also if it is diagonal) $\Lambda(G) = I$.
- Relative permutations in elements of G and its inverse are related by $d[G^{-1}]_{ji}/[G^{-1}]_{ji} = -\lambda_{ij}dg_{ij}/g_{ij}$.

These properties can be proven from the following expression for the individual elements of the RGA:

$$\lambda_{ij}(s) = (-1)^{i+j}\frac{g_{ij}(s)\det(G^{ij}(s))}{\det(G(s))} \tag{4.5}$$

Here $G^{ij}(s)$ denotes the matrix $G(s)$ with row i and column j removed.

4.3.2 The RGA as a general analysis tool

In this section we will consider the RGA as a general analysis tool.

The RGA and zeros in the right half plane. It has been shown [43] that if the RGA has different sign at steady state and at infinite frequency, then this is an indication of RHP zeros in either G,g_{ij} or G^{ij}. However, in order to evaluate the RGA as a function of frequency, one will generally need a state space representation of G. It would then make sense to calculate the zeros (and poles) from the state space representation rather than looking at the RGA.

The RGA and the optimally scaled condition number. Bristol [14] pointed out the formal resemblance between the RGA and the condition number $\gamma(G) = \overline{\sigma}(G)/\underline{\sigma}(G) = \overline{\sigma}(G)\overline{\sigma}(G^{-1})$. However, the condition number depends on scaling, whereas the RGA does not. Minimizing the condition number with respect to all input and output scalings yields the optimally scaled condition number,

$$\gamma^*(G) = \min_{D_1,D_2}\gamma(D_1GD_2) \tag{4.6}$$

The optimal scaling matrices can be obtained by solving a structured singular value problem[13]. However, formulating and solving this problem is quite complicated, and there is no readily available software with a simple function call for solving Eq.(4.6). Anyway, the main purpose for obtaining $\gamma^*(G)$ would be to get an indication of possible robustness problems - a large value of $\gamma^*(G)$ would indicate

that the control performance would be sensitive to small errors in the plant model G. However, Nett and Manousiouthakis [73] have proven that large elements in the RGA matrix imply a large value of $\gamma^*(G)$:

$$\|\Lambda_m(G)\| - \frac{1}{\gamma^*(G)} \leq \gamma^*(G) \tag{4.7}$$

where $\|\Lambda_m\| = 2\max\left\{\|\Lambda(G)\|_{i1}, \|\Lambda(G)\|_{i\infty}\right\}$ (i.e., twice the larger of $\max_i \sum_j |\lambda_{ij}(G)|$ and $\max_j \sum_i |\lambda_{ij}(G)|$)[3]. There is also a conjectured (but not rigorously proven) upper bound on $\gamma^*(G)$ based on the RGA [90], and it is therefore good reason to believe that $\gamma^*(G)$ cannot be large without some elements of the RGA matrix also being large.

The RGA and individual element uncertainty. It can be shown (e.g. [43]) that a matrix G becomes singular if the ij'th element is perturbed from g_{ij} to $g_{Pij} = (1 - \frac{1}{\lambda_{ij}})g_{ij}$. Some implications of this result are:

1. *Element uncertainty.* If the relative uncertainty in an element of a transfer function matrix at any frequency is larger than $|1/\lambda_{ij}(j\omega)|$, then the plant may have zeros on the imaginary axis or in the RHP at this frequency. However, independent, element-by-element uncertainty is often a poor uncertainty description from a physical point of view, since the elements of the transfer function matrix are usually coupled in some way.

2. *Model identification.* Models of multivariable plants $G(s)$ are often obtained by identifying one element at the time, i.e., by step or impulse responses. If there are large RGA element, such model identification is likely to give meaqningless results (e.g., wrong sign of $\det(G(0))$ or non-existing RHP zeros) if there are large RGA elements within the bandwidth where the model is intended to be used. Truly multivariable identification techniques may alleviate this problem, but physical knowledge about the process should always be used to validate and correct identified models.

3. *Uncertainty in the state matrix.* Consider a plant described by a linear state space model. If the state autotransition matrix A has large RGA elements, only small relative changes in the elements of A can make the plant unstable[4].

[3]The row and column sums of the RGA matrix only equal 1 if the actual (complex) values of the elements are added, not their absoulte values.

[4]The result above only tells the necessary relative change in an element to make an eigenvalue equal to zero. Even smaller perturbations in the elements may make a complex conjugate pair of eigenvalues move from the LHP to the RHP by crossing the imaginary axis (away from the origin).

RGA and diagonal input uncertainty. One type of uncertainty that is alway present is input uncertainty. This can be described by assuming that the true (perturbed) plant G_P is related to the nomminal (assumed) plant G by

$$G_P = G(I + \Delta), \qquad \Delta = \text{diag}\,(\Delta_i)$$

where the Δ_i's represent the relative uncertainty in each of the manipulated variables. If an "inverse-based" controller (decoupler) is used, $C(s) = G^{-1}(s)K(s)$, where $K(s)$ is a diagonal matrix, then the true open-loop gain G_PC is

$$G_PC = (I + G\Delta G^{-1})K$$

The diagonal elements of $G\Delta G^{-1}$ are directly given by the RGA [90]:

$$(G\Delta G^{-1})_{ii} = \sum_{j=1}^{n} \lambda_{ij}(G)\Delta_j$$

Since we cannot know the values of the Δ_i's during control system design, it is risky to use an inverse-based controller for plants with large RGA elements. On the other hand, a diagonal controller (consisting of SISO control loops) will be relatively insensitive to the diagonal uncertainty, but will not be able to counteract the strong interactions in the process (as indicated by the large RGA elements).

The RGA as an interaction measure. Since the elements of the RGA can be interpreted as the ratio of the open loop process gain to the gain when all other outputs are perfectly controlled, it should be intuitively clear that the RGA can serve as an interaction measure (i.e., a measure of to what extent the control of output i will be affected by the control of other outputs). If an element of the RGA differs significantly from one, the use of the corresponding input-output pair for control will imply that the control of that output will be affected by the control actions in the other loops to a significant degree. It is of most interest to consider the interactions in the (expected) bandwidth region for control. Provided the steady state RGA is positive, it is of less interest.

However, it should be noted that the RGA is only a measure of *two-way* interaction. If there is only one-way interaction (e.g., if the transfer function matrix is triangular), the relative gain matrix will be $\Lambda = I$. This is both a strength and a weakness of the RGA. It is a strength when it comes to relating the RGA to stability of the closed loop system, an issue which is addressed in the next section.

4.3.3 The RGA and stability

The RGA is a measure of *two-way* interaction, and thus also a measure of *potentially destabilizing* interaction. If $\Lambda(j\omega) = I, \quad \forall\omega$, stability of the individual loops will also imply stability of the overall system, since the interactions then cannot introduce

any additional feedback paths in the closed loop system. This is because $\Lambda = I$ can only occur if the transfer function matrix is triangular - or can be made triangular by simple row and column interchanges while keeping the same elements on the diagonal.

Let us next consider loop 1 in a plant control system (the generalisation to loop k is trivial). Introduce $G' = \text{diag}\{g_{11}, G^{11}\}$, where G^{11} is obtained from G by removing row 1 and column 1. Let G' have n'_U RHP poles (note that n'_U can be different for different loops), and let the controller transfer function matrix be K. Assume:

- The transfer function GK is strictly proper.
- The controller K is diagonal, has integral action in all channels, and is otherwise stable.
- The plant transfer function matrix G have n_U RHP poles.

Then a necessary condition for simultaneously obtaining

a) Stability of the closed loop system

b) Stability of loop 1 by itself

c) Stability of the system with loop 1 removed (e.g., loop 1 in manual)

is that

$$\text{sign}\{\lambda_{11}(0)\} = \text{sign}\{(-1)^{-n_U+n'_U}\} \tag{4.8}$$

For proof, see [43]. Note that for stable systems this implies the widely used criterion of pairing on positive RGA's, see e.g. [28].

The RGA and pairing of controlled and manipulated variables

The steady state RGA is a widely used criterion for pairing controlled and manipulated variables. Equation (4.8) provides a generalisation of the traditional pairing criterion based on the sign of the steady state RGA. The magnitude of the steady state RGA is also widely used as a pairing criterion, but the magnitude of the RGA in the bandwidth region for control is a more reliable pairing criterion. Ideally, we would like that in the bandwidth region $\Lambda = I$. Thus, it makes sense to select a pairing which minimizes $\|\Lambda - I\|$ in the bandwidth region. Often, this corresponds to selecting a pairing corresponding to RGA elements of magnitude close to 1 in the bandwidth region. However, for systems with more than two inputs and outputs, there may be some special cases where minimizing $\|\Lambda - I\|$ gives another pairing than selecting RGA elements of magnitude close to 1. In such cases, the minimization of $\|\Lambda - I\|$ appears to be the more reliable pairing criterion.

4.3.4 Summary of RGA-based input-output pairing

4.3.5 Alternative interaction measures

There have been proposed a number of alternative interaction measures. One family of such interaction measures is based on the width of the socalled *Gershgorin bands*. However, this type of interaction measure can be exceedingly conservative, and will not be discussed further. Measures more closely related to the RGA are the socalled *Rijnsdorp interaction measure* [80, 11]and the *Niederlinski Index* [74]. Both these measures are equivalent to the RGA for 2×2 systems. The Rijnsdorp interaction measure is in a form which makes it more straight forward to analyse tradeoffs between control performance in individual loops [10], but cannot strictly be generalised to larger-dimensional systems (without making the interaction measure dependent on the controller). The Niederlinski index applies also to systems of large dimension, but it is this authors (subjective) opinion that it is of less use than the RGA.

4.3.6 Input-output pairing for stabilization

4.4 Tuning of decentralized controllers

4.4.1 Introduction

In this section, we consider the case when the control configuration is fixed, and focus on fully decentralized control. That is, it is assumed that the overall controller consists of multiple single-input, single-output controllers, and the pairing of manipulated and controlled variables has been determined. Despite the prevalence of decentralized controllers in industry, the tuning (determination of controller parameters) of decentralized controllers is not a solved problem in mathematical terms. The well established controller synthesis methodologies, like H_2- or $H_\infty-$optimal control, cannot handle a pre-specified structure for the controller. In fact, a truly H_2- or $H_\infty-$optimal decentralized controller would have an infinite number of states[82]. This follows, since these controller synthesis procedures result in controllers which have the same number of states as the 'plant'. When synthesizing one decentralized controller element, all the other decentralized controllers would become a part of the 'plant' as seen from the controller to be synthesized, and this controller element would therefore have a large number of states. Now, with this new controller in operation, it becomes a part of the 'plant' as seen from the other controllers, and the other controllers may therefore be improved - thereby introducing yet more states. Sourlas et al. have looked at l_1-optimal [5] decentralized control [95, 94], and have developed a method for calculating the best achievable decentralized performance, both for decentralized control in general and for fixed order decentralized controllers. However, the computations involved are rather complex, and may well become hard to solve even for problems of moderate dimension. In the absence of any decentralized controller

[5] In l_1-optimal control, the ratio $\|y(t)\|_\infty / \|d(t)\|_\infty$ is minimized.

synthesis method that has both solid theoretical foundation and is easily applicable, a few practical approaches have been developed:

- Independent design. The individual decentralized controller elements are designed independently, but bouds on the controller designs are sought which ensure that the overall system will behave acceptably.
- Sequential design. The controller elements are designed sequentially, and the controllers that have been designed are assumed to be in operation when the next controller element is designed.
- Simultaneous design. Optimization is used to simultaneously optimize the controller parameters in all decentralized controller elements. A particular controller parametrization (e.g. PI-controllers) have to be chose *a priori.*

In the following, these three tuning approaches will be described in some detail, but first some methods for tuning conventional single-loop controllers will be reviewed.

4.4.2 Loop shaping basics

4.4.3 Tuning of single-loop controllers

There are a number of methods for tuning single-loop controllers, and no attempt will be made here at providing a comprehensive review of such tuning methods. Instead, a few methods will be described, which all are based on simple experiments or simple models, and do not require any frequency-domain analysis (although such analysis may enhance understanding of the resulting closed loop behaviour).

Ziegler-Nichols closed-loop tuning method

This tuning method can be found in many introductory textbooks, and is probably the most well-known tuning method. It is based on a simple closed loop experiment, using proportional control only. The proportional gain is increased until a sustained oscillation of the output occurs (which neither grows nor decays significantly with time). The proportional gain giving the sustained oscillation, K_u, and the oscillation period (time), T_u, are recorded. The proposed tuning parameters can then be found in Table 1. In most cases, increasing the proportional gain will provide a sufficient disturbance to initiate the oscillation - measurement noise may also do the trick. Only if the output is very close to the setpoint will it be necessary to introduce a setpoint change after increasing the gain, in order to initiate an oscillation. Note that for controllers giving positive output signals, i.e., controllers giving output signals scaled in the range $0-1$ or $0\%-100\%$, a constant bias must be included in the controller in addition to the proportional term, thus allowing a negative proportional term to have an effect. Otherwise, the negative part of the oscillation in the plant input will

be cut off, which would also effect the oscillation of the output - both the input and output of the plant may still oscillate, but would show a more complex behaviour than the single-frequency sinusoids that the experiment should produce.

Table 1. Tuning parameters for the closed loop Ziegler-Nichols method

Controller type	Gain, K_P	Integral time, T_I	Derivative time, T_D
P	$0.5 \cdot K_u$		
PI	$0.45 \cdot K_u$	$0.85 \cdot T_u$	
PID	$0.6 \cdot K_u$	$0.5 \cdot T_u$	$0.12 \cdot T_u$

Essentially, the tuning method works by identifying the frequency for which there is a phase lag of 180°. In order for the tuning method to work, the system to be controlled must therefore have a phase lag of 180° in a reasonable frequency range, and with a gain that is large enough such that the proportional controller is able to achieve a loop gain of 1 (0 dB). These assumptions are fulfilled for many systems. The tuning method can also lead to ambiguous results for systems with a phase lag of 180° at more than one frequency. This would apply for instance to a system with one slow, unstable time constant, and some faster, but stable time constants. Such a system would have a phase lag of 180° both at steady state and at some higher frequency. It would then be essential to find the higher of these two frequencies. Furthermore, the system would be unstable for low proportional gains, which could definitely lead to practical problems in the experiment, since it is common to start the experiment with a low gain. Despite its popularity, the Ziegler-Nicols closed loop tuning rule is often (particularly in the rather conservative chemical processing industries) considered to give somewhat aggressive controllers.

Tuning based on the process reaction curve

In process control, the term 'reaction curve' is sometimes used as a synonym for a step response curve. Many chemical processes are stable and well damped, and for such systems the step response curve can be approximated by a first-order-plus-deadtime model, i.e.,

$$y(s) = \frac{Ke^{-\theta s}}{1+Ts}u(s) \tag{4.9}$$

and it is relatively straight forward to fit the model parameters to the observed step response. This is illustrated in Figure 4.9. Assuming that the response in Fig. 4.9 is the result of a step of size B at time 0 in the manipulated variable, the model parameters are found as follows:

1. Locate the *inflection point*, i.e., the point where the curve stops curving upwards and starts to curve downwards.

2. Draw a straight line through the inflection point, with the same gradient as the gradient of the reaction curve at that point.

3. The point where this line crosses the initial value of the output (in Fig.4.9 this is assumed to be zero) gives the apparent time delay θ.

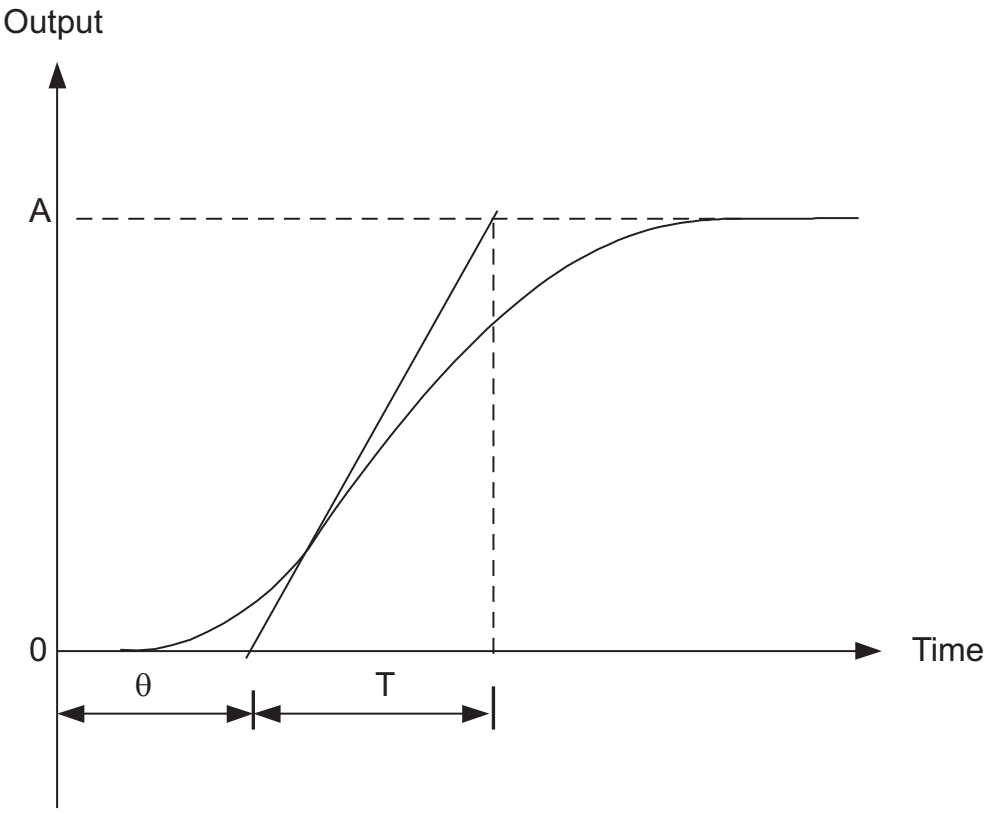

Figure 4.9: Estimating model parameters from the process reaction curve.

4. The straight line reaches the steady state value of the output at time $T + \theta$.

5. The gain K is given by A/B.

Ziegler-Nichols open loop tuning Ziegler and Nichols [105] propose the tuning rules in Table 2 based on the model in Eq. (4.9).

Table 2. Tuning parameters for the open loop Ziegler-Nichols method

Controller type	Gain, K_P	Integral time, T_I	Derivative time, T_D
P	$\frac{T}{K\theta}$		
PI	$\frac{0.9T}{K\theta}$	$\frac{\theta}{0.3}$	
PID	$\frac{4T}{3K\theta}$	$\frac{\theta}{0.5}$	0.5θ

Cohen-Coon tuning Cohen and Coon [16] have modified the Ziegler-Nichols open loop tuning rules. The modifications are quite insignificant when the deadtime θ small relative to the time constant T, but can be important for large θ. The Cohen-Coon tuning parameters are given in Table 3.

Table 3. Tuning parameters for Cohen-Coon method

Controller type	Gain, K_P	Integral time, T_I	Derivative time, T_D
P	$\frac{T}{K\theta}(1+\frac{\theta}{3T})$		
PI	$\frac{T}{K\theta}(0.9+\frac{\theta}{12T})$	$\theta(\frac{30+3\theta/T}{9+20\theta/T})$	
PID	$\frac{T}{K\theta}(\frac{4}{3}+\frac{\theta}{4T})$	$\theta(\frac{32+6\theta/T}{13+8\theta/T})$	$\theta\frac{4}{11+2\theta/T}$

IMC-PID tuning

In internal model control (IMC), the controller essentially includes a process model operating in "parallel" with the process, as illustrated in Figure 4.10. The IMC

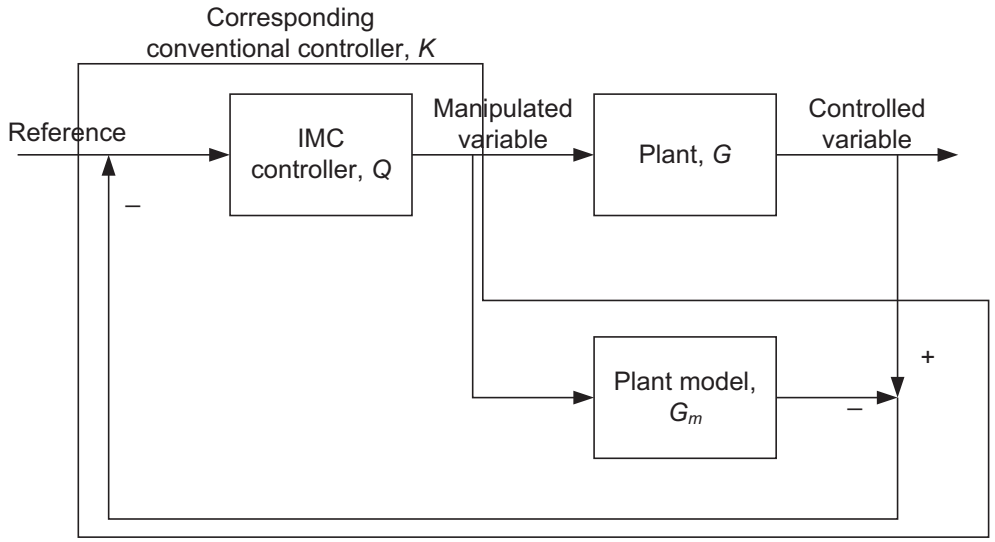

Figure 4.10: An internal model controller.

controller Q and the corresponding conventional feedback controller K are related by

$$K = Q(1 - G_m Q)^{-1} \tag{4.10}$$

Note that if the model is perfect, $G_m = G$, IMC control essentially results in an open loop control system. This means that it is not straight forward to use it for unstable systems, but for stable systems (and a perfect model) *any* stable IMC controller Q results in a stable closed loop system - this holds also for non-linear systems. In the following discussion on IMC controllers we will therefore assume the open loop system to be stable. Another advantage with IMC control is that the transfer function from reference r to controlled variable y is simply given by $y = Tr = GQr$. Designing the closed loop transfer function T (or $S = 1-T$) therefore becomes simple. Conventional IMC controller design consists of factoring the plant G into a minimum phase part G^m and a non-minimum phase part G^n, with G^n chosen such that $G^n(0) = 1$. For example, the plant

$$G = \frac{10(s-1)}{(10s+1)(30s+1)}$$

may be factorized to

$$G^m = \frac{-10(s+1)}{(10s+1)(30s+1)}; \qquad G^n = -\frac{(s-1)}{(s+1)}$$

The IMC controller Q is then chosen as $Q = (G^m)^{-1}F$, where F is a low pass filter which is used both to make Q proper[6], and to make the closed loop system robust. Normally, the filter F is chosen to be on the form

$$F = \frac{1}{(\lambda s+1)^n}$$

Clearly, the order n of the filter must be sufficiently large to make Q proper, but usually a low order is sufficient (i.e., n is in the range 1 to 3). This leaves only one free parameter, λ, which makes it feasible to tune the system on-line. A large λ make the system slow, decreasing it increases the speed of response. It is common to use simple, low order transfer function models of the system when designing feedback controllers. Rivera et al. [81] have shown that IMC controllers designed based on low-order transfer function models of the plant in most cases result in overall controllers K having the familiar PID structure, possibly with an additional lag. This additional lag would correspond to the time constant that is commonly applied to the derivative action in many PID controllers. In their paper, Rivera et al. list numerous such low-order plant transfer functions, the corresponding PID parameters, including the dependence of the PID parameters on the low pass filter time constant λ.

Autotuning

Many industrial PID controllers include some self-tuning or autotuning function, allowing the controller to find controller tuning parameters "by itself". In order to find tuning parameters, some sort of automated identification experiment is necessary. Although many different types of experiments and identification procedures in principle are possible, most autotuners use relay feedback, i.e., the ordinary controller is replaced by a relay, as shown in Fig. 4.11. Whith the use of relay feedback, most systems which are stable or integrating in open loop will enter a stable limit cycle, with a (dominant) oscillation frequency of $\omega_u = 2\pi/T_u$. Similarly, K_u can be found from $K_u = 4d/\pi a$, where d is the relay amplitude and a is the amplitude of oscillation of the output. The tuning of the controller can then be based on the Ziegler-Nichols closed-loop tuning, or modifications thereof. The relay based autotuning in its simplest form thus works by identifying the frequency at which the process has a phase lag of 180°, and the corresponding gain. Other points on the Nyquist curve may be identified by connecting a linear system in series with the relay. A more comprehensive treatment of relay-based autotuning can be found in articles by Åström and Hägglund [6, 8], or by Schei [83].

[6] A proper transfer function model has a denominator polynomial of order at least as high as the order of the numerator polynomial. A system has to be proper in order to be physically realizable.

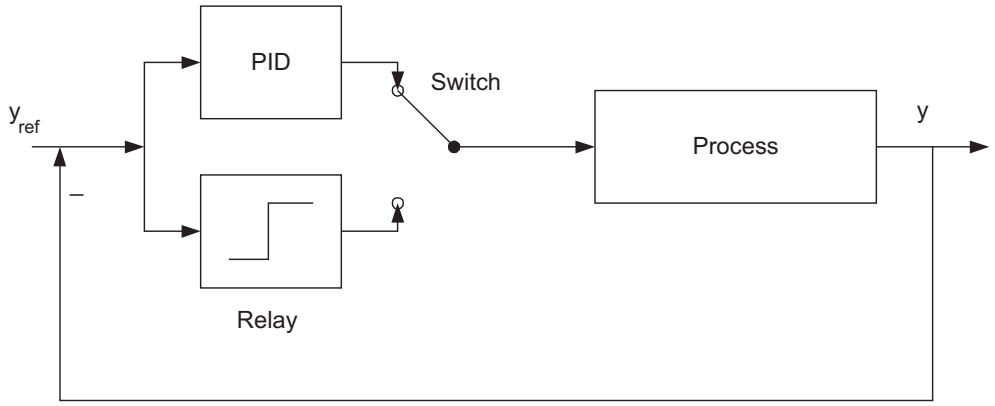

Figure 4.11: Block diagram showing controller with relay-based autotuner.

What type of controller should be used?

Åström et al. [9] studied PID control for a number of different monovariable systems that were either asymptotically stable or integrating. The dynamical behaviour of the systems studied were categorized in terms of four dimensionless numbers defined in the paper, and depending on the value of these dimensionless numbers, recommendations were made with respect to whether P, I, PI, PD or PID control should be used, or when more complex controllers should be used (e.g., deadtime compensation or feedforward). In addition, conclusions are drawn with respect to when the Ziegler Nichols closed loop tuning method can be expected to give good results.

4.4.4 Multiloop controller tuning

The term 'multiloop controller' is often used for decentralized controllers. Below, we will briefly discuss the three different tuning approaches listed in the Introduction. In addition to fulfilling the overall performance criteria (whatever they may be), a desireable property of multiloop controllers is that they exhibit *integrity*, i.e. that they remain stable when one or more of the loops are taken out of service. Ideally, they should also show a modest and predictable performance degradation when loops are taken out of service. One must clearly accept poor performance in the loops that are out of service, but preferably control quality will not be much affected in the loops that remain on-line. Whether such predictable performance degradation is achieved, may depend on both the system itself, the control structure chosen, and the tuning parameters.

Independent design

One may group independent design techniques into two categories:

- Naive independent design, where the individual loops are design without particular regard for the fact that hey have to operate in a multivariable control system. If it turns out that the overall system is unacceptable, there is hopefully some method for improving an initial design.

- Rigorous independent design. In this group of approaches, explicit bounds are derived for the behaviour of each individual loop. If these bounds are fulfilled when each individual loop is designed, it is guaranteed that the overall system will fulfill the performance criteria.

Naive independent design The most well known of tuning methods in this category, is the so-called 'BLT tuning'. It essentially consists of tuning each loop individually (typically with the Ziegler-Nichols closed loop tuning), and then to check the infinity norm of the multivariable complementary sensitivity function (the transfer function from reference to controlled variable), $T = GK(I + GK)^{-1}$. If the 'peak value' of this transfer function is too large, a common detuning factor is applied to the proportional gain for all loops. Typically, this peak value should be less than 2, possibly in the range 1.2-1.5. The term 'peak value' here refers to the infinity norm, i.e., the maximum value of the largest singular value over all frequencies.

$$\|T\|_\infty = \max_\omega \overline{\sigma}(T(j\omega))$$

Some problems with this tuning procedure are:

- Applying a common detuning factor to all loops is often not desireable, and the result may be that the loops are detuned more than necessary. This is typically the case when several loops have similar bandwidths, and there is unacceptable interaction in the bandwidth region. In such cases, it is often sufficient to detune only one of the interacting loops.

- Detuning can produce stability problems for loops that have a phase lag close to or more than 180° at low frequencies, which will occur for instance for unstable systems or for integrating processes controlled by an integrating controller.

- The tuning procedure does not address issues related to integrity or tuning modifications made by operators.

The main advantage of this tuning procedure is its simplicity. Despite its shortcomings, it is frequently used in the process control literature as a comparison against which other tuning procedures are compared. It should not be a surprise that most authors are able to find examples for which their proposed tuning procedure outperforms the BLT tuning.

Rigorous independent design Rigorous independent design was introduced by Skogestad and Morari[91]. They approach the tuning problem from a robust control viewpoint, using the structured singular value (μ) framework. Instead of finding the controller directly using DK-iteration (which anyway would not have resulted in a decentralized controller) they effectively include the controller in the uncertainty description for the robust stability (or robust performance) problem. This is done in two different ways, by

1. expressing the controller as an LFT of the sensitivity functions for the individual loops, and
2. expressing the controller as an LFT of the complementary sensitivity functionss for the individual loops.

The two different LFTs thus result in two different robust stability problems, in which the controller is replaced by a diagonal, complex-valued 'uncertainty'. Iteration is used (i.e., a skewed-μ problem) to find the largest magnitude for these 'uncertainties' for which the robust stability/performance can be guaranteed. Robust stability/performance will then be guaranteed provided all individual loops at all frequencies fulfill the magnitude bound on either the sensitivity function or the complementary sensitivity function. Some advantages of this approach include

- It can handle robustness issues rigorously.
- It places no unnecessary constraint on the controller type, only on the sensitivity and complementary sensitivity functions. Thus the design freedom for the individual loops is not compromised.
- Explicit bounds are derived for the individual loops, which could be used to indicate how much plant operators (or engineers) should be allowed to modify the controller tuning parameters.

Disadvantages include

- The theoretical and numerical complexity inherent in using the structured singular value framework, which makes the method inaccessible to many practising engineers.
- It provides the same bounds for all loops, and thus cannot take advantage of using different bandwidths in different loops. Differences between individual loops may be entered explicitly into the structured singular value problem, but the method itself provides no indication on what differences between the loops to use.
- It is inherently conservative, since it can only specify a magnitude bound for the uncertainties corresponding to the closed loop transfer functions. This is related to the fact that the method does not specify any particular controller type.

- It does not cover integrity issues explicitly, they have to be explored after designing the controllers.

In order to minimize the inherent conservatism in the rigorous independent design procedure of [91], Hovd and Skogestad [47] introduced independent design for Internal Model Controllers. In this work, bounds are found for the IMC filter time constant and its inverse. Thus, the uncertainty associated with the controller tuning can be assumed to be real-valued, leading to less conservative bounds. Clearly, this comes at the cost of numerically even more complex calculation, and the *à priori* determination of controller parametrization.

Sequential design

This is probably the most common design approach in industry for designing decentralized controllers. The controllers are designed and put into operation one at the time, and the controllers that have been designed are kept in operation when new controllers are designed. Thus, 'sequential design' does not necessarily imply any specific method for designing the individual controller elements, but merely that they are designed in a *sequence*. It therefore also allows controller design methods that does not require any explicitly formulated system model. Methods based on experimentation/feedback alone, like Ziegler-Nichols or autotuning, are also accommodated. More complex controller synthesis methods that do require a system model are also possible. Sequential design provides a limited extent of system integrity. Normally, one design requirement for the controller in each loop would be that the system should be stable after closing that loop. The system will therefore remain stable if loops are taken out of service in the reverse of the order in which they were designed. It is not uncommon that the controllers in some loops have to be re-designed when new controllers are put into service, due to unacceptable interactions between different control loops. In such cases, the limited integrity guarantee of sequential design no longer holds. The very term 'sequential design' begs the question 'In what sequence should the individual controllers be designed?' The conventional rule of thumb is to *close the fast loops first.* This is intuitively reasonable, as it is often the case that the faster loops are comparatively unaffected by the tuning in slower loops. However, in some cases there may be strong one-way interactions causing even slow loops to significantly disturb faster loops. This conventional rule also requires the engineer to have a good idea of what speed of control can be expected in the individual loops. Note that *closing the fast loops first* normally implies that the inner loops in a cascade shouild be designed first, which clearly makes sense.

Simultaneous design

Simultaneous design implies that the tuning parameters for all loops are determined simulataneously. Since formal controller synthesis methods will not lead to decentralized controllers, simultaneous design is done by choosing a particular controller

parametrization (e.g., PID control), and using optimization to find the controller parameters which optimizes some measure of system performance. Although such simultaneous design often works reasonably well, the optimization problems are typically non-convex, and convergence to a global optimum can not be guaranteed. If the optimization fails to find acceptable controller parameters, it need not be obvious whether this is because no acceptable parameters exist (for the perticular choice of controller parametrization), or whether it is simply due to an unfortunate initial guess of parameter values. Simultaneous design typically provides no integrity guarantee. Integrity may be enforced by introducing additional constraints in the formulation of the optimization problem. However, such constraints are typically non-linear, and the required number of additional constraints will grow quickly with system size.

4.4.5 Tools for multivariable loop-shaping

Shaping the Bode diagram of a monovariable transfer function is a well-known tool for designing monovariable controllers. Performance requirements usually dictate a high loop gain at low frequencies, whereas measurement noise and possibly robustness issues means that the loop gain has to be small at high frequencies. The closed loop transfer function from disturbance to control offset for a single-input single-output system is given by

$$e = r - y = \frac{-g_d}{1 + gk} d$$

and thus at frequencies well within the loop bandwidth (assuming $|gk| >> 1$), we get $y \approx (gk)^{-1} g_d d$. Therefore, a requirement like 'the effect of disturbances should be reduced by a factor of 10 at frequency w_d' implies that the magnitude of the loop gain,$|gk|$, should be 10 times the magnitude of the disturbance transfer function, $|g_d|$, at frequency w_d. Similarly, the transfer function from reference signal to control offset is given by

$$e = r - y = \frac{1}{1 + gk} r$$

which at frequencies where $|gk| >> 1$ can be approximated by $y \approx (gk)^{-1} r$.Thus, if the control offset should be no more than 10% of the reference signal at frequency w_e, then the loop gain $|gk|$ should be at least 10 at frequency w_e. In the following, loop gain requirements for individual loops in multivariable systems will be presented. In the same way as for the SISO case above, these loop gain requirements are reasonable accurate at low frequencies (well below the bandwidths of the individual loops), but the underlying approximation breaks down in the bandwidth region.

The Performance Relative Gain Array

The relative gain array, RGA, is a useful measure of two-way (i.e., potentially destabilizing) interactions, but severe one-way interactions can exist even if the RGA matrix $\Lambda = I$. The Performance Relative Gain Array, PRGA, is able to capture both

one-way and two-way interactions. To arrive at the PRGA, we introduce the matrix $\widetilde{G}(s)$, which is a diagonal matrix consisting of the elements on the diagonal of $G(s)$, i.e., the elements corresponding to the individual control loops. Then, the matrix of sensitivity functions for the individual loops is given by $\widetilde{S} = (I + \widetilde{G}K)^{-1}$, which is a diagonal matrix (since K is diagonal). Note that the diagonal elements of $\widetilde{S}$ are *not* the same as the diagonal elements of the sensitivity function $S = (I + GK)^{-1}$. The relationship between S and $\widetilde{S}$ by

$$S = (I + \widetilde{S}(\Gamma - I))^{-1}\widetilde{S}\Gamma$$

where $\Gamma = \widetilde{G}G^{-1}$ is the Performance Relative Gain Array (PRGA) matrix. At frequencies where the loop gains of the individual loops is large, $\widetilde{S}$ is small, and hence $S \approx \widetilde{S}\Gamma$. Thus, the effect of reference changes on control offset is given by

$$e = r - y = SRr \approx \widetilde{S}\Gamma Rr$$

where R is just a diagonal scaling matrix which is chosen such that the (scaled) reference changes $|r_j| \leq 1 \forall j$. Thus, the effect of a change in reference j on control offset i is given by

$$e_i = [SR]_{ij} r_j \approx [\widetilde{S}\Gamma R]_{ij} r_j = \widetilde{s}_i \gamma_{ij} R_j r_j \approx \frac{\gamma_{ij}}{g_{ii}k_i} R_j r_j$$

where $\widetilde{s}_i = 1/(1 + g_{ii}k_i)$ is the sensitivity function for loop i, γ_{ij} is element ij of Γ, and R_j is element j on the diagonal of R. The second approximation in the above equation holds provided $|g_{ii}k_i| >> 1$, and thus holds whenever the first approximation holds. Consequently, if the effect of reference change j on control offset i should be less than α at frequency ω_α, we require $|g_{ii}k_i| > \alpha |\gamma_{ij}R_j|$ at frequency ω_α. The PRGA (and our performance requirements) thus provide us with estimated loop gain requirements for achieving acceptable performance.

The Closed Loop Disturbance Gain

The Closed Loop Disturbance Gain (CLDG) is similar to the PRGA, with the difference that it looks at the effect of disturbances on control offset. The closed loop transfer function from disturbances to control offset is given by

$$e = -SG_d d \approx -\widetilde{S}\Gamma G_d d$$

where, as before, the approximation holds where the loop gains are large. The matrix $\Delta = \Gamma G_d$ is the Closed Loop Disturbance Gain. We get

$$e_i = -\left[SG_d\right]_{ij} d_j \approx \frac{\delta_{ij}}{g_{ii}k_i} d_j$$

where δ_{ij} is element ij of Δ.

The CLDG's thus provide estimates of loop gain requirements for disturbance rejection in much the same way as the PRGA's do for reference changes.

Example

A simplified model of a distillation column may be given by

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{1}{75s+1}\begin{bmatrix} 87.8 & -86.4 \\ 108.2 & -109.6 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \frac{1}{75s+1}\begin{bmatrix} 7.88 & 8.81 \\ 11.72 & 11.19 \end{bmatrix}\begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$

where y_1 is the composition of the top product, y_2 is the composition of the bottom product, u_1 is the reflux flowrate, u_2 is the boilup rate, d_1 is a feed flowrate disturbance, and d_2 is a feed composition disturbance.

Assuming that the variables are reasonably scaled, it is clear from looking at the disturbance model that control will be necessary, since the disturbances can cause composition offsets larger than 1 in magnitude. It would appear that both disturbances are approximately equally severe, and that output 2 is somewhat more affected by the disturbances than output 1. However, this only holds for open loop operation. The CLDG's and PRGA's for this example are shown in Figs. 4.12 and 4.13. The figures also show the loop gains resulting from using the PI controller $u_i(s) = \frac{75s+1}{75s}(y_i(s) - r_i(s))$ in both loops (for loop 2, a negative controller gain must be used, since the process gain is negative). The vertical distance between the loop gain and the CLDG's is an estimate of the degree of disturbance attenuation (inside the loop bandwidth). The figures indicate that the simple PI controllers are able to provide acceptable response to disturbances, but that disturbance 1 is much more difficult to reject than disturbance 2.

The predictions based on the CLDG's are shown to hold up reasonably well in Figs. 4.14 and 4.15. Disturbance 2 causes control offsets that are insignificant, whereas disturbance 1 causes larger - although still acceptable - control offsets.

Unachievable loop gain requirements

The PRGA and CLDG presented above provide us with approximate loop gain requirements for acceptable control (provided the variables are properly scaled). It may happen that it is impossible to fulfill these loop gain requirements, if there are significant bandwidth limitations in the system. One then has to choose between three alternatives

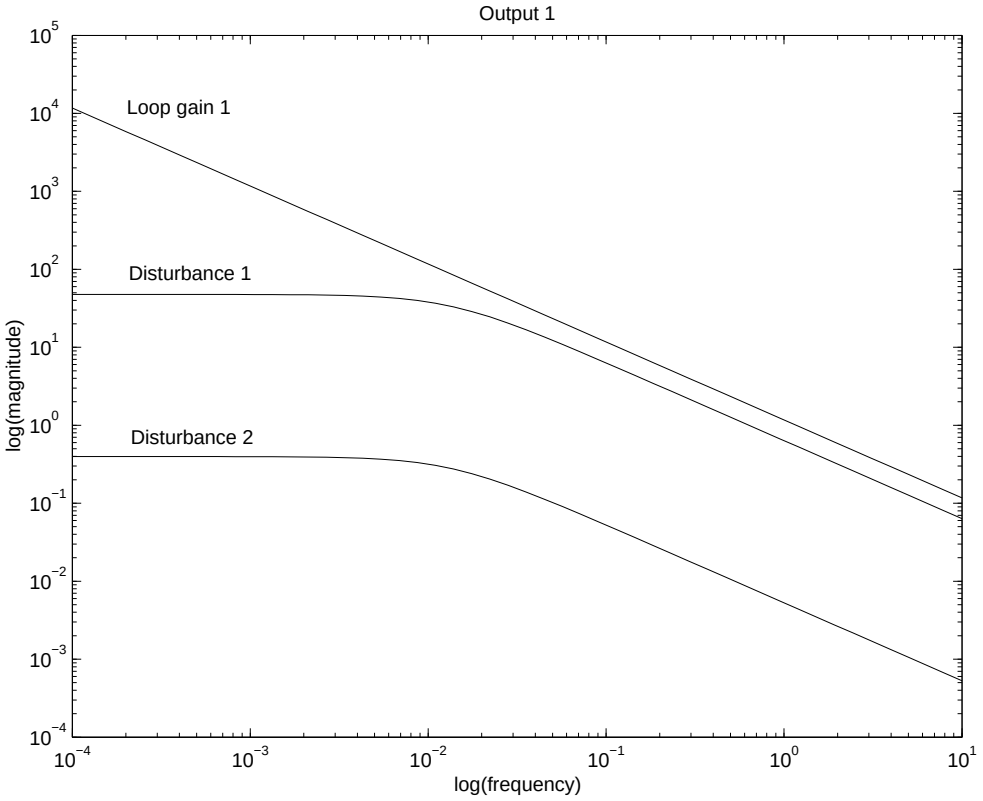

Figure 4.12: CLDG's and loop gain for loop 1.

1. Use more advanced controllers. This may help, at the cost of using more complex design and implementation. However, one should realize that even the most advanced controller cannot remove fundamental bandwidth limitations, like e.g. multivariable RHP transmission zeros.

2. Modify your performance requirements. The PRGA and CLDG, when analyzed together with relevant performance limitations, can indicate how much the performance requirements will need to be relaxed. The PRGA can indicate to what extent setpoint changes have to be filtered - which typically results in slower setpoint following, but also less interactions between loops.

3. Modify your system. The system may be modified to make control easier. Such modifications may include faster actuators, new and improved measurements (e.g., with less deadtime), or installing buffer tanks to filter disturbances. The CLDG can be used to estimate the required size for such buffer tanks.

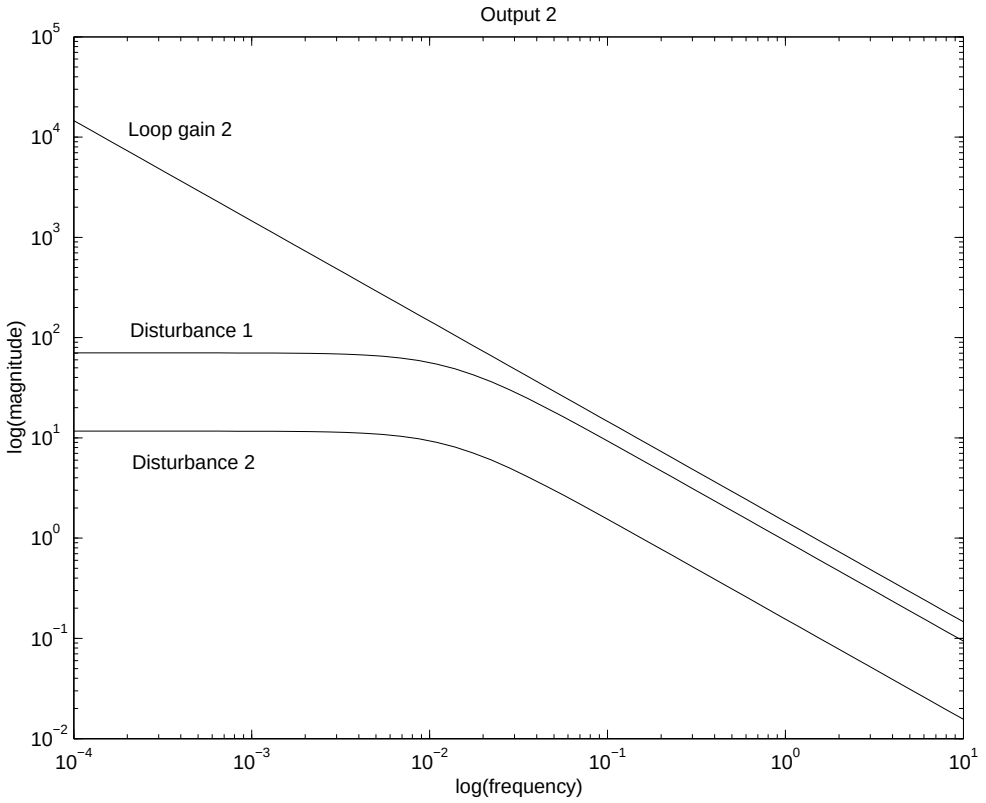

Figure 4.13: CLDG's and loop gain for loop 2.

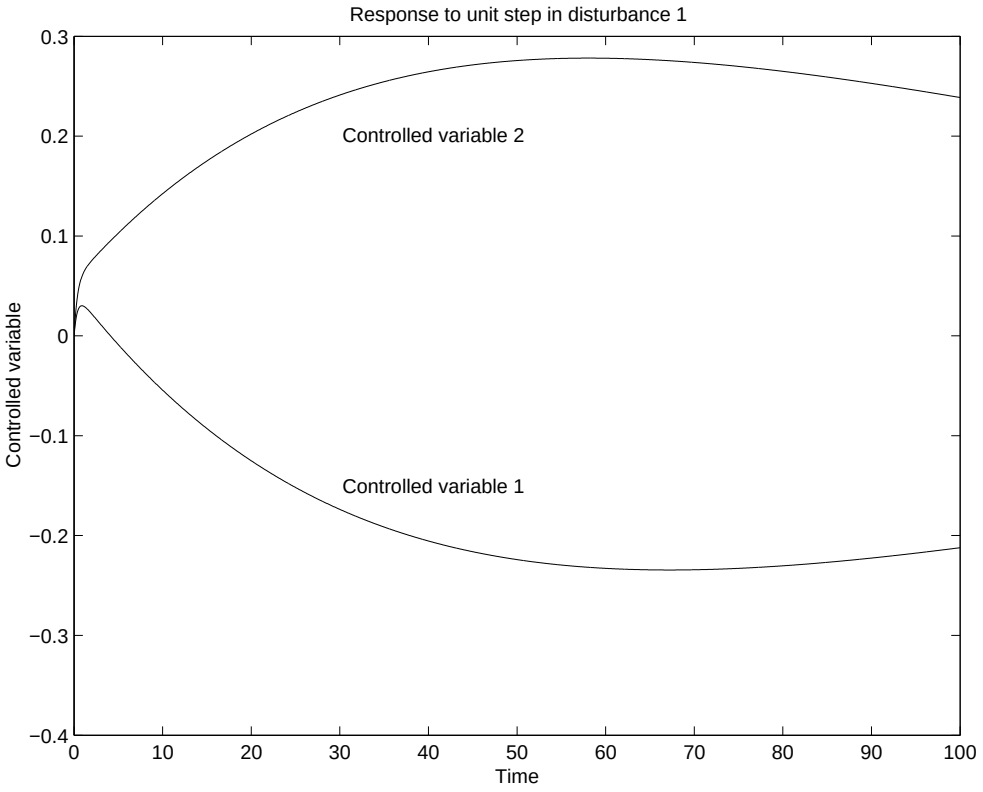

Figure 4.14: Response to a step in disturbance 1 of unit magnitude.

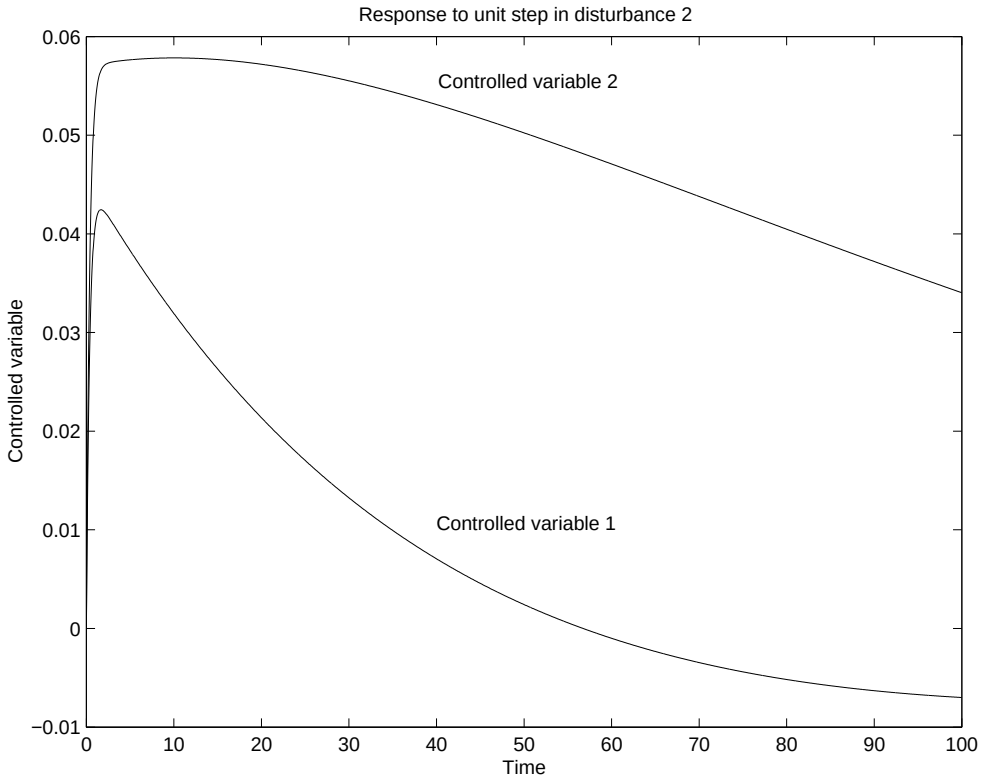

Figure 4.15: Response to a step in disturbance 2 of unit magnitude.

Chapter 5

Control structure selection and plantwide control

5.1 Introduction

The term *control structure design* refers to the structural decisions involved in control system design:

1. Selection of *controlled variables* c ('controlled outputs', with setpoints c_s).

2. Selection of *manipulated variables* u ('control inputs').

3. Selection of measurements y.

4. Selection of *control configuration* (the structure of the interconnections between the variables c_s, u, and y).

5. Selection of the *controller type* (PID, decoupler, MPC, ...)

The term *plantwide control* is only used in the process control community. Although the term is generally well understood within that community, it has lacked a clear, generally accepted definition. We will here (attempt to) adhere to the definition of Larsson and Skogestad [65]: *plantwide control* are *the structural and strategic decisions* involved in the control system design of a complete chemical plant.

The distinction between plantwide control and control structure design is thus somewhat vague. Larsson and Skogestad state that control structure design is the systematic (mathematical) approach to solving the plantwide control problem.

Like the other areas addressed by this note, the area of plantwide control is very large, worthy of a book on its own. This chapter is therefore by necessity incomplete. Larsson and Skogestad [65] provide a nice review of the area up to the year 2000, with a large number of references to relevant previous work. Other key sources for this chapter include [3], [4], [88], [31], [2] and [5].

5.2 General approach and problem decomposition

Considering the multi-layered control hierarchy described in section 1.3, one quickly realizes that when designing plantwide control structures one is faced with a 'hen-and-egg' type of paradox.

The *system*, as seen from the top layers, is not well defined until the lower layers of the control hierarchy have been designed. On the other hand, the *objectives* of the lower layers are not clearly defined until the higher layers of the control hierarchy have been designed.

It is clearly necessary to break this deadlock. Often, this is done by starting with a 'bottom-up' design, where the lower layers are designed first, with experience and process insight substituting for a clear objective formulation for the lower layers.

Although experience and process insight will give useful guidance, this bottom-up approach can easily result in design decisions with unfortunate consequences for the capabilities of the overall system. Larsson and Skogestad instead propose an initial top-down analysis, followed by a subsequent bottom-up design.

5.2.1 Top-down analysis

The top-down analysis seeks to clarify two issues:

1. What constitutes optimal operation, and what variables should be controlled in order to achieve (close to) optimal operation?

2. Where should the throughput (production rate) be set?

Defining and exploring optimal operation

In most cases, the objective of the overall plant is to achieve economically optimal operation, subject to environmental and safety constraints, and accommodating relevant disturbances (whether caused by market conditions or physical conditions).

It is assumed that this optimal operation is quantified in terms of a cost function J which should be minimized[1], and that the relevant constraints can be expressed mathematically as equality or inequality constraints.

It is further assumed that a plant model is available. Although detailed dynamical models often are not available, steady state models typically are. For most continuous chemical production plants, economics is dominated by steady state operation, and restricting the analysis to steady state is therefore usually acceptable.

The equality constraints should include the plant model, since the plant model must be fulfilled at any steady state operating point in order to ensure feasible operation. The inequality constraints will typically include operational constraints on variables such as temperature and pressure, product quality constraints, purity constraints on effluents, etc.

[1]Maximizing profit P may be formulated as minimizing the cost $J = -P$.

At this initial stage, major disturbances should also be identified. The number of steady-state degrees of freedom should also be identified. This determines how many variables can be specified (at steady state) in order to optimize operation.

The goal of this analysis is to determine how many and which variables should be selected as controlled variables, in order to achieve close to optimal operation. This is further discussed in sections 5.4 and 5.5 below.

The focus here is on specifying the controlled variables for the Supervisory control layer of the control hierarchy, see Fig. 1.1. The objectives of the Regulatory control layer are often linked to economics only in an indirect way, and at this layer there are typically many more variables that are controlled.

Determining where to set the throughput

The position of the throughput manipulator will greatly affect the structure of the remaining inventory control system. This issue is addressed further in section 5.8.

5.2.2 Bottom-up design

The bottom-up design starts with the lower layer of the control hierarchy, the regulatory control layer, and then works its way up the layers of the control hierarchy.

Whereas the top-down analysis attempts to keep the overall picture in focus to determine the throughput manipulator and controlled variables for optimal economic operation of the entire plant, further decomposition and a more local focus will frequently be necessary in the bottom-up design, especially at the lower layers of the control hierarchy.

In section 1.3 a hierarchical ('vertical') decomposition of the control system is presented. This decomposition is based on the observation that each layer has a different purpose - and that there is a corresponding timescale on which the individual layers operate.

One may also decompose the design problem 'horizontally', i.e., divide the design task at each layer into a set of smaller subtasks. Ideally the design of each such subtask will depend only weakly on each other.

The process structure or layout is often utilized to perform such 'horizontal' decomposition. The decomposition may be based on individual process units or small sets of closely connected units. However, one should be aware that process units that seem far apart may actually affect each other through plant recycles or utility systems (such as heating or cooling medium systems).

This horizontal decomposition is used more extensively at the lower layers of the control hierarchy. It is simply not practical (and hardly possible with foreseeable computing power) to account rigorously for the interactions between hundreds or thousands of control loops at the regulatory control layer. The purpose of the higher layers is to coordinate and optimize wider sections of the lower layers, and hence the extent of horizontal decomposition will decrease for the higher layers.

The design of the regulatory control layer is addressed next, in section 5.3. Sections 5.4 - 5.7 will address issues of more relevance to the higher layers of the control

hierarchy, in particular the supervisory and RTO layers. The chapter closes with a closer loop at inventory control in section 5.8.

5.3 Regulatory control

The top-down analysis should define the throughput manipulator as well as a (typically rather low) number of controlled variables used for keeping the plant close to optimal operation.

The number of variables that are controlled at the regulatory control layer will, however, be substantially higher. The purpose of the regulatory control layer may be said to be twofold:

1. To enable the operators to keep the plant in operation without the higher layers of the control hierarchy. The regulatory control layer uses simple algorithms and very reliable hardware, and will therefore be relatively reliable.

2. To make the design task at the higher layers simpler, by reducing the effects of uncertainty and nonlinearity.

The tasks of the regulatory control system may alternatively be described as

- *Stabilization.* In addition to stabilizing variables that are unstable in a strict system theoretic sense, this task will also include 'stabilizing' any variable that drifts over a wide operating range or otherwise shows unacceptably large variation.

- *Local rejection of disturbances.* Local control loops are used to reject disturbances before they can affect wider sections of the plant.

- *Linearization by feedback.* Feedback (when successful) typically has the effect of reducing the effect of nonlinearity within the loop. This is utilized in many circumstances, e.g., valve positioners to achieve the desired valve position, flow controllers to counteract valve nonlinearities, temperature controllers on heat exchangers, etc.

- *Reduction of uncertainty.* There will always be uncertainties and imperfections in out knowledge of the plant. Within the bandwidth of the feedback loop, feedback can reduce the effect of such uncertainty by moving its effect from an important controlled variable to a less important manipulated variable.

The tasks of the regulatory control layer are typically achieved using single loop controllers (PI/PID-controllers), with the occasional use of cascaded loops or feedforward. The other loop configurations of section 4.2 are used in more special cases.

Understanding of the tasks of the regulatory control layer, when combined with knowledge of how the plant works and is operated, will be of great help when selecting controlled and manipulated variables for regulatory control. The RGA and pole vectors introduced in section 4.3 will be of further help.

It should be noted that closing loops in the regulatory control layer (or in any other layer), although it 'uses up' manipulated variables, does not reduce the number of degrees of freedom available to the higher layers of the control system. Although the manipulated variable in a loop will be unavailable to the higher layers, the setpoint of the loop will be introduced as a new variable that may be used by the higher layers.

Example: Regulatory control of liquid level in a deaeration tower

Design of a regulatory control layer will here be illustrated on the example of a deaerator tower used in petroleum production. The aim is to illustrate how understanding of the tasks of the regulatory control layer and plant operation can be used in designing the regulatory control layer.

Plant description. It is common to inject water into petroleum reservoirs in order to maintain reservoir pressure and enhance oil production. Oxygen needs to be removed from the water before injection, as oxygen in the reservoir can result in bacterial growth and the production of acids that will corrode the production equipment.

The plant and a rudimentary control system is shown in Fig. 5.1. Vacuum is applied to the water in the deaerator tower, to liberate dissolved oxygen from the water. In the top of the deaerator tower, there is a packing which both increases the surface area and the retention time of the water, thereby improving oxygen removal. The deaerated water is collected in the 'sump' at the bottom of the tower.

High pressure is needed to inject the water in the reservoir. However, due to the low pressure in the deaerator tower, a specially designed booster pump is required to raise pressure up to an acceptable suction pressure for the main water injection pump.

The pumps run on constant speed, and require a minimum flowrate. There is therefore a minimum flow recycle control, which will open a recycle valve routing water from the pump outlet back to the deaerator tower sump in case of low flow.

The water level in the deaerator needs to be controlled. In case of low level, the suction pressure to the booster pump may become too low, causing cavitation, which may lead to excessive vibration and abrasion. Too high level can mean that the water covers part of the packing, reducing the deaeration efficiency and making the tower very heavy. A rudimentary regulatory control system is shown in Fig. 5.1. This rudimentary control system achieves stabilization of the liquid level, by controlling the liquid level by manipulating directly the feed water valve. In addition, there is the minimum flow recycle control mentioned above.

Plant dynamics and disturbances. Without control, the liquid level will have an almost purely integrating dynamics. This is easily stabilized by feedback, as indicated in Fig. 5.1. However, due to the residence time in the packing, there is a significant time delay from the feed water valve to the liquid level, limiting the achievable bandwidth for level control. At the same time there are significant disturbances both up- and downstream:

- On the upstream side, changing flowrates at other water consumers leads to disturbances in the inlet pressure, and hence disturbances in the feed water flowrate.

- On the downstream side, production engineers can change the openings of the injection water chokes, leading to large and fast disturbances to the outlet flowrate[2].

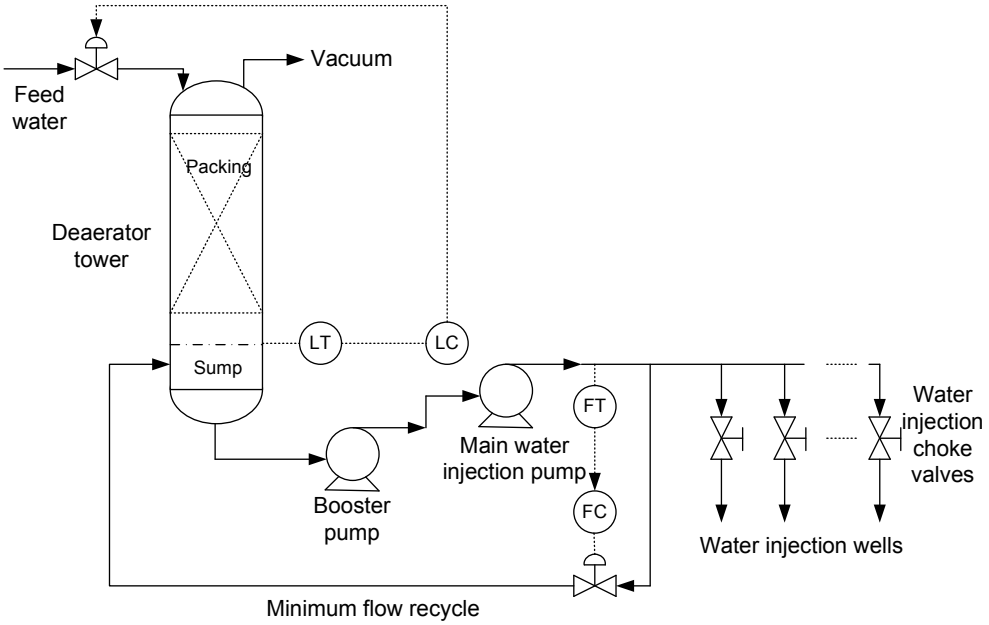

Figure 5.1: Deaerator tower with rudimentary regulatory control.

Improvements to the regulatory control. There is a conflict between the level control loop bandwidth required for disturbance rejection, and the bandwidth limitation resulting from the deadtime. The regulatory control system's ability to handle disturbances should therefore be improved. Two such improvements are relatively easy to achieve:

1. The disturbances in the upstream pressure may be rejected locally by using flow control on the feed water valve. Furthermore, this flow control loop will counteract any nonlinearity or uncertainty in the valve characteristic. This flow control loop is the inner loop in a cascade with the level control loop.

2. Feedforward from the outlet flowrate may be used to quickly counteract disturbances in the outlet flowrate, without being limited by the bandwidth of the level control loop. The feedforward signal is added to the output of the level controller, and changes the setpoint for the feed flowrate controller.

[2]In order to mantain reservoir pressure, it would be sufficient to adjust the water injection rate very slowly. From the control point of view, the obvious solution would be to reduce the flowrate disturbance by slow flowrate control on the water injection chokes. For what appears to be mainly psychological reasons, this appears to be unacceptable to production engineers, who insist on setting the injection choke opening directly. The production engineer 'makes the real money', and therefore decides on how the plant is operated.

With these improvements in handling disturbances, significantly lower bandwidth can be used in the level control loop, thus removing the conflict between the required bandwidth for disturbance rejection and the bandwidth limitation from the deadtime. The modified regulatory control structure is shown in Fig. 5.2.

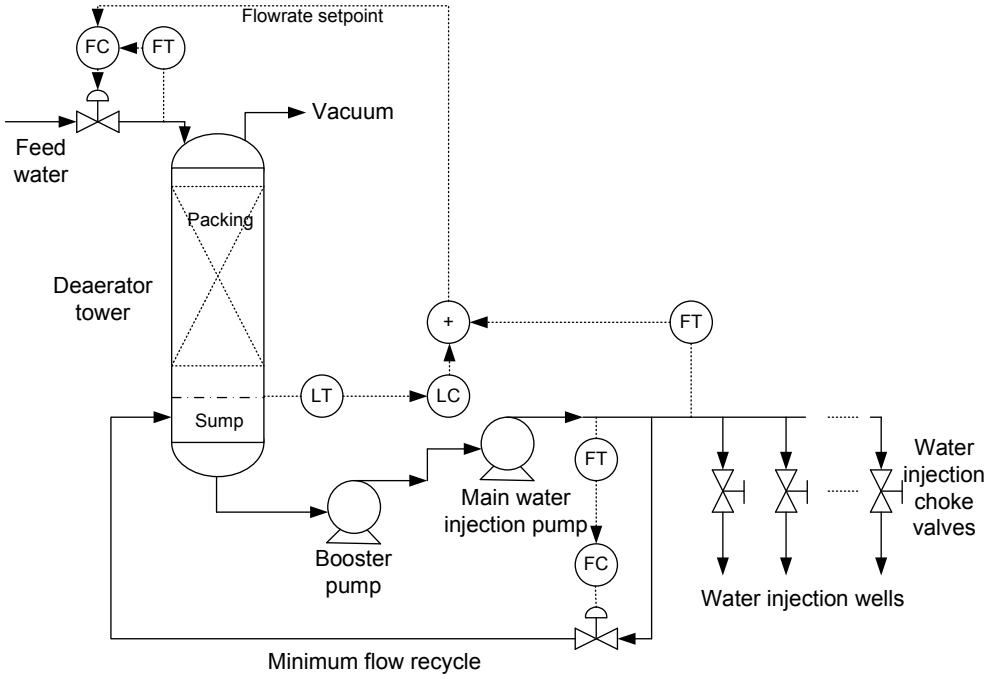

Figure 5.2: Deaerator tower with improved regulatory control.

Concluding remarks on the example. This example illustrates how understanding of the tasks of the regulatory control layer, combined with plant understanding, can help in designing the control structure for the regulatory control layer. A few additional comments may be in order:

1. The improvements in the regulatory control require two new flowrate sensors and a new controller. In general there is a cost issue as well as a maintenance issue with increasing instrumentation. In this case, avoiding a single shutdown due to improved control should more than justify the costs involved.

2. A mass balance on the deaerator sump yields

$$\frac{\rho A dh}{dt} = \rho F_{in}(t-T) - \rho F_{out}(t) \tag{5.1}$$

where ρ is the water density, A is the tower cross-sectional area, h is the liquid level, F_{in} is the flowrate through the feed water valve, T is the time delay, and F_{out} is the outlet flowrate. Thus, the time derivative of the level depends on the outlet flowrate. A commonly held misconception is therefore that feedforward from the outlet flowrate is equivalent to derivative action in the level

controller. However, (5.1) shows that the derivative of the level depends on *two* components, the outlet flowrate and the time-delayed inlet flowrate. Even with derivative action, the level controller is therefore limited in bandwidth by the time delay - and derivative action is seldom recommended for loops with significant time delay. No such bandwidth limitation arises for feedforward control.

3. Some readers may find it puzzling that the feedforward signal actually is transmitted *against the direction of flow*, i.e., 'the feedforward signal is transmitted backwards'. Drawing the control structure using ordinary control block diagrams (rather than a process flow diagram) may clarify this matter.

4. The improved control structure is simple to understand and to tune, and is a good alternative to more advanced controllers for this problem. Little would here be gained from using e.g. deadtime compensation or MPC.

Newer process designs for oxygen removal from injection water has replaced the deaerator tower in modern offshore platforms. This is due to the lower space requirements for the newer designs, rather than control problems.

5.4 Determining degrees of freedom

In order to obtain a well-defined operating point, all degrees of freedom have to be fixed. A simple and straight forward way to determine the degrees of freedom is to simply count the number of variables that may be freely set in the plant: the valve positions, pump and compressor speeds, heat inputs, etc. Let the resulting number of degrees of freedom be $\mathcal{N}_F$.

However, some variables (or combinations thereof) will have no steady state effect. These must be removed to find the number of steady state degrees of freedom. I.e., we have

$$\mathcal{N}_F = \mathcal{N}_{Fs} + \mathcal{N}_{Fd} \tag{5.2}$$

where $\mathcal{N}_{Fs}$ is the number of degrees of freedom which have a steady state effect, while $\mathcal{N}_{Fd}$ is the number of degrees of freedom with only dynamic effect. Following [65] we have

$$\mathcal{N}_{Fd} = \mathcal{N}_{m0} + \mathcal{N}_{y0} \tag{5.3}$$

where $\mathcal{N}_{m0}$ is the number of manipulated variables, or combinations thereof, with no steady state effect, and $\mathcal{N}_{y0}$ is the number of manipulated variables used to control variables with no steady state effect.

Typical cases when combinations of manipulated variables have no steady state effect include

- When there are multiple valves in the same pipeline. The steady-state mass flowrate must be the same everywhere along the pipeline.

- If a heat exchange has a bypass on both the hot and the cold side. Clearly, there will nevertheless be only one heat transfer rate, even though one may have two manipulated variables with which one may affect the heat transfer rate.

Identifying such (combinations of) manipulated variables will establish $\mathcal{N}_{m0}$.

Control of variables with no steady state effect is usually associated with control of liquid levels. Most liquid levels will have to be stabilized by feedback[3], and each such level control will 'consume' a manipulated variable. Sometimes a little thought is required to determine which levels do have a steady state effect.

- Most commonly, liquid levels have no steady state effect. This is the case for buffer tanks, etc.
- One example where the liquid level *will* have steady state effect is when the level affects available heat transfer area, such as in flooded condensers.
- In liquid-phase chemical reactors the liquid level will affect the effective reactor volume, and will thus have steady state effect.

This list is not exhaustive, but with proper understanding of the plant it should be clear what liquid levels have steady state effect, and determining $\mathcal{N}_{y0}$ therefore should not be difficult. Thus, $\mathcal{N}_{Fs}$ can be found, and we will know the number of variables which must be determined in order to achieve optimal (steady state) operation.

5.5 Selection of controlled variables

In section 5.3 we discussed the considerations behind selection of controlled and manipulated variables in the regulatory control layer. Consisting mainly of monovariable control loops, the measured variables and controlled variables are more or less the same in the regulatory control layer. In the higher layers of the control hierarchy the focus shifts towards economically (or otherwise) optimal operation, and more complex controllers are more often found. This also opens the possiblity that the controlled variables may differ from the variables that are actually measured.

Having determined the number of steady state degrees of freedom above, we have established the number of variables that need to be set in order to achieve optimal operation.

Basic insight into optimization will reveal that the optimal operating point can be equivalently specified in terms of different sets of variables, as long as the chosen variables can be set independently and the total number of variables specified equals the number of available degrees of freedom. This may lead to the belief that it does not matter what variables we control, provided the correct *number* of variables are controlled. This is a serious misunderstanding.

[3]Detailed modelling may well show that liquid levels are weakly self-regulating, and hence 'stable' in a strict system theoretic sense. This self-regulating effect comes from the effect of the liquid level on the outlet pressure. However, this self-regulating effect is very often to weak for the level to be considered 'stable' in a more practical sense - the level will vary too widely in response to common disturbances. The more common exception is when the level is 'controlled' by overflow over a weir. In such cases the level is typically strongly self-regulating, but on the other hand there is no way of manipulating the outlet flowrate anyway.

Consider again the control structure hierarchy in section 1.3, and the supervisory control layer receiving its specifications from the layer above. Ideally, achieving these specifications would be sufficient to achieve optimal operation. There are three reasons why this ideal situation rarely is achieved:

1. Model errors, or an ill-chosen optimality criterion in the higher layer, may result in errors in the specifications. Model errors will always be present. Engineering insight will hopefully guard against erroneous formulation of the optimization criterion in the higher layer, but inaccuracies such as inaccurate price data may occur.

2. The timescale separation between the layers may mean that the specifications received from the higher layer is *outdated*, based on old values of disturbances, etc.

3. There may be (most likely will be) an *implementation error*, i.e., the lower layer does not perfectly achieve the specifications of set by the higher layer. Even if integral action is used, which in the absence of active constraints should ensure that the specifications (setpoints) are achieved without steady state error, measurement bias will cause implementation error.

Each of these three errors will result in optimal operation not being achieved, and a *loss* is incurred. It turns out that the size of the loss can be highly dependent on what variables are used to specify the operating point[4].

It is therefore important to specify the desired operating point in terms of variables such that the loss will be small despite the three sources of error listed above. This is the main idea behind *self-optimizing control*, which will be presented next following the ideas in [88], which addresses points 2 and 3 above. Point 1 is not directly addressed, but uncertainty/variations in the *parameters* of the plant model or the optimality criterion may be handled in much the same way as the changes in disturbances that are covered by the proposed approach.

5.5.1 Problem formulation

It is assumed that

1. Optimal operation can be addressed using a steady state consideration, neglecting plant dynamics. This is reasonable for most continuous processes, but will not hold for batch processes.

2. The overall objective can be quantified in terms of a scalar objective $J_0(x, u, d)$, equality constraints $g_e(x, u, d) = 0$, and inequality constraints $g_{i0}(u, d) \leq 0$. The

[4]There is no contradiction between this statement and the statement above that the optimal operating point may be equivalently specified in terms of different sets of variables. The *optimal point* is identical for different sets of variables - but the cost of *deviating from the optimal point* can be strongly dependent on what variables are used to specify the optimal operating point.

objective $J_0(x, u, d)$ typically represents production cost or profit. Model equations lead to equality constraints, whereas the inequality constraints typically represents product quality and operational constraints.

3. The reference values r for the controlled variables z are kept constant for significant periods, i.e., between 're-optimization' at the higher layer the references r are independent of the disturbances d.

4. For any disturbance d, there are corresponding optimal values for the states $x = x_{opt}(d)$, manipulated variables $u = u_{opt}(d)$ and controlled variables $z = z_{opt}(d)$.

For a given disturbance $d = d^*$, the task of identifying the optimal operating point can thus be formulated as

$$\begin{aligned} \{x_{opt}(d^*), u_{opt}(d^*)\} &= \arg\min_{x,u} J_0(x, u, d^*) \\ g_e(x, u, d^*) &= 0 \\ g_{i0}(x, u, d^*) &\leq 0 \end{aligned} \tag{5.4}$$

The model equations (in the equality constraints) may be used to eliminate the state variables x from the problem formulation. The resulting expressions for the objective function and inequality constraints may be rather complex, and in implementation we may choose not to perform this elimination. However, here we will assume that the state variables x are eliminated from the formulation - mainly for notational convenience. This gives the following optimization problem formulation, equivalent to (5.4):

$$\begin{aligned} u_{opt}(d^*) &= \arg\min_{u} J(u, d^*) \\ g_i(u, d^*) &\leq 0 \end{aligned} \tag{5.5}$$

where the relationships between J and g_i in (5.5) and J_0 and g_{i0} in (5.4) should be clear from context.

Ideally, we want always to keep $u = u_{opt}(d)$ whenever d changes. However, we will not in practice manage to achieve this, and we get the loss

$$L(u, d) = J(u, d) - J(u_{opt}(d), d) \tag{5.6}$$

Instead of keeping manipulated variables u constant, we may use the manipulated variables to counteract changes in a chosen set of controlled variables z. In this case, the manipulated variables will change when disturbances change. Skogestad [88] defines self-optimizing control as follows:
Self-optimizing control is when we can achieve an acceptable loss with constant set-point values for the controlled variables without the need to reoptimize when disturbances occur.

5.5.2 Selecting controlled variables by direct evaluation of loss

Using direct evaluation of loss, we account rigorously for the nonlinearity in the problem formulation. The procedure is as follows:

1. List all possible sets of controlled variables. Note that we may choose to hold manipulated variables constant, and thus the manipulated variables should be included (in addition to measured variables) among the candidate controlled variables.
2. For each set of controlled variables in the list, evaluate the loss using (5.6).
3. Select the set of controlled variables that gives the smallest loss.

Step 2 above requires further explanation. We are faced with several design choices in this step:

- Whether to minimize the *worst case* loss or the *expected loss*. Minimizing the worst case loss may be the more 'robust' choice, but the worst case may seldom or never occur in practice.
- How to select the disturbances that are used in the loss evaluation. If we minimize the worst case loss, it is natural to include all extreme combinations of disturbance values[5]. Minimizing the average or expected value for the disturbance would imply including more of the disturbance combinations that are more likely to occur.
- Often, the loss evaluation is performed by keeping the reference values r constant at the optimal values for the optimal operating point. However, the optimal references may also be a result of the optimization, i.e., we wish to find the 'robust references' that minimize the loss.

Regardless of design choices, the direct evaluation of loss is often very demanding computationally. Certain design choices will further add to the computational load, in particular the calculation of robust references[6]. We will therefore in the following present controlled variable selection based on local analysis. This can be much less computationally intensive, and will also give insight into the desired characteristics of the controlled variables.

5.5.3 Controlled variable selection based on local analysis

When using local analysis, we explore how the loss depends on the choice of controlled variables in the vicinity of the nominal operating point. In addition to the assumptions made above, we further assume

[5]Although, since the optimization problem is nonlinear, we cannot really be sure that the worst case loss occurs at an extreme combination of disturbance values.

[6]On the other hand, references calculated for the nominal operating point need not allow a feasible solution for all disturbance values.

- The optimization problem is unconstrained. If a variable is at a constraint at the optimum, it is natural to use control to keep the variable at the constraint. The constrained variable is therefore assumed to be 'pre-selected' among the controlled variables, and the controlled variable selection problem can be addressed in the 'reduced space' with the constrained variable eliminated from the problem. Note, however, that the manipulated variable used to control the constrained variable should be included as a potential controlled variable in the remaining analysis.
- The cost function $J(u, d)$ is twice differentiable.
- We select as many controlled variables as the available degrees of freedom, and it is assumed that the selected controlled variables are independent (as seen from the manipulated variables).

We may then perform a Taylor series expansion of the cost function around the operating point (u^*, d^*), where $u^* = u_{opt}(d^*)$. Thus

$$J(u, d^*) = J(u^*, d^*) + J_u^T(u - u^*) + \frac{1}{2}(u - u^*)^T J_{uu}(u - u^*) + \cdots \tag{5.7}$$

where

$$J_u = \left(\frac{\partial J}{\partial u}\right)\Big|_{u^*,d^*}$$

$$J_{uu} = \left(\frac{\partial \partial J}{\partial u^2}\right)\Big|_{u^*,d^*}$$

Note that since we are addressing an unconstrained optimization problem, $J_u = 0$, and the loss related to non-optimal u therefore depends only on J_{uu}. To relate the loss due to non-optimal manipulated variables u to the output selection problem, we assume a linear (steady state) model relating disturbances, manipulated variables and controlled variables:

$$z = Gu + G_d d \tag{5.8}$$

The assumption on the number and independence of the selected controlled variables implies that G is invertible, and for a constant d we thus get

$$(u - u^*) = G^{-1}(z - z^*) \tag{5.9}$$

where $z^* = z_{opt}(d^*)$. Substituting this equation into to Taylor series expansion for the cost, we obtain

$$L = J(u, d^*) - J(u^*, d^*) \approx \frac{1}{2}(z - z^*)^T G^{-T} J_{uu} G^{-1}(z - z^*) \tag{5.10}$$

Clearly, we would like $z = z^*$, but as explained at the top of section 5.5 this will not be achieved in practice. Two important sources of error are addressed here

- *Optimization error* $r - z^*$. We get $r \neq z^*$ because the disturbance is not perfectly known, or because the disturbance has changed since the optimal references were calculated by the higher layer.

- *Implementation error* $z - r$.. The controlled variables do not achieve their reference values. Although integral action removes steady state offset in the controlled variable, measurement bias will cause a difference between the true value and the value read by the control system.

These two sources of error are normally independent of each other.

Equation (5.10) provides a criterion for selecting controlled variables. However, to use this equation we must estimate the expected magnitude of $z - z^*$. Thus, we scale G such that the scaled $\|z - z^*\|_2 \leq 1$. Halvorsen et al. [31] argue for scaling based on the vector 2-norm rather than the vector ∞ norm. This may be somewhat surprising, since assuming $\|z - z^*\|_2 \leq 1$ only allows one vector element at the time reaching its extreme value. The argument in [31] is partly based on mathematical convenience, but it is also supported by the reasonable assertion that multiple elements of the vector $z - z^*$ reaching their extreme values simultaneously is unlikely or rare.

Anyway, to perform the scaling, estimates of both the optimization error and the implementation error for each control variable must be obtained.

- The implementation error estimate should reflect the quality of the measurement of the given variable (or the expected quality of the *estimate* of the controlled variable, should it not be directly measurable). Maintenance quality may also be a consideration, even high quality sensors can become inaccurate if poorly maintained and poorly calibrated.

- The optimization error can be estimated by re-optimizing the cost function for a number of different disturbance values. The changes in disturbances used in this optimization should reflect the expected changes in disturbances *between each time the higher layer re-optimizes reference values*, it should not reflect the extreme range of disturbances over the lifetime of the plant.

For each element of the controlled variable vector, the sum of these two error components should be used in the scaling.

This gives the following procedure for selecting controlled variables:

- Scale all candidate controlled variables as outlined above.

- List all candidate *sets* of controlled variables. The number of controlled variables in each set should equal the number of steady state degrees of freedom, and the individual controlled variables in each set should be independent. Let k be the index identifying the candidate controlled variable set.

- For each candidate controlled variable set, evaluate $s_k = \bar{\sigma}\left(G_k^{-T} J_{uu} G_k^{-1}\right)$ (or, equivalently, $\tilde{s}_k = \bar{\sigma}\left(J_{uu}^{1/2} G_k^{-1}\right)$.

- Select the controlled variable set k corresponding to the smallest s_k.

In practice, one may wish to retain a few candidate sets with small s_k for further investigation using non-linear simulation.

The minimum singular value rule

With the appropriate scaling of the manipulated variables (in addition to the scaling of the controlled variables described above), one may base the controlled variable selection on G_k alone, without involving J_{uu}.

The ideal manipulated variable scaling in this context is such that $J_{uu} = \alpha U$, where U is a unitary matrix[7]. This scaling means that the effect of non-optimal manipulated variables ($u \neq u^*$) only depends on $\|u - u^*\|_2$, but is independent of the direction of $u - u^*$.

Due to the fact that $\bar{\sigma}(G^{-1}) = 1/\underline{\sigma}(G)$, we get [93]

$$\max_{\|z-z^*\|_2 \leq 1} L = \frac{\alpha}{2\underline{\sigma}^2(G)} \tag{5.11}$$

Thus, the controlled variable selection can be based on $\underline{\sigma}(G)$, which should be large. Efficient numerical procedures for selecting controlled variables to maximize $\underline{\sigma}(G)$ is investigated in [57].

Comment. We also prefer large $\underline{\sigma}(G)$ to avoid input saturation in the face of disturbances and reference changes. Note, however, that these two reasons for preferring a large $\underline{\sigma}^2(G)$ are not related, and that different scaling are used in these two settings.

Desirable characteristics of the controlled variables

At this point we are able to summarize some desirable characteristics for the controlled variable sets:

1. There should be a large gain from the manipulated to the controlled variables, it should be easy to control the chosen controlled variables independently. This will ensure that $\underline{\sigma}(G)$ is large.

2. The optimization error $r - z^*$ should be small. That is, the optimal values of the controlled variables should depend only weakly on the disturbances d.

3. The implementation error $z - r$ should be small. In addition to the desired 'ease of control' mentioned in point 1 above, this also implies that it should be possible to determine the value of the controlled variables with good accuracy, i.e., measurement error/bias should be small.

5.5.4 An exact local method for controlled variable selection

The minimum singular value rule for controlled variable selection is based on two critical assumptions:

[7]Note: i) A unitary matrix has all singular values equal to 1, ii) This manipulated variable scaling differs from the scaling used elsewhere in this note, iii) Whereas J_{uu} determines the optimal scaling, it is the effect of the scaling on G that is of interest when using the minimum singular value rule.

- Scaling of the manipulated variables such that $J_u u = \alpha U$, where U is a unitary matrix. Finding the appropriate scaling may be hard or even impossible. However, avoiding this assumption can easily be done by basing the measurement selection on $\bar{\sigma}\left(J_{uu}^{1/2} G_k^{-1}\right)$ (which should be small)instead of $\underline{\sigma}\left(G_k\right)$ (which should be large).

- The assumption that any combination of controlled variable errors such that $\|z - z^*\|_2 \leq 1$ may occur in practice.

The second assumption may not hold in practice. In Halvorsen et al. [31], an alternative local method is proposed. The method is based on a Taylor series expansion in terms of both u and d around the nominal operating point (u', d'), where $u' = u_{opt}(d')$. Thus, here u' and d' are *fixed*, whereas in (5.7) d^* could vary and u^* changed with changes in d^*.

The Taylor series expansion in terms of both u and d gives

$$J(u,d) = J(u',d') + \begin{bmatrix} J'_u \\ J'_d \end{bmatrix}^T \begin{bmatrix} (u-u') \\ (d-d') \end{bmatrix} + \frac{1}{2} \begin{bmatrix} (u-u') \\ (d-d') \end{bmatrix}^T \mathcal{H} \begin{bmatrix} (u-u') \\ (d-d') \end{bmatrix} + \mathcal{O}^3 \tag{5.12}$$

where

$$\begin{aligned}
J'_u &= \left.\frac{\partial J}{\partial u}\right|_{(u',d')} \\
J'_d &= \left.\frac{\partial J}{\partial d}\right|_{(u',d')} \\
\mathcal{H} &= \begin{bmatrix} J'_{uu} & J'_{ud} \\ J'_{du} & J'_{dd} \end{bmatrix} \\
J'_{uu} &= \left.\frac{\partial^2 J}{\partial u^2}\right|_{(u',d')} \\
J'_{dd} &= \left.\frac{\partial^2 J}{\partial d^2}\right|_{(u',d')} \\
J'_{ud} &= \left.\frac{\partial^2 J}{\partial u \partial d}\right|_{(u',d')} \\
J'_{du} &= \left.\frac{\partial^2 J}{\partial d \partial u}\right|_{(u',d')} = \left(J'_{ud}\right)^T
\end{aligned}$$

In [31] it is shown that the loss can be written as

$$L = \frac{1}{2}\|z\|_2^2 \tag{5.13}$$

where

$$z = \left(J'_{uu}\right)^{1/2} \left[\left(\left(J'_{uu}\right)^{-1} J'_{ud} - G^{-1} G_d\right)(d-d') + G^{-1} n\right] \tag{5.14}$$

where n is the implementation error. Introduce the diagonal scaling matrices W_d and W_n, where W_d represents the expected magnitudes of the disturbances and W_n represents the expected magnitude of the implementation error, such that

$$\begin{aligned}(d - d') &= W_d \tilde{d} \\ n &= W_n \tilde{n}\end{aligned}$$

where $\tilde{d}$ and $\tilde{n}$ are scaled to be less than 1 in magnitude. For reasons mentioned briefly above, and further explained in [31], it is in the following assumed that

$$\left\| \begin{bmatrix} \tilde{d} \\ \tilde{n} \end{bmatrix} \right\|_2 \leq 1 \tag{5.15}$$

This assumption leads to the following expression for the worst case loss:

$$L = \frac{1}{2}\bar{\sigma}^2(M) \tag{5.16}$$

where

$$\begin{aligned}M &= \begin{bmatrix} M_d & M_n \end{bmatrix} \\ M_d &= (J'_{uu})^{1/2} \left[(J'_{uu})^{-1} J_{ud} - G^{-1} G_d \right] W_d \\ M_n &= (J'_{uu})^{1/2} G^{-1} W_n\end{aligned}$$

5.5.5 Measurement combinations as controlled variables

It was concluded above that we would like the optimal value for our controlled variables to be insensitive to the value of disturbances. Previously, we have (implicitly) assumed that the controlled variables are *selected* among available measurements and manipulated variables[8]. In general, we may also consider *combinations* of variables as controlled variables. The nullspace method of Alstad and Skogestad [3] provides a method for finding controlled variables that are linear combinations of the candidate variables, such that the optimal values for the controlled variables are insensitive to changes in disturbances.

The nullspace method for selecting controlled variables

Neglecting measurement bias (implementation error) n, we see from (5.14) that the loss resulting from changes in disturbances will be zero provided $J'_{uu})^{-1}Jud - G^{-1}G_d = 0$, or, equivalently, if $G(J'_{uu})^{-1}Jud - G_d = 0$. Let G and G_d be factorized, respectively, as

$$G = HG^y, \qquad G_d = HG_d^y$$

[8]It may well turn out that it is optimal to keep a manipulated variable at a constant value - e.g., maximizing a flowrate - and the manipulated variables themselves should therefore be included among the candidate controlled variables.

where G^y is the steady state transfer function from the manipulated variables to *all* candidate controlled variables, and G_d^y is the steady state transfer function from the disturbances to candidate controlled variables. The matrix H is a matrix containing the linear relationships between the *candidate* controlled variables and the controlled variables *actually used*[9]. Thus, the optimal values of the controlled variables are insensitive to changes in disturbances provided

$$H\left(G^y(J_{uu}')^{-1}J_{ud}' - G_d^y\right) = HF = 0 \tag{5.17}$$

and we see immediately that the optimal values of the controlled variables are insensitive to changes in disturbances if H lies in the left nullspace of F. That is, the rows of H can be chosen as any linearly independent combination of the output singular vectors of F corresponding to singular values equal to zero.

It has been noted before that the number of controlled variables will equal the number of steady state degrees of freedom, i.e., $n_c = n_u$. The dimensions of F will be $n_y \times n_d$, where n_y is the number of *candidate* controlled variables and n_d is the number of disturbances. A sufficient condition for the existence of n_u controlled variables to exist, whose optimal values are independent of changes in disturbances, is therefore that $n_y \geq n_u + n_d$. If F has full column rank (which is normally the case), this sufficient condition is also necessary.

Extending the nullspace method to account for implementation error

A shortcoming of the nullspace method is that it ignores implementation error. Kariwala and coworkers [56] extends the method to account for implementation error, and also addresses the problem of minimizing the *average* loss, not only the worst-case loss. Interestingly, they find that when combinations of measurements are selected to minimize the average loss, the worst-case loss is also minimized (whereas minimizing the worst-case loss does not necessarily minimize the average loss).

The solution in [56] is reformulated by Alstad et al. in [4], and it is shown that an optimal H that minimizes (both average and worst-case) loss in the face of both implementation error and changes in disturbances is given by

$$H^T = (\tilde{F}\tilde{F}^T))^{-1}G^y(G^{yT}(\tilde{F}\tilde{F}^T))^{-1}G^y)^{-1}(J_{uu}')^{1/2} \tag{5.18}$$

5.5.6 The validity of the local analysis for controlled variable selection

The exact local method presented above, including the use of measurement combinations as controlled variables, is based on a Taylor series expansion around the nominal operating point. A relevant question is then whether the conclusions will hold also for non-optimal operating points, i.e., will changes in the disturbances invalidate the choice of controlled variables?

[9] If individual variables are selected as controlled variables, the matrix H would be a selection matrix consisting mostly of zeros, but with exactly one 1 in each row and at most one 1 in any column.

This issue is studied by Alstad [2], who found that the effect of a non-optimal operating point on the average cost is independent of the choice of controlled variables. That is, a non-optimal operating point will increase the average cost, but this increase is the same independent of what controlled variables are chosen. Thus, the ranking of sets of controlled variables based on average cost does not require the operating point to be optimal.

The conclusion in [2] is found to hold provided as long as (5.12) is a good approximation of the operating cost, and the linear plant and disturbance models are valid.

In this context we should bear in mind that although the second-order Taylor series expansion in (5.12) may be a good approximation to the cost function in 'the reduced space' over a significant operating region, this is not necessarily the same as it being a good approximation of the actual cost function in the same region. That is, we have assumed that any constraints that are active at the optimum are eliminated from the problem formulation, and the cost function is expressed in the 'reduced space' remaining after this elimination. When the operating point changes, the set of active constraints may change (active constraints may become inactive, or new constraints may become active). This will in general result in a 'break' in the cost function (in the 'full space'). Thus, at points where the set of active constraints changes, the cost function will be non-differentiable, and we must expect the Taylor series approximation to be significantly less accurate when moving beyond the region where the set of active constraints remains unchanged.

When the set of active constraints at optimum changes within the range of disturbances expected in operation, self-optimizing control has little to offer beyond the rather laborious exact evaluation of cost.

Another issue is what controller or controllers is used to implement the control of the selected controlled variables. Model Predictive Control (MPC) has a particular strength compared to most other controller types when it comes to managing constraints, in particular in accommodating changes in the set of active constraints.

5.6 Selection of manipulated variables

The manipulated variables are the variables that are manipulated directly by the control system. Sometimes these are referred to as 'physical' degrees of freedom, and typically include valve positions, electrical power inputs, etc.

With reference to the hierarchical control system structure in section 1.3, however, a somewhat more general interpretation of the term manipulated variable is often used: For any control layer, the manipulated variables are the variables that layer manipulates in the layer below. In this context, a setpoint to a loop in the layer below may well be regarded as a 'manipulated variable'. In this setting, the manipulated variables for one layer are thus not fully defined until the lower layers of the control system have been defined.

In this section we will briefly discuss the 'fundamental' manipulated variables / 'physical' degrees of freedom. It is noted in [65] that the selection of these typically

is not much of an issue, as they follow as a direct consequence of the design process itself. This is to a large extent true, as the control engineering discipline usually gets involved after the basic process design has been determined. It has frequently been stated that this state of affairs is unfortunate, and that control should be considered at every stage in the design process. Nevertheless, efforts to integrate control more tightly in the design process seems to have found very limited industrial application.

Although design limits the ability to choose manipulated variables for control, the process control engineer should attempt to

- verify as far as possible that the proposed manipulated variables make acceptable control possible, and
- review that the characteristics of the manipulated variables are appropriate.

In some cases, it may also be possible to provide additional manipulated variables (not available in the original plant design) at acceptable cost. This could involve, e.g., installing by-pass lines (with control valves) on heat exchangers.

Verifying that the proposed manipulated variables make acceptable control possible. Many plants are designed based on steady-state models only. In such circumstances, only steady state aspects of control are readily assessed. The ability to achieve a consistent inventory control system can often be assessed using only the Piping and Instrumentation Diagram (P&ID). Consistency of inventory control will be addressed more thoroughly in section 5.8, here we will only illustrate using a simple example (from an actual plant design) of how the P&ID can be used to assess whether consistent inventory control is possible.

Example. The wellstream in petroleum production typically consists of a mixture of gas, oil and water. These three phases have to be separated, before the oil and gas are transported to market, and the water (usually) discharged. There are 'purity' requirements for all these three phases. The main separation takes place in a series of three-phase separators. To further reduce the water content in the oil, the oil from the last separator is passed to a coalescer, where high voltage is used to force the remaining fine water droplets in the oil to coalesce. The coalesced, hence larger, water droplets separate easily from the oil. The final separator and coalescer, with proposed control system, are shown in Fig. 5.3. The oil level in the separator is controlled using a control valve in the oil export pipeline. This is OK, since the coalescer is filled with liquid, and no control of oil level in the coalescer is necessary. The water from the separator and coalescer are routed together to a produced water pump, which transports the water to treatment. The water levels in both the separator and the coalescer are measured, and the highest level signal is used for level control, manipulating the speed of the produced water pump. A manual value on the water pipeline from the separator is intended to adjust for different rates of water separation in the separator and coalescer.

The water levels in both the separator and coalescer are essentially integrating, and need to be stabilized by feedback. These integrators are in parallel, and we will therefore need two independent feedback paths to stabilize them. Here we have only

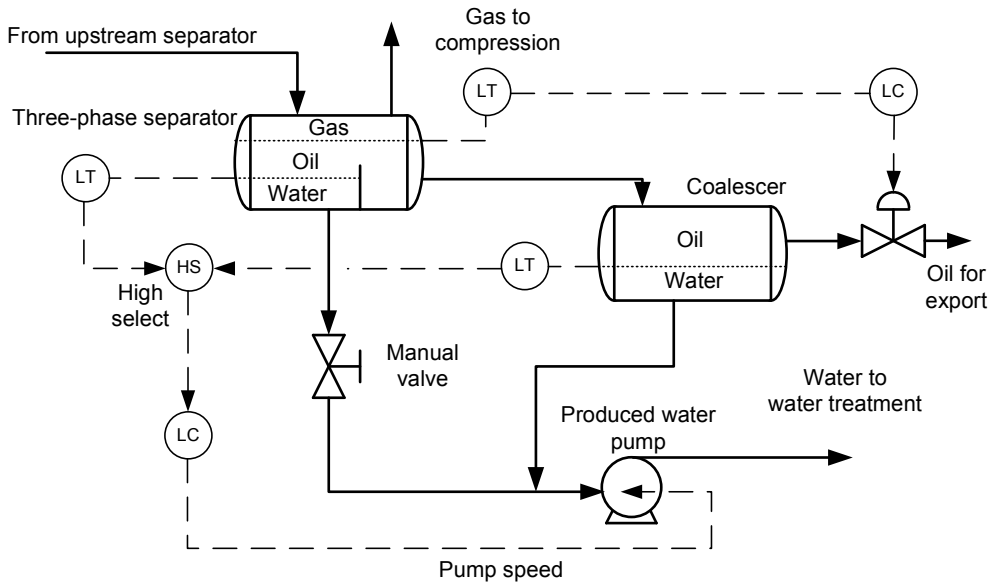

Figure 5.3: Final stage separator and coalescer with proposed level control.

one manipulated variable available to the control system for this stabilization task, and no controller can achieve the stabilization. The proposed control will avoid high water levels. However, for the vessel with the lower water level, the control provides no protection against emptying the vessel of water - consequently sending significant quantities of oil to the produced water treatment.

The manual valve will need to be replaced by an automatic valve to enable stabilization of both water levels - or one must expect the need for continuous monitoring and manual changes in the valve position by the operators.

Even when the need for additional manipulated variables is as blatantly obvious as in the above example, it may be a challenge to convince that additional manipulated variables need to be installed - in particular if the need for additional manipulated variables is discovered during the very hectic construction phase. Typically, when the need for additional or improved manipulated variables arise from dynamic rather than steady-state considerations, these are much harder both to identify and to argue convincingly for at the plant design stage - simply because a dynamic model often is not available.

To the extent that the available model allows, the limitations on achievable performance in section 3.4 should be assessed. This can provide clues also on how to improve the manipulated variables.

Reviewing the characteristics of the proposed manipulated variables. Three important aspects of a manipulated variable are:

1. *Size or capacity.* The manipulated variable should have the capacity to control throughout the expected range of operation. This implies that some range of manipulation should be available beyond the range of variation expected in steady state operation. Although this is an issue that may require some attention, typical steady-state plant designs will often fulfill this requirement. Typical manipulated variables (e.g., valves) are relatively inexpensive compared to the cost of major plant equipment, and therefore it is rarely economically optimal even from steady state considerations to let a manipulated variable limit throughput.

2. *Linearity of response.* Ideally, the response to a manipulated variable should be linear throughout its operating range, as this will minimize the need for changing controller parameters (retuning the controller) depending on operating conditions. This particularly relevant for that most common of manipulated variables, the control valve, which come in many different designs. A complete specification of control valves is (far) beyond the scope of this note. Nevertheless, we mention briefly that:

   - A *linear* valve characteristic is typically appropriate when the valve provides the main resistance to flow in the pipeline in which it is installed.
   - When there is other equipment providing significant resistance to flow, an *equal percentage* valve characteristic may be more appropriate, and may give an overall response (from valve position to flowrate) that is more linear than what would be achieved by a linear valve characteristic.

3. *Speed of response.* The control engineer should try to establish the required control bandwidth, and ensure that the manipulated variables are sufficiently fast to allow the required bandwidth. The speed of response of major disturbances, or the presence of unstable modes may give a clue to the required bandwidth.

5.7 Selection of measurements

When selecting measurements for control, one is often less restricted by the original design than what is the case for the selection of manipulated variables. Often, more measurements are available to choose from, and additional measurements may be installed at acceptable cost.

The understanding of the objectives of the different layers is important when selecting measurements. For example, the objectives of the regulatory control layer are, as described in section 5.3:

- *Stabilization.* Hence, it is important to select measurements that make the unstable mode(s) observable.

- *Linearization and removal of uncertainty by feedback.* For example, uncertainty or nonlinearity in a valve characteristic may be counteracted by flowrate control - which clearly implies that the flowrate needs to be measured.

- *Local rejection of disturbances* is much easier if the disturbances can be measured.

For the supervisory control layer, the selection of measurements will be influenced by

- the ability to *measure* or *estimate* the controlled variables of that layer (which preferably have been selected by considering plant economics, as described in preceding sections), and
- the ability to monitor important operational constraints (e.g., for constrained control using MPC).

Input-output controllability considerations are often useful when selecting measurements (at any layer).

- The selection of measured variables may affect the presence and location of (monovariable and multivariable) RHP zeros. For example, in [48] it is shown that the presence and location of multivariable RHP zeros in the FCC process[10] depend strongly on the choice of measured variables.
- The RGA may be used to avoid selecting measurement giving a very interactive system.
- Strong and direct responses from the manipulated to the measured variables are usually preferred. This typically implies having the measurements close to the manipulated variables. For example, if temperature control is used to control the mixing of a hot and a cold stream, positioning the measurement close to the mixing point will minimize time delay. However, positioning the measurement too close to the mixing point may make the measurement unreliable due to imperfect mixing.

5.8 Mass balance control and throughput manipulation

In most process plants there are a number of inventories (in particular liquid inventories, i.e., levels) that have to be stabilized by feedback. This is a major part of the 'stabilization objective' of the regulatory control layer. Stabilization of (liquid) inventories is often called 'mass balance control'[11].

Where the production rate or 'throughput' is set will greatly affect the structure of the inventory control system. There are generally three different cases:

- The production rate is set at the plant inlet. This is the case when the production is limited by raw material availability, or the plant receives its feedstock from an upstream plant with lower capacity. Sewage / waste water treatment plants also fall into this category.

[10] Fluid Catalytic Cracking (FCC) is an important process in refineries.

[11] A term this author actually finds somewhat inappropriate, since the dynamic mass balance necessarily will be fulfilled at all times.

- The production rate is set at the plant outlet. This is the 'produce-to-order' scenario, and is often the situation when product demand is weak.
- The throughput is set internally in the plant.

Selection of the throughput manipulator has very direct consequences for inventory control (also called mass balance control - typically level and pressure control loops). The throughput manipulator becomes unavailable for inventory control, and thus the inventory control loops have to 'radiate outwards' from the throughput manipulator. This is illustrated in Fig. 5.4 for a case where the throughput manipulator is located internally in the plant.

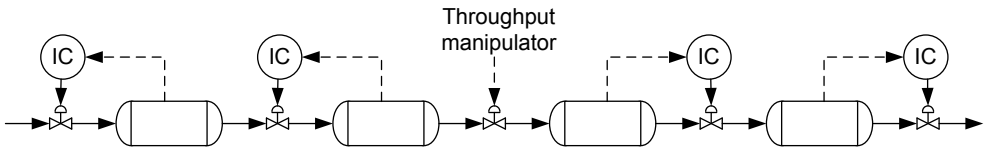

Figure 5.4: Throughput manipulator located internally in the plant - and the inventory control configuration 'radiating outwards' from the throughput manipulator.

According to Skogestad [89], the throughput has traditionally been set at the plant inlet. Price at al. [77] recommend using an internal stream for throughput manipulation instead. This is further specified in [89], where it is proposed to set the throughput at the bottleneck unit.

With hindsight, setting the throughput at the plant bottleneck seems the obvious choice when the plant is operating at maximum capacity, as should imply the ability to operate close to the capacity constraint. Setting the throughput several units away from the capacity constraint normally implies a 'long and slow loop' controlling the plant to its maximum capacity - which would imply a larger safety margin (or 'back off') to avoid violating operational constraints.

Aske [5] relates the rule of setting the throughput at the plant bottleneck to established results in network theory, discusses how to obtain estimates of the required back off, and how back off can be reduced.

When setting the throughput at the bottleneck unit in order to maximize production, the throughput is typically not set directly. Instead, the throughput manipulator is used to control some variable in the bottleneck unit to its constraint. For example, consider a process train where the capacity is limited by the available cooling capacity in a reactor, and cooling is required to stabilize the reactor. This is illustrated in Fig. 5.5. The stabilization is achieved by controlling reactor temperature using the opening of the cooling medium valve. The throughput may then be set by the feed to the reactor, but this is used to control the cooling medium valve opening to a setpoint. This setpoint should leave some range of operation for the temperature control, i.e., the setpoint for the cooling medium valve opening should be less than 100%, since some back off is required to avoid saturation of the temperature control loop.

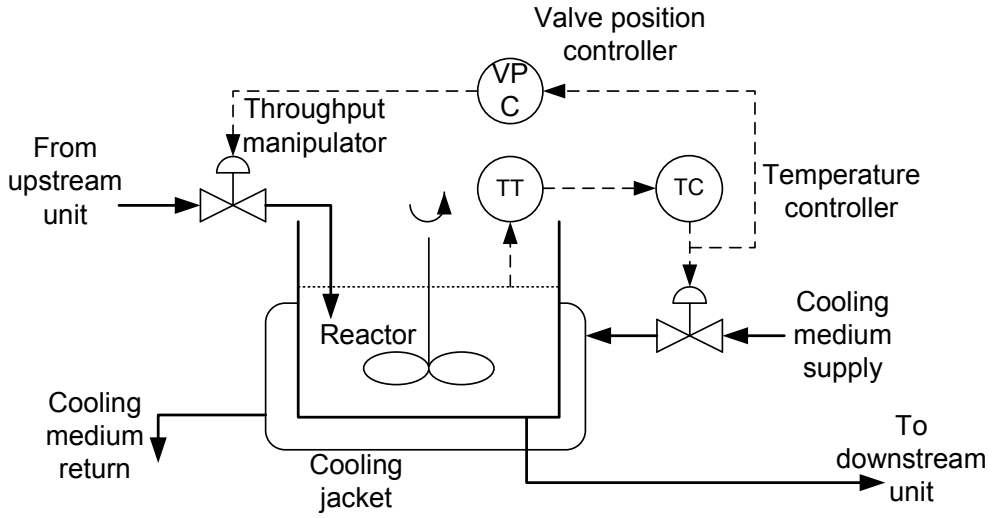

Figure 5.5: Throughput manipulator used to control cooling capacity to a safe distance from its constraint.

Changing operating conditions may cause the bottleneck unit to move, i.e., the constraint limiting production may change and move to another unit. With the rule of setting the throughput at the bottleneck unit, this would imply the need for reconfiguring the inventory control, which would be very impractical. An alternative may be to use MPC as a (multivariable) supervisory controller to handle the movement of the bottleneck.

5.8.1 Consistency of inventory control

Price and Georgakis [76] introduce the concept of *consistency* of inventory control, and state that "when an inventory control system is inconsistent, it cannot operate effectively by itself without additional control loops to supplement its action". This interpretation is modified by Aske [5]: "An inventory control system is consistent if the steady-state mass balances (total, components and phases) are satisfied for any part of the process, including the individual units and the overall plant".

Aske [5] similarly define the concept of *self-consistency* of inventory control: *A consistent inventory control system is said to be self-consistent ... if for each unit the local inventory control loops by themselves are sufficient to achieve steady-state mass balance consistency for that unit.* Clearly, consistency is required of any control system, while self-consistency is desired.

Further developing the ideas on consistency and self-consistency, Aske proposes the following self-consistency rule:

Self-consistency requires that:

1. *The total inventory of any part of the process must be "self-regulated" by its inflows or outflows, which implies that at least one flow in or one flow out of*

that part of the process must depend on the inventory inside that part of the process.

2. *For systems with several components, the inventory of each component foe any part of the process must be "self-regulated" by its in- or outflows or by chemical reaction.*

3. *For systems with several phases, the inventory of each phase for any part of the process must be "self-regulated" by its in- or outflows, or by phase transition.*

In this context, "self-regulation" means that the inventory is stabilized either by inherent feedback mechanisms in the process (the usual concept of self-regulation), *or* by local control loops.

Aske further develops the idea of self-regulation for some special cases:

- For *units in series* self-consistency implies that the inventory control must 'radiate outwards' from the throughput manipulator, as mentioned earlier.

- For *recycle loops*, the inventory *within the loop* must be "self-regulated" by the in- or outflows to the recycle loop.

- For *closed systems* (with no mass entering or leaving the system) one of the inventories must be left *uncontrolled*.

Illustrations of consistent and inconsistent inventory control structures for recycle loops are shoen in Fig. 5.6. For closed systems, the rule follows from noting that the outflow of one unit must be routed to other units, and that the total inventory is fixed.

Note: Aske's formulation of the self-consistency rule requires that *at least* one of the in- or outflows must depend on the inventory in the system. While this is correct, this still leaves room for conflicts between controllers when *more than one* in-or outflow is used to control the inventory. This is illustrated in Fig. 5.7. If both LC1 and LC2 are integrating, there is a possibility for conflict between the two controllers, leading one to increase output and the other to decrease output until a constraint is reached (open or closed valves). The conflict between the controllers can be caused by:

- Different level setpoints in the two loops.

- Different measurement noise in the two level measurements.

- Even if the two controllers were to use *the same* level sensor, there is a potential for differences in measurement noise if the sensor updates the measurement between the times when the two controllers are executed.

If either LC1 or LC2 is a P or PD controller (without integral action), the inventory control in Fig. 5.7 will be consistent.

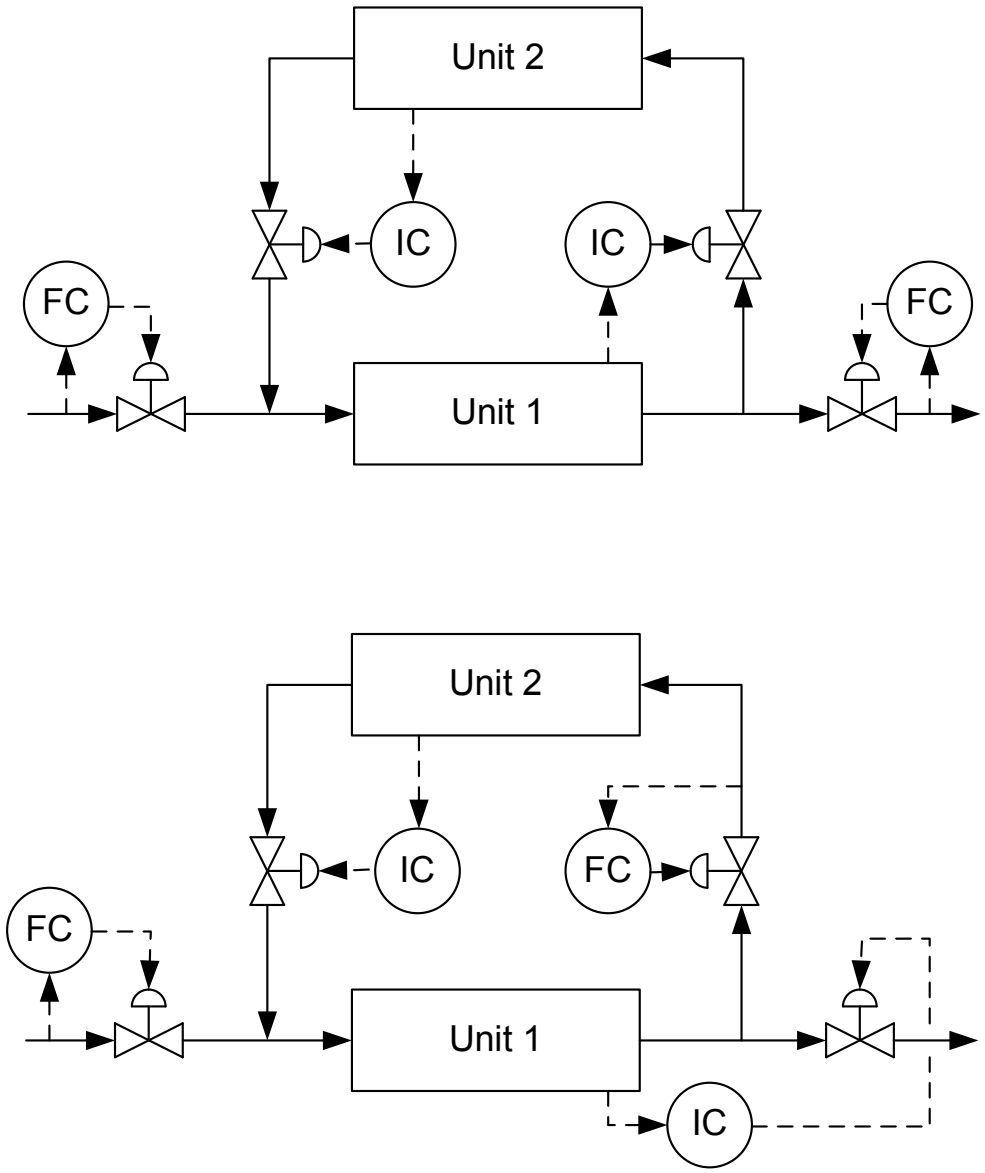

Figure 5.6: Illustrations of inventory control systems for recycle loops. Top: inconsistent inventory control, bottom: consistent inventory control.

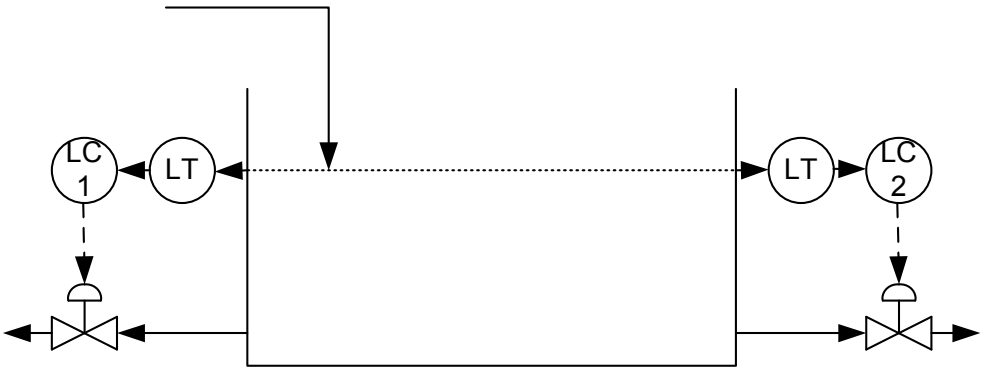

Figure 5.7: Inventory control which adheres to Aske's self-consistency rule, but where the consistency of the inventory control depends on the controllers used.

Chapter 6

Model-based predictive control

6.1 Introduction

Model-based predictive control (MPC) has become the most popular advanced control technology in the chemical processing industries. There are many variants of MPC controllers, both in academia and in industry, but they all share the common trait that an explicitly formulated process model is used to predict and optimize future process behaviour. Most MPC controllers are able to account for constraints both in manipulated variables and states/controlled variables through the formulation of the optimization problem.

When formulating the optimization problem in MPC, it is important to ensure that it can be solved in the short time available (i.e., the sampling interval is an upper bound on the acceptable time for performing the calculations). For that reason, the optimization problem is typically cast into one of two standard forms:

- Linear programming (LP) formulation. In an LP formulation, both the objective function and the constraints are linear.
- Quadratic programming (QP) formulation. In a QP formulation, the objective function is quadratic, whereas the constraints have to be linear. In addition, to ensure that there exists a unique optimal solution that can be found quickly with effective optimization solvers, the *Hessian matrix* in the objective function has to be *positive definite*[1].

LP problems can be solved more efficiently than QP problems, and an LP formulation may therefore be advantageous for very large optimization problems. However, a QP formulation generally leads to smoother control action and more intuitive effects of changes in the tuning parameters. The connection to 'traditional advanced control', i.e., linear quadratic (LQ) optimal control, is also much closer for a QP

[1]The Hessian matrix defines the quadratic term in the objective function, and is a symmetric matrix. Positive definiteness means that all eigenvalues are positive - for a monovariable optimization problem this implies that the coefficient for the quadratic term in the objective function is positive.

formulation than for an LP formulation. For these reasons, we will focus on a QP formulation in the following, and describe in some detail how a QP optimization problem in MPC may be formulated.

6.2 Formulation of a QP problem for MPC

A standard QP problem takes the form

$$\min_{v} \quad 0.5 v^T \widetilde{H} v + c^T v \tag{6.1}$$

subject to the constraints

$$Lv \leq b \tag{6.2}$$

Here v is the vector of free variables in the optimization, whereas $\widetilde{H}$ is the *Hessian matrix*, that was mentioned above, and which has to be positive definite. The vector c describes the linear part of the objective function, whereas the matrix L and the vector b describe the linear constraints. Some QP solvers allow the user to specify separate upper and lower bounds for v, whereas other solvers require such constraints to be included in L and b. For completeness, we will assume that these constraints have to be included in L and b.

The formulation of the MPC problem starts from a linear, *discrete-time* state-space model of the type

$$x_{k+1} = Ax_k + Bu_k + Ed_k \tag{6.3}$$
$$y_k = Cx_k + Fd_k \tag{6.4}$$

where the subscripts refer to the sampling instants. That is, subscript $k+1$ refers to the sample instant one sample interval after sample k. Note that for discrete time models used in control, there is normally no direct feed-through term, the measurement y_k does not depend on the input at time k, but it does depend on the input at time $k-1$ through the state x_k. The reason for the absence of direct feed-through is that normally the output is measured at time k before the new input at time k is computed and implemented. One may also argue that in most physically realistic system descriptions, inputs and disturbances affect the rate of change of states rather than the states themselves. To illustrate: mass transfer/flowrate disturbances affect the rate of accumulation of mass, heat transfer/temperature disturbances affect the rate of accumulation of energy, force disturbances affect acceleration (the rate of accumulation of momentum), etc.

In the same way as is common in control literature, the state x, input u, external disturbance d and measurement y above should be interpreted as *deviation variables.* This means that they represent the deviations from some consistent set of of variables

$\{x_L, u_L, d_L, y_L\}$ around which the model is obtained[2]. For a stable process, the set $\{x_L, u_L, d_L, y_L\}$ will typically represent a steady state - often the steady state we want to keep the process at. To illustrate, if in y_L represents a temperature of $330K$, a physical measurement of $331K$ corresponds to a devation variable $y = 1K$.

A typical optimization problem in MPC might take the form

$$\begin{aligned} \min_u \quad f(x,u) &= \sum_{i=0}^{n-1} \{(x_i - x_{ref,i})^T Q (x_i - x_{ref,i}) \\ &\quad + (u_i - u_{ref,i})^T P (u_i - u_{ref,i})^T \} \\ &\quad + (x_n - x_{ref,n})^T S (x_n - x_{ref,n}) \end{aligned} \tag{6.5}$$

subject to constraints

$$\begin{aligned} x_0 &= \text{given} \\ U_L &\le u_i \le U_U \quad \text{for } 0 \le i \le n+j \\ Y_{L,i} &\le H_i x_i \le Y_{U,i} \quad \text{for } 1 \le i \le n+j \end{aligned} \tag{6.6}$$

In the objective function Eq. (6.5) above, we penalize the deviation of the states x_i from some desired reference trajectory $x_{ref,i}$ and the deviation of the inputs u_i from some desired trajectory $u_{ref,i}$. These reference trajectories are assumed to be given to the MPC controller by some outside source. They may be constant or may also vary with time (subscript i). The constraints on achievable inputs or acceptable states are usually not dependent on the reference trajectories, and therefore these reference trajectories do not appear in the constraint equations (6.6). Usually, the state constraints represent constraints on process measurements (giving $H = C$), but constraints on other combinations of states are also possible (including constraints on combinations of inputs and states). Typically, the constraints are constant with time, but we will express the constraints for $i > n$ in terms of the (predicted) state at $i = n$, and for that reason the time index i also appear in the state constraints.

In the following, this formulation of the optimization problem will be recast into the standard QP formulation in Eqs.(6.1) and (6.2), but first a number of remarks and explanations to the optimization problem formulation in Eqs.(6.5) to (6.6) are needed.

- In addition to the above constraints, it is naturally assumed that the process follows the model in Eqs. (6.3) and (6.4).
- The matrices Q, P, and S are all assumed to be symmetric. P and S are assumed to be positive definite, whereas Q may be positive semi-definite.

[2]We do not here specify *how* the model is obtained, but typically it is either the result of identification experiments performed around the values $\{x_L, u_L, d_L, y_L\}$ or the result of linearizing and discretizing a non-linear, physical model around the values $\{x_L, u_L, d_L, y_L\}$.

- In many applications it may be more natural to put a weight (or cost) on the actual measurements rather than the states. This can easily be done by choosing $Q = C^T \widetilde{Q} C$, where $\widetilde{Q}$ is the weight on the measurements.

- One may also put constraints on the rate of change of the inputs, giving additional constraints on the form $\Delta U_L \leq u_i - u_{i-1} \leq \Delta U_U$.

- For the output constraints in Eq. (6.6) to be well defined, we must specify how the inputs u_i should behave in the interval $n \leq i \leq n+j$. Typical choices for this time interval are either that $u_i = u_{ref,i}$, $u_i = u_{i-1}$, or that $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$. The latter choice assumes that a (stabilizing) state feedback controller is used in this time interval. Note that this controller will never be used in practice (since the MPC calculations are re-computed at each sample instant), but it is needed to make the constraints well defined.

- Similarly, we must predict future values for disturbances. Good predictions may sometimes be available, due to e.g., knowledge about operation of upstream equipment. In the absence of such information, it is common to assume that the disturbance will keep its present (measured or estimated) value over the prediction horizon.

- If one assumes that $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$ for $n \leq i \leq n+j$, one should also include the input constraints in the problem formulation for the time interval $n \leq i \leq n+j$. These input constraints then effectively become state constraints for this time interval.

- Some MPC formulations use an objective function of the form $f(x,u) = \sum_{i=0}^{n_p} (x_i - x_{ref,i})^T Q (x_i - x_{ref,i}) + \sum_{i=0}^{n_u} (u_i - u_{ref,i})^T P (u_i - u_{ref,i})$, where $n_p > n_u$, and typically assume that $u_i = u_{ref,i}$ for $n_u < i < n_p$. Note that this corresponds to a particular choice for 'terminal state weight' S, since x_i for $n_u + 1 < i \leq n_p$ will then be given by x_{n_u+1} (and the process model).

- It is common to introduce integral action in MPC controllers by using the input *changes* at time i as free variables in the optimization, rather than the input itself. This follows, since the actual inputs are obtained by integrating the changes in the input. This can be done within the same framework and model structure as above, using the model

$$\begin{aligned} \widetilde{x}_{k+1} &= \begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \widetilde{A}\widetilde{x}_k + \widetilde{B}\Delta u_k + \begin{bmatrix} E \\ 0 \end{bmatrix} d_k \\ y_k &= \widetilde{C}\widetilde{x}_k \end{aligned}$$

where $\Delta u_k = u_k - u_{k-1}$, and

$$\widetilde{A} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix}, \quad \widetilde{B} = \begin{bmatrix} B \\ I \end{bmatrix}, \quad \widetilde{C} = \begin{bmatrix} C & 0 \end{bmatrix}$$

Alternatively, integral action may be included by including a disturbance estimate in the model update function, and adjusting state and input references according to the disturbance estimate. This approach provides more flexibility to account for different disturbance dynamics, the method described above is best suited for step-like disturbance or reference changes.

- To have a stable closed-loop system, it is necessary to have at least as many feedback paths as integrators. When integral action is included in the way described above, this means that one needs at least as many (independent) measurements as inputs. When the number of inputs exceeds the number of measurements, it is common to define 'ideal resting values' for some inputs. This essentially involves putting some inputs in the measurement vector, and defining setpoints for these.

In the following, we will recast the MPC optimization problem as a standard QP problem. We will assume that $u_i - u_{ref,n} = K(x_i - x_{ref,n})$ for $n \leq i \leq n+j-1$. To start off, we stack the state references $x_{ref,i}$, input references $u_{ref,i}$, input deviations $v_i = u_i - u_{ref,i}$, state deviations $\chi_i = x_i - x_{ref,i}$, and predicted disturbances d_i in long (column) vectors x_{ref}, u_{ref}, v, χ_{dev}, and δ:

$$u_{ref} = \begin{bmatrix} u_{ref,0} \\ u_{ref,1} \\ \vdots \\ u_{ref,n-2} \\ u_{ref,n-1} \end{bmatrix}; \quad x_{ref} = \begin{bmatrix} x_{ref,1} \\ x_{ref,2} \\ \vdots \\ x_{ref,n-1} \\ x_{ref,n} \end{bmatrix};$$

$$v = \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{bmatrix}; \quad \chi = \begin{bmatrix} \chi_1 \\ \chi_2 \\ \vdots \\ \chi_{n-1} \\ \chi_n \end{bmatrix}; \quad \delta = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

Introducing the matrices

$$\widehat{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 & 0 \\ 0 & Q & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & Q & 0 \\ 0 & 0 & \cdots & 0 & S \end{bmatrix}, \quad \widehat{P} = \begin{bmatrix} P & 0 & \cdots & 0 & 0 \\ 0 & P & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & P & 0 \\ 0 & 0 & \cdots & 0 & P \end{bmatrix} \tag{6.7}$$

$$\widehat{H} = \begin{bmatrix} H_1 & 0 & \cdots & 0 & 0 \\ 0 & H_2 & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & H_{n-1} & 0 \\ 0 & 0 & \cdots & 0 & H_n \end{bmatrix} \tag{6.8}$$

and the vectors

$$\widehat{U}_L = \begin{bmatrix} U_L \\ \vdots \\ U_L \end{bmatrix}; \quad \widehat{U}_U = \begin{bmatrix} U_U \\ \vdots \\ U_U \end{bmatrix}; \quad \widehat{Y}_L = \begin{bmatrix} Y_L \\ \vdots \\ Y_L \end{bmatrix}; \quad \widehat{Y}_U = \begin{bmatrix} Y_U \\ \vdots \\ Y_U \end{bmatrix} \tag{6.9}$$

The future state trajectory may now be expressed as

$$\begin{aligned} \chi + x_{ref} &= \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{n-1} \\ A^n \end{bmatrix} x_0 + \begin{bmatrix} B & 0 & \cdots & 0 & 0 \\ AB & B & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ A^{n-2}B & A^{n-3}B & \cdots & B & 0 \\ A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} (v + u_{ref}) \\ &+ \begin{bmatrix} E & 0 & \cdots & 0 & 0 \\ AE & E & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ A^{n-2}E & A^{n-3}E & \cdots & E & 0 \\ A^{n-1}E & A^{n-2}E & \cdots & AE & E \end{bmatrix} \delta \\ &= \widehat{A}x_0 + \widehat{B}(v + u_{ref}) + \widehat{B}_d\delta \end{aligned} \tag{6.10}$$

Next, three nominally equivalent formulations of the QP optimization problem in MPC will be described.

6.2.1 Future states as optimization variables

The optimization problem may now be expressed as

$$\min_{v,\chi} \begin{bmatrix} \chi^T & v^T \end{bmatrix} \begin{bmatrix} \widehat{Q} & 0 \\ 0 & \widehat{P} \end{bmatrix} \begin{bmatrix} \chi \\ v \end{bmatrix} \tag{6.11}$$

subject to constraints

$$\begin{aligned} \begin{bmatrix} 0 & -I \\ 0 & I \\ -\widehat{H} & 0 \\ \widehat{H} & 0 \end{bmatrix} \begin{bmatrix} \chi \\ v \end{bmatrix} &\leq \begin{bmatrix} -\widehat{U}_L + u_{ref} \\ \widehat{U}_U - u_{ref} \\ -\widehat{Y}_L + x_{ref} \\ \widehat{Y}_U - x_{ref} \end{bmatrix} \\ \begin{bmatrix} I - \widehat{B} \end{bmatrix} \begin{bmatrix} \chi \\ v \end{bmatrix} &= \widehat{A}x_0 + \widehat{B}u_{ref} + \widehat{B}_d\delta \end{aligned}$$

If the particular QP solver in use does not accept equality constraints, these can always be specified using two inequalities[3].

[3] I.e., $ax = b \Leftrightarrow \{ax \leq b \text{ and } ax \geq b\}$.

In this case, the optimization variables are both the future plant inputs v and future plant states χ. The model equations (expressed as equality constraints) guarantee that the relationship between inputs and states are fulfilled, and the *de facto* maximum number of degrees of freedom in the optimization is given by the number of inputs times the prediction horizon n.

6.2.2 Using the model equation to substitute for the plant states

We will use the *superposition principle,* which states that the total effect of several inputs can be obtained simply by summing the effects of the individual inputs. The superposition principle is always valid for linear systems, but typically does not hold for non-linear systems. This allows us to split χ into to components, $\chi = \chi_{dev} + \chi_v$. Here χ_{dev} is the deviation from the desired state trajectory that would result, given the initial state x_0 and assuming that the nominal reference input u_{ref} is followed, and that the predicted future disturbances are correct. Similarly, χ_v is the effect on the future state trajectory from the future deviations from the reference input. The model equations then give

$$\chi_{dev} = \widehat{A}x_0 + \widehat{B}u_{ref} + \widehat{B}_d\delta - x_{ref} \tag{6.12}$$

$$\chi_v = \widehat{B}v \tag{6.13}$$

Adding (6.12) and (6.13) we get the model equations used in the preceding subsection.

The objective function can be written as

$$\begin{aligned} f(x,u) = f(\chi_{dev},\chi_v,v) &= (x_0 - x_{ref,0})^T Q(x_o + x_{ref,0}) + \\ &\quad (\chi_{dev}+\chi_v)^T\widehat{Q}(\chi_{dev}+\chi_v) + v^T\widehat{P}v \\ &= (x_0 - x_{ref,0})^T Q(x_o + x_{ref,0}) + \chi_{dev}^T\widehat{Q}\chi_{dev} + \\ &\quad 2\chi_{dev}^T\widehat{Q}\chi_v + \chi_v^T\widehat{Q}\chi_v + v^T\widehat{P}v \end{aligned}$$

which should be minimized using the vector v as free variables.

Now, the terms $(x_0 - x_{ref,0})^T Q(x_o + x_{ref,0}) + \chi_{dev}^T\widehat{Q}\chi_{dev}$ will not be affected by the optimization, and may therefore be removed from the objective function. This is because we are primarily interested in finding the inputs that minimize the objective function, and not in the optimal value of the objective function. Thus, the objective function is in the form of a standard QP problem as defined in Eqs. (6.1) and (6.2) if we define

$$\begin{aligned} \widetilde{H} &= \widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P} \\ c^T &= \chi_{dev}^T\widehat{Q}\widehat{B} \end{aligned} \tag{6.14}$$

It now remains to express the constraints in the MPC problem in the form of a standard QP problem. Using (6.12) and (6.13) to substitute the model equations into the inequality constraints, we obtain

$$\begin{bmatrix} -\widehat{B} \\ \widehat{B} \\ -\widehat{H}\widehat{B} \\ \widehat{H}\widehat{B} \end{bmatrix} v \leq \begin{bmatrix} -\widehat{U}_L + u_{ref} \\ \widehat{U}_U - u_{ref} \\ -\widehat{Y}_L + x_{ref} \\ \widehat{Y}_U - x_{ref} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \widehat{H}\left(\widehat{A}x_0 + \widehat{B}u_{ref} + \widehat{B}_d\delta - x_{ref}\right) \\ -\widehat{H}\left(\widehat{A}x_0 + \widehat{B}u_{ref} + \widehat{B}_d\delta - x_{ref}\right) \end{bmatrix} \tag{6.15}$$

6.2.3 Optimizing deviations from linear state feedback

The main reason for using model predictive control is usually it's ability to handle constraints. If constraints are not a problem, linear state feedback (using e.q. LQ-optimal control) would often be preferred. Indeed, if no constraints are active, many MPC formulations can be shown to be equivalent to linear state feedback. This has lead Rossiter (XXXX cite Rossiter) to propose an MPC formulation where the degrees of freedom in the optimization are the deviations from linear state feedback that are necessary to adhere to the constraints. Thus, the input is parameterized as

$$u_i - u_{ref,i} = K(x_i - x_{ref,i}) + c_i \tag{6.16}$$

for some given state feedback controller K. Here c_i are the deviations from linear state feedback that are to be minimized, and it is assumed that $c_i = 0$ for $i \geq n$. We introduce the notation

$$\widehat{K} = \text{diag}\{K\}; \quad \widehat{c} = \left[c_0^T, c_1^T, \cdots, c_{n-1}^T\right]^T$$

Next, we use the model equations to express the future manipulated variables. When using (6.10) one needs to keep in mind that v starts from time $i = 0$, whereas χ starts from time $i = 1$. Thus we define

$$\begin{aligned} \widehat{A}' &= \text{the } n \text{ first blocks of rows of } \begin{bmatrix} I \\ \widehat{A} \end{bmatrix} \\ \widehat{B}' &= \text{the } n \text{ first blocks of rows of } \begin{bmatrix} 0 \\ \widehat{B} \end{bmatrix} \\ \widehat{B}_d' &= \text{the } n \text{ first blocks of rows of } \begin{bmatrix} 0 \\ \widehat{B}_d \end{bmatrix} \\ x_{ref}' &= \text{the } n \text{ first blocks of rows of } \begin{bmatrix} x_{ref,0} \\ x_{ref} \end{bmatrix} \end{aligned}$$

where the $'$ sign should not be confused with the transposition of the matrix. Future plant inputs may thus be expressed as

$$v = \widehat{K}\left(\widehat{A}'x_0 + \widehat{B}'(v + u_{ref}) + \widehat{B}_d'\delta - x_{ref}'\right) + \widehat{c} \tag{6.17}$$

Rearranging, we obtain

$$v = (I - \widehat{K}\widehat{B}')^{-1}\widehat{K}\left(\widehat{A}'x_0 + \widehat{B}'u_{ref} + \widehat{B}'_d\delta - x'_{ref}\right) + (I - \widehat{K}\widehat{B}')^{-1}\widehat{c} \tag{6.18}$$

where we note that $(I - \widehat{K}\widehat{B}')$ is always invertible since $\widehat{B}'$ have all non-zero elements below the main diagonal. It is trivial (but tedious) to substitute (6.18) into (6.14) and (6.15) to obtain the corresponding standard QP formulation in terms of the new optimization variables $\widehat{c}$.

6.2.4 Constraints from time n to $n + j$

For time n to $n + j$, it is relatively straight forward to use the model equation (6.3), together with the specified input usage (usually $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$) to express the states for $i > n$ and plant inputs for $i \geq n$ in terms of the predicted state x_n, predicted input and state references $u_{ref,i}$ and $x_{ref,i}$ and predicted future disturbances d_i. Thus, constraints in states and inputs for $i \geq n$ can be expressed as constraints on x_n. Thus, all constraints specified in (6.6) are well defined, even though the prediction horizon is of length n.

Many of these state constraints at time n representing state or input constraints in the interval $n \leq i \leq n + j$ may be redundant. One would ideally like to remove redundant constraints to ensure that the optimization problem is as small as possible. This can be done using the procedure described in Appendix 2. However, in applications where references ($u_{ref,i}$ and $x_{ref,i}$) or disturbances d_i vary, one will either have to determine redundant constraints on-line (prior to the optimization), or only remove constraints that are *always* redundant, i.e., constraints that are redundant for all conceivable values for $u_{ref,i}$, $x_{ref,i}$ and d_i, for $i \geq n$.

This is not as hopeless as it may seem. The constraints are linear, and this allows us to check for redundancy only at the extreme values of the variables. Furthermore, the prediction horizon is also commonly chosen sufficiently long for the plant to reach steady state, and thus it is reasonable to assume that $u_{ref,i} = u_{ref,n}$, $x_{ref,i} = x_{ref,n}$ and $d_i = d_n$, for $i \geq n$. This will reduce the number of variables that need to be considered.

Furthermore, if the control is supposed to remove offset at steady state, the references have to be consistent, i.e., at steady state (denoted by subscript ss), input $u_{ref,ss}$ and disturbance d_{ss} must result in the state $x_{ref,ss}$. Normally, one would consider d_{ss} and $x_{ref,ss}$ as independent variables, and $u_{ref,ss}$ as a dependent variable. Calculating consistent steady state references for given disturbances is the task of the *target calculation*, addressed in Section XXXX.

Many control problems are formulated based on the assumption that the reference values for states and inputs are zero, and reference changes are implemented by 'shifting the origin' for the deviation variables. However, the constraints are typically independent of the references, and shifting the origin will result in a corresponding shift in the constraints. Thus, shifting the origin does not remove the problem of variable references when constraints have to be considered.

6.2.5 Required value for j

The purpose of imposing constraints over j time steps longer than the prediction horizon is to ensure that optimizing over horizon of n steps does not lead to future optimization problems with no feasible solution. Furthermore, one of the ways of ensuring closed loop stability with MPC, is to design the MPC to correspond to a stabilizing, unconstrained state feedback controller (i.e., $(u_i - u_{ref,i}) = K(x_i - x_{ref,i}$, again) after the prediction horizon n. However, for the MPC to correspond to the unconstrained state feedback controller, the state feedback controller must not violate any constraints.

Thus, we want the predicted state at $i = n$ to lie within a set within which the state feedback controller does not violate any constraints, and the state feedback controller should be able to keep the state within that set for all $i > n$. Ideally, we would like to identify the largest such set in the state space, since this leads to the largest feasible region for a given prediction horizon n.

This set is known as the *maximal output admissible set*, often denoted $\mathcal{O}_\infty$. The properties and the determination of $\mathcal{O}_\infty$ are studied by Gilbert and Tan [27]. We will assumed that the state constraints in (6.6) constitute a closed and bounded polyhedron in the state space, and that the origin is in the interior of this polyhedron. Operation arbitrarily far from the origin is of no practical interest, and if the assumption above is not fulfilled it is therefore possible to add very lax state constraints to fulfill the assumption. This allows us to use the results of [27] for rather straight forward determination of the required j.

Let $\mathcal{O}_t$ denote the set in the state space for which the constraints are feasible over t time steps using the state feedback controller. Obviously, $\mathcal{O}_\infty \subseteq \mathcal{O}_{t+1} \subseteq \mathcal{O}_t$. We will use the following results from [27]:

R1 $\mathcal{O}_\infty$ is closed and bounded (and is convex due to the linearity of the constraints).

R2 $\mathcal{O}_\infty$ is *finitely determined* if $\mathcal{O}_\infty = \mathcal{O}_t$ for finite t. For the cases studied here, $\mathcal{O}_\infty$ is finitely determined by construction.

R3 If $\mathcal{O}_t = \mathcal{O}_{t+1}$ then $\mathcal{O}_\infty = \mathcal{O}_t$.

This leads to the following algorithm for determination of $\mathcal{O}_\infty$, and simultaneously determining the required value of j:
Algorithm 1. Maximal Output Admissible Set.

1. Set $t = 0$, and let $\mathcal{O}_t$ be parameterized by (6.6) for $k = n$. The constraints considered should be both the state constraints, and the constraints on the states implied by the input constraints, due to the use of the state feedback controller.

2. Increment the time index t, and express the constraints at time t in terms of x_n, using the system model (6.3) and the equation for the state feedback controller.

3. Remove any redundant constraints for time t. If all constraints for time index t are redundant, $\mathcal{O}_{t-1} = \mathcal{O}_t$, and hence $\mathcal{O}_\infty = \mathcal{O}_{t-1}$. Stop. Otherwise, augment

the set of constraints describing $\mathcal{O}_{t-1}$ by the non-redundant constraints for time t to define $\mathcal{O}_t$. Go to Step 2.

Due to R2 above, this algorithm will terminate in finite time for the problems considered here. Checking for redundancy of constraints is also straight forward for linear systems subject to linear inequality constraints, as explained in Appendix 2.

For problems where references or disturbances may vary, it is necessary to verify the redundancy of the constraints for all combinations of extreme values of these variables, as explained in the preceding subsection. The determination of $\mathcal{O}_\infty$ for systems with disturbances has been addressed in [62].

6.2.6 Feasible region and prediction horizon

It was explained above that in order to guarantee closed loop stability, we will want the state at time x_n to lie within the maximal output admissible set $\mathcal{O}_\infty$. The feasible region for an MPC controller is therefore the set of states from which the state can be brought to $\mathcal{O}_\infty$ in n steps, without violating any constraints. The feasible region for a given n and given $\mathcal{O}_\infty$ can be found using Fourier-Motzkin elimination (Appendix 1), as noted in [58]. However, the Fourier-Motzkin procedure produces a number of redundant constraints which subsequently has to be removed. To minimize this problem, it is recommended to start from a prediction horizon $n = 0$ (i.e., the feasible region = $\mathcal{O}_\infty$) and gradually increment the prediction horizon, and remove redundant constraints along the way, see [44] for more detail.

6.3 Step response models

In industrial practice, process models based on step response descriptions have been very successful. Whereas step response models have no theoretical advantages, they have the practical advantage of being easier to understand for engineers with little background in control theory.

With a soild understanding of the material presented above, the capable reader should have no particular problem in developing a similar MPC formulation based on a step response model. Descriptions of such formulations can also be found in available publications, like Garcia and Morshedi's [26] original paper presenting "Quadratic Dynamic Matrix Control". Alternatively, step response models may also be expressed in state space form (with a larger number of states than would be necessary in a "minimal" state space model), see e.g. [46] for details.

The reader should beware that step-response models have "finite memory", and hence should only be used for asymptotically stable processes, that is, processes where the effect of old inputs vanish over time. Most industrially successful MPC controllers based on step response models are modified to handle also integrating processes, whereas truly unstable processes cannot be handled. Handling unstable processes

using step response models would require more complex modifications to the controllers and model description, and would thereby remove the step response model's advantage of being easy to understand.

Partly due to these reasons, MPC controllers are seldom used on unstable processes. If the underlying process is unstable, it is usually first stabilised by some control loops, and the MPC controller uses the setpoint of these loops as "manipulated variables".

In academia, there is widespread resentment against step response models - and in particular against their use in MPC controllers. Although there are valid arguments supporting this resentment, these are usually of little practical importance for asymptotically stable processes - although in some cases the computational burden can be reduced by using a state space model instead.

Step response *identification* is another matter. A step input has Laplace transform $u(s) = \frac{k}{s}$, and hence excites the process primarily at low frequencies. The resulting model can therefore be expected to be good only for the slow dynamics (low frequencies). If medium to high bandwidth control is desired for an MPC application, one should make sure that any identification experiment excites the process over the whole desired bandwidth range for the controller.

6.4 Updating the process model

The MPC controller essentially controls the *process model*, by optimizing the use of the inputs in order to remove the predicted deviation from some desired state (or output) trajectory. Naturally, good control of the *true process* will only be obtained if the process model is able to predict the future behaviour of the true process with reasonable accuracy. Model errors and unknown disturbances must always be expected, and therefore it will be necessary to update the process model to maintain good quality predictions of the future process behaviour.

The design of state estimators or -observers is itself a vast area, and is the subject of numerous books. Furthermore, this is an area that has seen a lot of interesting developments recently. No attempt will therefore be made at giving a comprehensive treatment of this subject. Instead, a short description of techniques that are particularly relevant for MPC applications will be given - but readers are certainly encouraged to obtain more thorough insight elsewhere.

6.4.1 Bias update

For asymptotically stable systems, a particularly simple model updating strategy is possible for MPC formulations that only use process inputs and measurements in the formulation (i.e., when unmeasured states do not appear in the objective function or in the constraints). In such cases, it would be natural to calculate the predicted *deviations from the desired output trajectory* (which may be called, say, $_{dev}$), rather than the predicted deviations from the desired *state* trajectory χdev. Then, the model

can be 'updated' by simply adding the present difference between process output and model output to the model's prediction of the future outputs. This is known as a 'bias update', and is widespread in industrial applications. Note, however, that the bias update

- is only applicable to asymptotically stable systems, and may result in poor control performance for systems with slow disturbance dynamics, and that
- it may be sensitive to measurement noise. If a measurement is noisy, one should attempt to reduce the noise (typically by a simple low-pass filter) before calculating the measurement bias.

6.4.2 Kalman filter and Extended Kalman Filters

The Kalman filter is probably the model updating technique of choice for the 'purist', as it is 'optimal' in the sense of minimizing the variance of the estimation error for linear systems subject to Gaussian noise[4].

In order to present the Kalman filter equations, some nomenclature must be introduced:

$\hat{x}_{k\|k-n}$	The n step ahead prediction of the state at time k.
$\hat{x}_{k\|k-1}$	The 1 step ahead prediction of the state at time k, i.e., the best estimate of the state at time k using information available up to and including time $k-1$ (also known as the *a priori* estimate).
$\hat{x}_{k\|k}$	The estimate of the state at time k, accounting for information available up to and including time k (also known as the *a posteriori* estimate).
w_k	State excitation noise at time k, assumed to be normally distributed with zero mean, and to have no correlation between values at different times k.
v_k	Measurement noise at time k, also assumed to be normally distributed with zero mean, without correlation in time.
W	Variance of the state exitation noise w.
V	Variance of the measurement noise v.
$\Pi_{k\|k-n}$	Variance in the state estimate for time k, when accounting for information up to and including time $k-n$.
$\Pi_{k\|k}$	Variance in the state estimate for time k, when accounting for information up to and including time k.
$\Pi_0 = \Pi_{0\|0}$	Variance in initial state estimate (given or estimated).

The state excitation noise and measurement noise are included in the plant model as fol-

[4]The use of 'inverted commas' around *purist* and *optimal* should not be interpreted as any disregard of control theory. One should, however, keep in mind that most real-life systems are not linear, and that Gaussian noise cannot capture all the observed differences between model predictions and actual observations. Despite these reservations, the Kalman filter has proven valuable in numerous applications.

lows:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Ew_k \\ y_k &= Cx_k + v_k \end{aligned} \tag{6.19}$$

The Kalman filter equations are then given by (see, e.g., [1]):

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} + Bu_k \tag{6.20}$$

$$\Pi_{k+1|k} = A\Pi_{k|k}A^T + EWE^T \tag{6.21}$$

$$\Pi_{k+1|k+1} = \Pi_{k+1|k} - \Pi_{k+1|k}C^T(C\Pi_{k+1|k}C^T + V)^{-1}C\Pi_{k+1|k} \tag{6.22}$$

$$\tag{6.23}$$

When the measurement y_{k+1} is obtained, this is used to update the state estimate:

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_{k+1}(y_{k+1} - C\hat{x}_{k+1|k}) \tag{6.24}$$

where K_{k+1} is the Kalman filter gain at time $k+1$, and is given by

$$K_{k+1} = \Pi_{k+1|k+1}C^TV^{-1} \tag{6.25}$$

From (6.21) we see that the uncertainty (represented by the variance of the state estimate) in stable states reduces with time, and that the uncertainty for unstable states increase. Similarly, the same equation tells us that the state excitation noise increases uncertainty. Equation (6.22) shows that the uncertainty is reduced by taking new measurements, but the reduction in uncertainty is small if the measurement noise is large. All this does of course agree with intuition.

Provided some technical assumptions are met (like detectability - all unstable states must show up in the measurements), the variances will converge to steady values, which may be found by setting $\Pi_{k+1|k} = \Pi_{k|k-1}$ and $\Pi_{k+1|k+1} = \Pi_{k|k}$. Equations (6.21, 6.22) then give

$$\Pi_{k+1|k} = A\Pi_{k+1|k}A^T + A\Pi_{k+1|k}C^T(V + C\Pi_{k+1|k}C^T)^{-1}C\Pi_{k+1|k}A^T + EWE^T \tag{6.26}$$

and the corresponding steady state value of $\Pi_{k|k}$ can be found from (6.22), and the steady state Kalman gain from (6.25).

Although it is natural to assume that the state estimates are more uncertain initially, it is quite common to ignore the transient behaviour described by (6.21, 6.22), and only use the steady state solution to the Kalman filter. Software for calculating the steady state Kalman filter is readily available, (6.26) is cumbersome and difficult to solve by hand for systems with more than one state.

Augmenting a disturbance description

In many applications, assuming disturbances (represented by the state excitation noise, w), to be a sequence of zero mean, normally distributed, independent impulses (a 'white noise' description) is a poor representation of how disturbances actually enter the system. Often, there is strong temporal correlation in how disturbances

affect system states and outputs. Such disturbances may be modelled by augmenting states representing the slow disturbance dynamics to the plant model. A good representation of disturbance dynamics is often a crucial element in achieving good closed-loop control performance.

A Kalman filter using an augmented disturbance description is often termed an Augmented Kalman Filter (AKF). In section 6.5.2 an example of a state space model, augmented with integrating states to represent disturbances both at the plant inlet and at the plant outlet, is shown in (6.89 - 6.91). When augmenting the model with integrating states, it is important that the augmented model is detectable. This point is further elaborated in section 6.5.2.

The Extended Kalman Filter

The Extended Kalman Filter (EKF) is an extension of the Kalman filter to non-linear systems. Although this extension seems quite natural and sensible, it is nevertheless somewhat ad hoc.

We start from a non-linear plant model, with additive measurement noise:

$$x_{k+1} = f(x_k, u_k, w_k) \tag{6.27}$$

$$y_k = h(x_k, u_k) + v_k \tag{6.28}$$

Equation (6.27) is used directly (assuming $w_k = 0$) to calculate $\hat{x}_{k+1|k}$. Similarly, (6.28) is used to calculate $\hat{y}_{k+1|k}$ (using $v_{k+1} = 0$, $\hat{x}_{k+1|k}$, and u_k)[5]. The value of $\hat{y}_{k+1|k}$ then enters instead of $C\hat{x}_{k+1|k}$ in (6.24). On the other hand, the propagation of estimate variances (6.21, 6.22) and calculation of the Kalman filter gain (6.25) are done with local linearizations of the nonlinear model. Thus, we use:

$$A_k = \left.\frac{\partial f}{\partial x}\right|_{w_k=0,\hat{x}_{k|k},u_k} \tag{6.29}$$

$$B_k = \left.\frac{\partial f}{\partial u}\right|_{w_k=0,\hat{x}_{k|k},u_k} \tag{6.30}$$

$$E_k = \left.\frac{\partial f}{\partial w}\right|_{w_k=0,\hat{x}_{k|k},u_k} \tag{6.31}$$

$$C_{k+1} = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k+1|k},u_k} \tag{6.32}$$

The EKF is commonly used for state estimation for nonlinear plants, and often performs well if the linearization is a fairly accurate approximation to the non-linear system over a single time step.

[5]Normally, the state estimate is updated *before* a new input is calculated, and therefore the input which is applied when the measurement y_{k+1} is obtained is actually u_k (assuming that a 'zero order hold' is used).

The Iterated Extended Kalman Filter

The Iterated Extended Kalman Filter (IEKF) is and attempt to enhance the ability of the EKF to handle non-linearity. We note that C_{k+1} in (6.32) is obtained by linearizing around the *a priori* state estimate $\hat{x}_{k+1|k}$. If the system is strongly non-linear, the resulting value of C_{k+1} may therefore be inaccurate, and a more accurate linearized measurement equation may be obtained by linearizing around the *a posteriori* estimate $\hat{x}_{k+1|k+1}$ - once that estimate is available. Further iterations would allow forther improvements in state estimation accuracy.

To present the IEKF, we will need a second subscript on several of the matrices in the EKF formulation, as well as the *a posteriori* state estimate. This second subscript represents the iteration number (at time $k+1$), with iteration number 0 representing the initial EKF calculations. Thus, from the initial EKF calculations we have $\hat{x}_{k+1|k,0} = h(\hat{x}_{k|k,N}, u_k)$ and $\Pi_{k+1|k}$, where N is the number of iterations of the IEKF at each timestep. For iteration i at time $k+1$ the IEKF calculations then proceed as follows:

$$
\begin{aligned}
C_{k+1,i} &= \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k+1|k+1,i-1},u_k} && (6.33)\\
K_{k+1|i} &= \Pi_{k+1|k} C_{k+1,i}^T \left(C_{k+1,i}\Pi_{k+1|k}C_{k+1,i}^T + V\right)^{-1} && (6.34)\\
\Pi_{k+1|k+1,i} &= (I - K_{k+1,i}C_{k+1,i})\Pi_{k+1|k} && (6.35)\\
\hat{x}_{k+1|k+1,i} &= \hat{x}_{k+1|k} && (6.36)\\
&+ K_{k+1,i}\left[y_{k+1} - h(\hat{x}_{k+1|k+1,i-1}, u_k) - C_{k+1,i}(\hat{x}_{k+1|k} - \hat{x}_{k+1|k+1,i-1})\right]
\end{aligned}
$$

The calculations proceed a predetermined number of iterations, or terminate when the change in state estimate between subsequent iterations is sufficiently small. At that point, after N iterations, one specifies

$$
\begin{aligned}
\hat{x}_{k+1|k+1} &= \hat{x}_{k+1|k+1,N}\\
\Pi_{k+1|k+1} &= \Pi_{k+1|k+1,N}
\end{aligned}
$$

which allows initiating the IEKF at time $k+2$ using the ordinary EKF. Although it is no general rule, it is often found that most of the improvement in the state estimate is achieved with a low number of iterations in the IEKF - often only one iteration after the EKF calculations.

6.4.3 Unscented Kalman filter

The Unscented Kalman Filter is a more recent modification to the Kalman filter, to better handle nonlinear models. The UKF avoids using a local linearization of the nonlinear model, but instead uses the model directly to propagate state estimates and (approximations of) probability distributions forward in time. Although not many industrial applications are reported, it seem that the UKF compares well with the more common EKF, in particular when the nonlinearities are pronounced. The presentation of the UKF in this note is based on Simon [87], who gives an accessible

introduction to both traditional state estimation and more recent developments in the area, and includes extensive references to the state estimation literature.

For simplicity of presentation, we assume that both the state excitation noise w and the measurement noise v enter the equations linearly, i.e.

$$x_{k+1} = f(x_k, u_k) + w_k \tag{6.37}$$
$$y_k = h(x_k) + v_k \tag{6.38}$$

The noises w and v are both assumed to be zero mean, normally distributed, with known covariances W and V, respectively. Let n denote the number of states in the model (the dimension of the state vector x).

The UKF is initialized with known (or assumed) initial values for the mean value of the state $x_{0|0}$ and the state covariance $\Pi_{0|0}$.

The UKF then proceeds as follows:

- *Propagate the mean state estimate from time $k-1$ to time k.* Instead of propagating the mean value $\hat{x}_{k-1|k-1}$ directly through the system dynamics, $2n$ perturbed state values are perturbed, to better capture how the system non-linearity affects the mean.

  1. Select the perturbed states as follows:

$$\hat{x}^{(i)}_{k-1} = \hat{x}_{k-1|k-1} + \tilde{x}^{(i)} \qquad i = 1, \ldots, 2n \tag{6.39}$$
$$\tilde{x}^{(i)} = \left(\sqrt{n\Pi_{k-1|k-1}}\right)^T_i \qquad i = 1, \ldots, n \tag{6.40}$$
$$\tilde{x}^{(n+i)} = -\left(\sqrt{n\Pi_{k-1|k-1}}\right)^T_i \qquad i = 1, \ldots, n \tag{6.41}$$

     where $\left(\sqrt{n\Pi}\right)_i$ denotes the i'th row of the matrix square root of $n\Pi$, defined such that $(\sqrt{n\Pi})^T(\sqrt{n\Pi}) = n\Pi$. The matrix square root may be calculated by the Matlab functions `sqrtm` or `chol`[6]. These perturbed state values $\hat{x}^{(i)}$ are often termed *sigma points*.

  2. Propagate each sigma point through the system dynamics:

$$\hat{x}^{(i)}_{k|k-1} = f(\hat{x}^{(i)}_{k-1}, u_{k-1}) \tag{6.42}$$

  3. Combine the points $\hat{x}^{(i)}_{k|k-1}$ to obtain the *a priori* state estimate:

$$x_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} \hat{x}^{(i)}_{k|k-1} \tag{6.43}$$

- *Calculate the a priori state covariance estimate:*

$$\Pi_{k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} \left(\hat{x}^{(i)}_{k|k-1} - x_{k|k-1}\right)\left(\hat{x}^{(i)}_{k|k-1} - x_{k|k-1}\right)^T + W \tag{6.44}$$

[6]The matrix square root is not uniquely defined (even for the positive definite covariance matrices considered here), and the two functions may therefore give different results. This is thought to be of little consequence here.

- *Implement the measurement equation.*

  1. Determine new sigma points around x_{k_k-1}:[7]

$$\begin{aligned} \hat{x}_k^{(i)} &= x_{k|k-1} + \tilde{x}^{(i)} \qquad i = 1, \ldots, 2n && (6.45)\\ \tilde{x}^{(i)} &= \left(\sqrt{n\Pi_{k|k-1}}\right)_i^T \qquad i = 1, \ldots, n && (6.46)\\ \tilde{x}^{(n+i)} &= -\left(\sqrt{n\Pi_{k|k-1}}\right)_i^T \qquad i = 1, \ldots, n && (6.47) \end{aligned}$$

  2. Pass each of the new sigma sigma points through the measurement equation:

$$\hat{y}_k^{(i)} = h(\hat{x}_k^{(i)}) \tag{6.48}$$

  3. Calculate the predicted measurement at time k:

$$\hat{y}_k = \frac{1}{2n}\sum_{i=1}^{2n} \hat{y}_k^{(i)} \tag{6.49}$$

- *Estimate the measurement covariance:*

$$\Pi_{y,k} = \frac{1}{2n}\sum_{i=1}^{2n} \left(\hat{y}_k^{(i)} - \hat{y}_k\right)\left(\hat{y}_k^{(i)} - \hat{y}_k\right)^T + V \tag{6.50}$$

- *Estimate the cross covariance between the state estimate $\hat{x}_{k|k-1}$ and the measurement estimate $\hat{y}_k$:*

$$\Pi_{xy,k} = \frac{1}{2n}\sum_{i=1}^{2n} \left(\hat{x}_{k|k-1}^{(i)} - x_{k|k-1}\right)\left(\hat{y}_k^{(i)} - \hat{y}_k\right)^T \tag{6.51}$$

The *a posteriori* state estimate and covariance are now obtained from

$$\begin{aligned} K_k &= P_{xy,k}P_{y,k}^{-1} && (6.52)\\ \hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k(y_k - \hat{y}_k) && (6.53)\\ \Pi_{k|k} &= \Pi_{k|k-1} - K_k\Pi_{y,k}K_k^T && (6.54) \end{aligned}$$

Remark: Note that the UKF applies also to time-varying systems. Both the system dynamics $f(\cdot)$, the measurement equation $h(\cdot)$ and the noise covariances W and V may be time varying (as long as they are known).

Many feedback systems are characterized by continuous-time system dynamics and discrete-time control and estimation. For such systems, so-called 'hybrid' EKFs have been developed, see, e.g., Simon [87]. Note that for the UKF the functions $f(\cdot)$ and $h(\cdot)$ need not be explicitly given as discrete-time functions - they may just as well result (implicitly) from the integration of ordinary differential equations. It is

[7] This step may be omitted, and the propagated sigma points $\hat{x}_{k|k-1}^{(i)}$ calculated above used instead, if reducing the computational load is essential.

therefore rather straight forward to apply the UKF also to continuous-time systems with discrete-time estimation and control.

In some cases the state excitation noise and the measurement noise may enter non-linearly, i.e., we have

$$x_{k+1} = f(x_k, u_k, w_k) \tag{6.55}$$
$$y_k = h(x_k, v_k) \tag{6.56}$$

In such cases, the state vector x can be augmented with the noise vectors, giving

$$x_k^a = \begin{bmatrix} x_k \\ w_k \\ v_k \end{bmatrix} \tag{6.57}$$
$$\hat{x}_{0|0}^a = \begin{bmatrix} \hat{x}_{0|0} \\ 0 \\ 0 \end{bmatrix} \tag{6.58}$$
$$\Pi_{0|0}^a = \begin{bmatrix} \Pi_{0|0} & 0 & 0 \\ 0 & W & 0 \\ 0 & 0 & V \end{bmatrix} \tag{6.59}$$

Thus the UKF procedure described above can be used. Note, however, that the state excitation noise w_k and the measurement noise v_k are now accounted for when calculating the sigma points. Therefore W should not be added when calculating the *a priori* covariance estimate nor should V be added when calculating the measurement covariance.

The IEKF and UKF are both modifications of the (E)KF for the purpose of improved handling of nonlinearity. Another such modification is the second-order EKF (see, e.g., [87]). This author is not aware of systematic comparisons of performance and computational requirements for these state estimation methods. Clearly, the UKF can be computationally rather demanding, if propagating the sigma points through (6.37) is demanding. This can occur, e.g., if $f(\cdot)$ in (6.37) results implicitly from the integration of a high order, stiff continuous-time model. However, for such problems, the rigorous propagation of the covariance matrix Π for a hybrid (continuous - discrete) EKF is also likely to be demanding.

6.4.4 Receding Horizon Estimation

Receding Horizon Estimation (RHE, a.k.a. Moving Horizon Estimation, MHE) is inspired by the success of MPC in control problems where constraints are important.

There are also many estimation problems where knowledge about the plant is easily formulated as constraints, and where such constraints will improve on the plant knowledge that is captured by the model alone. An opinion commonly held in academia seems to be that a sufficiently detailed plant model will capture all relevant constraints. Whereas this may be true (and often relatively straight forward to capture) in a simulation model, the way models are used in estimation may often destroy such model features. Two examples:

- The EKF uses a local linearization of the plant model, and the state update may easily result in infeasible state estimates.
- When propagating probability distributions for the UKF, the states are perturbed. These perturbed states may be infeasible.

There does exist approaches for ensuring feasible state estimates both for the EKF and the UKF, usually involving the 'projection' of the state estimate onto a feasible region of the state space. However, it may be better to embed the knowledge about what state estimates are possible directly into the state estimation. In such cases, RHE seems to be an obvious choice.

We assume as before that the state excitation noise and measurement noise are zero mean, independent and normally distributed, with covariances W and V, respectively. We also assume that an estimate $\hat{x}_0$ of the initial state is available, with a known covariance Π_0.

At time k, a natural formulation of the state estimation problem would then be to solve

$$\min_{\tilde{\mathbf{x}},\mathbf{w},\mathbf{v}} \quad (\hat{x_0} - \tilde{x_0})^T \Pi_0^{-1} (\hat{x_0} - \tilde{x_0}) + \sum_{i=1}^{k} v_i^T V^{-1} v_i + w_{i-1}^T W^{-1} w_{i-1} \Bigg) \tag{6.60}$$

subject to constraints

$$\begin{aligned}
\hat{x}_0 \qquad &\text{given} \\
y_i \qquad &\text{given}; \qquad i = 1, \cdots, k \\
u_i \qquad &\text{given}; \qquad i = 0, \cdots, k-1 \\
y_i \quad &= \quad C\tilde{x}_i + v_i; \qquad i = 1, \cdots, k \qquad (6.61) \\
\tilde{x}_{i+1} \quad &= \quad A\tilde{x}_i + Bu_i + Ew_i; \qquad i = 0, \cdots, k-1 \qquad (6.62) \\
X_L \leq \quad \tilde{x}_i \quad &\leq X_U; \qquad i = 0, \cdots, k \qquad (6.63)
\end{aligned}$$

Here

$$\begin{aligned}
\tilde{\mathbf{x}} &= \begin{bmatrix} \tilde{x}_0^T & \cdots & \tilde{x}_k^T \end{bmatrix}^T \\
\mathbf{w} &= \begin{bmatrix} w_0^T & \cdots & w_{k-1}^T \end{bmatrix}^T \\
\mathbf{v} &= \begin{bmatrix} v_1^T & \cdots & v_k^T \end{bmatrix}^T
\end{aligned} \tag{6.64}$$

One may also put constraints explicitly on w_i and v_i. Note, however, that when both w_i and v_i (or $\tilde{x}_i$ and v_i) are constrained, outliers in measurements, etc., may result in an infeasible optimization problem.

The optimization problem above is called a 'Full Information' problem, at each step in time it accounts for all the information that is available at that time. Converting the Full Information problem to a standard QP should not be difficult, following

the lines of what was done above for the MPC formulation. However, one problem is apparent: the size of the optimization problem grows without bounds as time progresses. The typical way of handling this problem, is to consider only a 'window' in the recent past in the optimization problem. The information available from the time before the start of the current window is accounted for by a weight on the given state estimate at the beginning of the window.

Below, the problem formulation for a fixed window length of N timesteps is presented. Following what is conventional in the literature, the time indices on the variables are changed to reflect 'standing at time $t = k$ and looking backwards in time', rather than 'standing at time $t = 0$ and looking forward in time'. This gives the following problem formulation:

$$\begin{aligned} \min_{\tilde{\mathbf{x}},\mathbf{w},\mathbf{v}} \quad & (\hat{x}_{k-N} - \tilde{x}_{k-N})^T \S (\hat{x}_{k-N} - \tilde{x}_{k-N}) \qquad (6.65) \\ + \quad & \sum_{i=1}^{N} \left(v_{k-N+i}^T V^{-1} v_{k-N+i} + w_{k-N-1+i}^T W^{-1} w_{k-N-1+i} \right) \end{aligned}$$

subject to constraints

$$\begin{aligned} \hat{x}_{k-N} &= \text{given} \\ y_{k-N+i} &= \text{given}; \quad i = 1, \cdots, N \\ u_{k-N+i} &= \text{given}; \quad i = 0, \cdots, N-1 \\ y_{k-N+i} &= C\tilde{x}_{k-N+i} + v_{k-N+i}; \quad i = 1, \cdots, N \qquad (6.66) \\ \tilde{x}_{k-N+i+1} &= A\tilde{x}_{k-N+i} + Bu_{k-N+i} + Ew_{k-N+i}; \quad i = 0, \cdots, N-1 \qquad (6.67) \\ X_L \leq \tilde{x}_{k-N+i} &\leq X_U; \quad i = 0, \cdots, N \qquad (6.68) \end{aligned}$$

Clearly the definitions of $\tilde{\mathbf{x}}$, $\mathbf{w}$ and $\mathbf{v}$ need to be modified:

$$\begin{aligned} \tilde{\mathbf{x}} &= \begin{bmatrix} \tilde{x}_{k-N}^T & \cdots & \tilde{x}_k^T \end{bmatrix}^T \qquad (6.69) \\ \mathbf{w} &= \begin{bmatrix} w_{k-N}^T & \cdots & w_{k-1}^T \end{bmatrix}^T \qquad (6.70) \\ \mathbf{v} &= \begin{bmatrix} v_{k-N+1}^T & \cdots & v_k^T \end{bmatrix}^T \qquad (6.71) \\ & \qquad (6.72) \end{aligned}$$

Note that

- The problem formulation above reflects the situation where, at each timestep, the state estimation is performed after receiving new measurements, before the MPC calculations are performed. Thus, the MPC calculations are performed with the *a posteriori* state estimate as a initial condition. To reduce computational delay before a new manipulate variable is available, one may instead in the MPC use the *a priori* state estimate as the initial condition - and at each timestep perform the MPC calculations before the state estimation. This may be particularly relevant for some nonlinear MPC problems, where the model

at each timestep is linearized around a predicted future state and input trajectory. Using the *a priori* state estimate in the MPC allows the linearization and subsequent problem formulation to be performed 'at the end of the previous timestep' rather than before solving the MPC optimization 'at the start of the new timestep'.

- In the problem formulation above, the effect of v_0 is assumed accounted for in the estimate $\hat{x}_0$, and v_0 therefore does not enter the optimization problem.
- Likewise, no effect of w_k can be observed before time $k+1$, and w_k therefore does not enter the optimization problem.
- In the MPC formulation, the state constraints represent undesirable operating conditions. In the estimation formulation, however, the state constraints represent *impossible* (or highly improbable) operating conditions - typically constraints such as *'the concentration of any chemical component cannot be negative'*. That is, the state constraints typically are not the same in MPC and RHE. If an operating condition is undesirable, it is important to get away from that operating condition as quickly as possible. Therefore, the RHE must be able to detect such an operating condition - and the state constraint introduced in the MPC to avoid the undesirable operating condition therefore should not be included in the RHE.

The arrival cost

In the RHE formulation above, the term $(\hat{x}_{k-N} - \tilde{x}_{k-N})^T \S (\hat{x}_{k-N} - \tilde{x}_{k-N})$ accounts for the information hat has been available about the system *before the start of the estimation window*. This term is often called the *arrival cost*. The ideal arrival cost would make the fixed window length problem in (6.65-6.68) identical to the Full Information problem in (6.60-6.63). In general, we are not able to determine such an arrival cost. The exception is the linear, unconstrained case, where the Kalman filter can provide us with the arrival cost. However, the arrival cost also depends on how information is passed between subsequent timesteps of the RHE, which will be further explained in the next two subsections.

The filtering formulation of RHE

In the filtering formulation of the RHE, we use the estimate

$$\hat{x}_{k-N} = \tilde{x}_{k-N|x-N} \tag{6.73}$$

That is, $\hat{x}_{k-N}$ is the (*a posteriori*) estimate obtained the first time the time instant $k-N$ was included in the estimation window, and is based only on information available at time $k-N$. With this formulation, we use

$$S = \Pi^{-1}_{k-N|k-N} \tag{6.74}$$

where $\Pi_{k-N|k-N}$ is the *a posteriori* estimate covariance at time $k-N$.

The smoothing formulation of RHE

In the smoothing formulation, we use instead the most recent estimate of x_{k-N}. Thus, at time k we use

$$\hat{x}_{k-N} = \tilde{x}_{k-N|k-1} \tag{6.75}$$

This means that the estimate $\hat{x}_{k-N}$ is 'smoothed' (and improved) using measurements obtained *after* time $k-N$. In this case, it has been shown (see, e.g., [79]), that the arrival cost should consist of two terms. The first term represents the uncertainty (covariance) of the estimate $\hat{x}_{k-N}$, represented by $\Pi_{k-N|k-1}$. The second term is added to prevent the information in $y_{k-N}, \cdots, y_{k-1}$ to be used twice (both in $\hat{x}_{k-N}$ and in the RHE calculations at time k).

To calculate the second term, we need the covariance of the estimate of the measurement sequence $Y_{N-1} = \left[\begin{array}{cccc} y_{k-N+1}^T & \cdots & y_{k-1}^T \end{array}\right]^T$, given x_{k-N}.

Manipulating the model equations, we get

$$\begin{aligned} Y_{N-1} &= \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N-2} \\ CA^{N-1} \end{bmatrix} x_{k-N} \\ &+ \begin{bmatrix} 0 & 0 & \cdots & 0 & CB \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & CB & \cdots & CA^{N-3}B \\ 0 & CB & \cdots & CA^{N-3}B & CA^{N-2}B \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-N+1} \\ u_{k-N} \end{bmatrix} \\ &+ \begin{bmatrix} 0 & 0 & \cdots & 0 & CE \\ 0 & 0 & \ddots & CE & CAE \\ \vdots & \ddots & \cdots & \ddots & \vdots \\ 0 & CE & \cdots & CA^{N-3}E & CA^{N-2}E \end{bmatrix} \begin{bmatrix} w_{k-1} \\ w_{k-2} \\ \vdots \\ w_{k-N+1} \\ w_{k-N} \end{bmatrix} \\ &+ \begin{bmatrix} 0 & I_{(N-1)\cdot n_y \times (N-1)\cdot n_y} \end{bmatrix} \begin{bmatrix} v_k \\ v_{k-1} \\ \vdots \\ v_{k-N+1} \end{bmatrix} \end{aligned} \tag{6.76}$$

Noting that Y_{N-1} is independent of v_k, this may be reformulated as

$$Y_{N-1} - \mathcal{O}_{N-1} x_{k-N} - \tilde{B}\mathbf{u} = \tilde{E}\mathbf{w} + \tilde{I}\mathbf{v}_{k-1} \tag{6.77}$$

where

$$\tilde{I} = \left[\begin{array}{cc} I_{(N-1)\cdot n_y \times (N-1)\cdot n_y} & 0_{(N-1)n_y \times \cdot n_y} \end{array}\right]; \quad \mathbf{v}_{k-1} = \begin{bmatrix} v_{k-1} \\ \vdots \\ v_{k-N} \end{bmatrix}$$

The variance of the left hand side of (6.77) (for a given x_{k-N}) and fixed $\mathbf{u}$ is therefore

$$S_2^{-1} = \tilde{E}\tilde{W}\tilde{E}^T + \tilde{I}\tilde{V}\tilde{I}^T \tag{6.78}$$

where $\tilde{W} = \mathrm{d}iag\{W\}$ and $\tilde{V} = \mathrm{d}iag\{V\}$.

The above expression corrects some minor mistakes in the expression in [79]. Next, we need to account for the fact that the inputs u, although known, depend on the noises w and v. To express this dependency, we need to account for feedback in both control and estimation. An explicit formulation of MPC and RHE would enable accounting for the constraints active at each timestep. However, the explicit solution often is not available, and the formulation would become both complex and time-varying. Instead, we will assume that a 'conventional' QP-based MPC formulation is in use, which when constraints are not active corresponds to (an easily computable) LQ-optimal controller K. Similarly, the state estimation will be represented by the steady-state Kalman filter gain L.

The plant model, together with the (unconstrained) control and estimation, then yields

$$\begin{aligned} x_{k+1|k+1} &= Ax_{k|k} + Bu_k + L(y_k - Cx_{k|k}) + w_k \\ &= (A+BK)x_{k|k} + Lv_k + w_k \end{aligned} \tag{6.79}$$

Starting from a given value of x_{k-N}, we then obtain

$$\begin{aligned} u_{k-N} &= Kx_{k-N} \\ u_{k-N+1} &= K(A+BK)x_{k-N} + KLv_k + Kw_k \\ u_{k-N+2} &= K(A+BK)^2x_{k-N} + K(A+BK)Lv_k + KLv_{k+1} + K(A+BK)w_k + Kw_{k+1} \\ u_{k-N+i} &= K(A+BK)^ix_{k-N} \\ &+ K\begin{bmatrix} I & (A+BK) & \cdots & (A+BK)^{i-1} \end{bmatrix} \begin{bmatrix} Lv_{k-N+i} + w_{k-N+i} \\ \vdots \\ Lv_{k-N} + w_{k-N} \end{bmatrix} \end{aligned} \tag{6.80}$$

Thus, we get

$$\begin{aligned} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-N} \end{bmatrix} &= \tilde{K} \begin{bmatrix} (A+BK)^{N-1} \\ (A+BK)^{N-2} \\ \vdots \\ (A+BK) \\ I \end{bmatrix} x_{k-N} \\ &+ \tilde{K} \begin{bmatrix} I & (A+BK) & \cdots & (A+BK)^{N-2} & (A+BK)^{N-1} \\ 0 & I & (A+BK) & \cdots & (A+BK)^{N-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & I & (A+BK) \\ 0 & 0 & \cdots & 0 & I \end{bmatrix} \begin{bmatrix} Lv_{k-1} + w_{k-1} \\ \vdots \\ Lv_{k-N} + w_{k-N} \end{bmatrix} \\ &= \tilde{K}A_K x_{k-N} + \tilde{K}B_K\mathbf{w} + \tilde{K}B_K\tilde{L}\mathbf{v}_{k-1} \end{aligned} \tag{6.81}$$

Substituting (6.81) into (6.77) we obtain

$$Y_{N-1} - \left(\mathcal{O}_{N-1} + \tilde{K}A_K\right) x_{k-N} = \left(\tilde{E} + \tilde{K}B_K\right) \mathbf{w} + \left(\tilde{I} + \tilde{K}B_K\tilde{L}\right) \mathbf{v}_{k-1} \tag{6.82}$$

This corresponds to the arrival cost

$$\begin{aligned} \Gamma(\tilde{x}_{k-N}) &= (\hat{x}_{k-N} - \tilde{x}_{k-N})^T S_1 (\hat{x}_{k-N} - \tilde{x}_{k-N}) \qquad (6.83) \\ &- \left(Y_{N-1} - \left(\mathcal{O}_{N-1} + \tilde{K}A_K\right)\tilde{x}_{k-N}\right)^T S_2 \left(Y_{N-1} - \left(\mathcal{O}_{N-1} + \tilde{K}A_K\right)\tilde{x}_{k-N}\right) \end{aligned}$$

with

$$S_1^{-1} = \Pi_{k-N|k-1} \tag{6.84}$$

$$S_2^{-1} = \left(\tilde{E} + \tilde{K}B_K\right)\tilde{W}\left(\tilde{E} + \tilde{K}B_K\right)^T + \left(\tilde{I} + \tilde{K}B_K\tilde{L}\right)\tilde{V}\left(\tilde{I} + \tilde{K}B_K\tilde{L}\right)^T \tag{6.85}$$

To obtain the smoothed covariance $\Pi_{k-N|k-1}$ we must first propagate the Kalman filter covariances *forward* in time to obtain $\Pi_{k-N+i|k-N+i}$ and $\Pi_{k-N+i|k-N+i-1}$. The smoothed covariance is then obtained by propagating *backwards* from $k-1$ to $k-N$ using the following relationships [1]:

$$\Pi_{T-i|T} = \Pi_{T-i|T-i} - Z_{T-i}(\Pi_{T-i+1|T-i} - \Pi_{T-i+1|T})Z_{T-i}^T \tag{6.86}$$

$$Z_{T-i} = \Pi_{T-i|T-i}A^T\Pi_{T-i+1|T-i}^{-1} \tag{6.87}$$

starting with $\Pi_{k-1|k-1}$ and $T = k-1$.

6.4.5 Concluding comments on state estimation

It is clearly impossible to cover all relevant formulations of state estimators in a chapter of this note. Other relevant and interesting estimator types include

- The second order EKF[87], mentioned briefly above.
- The particle filter [87]. This is essentially a Monte Carlo approach to state estimation, and may be particularly relevant for systems where the probability density function of the state estimate is multi-modal. For such systems it is clearly misleading to represent the state estimation accuracy using the state estimate covariance only.
- The Ensemble Kalman Filter (EnKF), [20, 21], a modification of the Kalman filter for applications to systems of very high order, such as meteorological models and petroleum reservoir models.

In addition, there is also a large area of observer design for deterministic systems.

Another area that has not been addressed, is the practical implementation of the state estimators, both for computational efficiency and robustness. For all of these topics, the reader is referred to more specialized literature. The book by Simon [87] is proposed as a good place to look for information and references to other works on many of these issues.

6.5 Disturbance handling and offset-free control

In most control applications, the ability to handle disturbances is important. In addition, differences between the model and the actual plant will lead to erroneous prediction, and hence to steady state offset.

Disturbances thet can be measured directly, and whose effect on the controlled variables are known, can be handled by feedforward, which is easily included in MPC. This is addressed briefly in the next subsection.

Unmeasured disturbances and plant-model mismatch require integral action for offset-free control at steady state. The simplest way of including integral action is to formulate the MPC in terms of the *changes* in manipulated variables, as described in Section 6.2, combined with a 'bias update'. Provided the actuation limits for the manipulated variables are included in the constraints, to avoid windup, this is a fairly straight forward way of achieving offset-free control.

The problem with this simple way of achieving offset-free control is that it can result in poor control performance. It implicitly assumes that the effects of disturbances is modelled well as steps in the measured output. In many applications, disturbances show dynamics over a significant timescale - typically the same timescale as for the manipulated variables. That is, disturbances often enter at the plant inputs rather than at the plant outputs. Good performance for MPC requires the effects of disturbances to be modelled well. For disturbances entering at the plant inputs, the simple way of introducing integral action described above will lead to poor performance in the face of disturbances. A more general way of ensuring offset-free control, which is able to handle disturbances entering both at the plant inputs and at the plant outputs, will be described below. This is based on [71], where a more complete description may be found.

6.5.1 Feedforward from measured disturbances

With MPC it is very simple to include feedforward from measured disturbances, provided one has a model of how the disturbances affect the states/outputs.

Feedforward is naturally used to counteract the future effects of disturbances on the controlled variables (it is too late to correct the present value). Thus, feedforward in MPC only requires that the effect on disturbances on the controlled variables are taken into account when predicting the future state trajectory in the absence of any control action. Feedforward from measured disturbances is included in the MPC formulation above, through the term $\hat{B}_d\delta$ in (6.10), where δ represents the *present* and *future* disturbances. If no other information is available, it is usually assumed that the future disturbances are equal to the present disturbance. Control performance will of course be affected by the accuracy of this assumption. In some cases information from upstream units, or knowledge of planned production changes, can provide better information about future disturbances.

The benefit obtained by using feedforward will (as always) depend on what bandwidth limitations there are in the system for feedback control. Furthermore, effective

feedforward requires both the disturbance and process model to be reasonably accurate.

6.5.2 Disturbance estimation and offset-free control

If offset-free control is desired, it is necessary to account for differences between the model and the actual plant. This can be done by estimating *unmeasured* disturbances affecting the plant. The 'bias update' is a simple way of doing this, but it is often desired to be able to account for more general disturbance dynamics. This is done by augmenting the plant model with additional states $d_{i,k}$ representing disturbances entering at the plant inputs, and $d_{o,k}$ representing disturbances entering at the plant output. Thus, the augmented state space model becomes

$$\begin{aligned} \tilde{x}_{k+1} &= \tilde{A}_k\tilde{x}_k + Bu_k + Ed_k \\ y_k &= \tilde{C}\tilde{x}_k + Fd_k \end{aligned} \tag{6.88}$$

Here d_k represent *measured* disturbances, whereas the estimated disturbances are included in the augmented state vector $\tilde{x}$. The augmented state vector and the correspondingly modified state space matrices are given by

$$\tilde{x} = \begin{bmatrix} x \\ d_i \\ d_o \end{bmatrix} \tag{6.89}$$

$$\tilde{A} = \begin{bmatrix} A & E_i & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \tag{6.90}$$

$$\tilde{C} = \begin{bmatrix} C & 0 & C_{do} \end{bmatrix} \tag{6.91}$$

This model can be used for state estimation, using, e.g., a Kalman filter or Receding Horizon Estimation. Muske and Badgwell [71] show that it is always possible to choose E_i and C_{do} such that the augmented state space model is detectable, provided

- the original model $\{C, A\}$ is detectable, and
- the number of estimated disturbance states (the sum of the number of elements in d_i and d_o) is no larger than the number of independent measurements used for estimation.

Naturally, the matrices E_i and C_{do}, as well as the dimensions of d_i and d_o, should be chosen to reflect the observed disturbance dynamics as well as possible. However, unfortunate choices for E_i and C_{do} may make the augmented model undetectable.

If $\{C, A\}$ is detectable, detectability of the overall system is determined by

$$\mathrm{R}ank \begin{bmatrix} (I - A) & -E_i & 0 \\ C & 0 & C_{do} \end{bmatrix}$$

which should equal the number of states in the *augmented* model. From this is derived a few conditions for detectability of the augmented system:

- The augmented system $\{\tilde{C}, \tilde{A}\}$ is not detectable if E_i and/or C_{do} are not full column rank.
- The augmented system $\{\tilde{C}, \tilde{A}\}$ is not detectable if the number of disturbance states exceeds the number of linearly independent outputs.
- The augmented system $\{\tilde{C}, \tilde{A}\}$ is not detectable if the range of E_i contains an unobservable mode of $\{C, A\}$.
- The augmented system $\{\tilde{C}, \tilde{A}\}$ is not detectable if the range of C_{do} contains the output space spanned by an integrating mode of A.

If a detectable augmented state-space model is used for estimation, the estimated input disturbances d_i can be used just like measured disturbances in the MPC. That is, the following state space equation should be used

$$x_{k+1} = Ax_k + Bu_k + \begin{bmatrix} E & E_i \end{bmatrix} \begin{bmatrix} d_k \\ d_{i,k} \end{bmatrix} \tag{6.92}$$

In addition, one must ensure that the state references x_{ref} and manipulated variable references u_{ref} are consistent at steady state with the steady state (measured and estimated) disturbances and the input and output targets specified by higher levels of the operational hierarchy. This is further addressed in the section on *Target calculation* below. If the references are consistent at steady state, and the system is stable in closed loop, the disturbance estimation scheme described above will result in offset-free control at steady state.

6.6 Feasibility and constraint handling

For any type of controller to be acceptable, it must be very reliable. For MPC controllers, there is a special type of problem with regards to *feasibility* of the constraints. An optimization problem is *infeasible* if there exists no exists no set of values for the free variables in the optimization for which all constraints are fulfilled. Problems with infeasibility may occur when using MPC controllers, for instance if the operating point is close to a constraint, and a large disturbance occurs. In such cases, it need not be possible to fulfill the constraint at all times. During startup of MPC controllers, one may also be far from the desired operating point, and in violation of some constraints. Naturally, it is important that the MPC controller should not 'give up' and terminate when faced with an infeasible optimization problem. Rather, it is desirable that the performance degradation is predictable and gradual as the constraint violations increase, and that the MPC controller should effectively move the process into an operating region where all constraints are feasible.

If the constraints are *inconsistent*, i.e., if there exists no operating point where the MPC optimization problem is feasible, then the problem formulation in meaningless, and the problem formulation has to be modified. Physical understanding of the process is usually sufficient to ensure that the constraints are consistent. A simple

example of an inconsistent set of constraints is if the value of the minimum value constraint for a variable is higher than the value of the maximum value constraint.

Usually, the constraints on the inputs (manipulated variables) result from true, physical constraints that cannot be violated. For example, a valve cannot be more than 100% open. On the other hand, constraints on the states/outputs often represent operational desireables rather than fundamental operational constraints. State/output constraints may therefore often be violated for short periods of time (although possibly at the cost of producing off-spec products or increasing the need for maintenance). It is therefore common to modify the MPC optimization problem in such a way that output constraints may be violated if necessary. There are (at least) three approaches to doing this modification:

1. Remove the state/output constraints for a time interval in the near future. This is simple, but may allow for unnecessarily large constraint violations. Furthermore, it need not be simple to determine for how long a time interval the state/output constraints need to be removed - this may depend on the operating point, the input constraints, and the assumed maximum magnitude of the disturbances.

2. To solve a separate optimization problem prior to the main optimization in the MPC calculations. This initial optimization minimizes some measure of how much the output/state constraints need to be moved in order to produce a feasible optimization problem. The initial optimization problem is usually a LP problem, which can be solved very efficiently.

3. Introducing *penalty functions* in the optimization problem. This involves modifying the constraints by introducing additional variables such that the constraints are always feasible for sufficiently large values for the additional variables. Such modified constraints are termed *soft constraint*s. At the same time, the objective function is modified, by introducing a term that penalizes the magnitude of the constraint violations. The additional variables introduced to ensure feasibility of the constraints then become additional free variables in the optimization. Thus, feasibility is ensured by increasing the size of the optimization problem.

The two latter approaches are both rigorous ways of handling the feasibility problem. Approach 3 has a lot of flexibility in the design of the penalty function. One may ensure that the constraints are violated according to a strict list of priorites, i.e., that a given constraint will only be violated when it is impossible to obtain feasibility by increasing the constraint violations for less important constraints. Alternatively, one may distribute the constraint violations among several constraints. Although several different penalty functions may be used, depending on how the magnitude of the constraint violations are measured, two properties are desireable:

- That the QP problem in the optimization problem can still be solved efficiently. This implies that the Hessian matrix for the modified problem should be positive

definite, i.e., that there should be some cost on the *square* of the magnitude of the constraint violations.

- That the penalty functions are *exact*, which means that no constraint violations are allowed if the original problem is feasible. This is usually obtained by putting a sufficiently large weight on the magnitude of the constraint violations (i.e., the linear term) in the objective function.

The use of penalty functions is described in standard textbooks on optimization (e.g. [23]), and is discussed in the context of MPC in e.g. [17, 85, 45].

Feasibility at steady state is discussed in more detail in the section on 'Target calculation' below. The techniques used there closely resemble those that are applied to the dynamic optimization problem in MPC, with the simplification that only steady state is addressed i.e., there is no prediction horizon involved and the variation in constraint violations over the prediction horizon is not an issue. Thus, only the techniques of points 2 and 3 above are relevant for target calculation.

In addition to the problem with feasibility, hard output constraints may also destabilize an otherwise stable system controlled by an MPC controller, see [104]. Although this phenomenon probably is quite rare, it can easily be removed by using a soft constraint formulation for the output constraints [17]. The following section will discuss closed loop stability with MPC controllers in a more general context.

6.7 Closed loop stability with MPC controllers

The objective function in Eq. (6.5) closely resembles that of discrete-time Linear Quadratic (LQ) - optimal control. For stabilizable and detectable[8] systems, infinite horizon LQ-optimal control is known to result in a stable closed loop system. Note that the requirement for detectability does not only imply that unstable modes must be detectable from the physical measurements (i.e., that (C, A) is detectable), but also that the unstable modes must affect the objective function, i.e., $(Q^{1/2}, A)$ must be detectable.

With the stabilizability and detectability requirements fulfilled, a *finite horizon* LQ-optimal controller is stable provided the weight on the 'terminal state', S, is sufficiently large. How large S needs to be is not immediately obvious, but it is quite straight forward to calculate an S that is sufficiently large. In the MPC context, this can be done by designing a stabilizing state feedback controller K , and then calculate the S that gives the same contribution to the objective function that would be obtained by using the controller K, and summing the terms $(x_i - x_{ref,n})^T Q(x_i - x_{ref,n})$ from $i = n$ to infinity. Since the controller K results in an asymptotically

[8]Stabilizability is a weaker requirement than the traditional state controllability requirement, since a system is stabilizable if and only if all unstable modes are controllable, i.e., a system can be stabilizable even if some stable modes are uncontrollable. Similarly, a system is detectable if all unstable modes are observable.

stable system, this sum is finite, and hence S is finite. The value of S can be obtained by solving a discrete Lyapunov equation

$$S - (A + BK)^T S(A + BK) = Q$$

Note that if one chooses to use the infinite horizon LQ-optimal controller, solving the Riccati equation gives both the controller K and the terminal state weight S:

$$S = A^T SA + Q - A^T SB(P + B^T SB)^{-1} B^T SA$$

and the corresponding controller is given by

$$K = -(B^T SB + P)^{-1} B^T SA$$

Whereas the solution from teh discrete Lyapunov equation sums only the 'state cost' over the infinite time horizon, the solution from the Riccati equation accounts for both the state and manipulated variable costs over the infinite horizon. The Riccati equation solution for S is thus preferable.

With a sufficiently large S, obtained as described above, the remaining requirement for obtaining closed loop stability is that constraints that are feasible over the horizon $n \leq i \leq n + j$ will remain feasible over an infinite horizon (assuming no new disturbances enter). How to find a sufficiently large j has been described above.

The above results on how to find values for S and j to guarantee stability, are not very useful if, e.g., a step response model is used, since the values of the states are then unavailable. Step response-based MPC controllers therefore do not have a terminal state weight S, but rather extend the prediction of the outputs further into the future than the time horizon over which the inputs are optimized (corresponding to $n_p > n_u$ in the comments following Eq. (6.6). Although a sufficiently large prediction horizon n_p compared to the "input horizon" n_u will result in a stable closed loop system (the open loop system is assumed asymptotically stable, since a step response model is used), there is no known way of calculating the required n_p. Tuning of step-response based MPC controllers therefore typically rely heavily on simulation. Nevertheless, the industrial success of step response-based MPC controllers show that controller tuning is not a major obstacle in implementations.

6.8 Target calculation

It is common for MPC controllers perform a 'target calculation' prior to the main optimization described above. The purpose of this target calculation is to determine consistent steady-state values for the state references $x_{ref,\infty}$ and input references $u_{ref,\infty}$. Most MPC implementation have infrequently changing setpoints, and will use reference values that are constant throughout the prediction horizon, i.e. $x_{ref,i} = x_{ref,\infty} \forall i$ and $u_{ref,i} = u_{ref,\infty} \forall i$. This covers industrial practice in the majority of installations, but will not be applicable to some problems, e.g. batch processes or cyclically operated plants. We will use a linear plant model, which is also common

industrial practice. Extending the following to non-linear plant models should in principle not be difficult for the competent reader. However, performing the target calculation at each timestep means that one should be concerned with being able to do the calculations quickly and reliably, and using linear models makes it much simpler to ascertain that will actually the case.

One prerequisite for offset-free control is that the minimum value of the objective function is at the desired references, and to ensure that one desires that

$$(I - A)x_{ref,\infty} = Bu_{ref,\infty} + \tilde{E}\tilde{d}_\infty \quad (6.93)$$

$$y_{ref} = Cx_{ref,\infty} + \tilde{F}\tilde{d}_\infty \quad (6.94)$$

Here $y_{ref,\infty}$ is the desired steady state value of some variables, the desired values of which are determined by higher levels in the operational hierarchy[9].

The disturbance variable vector $\tilde{d}_\infty$ is the expected/predicted/estimated steady state value of *all* disturbances affecting the process, i.e., it should contain the steady state values of measured disturbances d, estimated input disturbances d_i, and estimated output disturbances d_o. Thus,

$$\begin{aligned} \tilde{d}_\infty &= \begin{bmatrix} d_\infty \\ d_{i,\infty} \\ d_{o,\infty} \end{bmatrix} \\ \tilde{E} &= \begin{bmatrix} E & E_i & 0 \end{bmatrix} \\ \tilde{F} &= \begin{bmatrix} F & 0 & C_{do} \end{bmatrix} \end{aligned}$$

In the (rare) unconstrained case, and with as many inputs u as controlled outputs y, the state and input targets can be found from a simple matrix inversion

$$\begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \end{bmatrix} = \begin{bmatrix} -(I-A) & B \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} 0 & -\tilde{E} \\ I & -\tilde{F} \end{bmatrix} \begin{bmatrix} y_{ref,\infty} \\ \tilde{d}_\infty \end{bmatrix} \quad (6.95)$$

$$= M^{-1} \begin{bmatrix} 0 & -\tilde{E} \\ I & -\tilde{F} \end{bmatrix} \begin{bmatrix} y_{ref,\infty} \\ \tilde{d}_\infty \end{bmatrix} \quad (6.96)$$

Clearly, for the targets $x_{ref,\infty}$ and $u_{ref,\infty}$ to be well defined, the matrix M above needs to be of full rank. Many factors may make it impossible to obtain the targets by the simple calculations above:

- There may be more inputs than outputs.

[9] In general, the higher levels of the operational hierarchy may specify targets in terms of different measurements than those that are used for control/estimation at the supervisory control level. In such cases, the relationships between the variables used for supervisory control (including estimated output disturbances) and the variables for which targets are specified, will need to be modelled.

- There may be more controlled variables than inputs.
- In addition to desired values for the controlled variables, one may wish to keep the inputs close to specific values.
- Achieving the desired values for the controlled variables may be impossible (or otherwise unacceptable) due to constraints.

When such problems of concern (and if they are not, there is probably little reason to use MPC in the first place), the target calculations are performed by solving an optimization problem or a series of such problems. In the following, we will use the subscript $_d$ to denote *desired* values of controlled variables y and inputs u, whereas the subscript $_{ref}$ will still refer to the reference values or targets used in the MPC calculations. The desired values are set by operators or higher level plant optimization, whereas the MPC targets are the result of the target calculation.

The most straight forward formulation will cast the target calculation as a QP problem:

$$\min_{x_{ref,\infty},u_{ref,\infty}} \quad \left(y_d - Cx_{ref,\infty} - \tilde{F}\tilde{d}_\infty\right)^T Q \left(y_d - Cx_{ref,\infty} - \tilde{F}\tilde{d}_\infty\right) \tag{6.97}$$

$$+(u_d - u_{ref,\infty})^T W (u_d - u_{ref,\infty}) \tag{6.98}$$

subject to given values for y_d, u_d and d_∞ , the model equations Eq. (6.93) and the relevant maximum and minimum value constraints on $x_{ref,\infty}$ and $u_{ref,\infty}$. The matrix Q is assumed to be positive definite. A positive definite W will in general result in offset in the controlled variables even in cases when the desired values $\widehat{y}_d$ can be achieved. The matrix W may therefore be chosen to be positive semi-definite. Muske [70] shows how to specify a semi-definite W which does not introduce offset in the controlled variables. Note, however, that when there are more inputs than controlled variables, the number of inputs whithout any weight in the optimization problem must not exceed the number of controlled variables. Also, in many cases there may be reasons for keeping the inputs close to a specified value, and in such cases the inputs concerned should be given a weight in the optimization problem above. Ideally, the target values should comply with the same maximum and minimum value constraints as that of the MPC problem, c.f. Eq. (6.6), but there may also be other constraints. Let us assume that all such constraints can be described by the inequality

$$\widehat{H} \begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \end{bmatrix} \geq \widehat{b} \tag{6.99}$$

Difficulties will arise whenever there is no feasible region in which the constraints of Eq. (6.93) and Eq. (6.99) cannot all be fulfilled. This is indeed often the case when operating in a highly constrained region (which is the major advantage of MPC), but may also result from operators specifying overly stringent constraints. For *any* sort of control to be feasible in such a case, it becomes necessary to relax some of

the constraints. It should be obvious that the process model Eq. (6.93) cannot be relaxed, since it is given by the physics of the problem at hand. Likewise, most input constraints are hard constraints that cannot be relaxed, such as actuator limitations. On the other hand, many state or output constraints represent operational desirables rather than physical necessities, and violation of such constraints may be possible without putting the safety of the plant in jeopardy. Allowing violations in selected constraints can be achieved by introducing additional variables into the optimisation problem. Thus, instead of Eq. (6.97) we get

$$\min_{x_{ref,\infty},u_{ref,\infty},p} \quad (y_d - Cx_{ref,\infty} - \tilde{F}\tilde{d}_\infty)^T Q(y_d - Cx_{ref,\infty} - \tilde{F}\tilde{d}_\infty) \tag{6.100}$$

$$+(u_d - u_{ref,\infty})^T W(u_d - u_{ref,\infty}) + l^T p + p^T Z p \tag{6.101}$$

where l is a vector of positive constraint violation costs and Z is positive definite. The vector p gives the magnitude of the constraint violations. The model equations in Eq. (6.93) are assumed to hold as before, whereas the constraints in Eq. (6.99) are modified to

$$\begin{aligned} \widehat{H}\begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \end{bmatrix} + \widehat{L}p &\geq \widehat{b} \\ p &\geq 0 \end{aligned} \tag{6.102}$$

The matrix $\widehat{L}$ determines which constraints are relaxed. Its elements will take the values 0 or 1, with exactly one element equal to 1 for each column, and at most one element equal to 1 for each row. If a row of $\widehat{L}$ contains an element equal to 1, this means that the corresponding constraint may be relaxed.

For a sufficiently large l, the optimal solution to Eq. (6.100) is also the optimal solution to Eq. (6.97), provided a feasible solution for Eq. (6.97) exists.

The target calculation formulation in Eqs. (6.100 - 6.102) will distribute the constraint violations between the different relaxable constraints. If one instead wishes to enforce a strict priority among the constraints, so that a given constraint is violated only if feasibility cannot be achieved even with arbitrarily large constraint violations in the less important constraints, this may be achieved by solving a series of LP problems [10], followed by a QP problem for the target calculation. The following algorithm may be used:

1. Simple inspection at the design stage will often ensure that the non-relaxable constraints are always feasible. If not, it may be necessary to check that

[10] A series of QP problems may sometimes be preferable, if one wishes to distribute constraint violations between several constraints of the same importance. Using a QP formulation only affects the criterion functions of the following optimization problems, not the constraints.

there exists a feasible solution to the problem when only considering the non-relaxable constraints. Set $\widehat{H}_r$ to the rows of $\widehat{H}$ corresponding to the non-relaxable constraints, and $\widehat{b}_r$ to the corresponding elements of $\widehat{b}$. Set c_r to $\begin{bmatrix} 0 & 0 & 1 & \cdots & 1 \end{bmatrix}^T$,where the leading zeros should be interpreted as zero vectors of dimensions corresponding to the dimensions of the state and input vectors, respectively. Solve the LP problem

$$\min_{x_{ref,\infty},u_{ref,\infty},p} c_r^T \begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \\ p \end{bmatrix}$$

subject to the constraints

$$\begin{bmatrix} \widehat{H}_r & I \end{bmatrix} \begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \\ p \end{bmatrix} \geq \widehat{b}_r$$
$$p \geq 0$$

If the optimal value for this LP problem is larger than 0, the non-relaxable constraints are infeasible, which would indicate serious mistakes in the constraint specifications or abnormally large disturbances (the latter of which could affect $\widehat{b}_r$). Proceed to the next step in the algorithm if the non-relaxable constraints are feasible, if not, there is reason to activate an alarm to get operator attention.

2. Add the most important of the remaining relaxable constraints and find the minimum constraint violation in that constraint only which results in a feasible solution. This is done by adding the corresponding row of $\widehat{H}$ and $\widehat{b}$ to $\widehat{H}_r$ and $\widehat{b}_r$, respectively, using a scalar 'dummy variable' p, and setting c_r to $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$The zeros in c_r are still zero vectors of appropriate dimension, whereas the 1 is scalar. The LP problem to solve at this stage becomes

$$\min_{x_{ref,\infty},u_{ref,\infty},p} c_r^T \begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \\ p \end{bmatrix}$$

subject to the constraints

$$\begin{bmatrix} & & 0 \\ \widehat{H}_r & & \vdots \\ & & 0 \\ & & 1 \end{bmatrix} \begin{bmatrix} x_{ref,\infty} \\ u_{ref,\infty} \\ p \end{bmatrix} \geq \widehat{b}_r$$
$$p \geq 0$$

3. Move the contribution of the dummy variable p into $\widehat{b}_r$. That is, set $\widehat{b}_r \leftarrow \widehat{b}_r + \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix}^T p$. If there are more relaxable constraints, go to point 2 above.

4. When all constraints are accounted for, and a feasible solution is known to exist, solve the QP problem for target calculation with modified constraints.

Instead of solving a series of LP problems, the solution may be found by solving a single LP problem [103]. However, the required LP problem is quite complex to design. Although this design problem is solved off-line, it will need to be modified whenever the constraint specifications change. At the time of writing, no reliable software is known to exist for solving this LP design problem.

6.9 Robustness of MPC controllers

The main advantage of MPC controllers lie in their ability to handle constraints. On the other hand, they may be sensitive to errors in the process model. There have been tales about processes which are controlled by MPC controllers when prices are high (and it is important to operate close to the process' maximum throughput), but are controlled by simple single-loop controller when prices are low (and production is lower, leading to no active constraints). The potential robustness problems are most easily understood for cases when no constraints are active, i.e., when we can study the objective function in Eq. (6.1) with H and c from Eq. (6.14). We then want to minimize

$$f(v) = 0.5v^T(\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P})v + \chi_{dev}^T\widehat{A}^T\widehat{Q}\widehat{B}v$$

with respect to v. The solution to this minimization can be found analytically, since no constraints are assumed to be active. We get[11]

$$v = -(\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P})^{-1}\widehat{B}^T\widehat{Q}\widehat{A}\chi_{dev}$$

Clearly, if the model contains errors, this will result in errors in $\widehat{B}$ and $\widehat{A}$, and hence the calculated trajectory of input moves, v, will be different from what is obtained with a perfect model. If the Hessian matrix $\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P}$ is *ill-conditioned*[12], the problem is particularly severe, since a small error in the Hessian can then result in a large error in its inverse. For a physical motivation for problems with ill-conditioning consider the following scenario:

- The controller detect an offset from the reference in a direction for which the process gain is low.

[11] Note that $\widehat{Q} = \widehat{Q}^T$, and that the assumptions on Q, S and P ensures that $(\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P})$ is of full rank, and hence invertible.

[12] A matrix is ill-conditioned if the ratio of the largest singular value to the smallest singular value is large. This ratio is called the *condition number*.

- To remove this offset, the controller calculates that a large process input is needed in the low gain input direction.
- Due to the model errors, this large input actually slightly "misses" the low gain input direction of the true process.
- The fraction of the input that misses the low gain direction, will instead excite some high gain direction of the process, causing a large change in the corresponding output direction.

Now, there are two ways of reducing the condition number of $\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P}$:

1. Scaling inputs and states in the process model, thereby changing $\widehat{B}$.
2. Modifying the tuning matrices $\widehat{Q}$ and $\widehat{P}$.

Scaling inputs and states (or outputs, if the objective function uses outputs instead of states) is essentially the same as changing the units in which we measure these variables. In some cases this sufficient, but some processes have inherent ill-conditioning that cannot be removed by scaling.

In theory, one may use non-zero values for all elements in the tuning matrices $\widehat{Q}$ and $\widehat{P}$, with the only restriction that $\widehat{Q}$ should be positive semi-definite[13] and $\widehat{P}$ should be positive definite (and hence both should be symmetric). However, little is known on how to make full use of this freedom in designing $\widehat{Q}$ and $\widehat{P}$, and in practice they are obtained from Q ,P and S as shown in Eq. (6.7), and typically Q and P are diagonal. It is common to try to reduce the ill-conditioning of the Hessian matrix by multiplying all elements of $\widehat{P}$ by the same factor. If this factor is sufficiently large, the condition number of the Hessian matrix will approach that of P - which can be chosen to have condition number 1 if desired. However, increasing all elements of $\widehat{P}$ means that the control will become slower in all output directions, also in directions which are not particularly sensitive to model uncertainty.

If the above ways of reducing the condition number of the Hessian matrix are insufficient or unacceptable, one may instead modify the process model such that the controller "does not see" offsets in the low gain directions. Inherent ill-conditioning (which cannot be removed by scaling) is typically caused by physical phenomena which make it difficult to change the outputs in the low gain direction. Fortunately, this means that disturbances will also often have a low gain in the same output direction. It may therefore be acceptable to ignore control offsets in the low gain output directions. In terms of the MPC formulation above, the controller can be forced to ignore the low gain directions by modifying $\widehat{B}$ by setting the small singular values of $\widehat{B}$ to zero. This is known as *singular value tresholding*, since we remove all singular values of $\widehat{B}$ that is smaller than some treshold. If we term this modified matrix $\widehat{B}$ for $\widehat{B}_m$, we find that the trajectory of input moves calculated by the (unconstrained) MPC optimization now becomes

[13]The lower right diagonal block of $\widehat{Q}$, corresponding to the terminal state weight S, should be strictly positive definite (and sufficiently large).

$$v = -(\widehat{B}_m^T \widehat{Q} \widehat{B}_m + \widehat{P})^{-1} \widehat{B}_m^T \widehat{Q} \widehat{A} \chi_{dev} = -(\widehat{B}_m^T \widehat{Q} \widehat{B}_m + \widehat{P})^{-1} \chi_m$$

Note that the conditioning of the Hessian matrix is not improved by setting the small singular values of $\widehat{B}$ to zero, but the vector χ_m does not show any control offset in the corresponding output directions, and hence the vector v will contain no input moves in the corresponding input directions.

Singular value tresholding is effective in improving robustness to model errors, but it clearly causes nominal control performance (the performance one would get if the model is perfect) to deteriorate, since the controller ignores control offsets in some output directions. Removing too many singular values from $\widehat{B}$ will result in unacceptable control performance.

6.10 Using rigorous process models in MPC

Most chemical processes are inherently nonlinear. In some cases, rigorous dynamical models based on physical and chemical relationships are available, and the process engineers may wish to use such a model in an MPC controller. This would for instance have the advantage of automatically updating the model when the process is moved from one operating point to another.

However, to optimize directly on the rigorous model is not straight forward. The non-linearity of the model typically results in optimization problems that are non-convex. Optimization of non-convex problems is typically *a lot* more time consuming than optimization of convex problems and the time required to find a solution can vary dramatically with changing operating point or initial states. This means that direct optimization of non-linear models is usually ill-suited for online applications like MPC. Furthermore, it is often the case that the most important 'non-linearities' in the true system are the constraints, which are handled effectively by MPC.

This does not mean that rigorous models cannot be utilized by MPC controllers, but it means that one can make only partial use of such models. The idea is to utilize these models to the extent that the available time permits. One may then approximate the true optimization problem by a modified, convex problem, or a series of such problems.

Predict using the rigorous model. The simplest way of (partially) accounting for non-linearity in the process model, is to calculate the deviation from the desired state (or output) trajectory from a rigorous, non-linear model, whereas the other parts of the optimization formulation uses a linearized model. In this way, the calculated input trajectory v will to some extent account for the non-linearities.

Line search If greater accuracy is needed, one may do a line search using the non-linear model to optimize what multiple of v should be implemented, i.e., perform a search to optimize (while taking the constraints into account)

$$\min_{\alpha} f(x,u) = \min_{\alpha} f(x_0, u_{ref} + \alpha v) \tag{6.103}$$

where α is a positive real scalar. Such line searches are a standard part of most non-linear optimization methods, and are covered in many textbooks on optimization e.g. in [23]. When performing the minimization in Eq. (6.103) above, the full non-linear model is used to calculate future states from $(x_0, u_{ref} + \alpha v)$.

Iterative optimization. Even with the optimal value of α, one probably has not found the optimal solution to the original non-linear optimization problem. Still better solutions may be found by an iterative procedure, where the predicted deviation from the desired state trajectory x_{ref} is found using the best available estimate of the future input trajectory. That is, for iteration number k, use the model to calculate the resulting vector $\chi_{dev,k}$ when the input trajectory $u_{ref} + v_t$ is applied, where $v_t = \sum_{l=0}^{k-1} v_l$, and minimize

$$\min_{v_k} f(v) = (v_t + v_k)^T (\widehat{B}^T \widehat{Q} \widehat{B} + \widehat{P})(v_t + v_k) + \chi_{dev,k}^T \widehat{A}^T \widehat{Q} \widehat{B}(v_t + v_k)$$

subject to constraints that should be modified similarly. It is also assumed that a line search is performed between each iteration. The iterations are initialized by setting $v_0 = 0$, and are performed until the optimization converges, or until the available time for calculations is used up. The iterative procedure outlined above need not converge to a globally optimal solution for the original problem, it may end up in a local minimum. Furthermore, there is no guarantee that this is a particularily efficient way of solving the original optimization problem (in terms of the non-linear model). It does, however, have the advantage of quickly finding reasonable, and hopefully feasible, input sequences. Even if the optimization has to terminate before the optimization has converged, a 'good' input has been calculated and is available for implementation on the process.

Linearize around a trajectory. If the operating conditions change significantly over the time horizon (n) in the MPC controller, the linearized model may be a reasonable approximation to the true process behaviour for only a part of the time horizon. This problem is relatively rare when constant reference values are used, but may be relevant when moving from one operating point to another. It is then possible to linearize the process around the predicted process trajectory $(x_{ref} + \chi_{dev})$ rather than around a constant state. One then gets a time-varying (but still linear) model, i.e., a "new model" for each time interval into the future. Conceptually, linearizing around a trajectory does not add much complexity compared to linearizing around a constant state, but it does add significantly to the notational complexity that is necessary in the mathematical formulation of the optimization problem. Furthermore, analytical representations of the linearized models are typically not available, and the linearization has to be performed by numerically perturbing the process around the predicted process trajectory. This can clearly add significantly to the computational

burden. Linearizing around a trajectory can be combined with iterative optimization as outlined above - which would further add to the computational burden.

Chapter 7

Some practical issues in controller implementation

This short chapter will address a few practical issues in controller implementation that can be crucial for achieving good control performance. For an experienced control engineer at a production plant, the issues discussed here may be trivial and self-evident. However, after having seen trivial mistakes in controller implementation leading to seriously reduced performance or controller malfunction, and having heard tales of many more cases of the same, it appears necessary to address these issues.

Suggestions and motivation for extending the list of issues are welcome.

7.1 Discrete time implementation

Although many controller design procedures use continuous-time plant and controller descriptions, controllers are nowadays invariably implemented on digital computers, resulting in a discrete time implementation. This gives rise to the two issues that are briefly addressed below.

7.1.1 Aliasing

Aliasing occurs when a high frequency signal (beyond the sampling frequency) , due to slow sampling, in interpreted as a low frequency signal (below the sampling frequency. This phenomenon is easy to understand, simply by inspecting a figure like Fig. 7.1. The continuous curve represents the high frequency signal, and the x's represent sampled values. Clearly, if the signal in Fig. 7.1 is a controlled variable in a control loop, the controller will attempt to counteract the slow oscillations it sees in the controlled variable. Since the true oscillation is at a frequency beyond the sampling frequency, counteracting the oscillations by control is impossible, and the controller will merely excite the plant without achieving improved control.

Once a continuous-time signal has been sampled, there is no way of distinguishing low-frequency signal components due to aliasing from 'true' low frequency signal components. High frequency signal components must therefore be removed from the

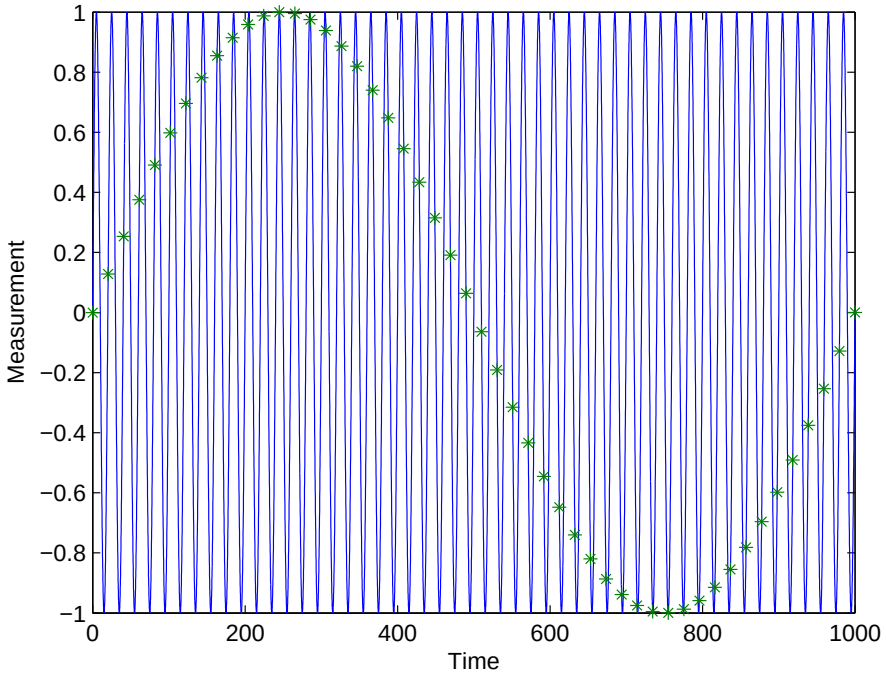

Figure 7.1: High frequency being mistaken for a low frequency signal, due to too slow sampling.

continuous-time signal, prior to sampling (also known as a 'presampling filter'). Usually, the continuous-time signal from a measurement device is an electrical signal, and a presampling filter is made from a simple RC network with low pass characteristics.

7.1.2 Sampling interval

Converting from a continuous- to a discrete-time control description is fairly standard, and covered in most books on digital control. Continuous-to-discrete conversion is therefore not described here. We will only note that this can be done in several different ways, among which discretization with zeroth order hold on the manipulated variables (assuming the manipulated variables to remain constant between sampling instances) appears to be the more common, and to work well in most cases.

Many introductory books will also provide the following rule-of-thumb for selecting the sampling interval: The sampling interval should be at least ten times faster than the closed loop bandwidth. Denoting the (continuous-time) crossover frequency ω_c, this means that the sampling interval t_s should be chosen according to

$$t_s \leq \frac{2\pi}{10\omega_c} \tag{7.1}$$

This is not an absolute rule, slower sampling may be possible. Furthermore, adhering to this rule is no guarantee against problems related to the discrete-time implementa-

tion. However, if slower sampling is attempted, there is particular reason for considering the possibility of performance degradation or even instability due to infrequent sampling.

Sampling too fast is primarily a waste of computing power. For systems where the computing power is limited, too fast sampling should therefore be avoided. Note that emergency situations may put significantly higher demands on the computing power of a control system than normal operations.

Most control functions in a large plant is implemented in a Distributed Control System (DCS). The engineer will then not have full freedom in selecting the sampling time, it has to be in integer multiples of the basic cycle time for the DCS. Control functions that require faster sampling than the basic sample time, will need to be implemented in dedicated hardware. For some control problems, e.g., compressor anti-surge control, this is often the case.

7.1.3 Execution order

Each time a controller executes, the following tasks have to be performed:

1. Read in new plant measurements.

2. Perform controller calculations, i.e., calculate new values for the manipulated variable. For observer/state feedback type controllers, the observer or state estimation calculations should be performed *before* the state feedback control calculations.

3. Implement the new manipulated variable values.

Clearly, these tasks should be executed in the order indicated above. Executing the tasks in the wrong order will introduce a totally unnecessary time delay into the control loop. With reasonable sampling intervals, a wrong execution order can be very detrimental for control. Only if sampling is very fast compared to the closed loop bandwidth, can one safely neglect this additional deadtime.

7.2 Pure integrators in parallel

Whereas a multiple integrators in series can be stabilized by a single feedback path, the same is not true for integrators in parallel. Thus, if there are n_i integrators in parallel, and n_m independent feedback paths (the number of independent feedback paths often corresponds to the number of independent measurements or the number of manipulated variables, whichever is lower), there will be $n_i - n_m$ integrators that are not possible to stabilize by feedback.

Often such integrators in parallel occur because of using several controllers (with integral action), controlling the same measurement, while using different manipulated variables. The safe way of implementing such parallel control can be found in Section 4.2.7 and Fig. 4.6.

To better understand the problem with integrators in parallel, consider Figure 7.2. The two integrating controllers integrate the opposite way of each other. The combined effect on the output is zero. That is, the two integrators are not both observable from the output (only their sum is), and they can therefore not both be stabilized by feedback.

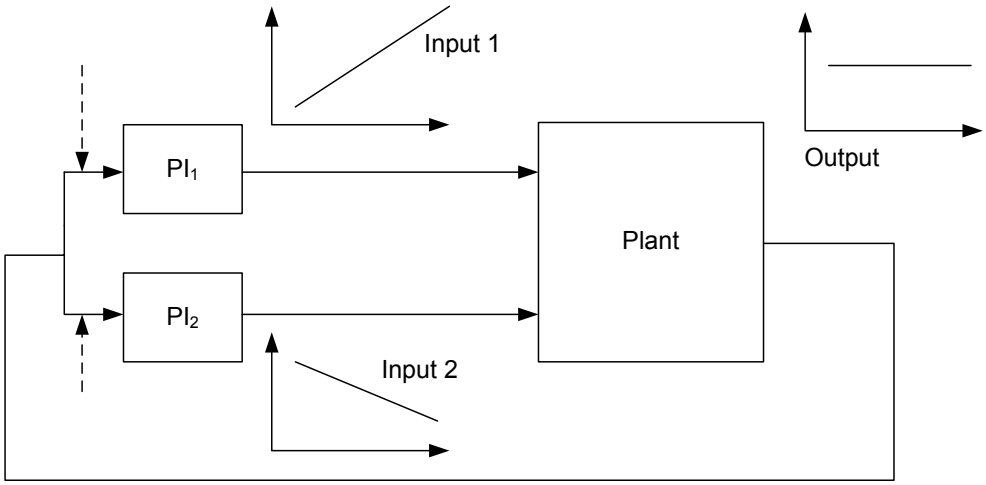

Figure 7.2: Multiple integrating controllers with a single measurement.

There are at least three different reasons why two integrating controllers in parallel may drift opposite ways, as illustrated in Fig. 7.2:

1. They may be given different setpoints. This is a rather stupid error that should be avoided.

2. The transmission of the feedback measurement may be affected by noise - and by different noise values for the different controllers. This was a very common problem with analog signal transmission, but is less of a problem with digital communications.

3. The two controllers will in practice not execute simultaneously. It is possible that the plant measurement is updated between the times when the two controllers execute, and the measurement may be updated several times for each time the controllers execute. The effect will be that the two controllers see *different* measurement and quantization noises. The result will be that the controller outputs drift. This cause for drifting controllers is every as likely with modern control systems as with older systems.

7.3 Anti-windup

In virtually all practical control problems, the range of actuation for the control input is limited. Whenever the input reaches the end of its range of actuation (the control

input is *saturated*), the feedback path is broken. If the controller has been designed and implemented without regard for this problem, the controller will continue operating as if the inputs have unlimited range of actuation, but further increases in the controller output will not be implemented on the plant. The result may be that there is a large discrepancy between the internal states of the controller and the input actually applied to the plant. This problem often persists even after the controlled variable has been brought back near its reference value, and controllers that would work fine with unlimited inputs or with small disturbances, may show very poor performance once saturation is encountered.

The problem described is typically most severe when the controller has slow dynamics - integral action is particularly at risk (since a pure integration corresponds to a time constant of infinity). An alternative term for integral action is '*reset action*, since the integral action 'resets' the controlled variable to its reference value at steady state. When the input saturates while there remains an offset in the controlled variable, the integral term will just continue growing, it 'winds up'. The problem described above is therefore often termed *reset windup*, and remedial action is correspondingly termed *anti-reset windup* or simply *anti-windup*.

Anti-windup techniques remain an active research area, and no attempt is made here to give an up-to-date review of this research field. The aim is rather to present some important and useful techniques that should be known to practicing control engineers.

7.3.1 Simple PI control anti-windup

A simple PI controller with limited actuation range for the control inputs (i.e., controller *outputs*), may be implemented as illustrated in Fig. 7.3. Here, the actual input implemented on the plant is feed back to the controller through the low pass filter $1/(\tau_I s + 1)$. If the actual plant input is not measured, it suffices to know the range of actuation for the input. The actual input can then easily be calculated.

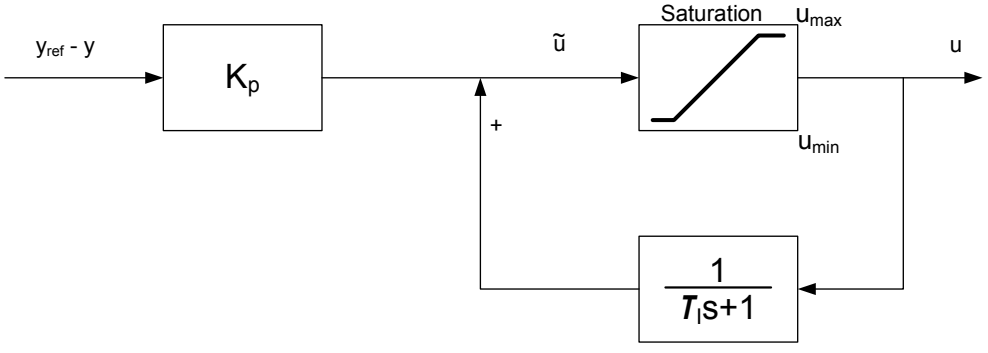

Figure 7.3: Simple anti-windup scheme for a PI controller.

From Fig. 7.3, it is easy to see that when the plant input is not saturated (when

$\tilde{u} = u$), we get

$$u = K_p \frac{\tau_I s + 1}{\tau_I s}(y_{ref} - y) \tag{7.2}$$

That is, we get the normal behaviour of a PI controller. On the other hand, consider the case when the input is in saturation at its upper limit u_{max}:

$$\tilde{u} = K(y_{ref} - y) + \frac{1}{\tau_I s + 1} u_{max} \tag{7.3}$$

The internal feedback path in the controller is now broken, there is no open integrator in the controller, and the controller state goes to u_{max} with a time constant τ_I. Thus, the integrating state does not wind up. Note also that when the controller state has reached its stationary value of u_{max}, the controller output will stay at its maximum value until the measurement y has crossed the reference value y_{ref}.

This anti-windup scheme is straight forward and simple to implement provided any actuator dynamics is fast compared to the PI controller time constant τ_I.

7.3.2 Velocity form of PI controllers

The PI controller in (refEq:PI) is in*position form*, i.e., the controller output corresponds to the desired position/value of the plant input. Alternatively, the controller output may give the desired *change* in the plant input.

Whereas the equations for PI controllers in position form are often expressed in continuous time (even though the final implementation in a plant computer will be in discrete time), the velocity form of the PI controller is most often expressed in discrete time. Let the subscript denote the discrete time index, and $e_k = y_{ref} - y_k$ be the control offset at time k. The discrete time equvivalent of (7.2) may then be expressed as

$$\Delta u_k = u_k - u_{k-1} = \frac{T}{\tau_I} e_{k-1} + K_p(e_k - e_{k-1}) \tag{7.4}$$

where T is the sample interval. Here Δu_k represents the *change* in the plant input at time k. If this change is sent to the actuator for the plant input, instead of the desired position of the input, the windup problem goes away. This is because desired changes that violate the actuation constraints simply will not have any effect.

The velocity form can also be found for more complex controllers, in particular for PID controllers. However, derivative action is normally rather fast, and the effects thereof quickly die out. It is therefore often not considered necessary to account for the derivative action in anti-windup of PID controllers.

7.3.3 Anti-windup in cascaded control systems

For ordinary plant input, it is usually simple to determine the range of actuation. For instance, a valve opening is constrainted to be within 0 and 100%, maximum and minimum operating speeds for pumps are often well known, etc. In the case of cascaded control loops, the 'plant input' seen by the outer loop is actually the reference

signal to the inner loop, and the control is typically based on the assumption that the inner loop is able to follow the reference changes set by the outer loop. In such cases, the 'available range of actuation' for the outer loop may be harder to determine, and may depend on operating conditions. An example of this problem may be a temperature control system, where the temperature control loop is the outer loop, and the inner loop is a cooling water flow control loop with the valve opening as the plant input. In such an example, the maximum achievable flowrate may depend on up- and downstream pressures, which may depend on cooling water demand elsewhere in the system.

Possible ways of handling anti-windup of the outer loop in such a situation include

- Using conservative estimates of the available range of actuation, with the possibility of not fully utilizing plant capacity in some operating scenaria.
- The controller in the inner loop may send a signal informing the controller in the outer loop when it is in saturation (and whether it is at its maximum or minimum value). The controller in the outer loop may then stop the integration if this would move the controller output in the wrong direction.
- Use the velocity form of the controller, provided the reference signal for the inner loop is calculated as *present plant output + change in reference from outer loop*. If the reference signal is calculated as 'reference at last time step + change in reference from outer loop', windup may still occur.
- For PI controllers, use the implementation shown in Fig. 7.3, where the 'plant input' used in the outer loop is the plant measurement for the inner loop.

Note that the two latter anti-windup schemes above both require a clear timescale separation between the loops, otherwise performance may suffer when the plant input (in the inner loop) is not in saturation. There is usually a clear timescale separation between cascaded loops.

7.3.4 Hanus' self-conditioned form

Hanus' self-conditioned form [32, 93] is a quite general way of preventing windup in controllers. Assume a linear controller is used, with state space realization

$$\dot{v} = A_K v + B_K e \tag{7.5}$$
$$\tilde{u} = C_K v + D_K e \tag{7.6}$$

where v are the controller states, e are the (ordinary) controller inputs, and $\tilde{u}$ is the calculated output from the controller (desired plant input). The corresponding controller transfer function may be expressed as

$$K(s) \stackrel{s}{=} \left[\begin{array}{c|c} A_K & B_K \\ \hline C_K & D_K \end{array}\right] = C_K(sI - A_K)^{-1}B_K + D_K \tag{7.7}$$

The corresponding implementation of the same controller in Hanus' self-conditioned form is illustrated in 7.4, where $\tilde{K}(s)$ given by

$$\begin{aligned} \tilde{u} &= \tilde{K}(s) \begin{bmatrix} e \\ u \end{bmatrix} \\ K(s) &\stackrel{s}{=} \left[\begin{array}{c|cc} A_K - B_K D_K^{-1} C_K & 0 & B_K D_K^{-1} \\ \hline C_K & D_K & 0 \end{array}\right] \end{aligned} \tag{7.8}$$

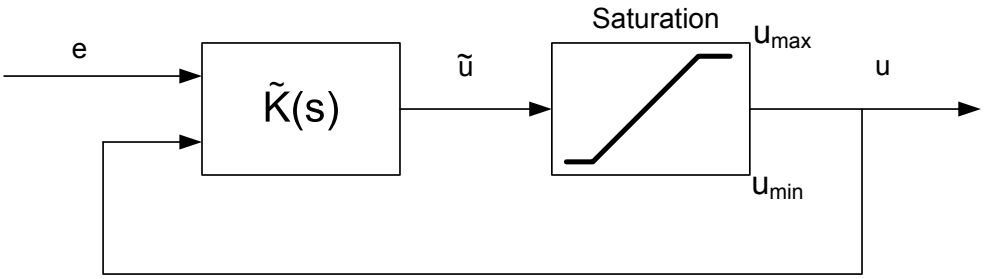

Figure 7.4: Illustration of anti-windup with the controller $K(s)$ implemented in its self-conditioned form $\tilde{K}(s)$.

From (7.8) we see that when the plant input u is not saturated, i.e., when $\tilde{u} = u$, the controller dynamics are given by (7.6). When the plant input is saturated, the steady state controller output will be

$$\tilde{u} = -C_K(A_K - B_K D_K^{-1} C_K)^{-1} u + De \tag{7.9}$$

If $B_K D_K^{-1} C_K >> A_K$, we get

$$\tilde{u} \approx u + D_K e \tag{7.10}$$

and thus the plant input will stay at its limit until the corresponding element of $D_K e$ changes sign.

Clearly, the use of this anti-windup methodology requires D_K to be invertible, and hence also of full rank. Thus, the controller must be semi-proper. The rate at which the controller states converge towards the steady state solution (when in saturation) is given by the eigenvalues of $A_K - B_K D_K^{-1} C_K$. This matrix obviously has to be stable. A small (but non-singular) D_K will generally make the convergence fast.

In [32], self-conditioning is presented in a more general setting, potentially accounting also for time-varying or non-linear controllers. However, only in the case of linear time-invariant controllers do the resulting controller equations come out in a relatively simple form.

7.3.5 Anti-windup in observer-based controllers

Many advanced controllers are (or may be) implemented as a combination of static state feedback controllers and a state observer/estimator. This is the case for LQG/H_2-optimal controllers as well as H_∞-optimal controllers.

For such controllers, anti-windup is achieved by ensuring that the state observer/estimator receives the *actual plant input that is implemented on the plant*. This is illustrated in Fig. 7.5

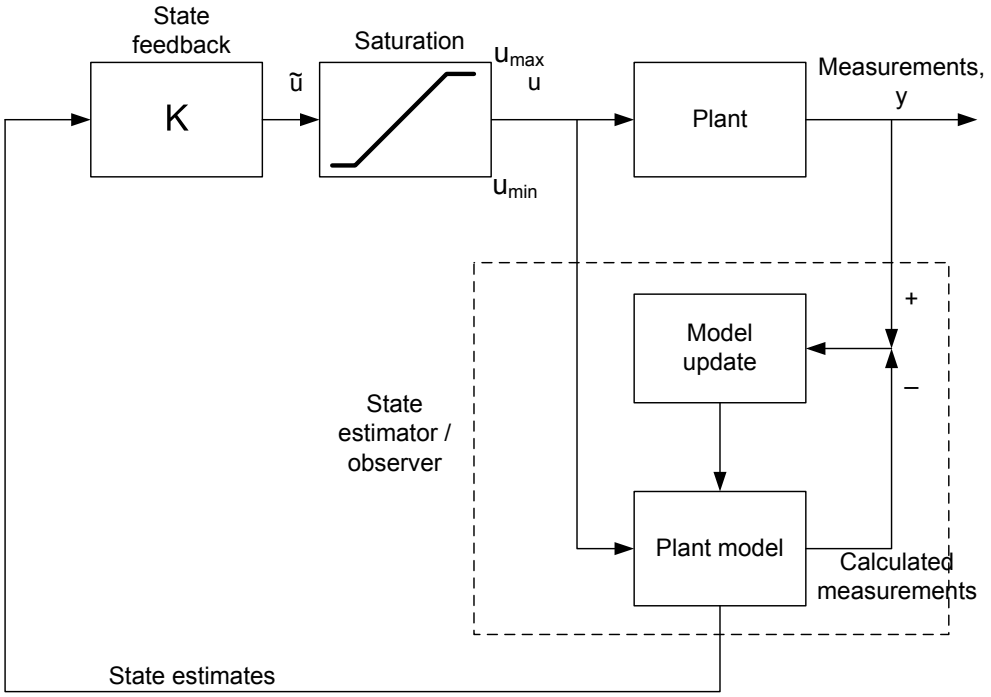

Figure 7.5: Illustration of anti-windup for controllers based on static state feedback combined with state estimation.

In many applications it is desired to have offset-free control at steady state. This requires the use of integral action. This is often incorporated in a state estimator/state feedback control design as illustrated in Fig. 7.6.

The state estimator only estimates actual plant states, whereas the state feedback is designed for a model where integrators (which integrate the control offset) are appended to the plant model. When implementing the controller, the integrators are a part of the controller (in the control system). The values of the integrators are thus directly available in the control system, and clearly there is no need to estimate these states.

However, when integration is incorporated in this way, the integrating states may wind up even if the actual input values are sent to the state estimator. Figure 7.7 illustrates how the anti-windup signal to the integrators must represent the range of movement available for the integrating states, i.e., with the contribution from the (actual) state feedback removed.

Remark. Note that if Hanus' self-conditioned form is used for the anti-windup, this requires a non-singular D-matrix, resulting in a PI block instead of a purely integrating block. The size of this D-matrix may affect controller performance (depending on how and whether it is accounted for in the 'state' feedback control design).

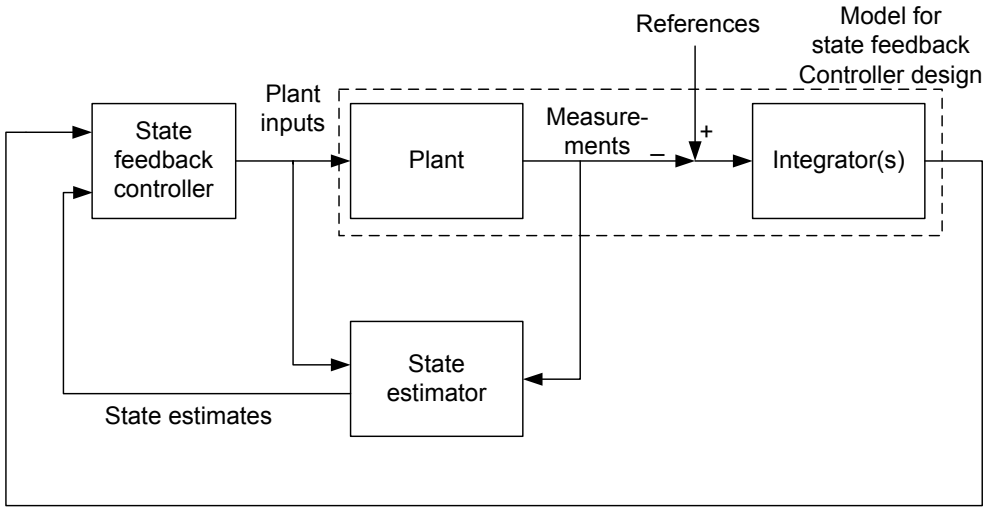

Figure 7.6: State estimator and static state feedback augmented with integral action.

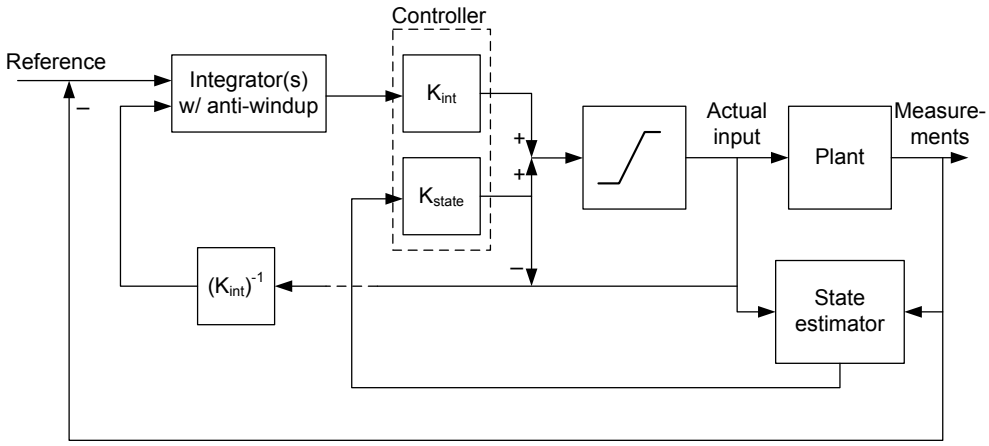

Figure 7.7: Implementation of anti-windup for state estimator and static state feedback augmented with integral action.

7.3.6 Decoupling and input constraints

Decouplers are particularly prone to performance problems due to input constraints. This is not easily handled by standard anti-windup, because much of the input usage can be related to counteracting interactions. Therefore, if an output is saturated, but other outputs are adjusted to counteract the effects of the 'unsaturated' output, severe performance problems may be expected.

One way of ensuring that the decoupler only tries to counteract interactions due to the inputs that are actually implemented on the plant, is to implement the decoupler as illustrated in Fig. 7.8.

The implementation in Fig. 7.8 is easily extended to systems of dimension higher than 2×2. When the inputs are unsaturated, the 'Decoupler with saturation' in

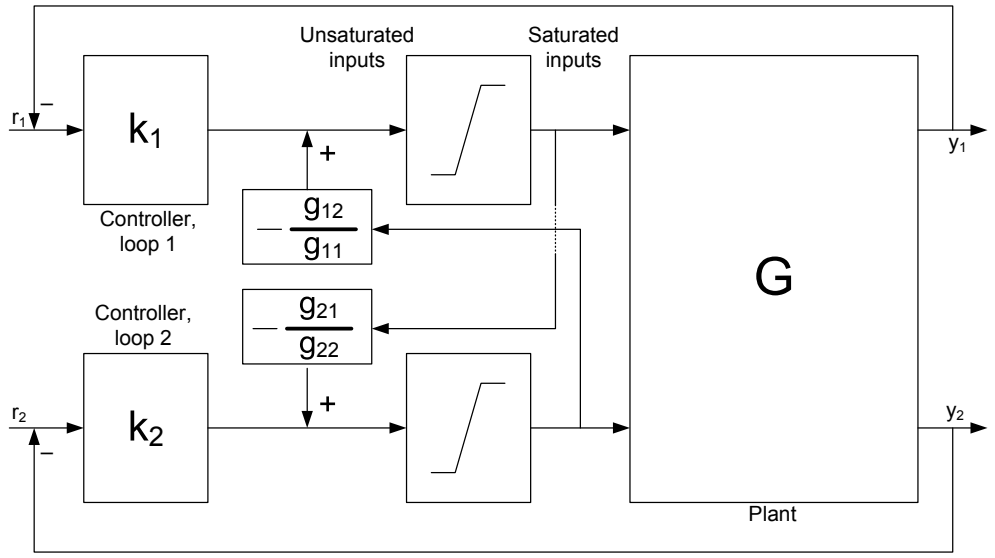

Figure 7.8: Implementation of decoupler in order to reduce the effect of input saturation. The decoupler will only attempt to counteract interactions due to inputs that are actually implemented on the plant.

Fig. 7.3 corresponds to the decoupling compensator $W(s) = G(s)^{-1}\tilde{G}(s)$, where $\tilde{G}(s)$ denotes the diagonal matrix with the same diagonal elements as $G(s)$. The precompensated plant therefore becomes $GW = \tilde{G}$, i.e., we are (nominally) left only with the diagonal elements of the plant.

Note that if the individual loop controllers $k_i(s)$ contain slow dynamics (which is usually the case, PI controllers are often used), they will still need anti-windup. In this case the anti-windup signal to the controller should not be the saturated input, but the saturated input *with the contribution from the decoupling removed*, i.e., the decoupling means that the saturartion limitations for the individual loop controllers $k_i(s)$ are time variant.

7.4 Bumpless transfer

The term 'bumpless transfer' refers to the 'bumps' that may occur in the controller output (and consequently in the plant output) when changing controller parameters, switching between different controllers, or switching the control between manual and automatic operation.

7.4.1 Switching between manual and automatic operation

If we want bumpless transfer in this case, we must ensure that the controller output remains unchanged if the controller input is unchanged. For proportional controllers this requires setting/modifying the bias on the controller output. For controllers with dynamic states, the controller states must be set such that the states agree with both

the controller output prior to switching to manual and the observed plant outputs prior to switching.

Assume that a discrete time implementation of the controller is used, and that switching from manual to automatic occurs before executing the controller calculations at time $k = 1$.

- *Proportional control.* The controller output bias is set such that the controller output at time $k = 0$ (if the controller had been in automatic) equals the manual controller output value for the plant output observed at time $k = 0$.
- *PI control.* The calculation is similar to the case for proportional-only control. However, in this case one has the choice of either calculating an output bias, and set the integral (i.e., state) value to zero, or *vice versa*.
- *PID control.* In this case, the controller output at time $k = 0$ must agree with both the observed plant output at time $k = 0$ and the derivative of the plant output at that time. As there are differences in how PID controllers are implemented, particularly the derivative term, the detailed calculations are not described further here.
- For SISO controllers with n states, it is generally necessary to consider the n most recent plant outputs to calculate the controller states giving bumpless transfer.

Note that

- It is well known that integral action is grenerally needed to get offset-free control at steady state. For controllers without integral action, setting the output bias to an unfortunate value will make the steady state offset worse.
- Bumpless transfer is irrelevant for PI controllers in the velocity form.

7.4.2 Changing controller parameters

The calculations are very similar to what is described for switching between manual and automatic, the difference is only that the need for bumpless transfer arises for a different reason.

7.4.3 Switching between different controllers

Many advanced controllers can be decomposed in a state estimator and a static state feedback controller. Often different estimators/controllers are designed for different operational regions or different modes of operation. In this case it is essential that when switching to a new controller, the state estimates used are appropriate for the state feedback controller used. Therefore, all state estimators should be run in parallel - also the estimators corresponding to inactive controllers. *The estimators should receive the input that is actually implemented on the plant*, which for estimators

corresponding to inactive controllers typically means a plant input different from that the corresponding state feedback controller would generate. This way, the estimator can provide an updated state estimate when the corresponding controller is put in operation. This is illustrated in Fig. 7.9.

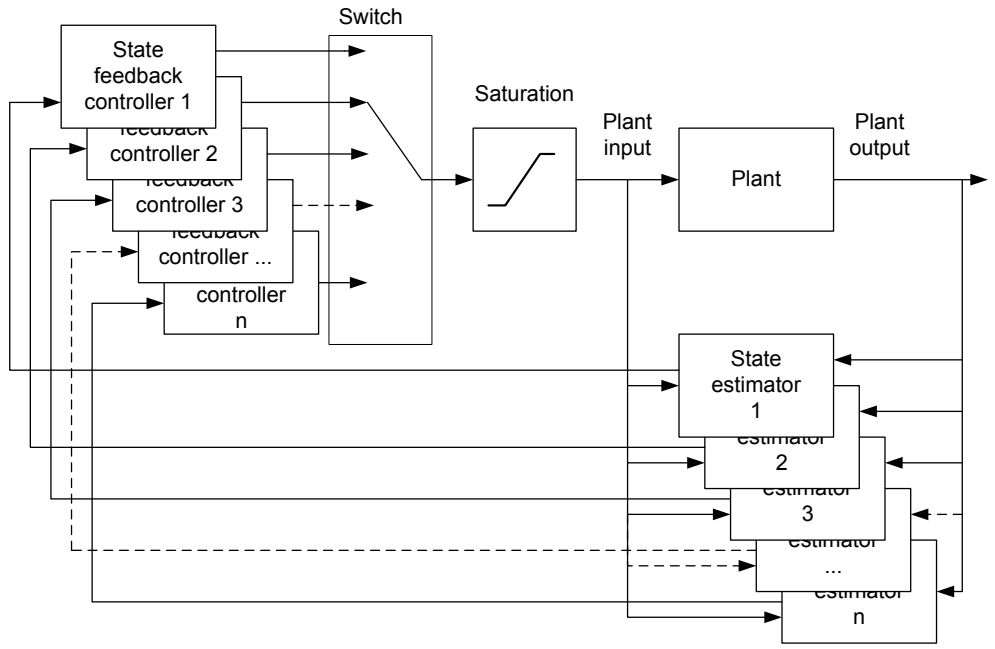

Figure 7.9: Switching between controllers that can be separated into a static state feedback and a state estimator.

Chapter 8

Controller Performance Monitoring and Diagnosis

8.1 Introduction

It is a sad fact that many control loops in industrial processes actually degrade system performance, by increasing the variability in the controlled variable rather than decreasing it. Still more control loops do actually work, but are very far from optimal. Some causes for poor controller performance are:

- Operating conditions have changed after the controller was tuned.
- The actual process has changed, some process modifications have been made after the controller was tuned.
- The controller has never actually been tuned, it is still using the manufacturer's default tuning parameters.
- A poor (or even inconsistent) control structure, causing severe interactions between control loops.
- Some equipment in the control loop may be in need of maintenance or replacement, e.g., faulty measurements, control valves with excessive stiction, severe fouling in heat exchangers, etc.

There are many reasons why such a situation may be allowed to last. Often, plant operators are aware of what parts of the process are oscillating or show large control offsets. However, this information often stays with the operators, and they learn to cope with the process as it is. The typical operator will lack the competence to assess whether the observed control performance is much worse than what should be expected. When asked a general question about whether control of the process is

acceptable, they may therefore very well confirm that the control is good even if that is not the case.

The automation department of a large plant is normally very small. The typical automation department is fully occupied with keeping the various automation and control system in operation, with little time for improving the control system. Most industrial automation engineers are therefore also trained to keep the control system running, and have little relevant background for evaluating controller performance or improving controllers. After an initial commissioning phase, most controllers are therefore "left alone" for long periods.

Considering the large number of control loops in an industrial plant, there is a need for tools which ensure efficient use of what little time is available for improving the control system, that is, tools which help the engineer to

- focus on where the control problems are most acute
- quickly assess whether significant improvements are easily achievable, e.g. by retuning the controller
- diagnose the cause for poor control performance.

Here, Control Performance Monitoring (CPM) is understood as tools and systematic methods for

- Assessing control loop performance, by comparison with a well-defined performance benchmark.
- Detecting oscillating control loops, and diagnosing the cause for oscillations.
- Root cause analysis for distributed oscillations (i.e., when multiple loops are oscillating, to arrange the loops into groups which oscillate in the same pattern, and then locate - and preferably also diagnose - the cause for oscillation for each of the groups).

In the open literature, the performance *assessment* part has received by far the most attention. This issue was brought to the attention of the academic community by an influential paper by Harris [34] in 1989, although similar ideas had been proposed earlier, e.g. by Fjeld [22]. These papers, as well as most publications on performance assessment, consider performance assessment in a stochastic setting, by comparing the observed variance in the controlled variable to the variable that can be achieved by an ideal controller (typically a minimum variance controller). Deterministic performance assessment has received much less attention, with Åström et al. [9] and Swanda and Seborg [97] as exceptions. Another interesting approach to performance monitoring is presented by Tyler and Morari [102], who show how many performance specifications can be formulated as bounds on the system's impulse response coefficients. The performance monitoring then consists of testing the relative likelihood of the system fulfilling the performance bounds, compared to the likelihood of it not doing so.

Oscillation detection and diagnosis has received less attention, whereas only recently has there appeared significant publications in the open literature on root cause detection for distributed oscillations.

This report will first consider the issue of oscillation detection, and then address oscillation diagnosis and root cause detection for distributed oscillations. The rationale for this is that loops with significant persistent oscillations will certainly fail any performance assessment test, and should always be examined. Thereafter, performance assessment is described. Issues relating to the relevance of a minimum variance benchmark are discussed, and a brief discussion about requirements for successful CPM is given. Finally, available techniques for CPM are discussed, with a focus on issues that need to be clarified, and needs for further development in analysis techniques.

There are some available literature surveys on Control Performance Monitoring, notably by Qin [78] and Harris et al. [35]. Industrial experience is described in many papers, this authors favourites are probably the papers by Kozub [63] and Thornhill et al. [99]. A recent update on multivariable CPM is given by Shah et al. [86]. The only textbook on the subject so far is that of Huang and Shah [50], a review of which can be found in [64].

There are several commercial suppliers of CPM tools. However, there is relatively little available in the open literature on how the CPM activities should be organized and coordinated with other activities involved in plant operation in the processing industries. Useful information on such issues is found in papers from the CPM team at Honeywell, e.g. [19, 67] (on the other hand, authors from ABB and Matrikon appear more eager to publish new tools and methodologies for CPM). Some of the complications involved in correctly diagnosing control problems and proposing corrective measures are illustrated in Owen et al. [75].

8.2 Detection of oscillating control loops

For the trained human eye, detection of oscillations may seem a trivial task. However, it is far from trivial to define and describe oscillations in a typical signal from a process plant in such a way that it can reliably be automated (in either on-line or off-line tools). We will here present a few tools that have been proposed, but first present some statistical tools. It is assumed that the signals under study are stable, or at least only marginally unstable, as otherwise the control loops in question will have to be taken out of service (and it should then be apparent that the control loop needs attention). Any exponentially growing signal will eventually hit some system constraint or cause some malfunction. Note that control loops are here classified as oscillatory if they show an unacceptable tendency to oscillate, a perfect limit cycle is not a requirement. Stable loops with insufficient damping will also be classified as oscillatory in this context.

8.2.1 The autocorrelation function

The autocorrelation function is essentially a measure of how closely the values of a variable, when measured at different times, are correlated. For a variable y and a data set of N datapoints, the autocorrelation function is given by

$$\rho_k = \frac{\sum_{t=1}^{N-k}(y_t - \overline{y})(y_{t+k} - \overline{y})}{\sum_{t=1}^{N}(y_t - \overline{y})^2}$$

The autocorrelation function is 1 for lag 0, that is, $\rho_0 = 1$. For stable signals, it generally decays with increasing lags, whereas it will oscillate for systematically oscillating signals, and a periodic signal will have a periodic autocorrelation function.

In principle, one should be able to detect oscillations directly from the autocorrelation function. However, it need not be so straight forward if the signal contains multiple frequencies, measurement noise, assymmetric oscillations, etc. Nonlinear effects may also introduce oscillations at frequencies that are multiples of the base oscillation frequency. Nevertheless, Moiso and Piipponen [68] propose an oscillation index calculated from the roots of a second order AR model fitted to the autocorrelation function. The method of Miao and Seborg, which is described below, is also based on the autocorrelation function.

8.2.2 The power spectrum

The power spectrum results from a Fourier transform of the autocorrelation function, and in essence it is the frequency domain equivalent of the autocorrelation function. If the signal exhibits a pure sinusoidal oscillation at a particular frequency, the power spectrum will have a peak at that frequency. An oscillation that does not decay with time, will have a very large peak at that frequency in the power spectrum. The problems of using the power spectrum for oscillation detection are similar to those of using the autocorrelation function. Instead of the power spectrum having a single spike at the oscillating frequency, the signal may be corrupted by noise and nonlinear effects that the power spectrum is blurred or contains numerous spikes.

8.2.3 The method of Miao and Seborg

Miao and Seborg[66] uses the autocorrelation function to detect oscillations. It calculates a somewhat non-standard 'decay ratio', as illustrated in Fig. 8.1.

The Miao-Seborg oscillation index is simply the ratio given by $R = a/b$. Miao and Seborg propose a treshold value of $R = 0.5$, a larger value will indicate (unacceptable) oscillations. Little justification is provided for this measure. In particular, it is not explained why this measure is better than simply comparing the magnitude of neighbouring peaks in the autocorrelation function.

Nevertheless, industrial experience appears to be favourable, and oscillations are detected with reasonable reliability. Some drawbacks are

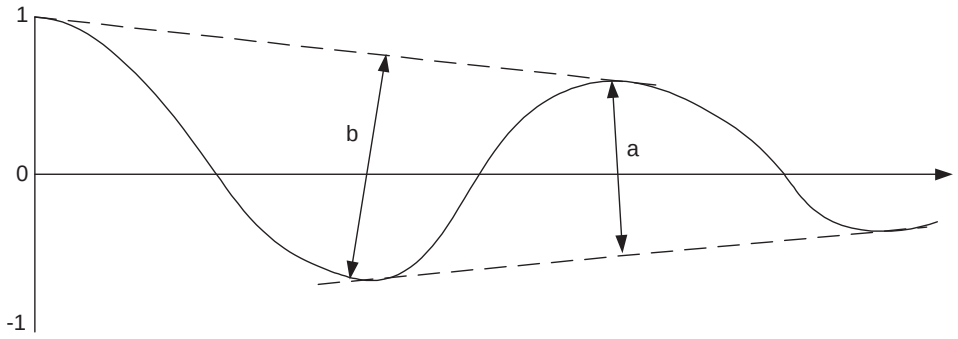

Figure 8.1: Calculation of the Miao-Seborg oscillation index from the autocorrelation function.

- it is somewhat complicated for on-line oscillation detection, it is better suited for offline analysis of batches of data.
- it does not take the amplitude of oscillations directly into account. Some oscillations of small amplitude may be acceptable, but this method will classify also loops with acceptable oscillation as oscillatory.
- it assumes that the oscillations are the main cause of variability in the measured variable. If a control loop experiences frequent (and irregular) setpoint changes of magnitude larger than the amplitude of the oscillations, it may fail to detect the oscillations.

8.2.4 The method of Hägglund

Hägglunds measure[29] may be said to be a more general measure of control performance rather than an oscillation detection method. The basic idea behind the measure is that the controlled variable in a well-functioning control loop should fluctuate around the setpoint, and that long periods on one side of the setpoint is a sign of poor tuning.

Hägglund's performance monitor looks at the control error $e(t) = r(t) - y(t)$, and integrates the absolute value of $e(t)$ for the period between each time this signal crosses zero:

$$IAE = \int_{t_{i-1}}^{t_i} |e(t)|\, dt$$

where t_{i-1} and t_i are the times of two consequtive zero crossings. Whenever this measure increases beyond a treshold value, a counter is incremented, and an alarm

is raised when the counter passes some critical value. It is shown in [29] how a forgetting factor can be used to avoid alarms from well-functioning loops which are exposed to infrequent, large disturbances (or setpoint changes).

Critical tuning parameters for this monitoring method are the IAE treshold value and the counter alarm limit. Typical choices for the IAE treshold value are

$$\begin{aligned} IAE_{\text{lim}} &= 2a/\omega_u \\ IAE_{\text{lim}} &= aT_I/\pi \end{aligned}$$

where a is an acceptable oscillation magnitude, ω_u is the ultimate frequency (the oscillation frequency found in a closel loop Ziegler Nichols experiment), and T_I is the integral time in a PI(D) controller. The more rigorous of the two treshold values is the first, and ω_u would be available if the loop was tuned with e.g. Hägglund's relay-based autotuning procedure. However, often ω_u will not be available, and the second expression for IAE_{lim} will then have to be used - this expression is intended to work as a reasonable approximation of the first expression for IAE_{lim} for a reasonably tuned loop. Naturally, this may be misleading if the cause of poor control performance is poor choice of controller tuning parameters.

The counter alarm limit is simply a tradeoff between the sensitivity of the monitoring method and the rate of "unnecessary" alarms. This monitoring method is

- Simple and appliccable for on-line implementation.
- It takes oscillation amplitude into account - it is ignores small oscillations unless the oscillation period is very long.
- Some tuning of the monitoring method must be expected. The guidelines for choosing IAE_{lim} is based on knowledge of the ultimate frequency of the control loop - which typically is not known unless a Ziegler-Nichols type tuning experiment or a Hägglund type autotuner is used. Alternatively, it is proposed to base IAE_{lim} on the controller integral time - which is only reasonable if the loop is well tuned.

8.2.5 The regularity index

Hägglund's monitoring method is extended in [98] for off-line oscillation detection, resulting in a new oscillation measure called the regularity index.

To calculate the regularity index, the integral absolute error is calculated, and when the control error crosses zero, the measure

$$\frac{IAE_i}{\Delta T_i \sigma} \tag{8.1}$$

is plotted together with the time t_{i+1} for the most recent zero crossing. Here IAE_i is the integral absolute error between the two most recent zero crossings, ΔT_i is the time between the zero crossings, and σ is an estimate of the r.m.s. value of the

noise. It is recommended to filter the measurements by estimating an AR model for the measurement, and to base the analysis (calculation of IAE) based on a one step ahead prediction from the AR model rather than the raw measurement. This will reduce the influence of measurment noise, and the AR model estimation can also give an estimate of the measurement noise, from which σ can be calculated.

Next, a treshold value ξ is chosen, and a *regularity factor* is derived from the time intervals Δk_i between each time the measure in Eq. (8.1) crosses the treshold value. Thus,

$$R_i = \frac{\Delta k_{i+1}}{\Delta k_i}; \qquad q(\xi) = \frac{Mean\,value\,of\,R}{Standard\,deviation\,of\,R} \tag{8.2}$$

The regularity index is then

$$q = \max_{\xi} q(\xi) \tag{8.3}$$

The period of oscillation is estimated from the number of times the measure in Eq. (8.1) crosses the treshold ξ between the first and last instance of crossing the treshold.

8.2.6 The method of Forsman and Stattin

This method also looks at the control error $e(t) = r(t) - y(t)$, but it is strictly an oscillation detection method and not a general performance measure. Forsman and Stattin [24] proposes comparing both the areas between the control error and zero and the time span that the error has the same sign. However, the resulting area and time span is not compared with the immediately previous area/timespan (when the control error had opposite sign), rather the comparison is made with the preceding period when the control offset had the same sign. This is illustrated in Fig. 8.2.

The method uses two tuning constants α and γ, that both should be in the range between 0 and 1, and simply counts the number of times h_A in a data set that

$$\alpha < \frac{A_{i+1}}{A_i} < \frac{1}{\alpha} \text{ and/or } \gamma < \frac{\delta_{i+1}}{\delta_i}$$

and the number of times h_B that

$$\alpha < \frac{B_{i+1}}{B_i} < \frac{1}{\alpha} \text{ and/or } \gamma < \frac{\varepsilon_{i+1}}{\varepsilon_i}$$

where A_i, B_i, δ_i and ε_i are defined in Fig. 8.2. The oscillation index is then given by $h = (h_A + h_B)/N$, where N is the number of times in the data set that the control offset crosses zero.

Forsman and Stattin recommend closer examination of loops having $h > 0.4$, and if $h > 0.8$ a very clear oscillative pattern can be expected.

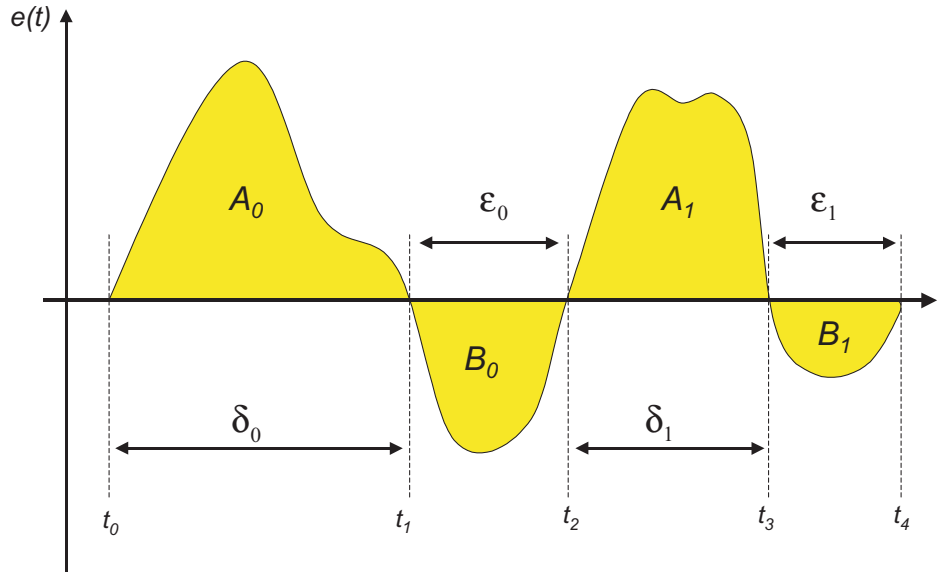

Figure 8.2: The oscillation detection method of Forsman and Stattin.

8.2.7 Pre-filtering data

All methods presented above may be ineffective for noisy data, and both Miao and Seborg [66] and Forsman and Stattin [24] discuss pre-filtering the data with a low pass filter to reduce the noise. Thornhill and Hägglund [98] propose filtering through using the one-step-ahead prediction from an AR model, as described previously. Clearly, the filter should be designed to give a reasonable tradeoff between noise and oscillation detection in the frequency range of interest. The interested reader should consult the original references for a more comprehensive treatment of this issue.

8.3 Oscillation diagnosis

Once an oscillating control loop has been detected, it is naturally of interest to find the cause of the oscillations, in order to come up with some effective remedy. There is no general solution to the diagnosis problem, the proposed methods can at best handle parts of the problem. We will present diagnosis procedures proposed by Hägglund [29, 98], and passive procedures (that may be automated) for detecting valve stiction proposed by Horch [39].

8.3.1 Manual oscillation diagnosis

Hägglund [29] proposes the manual oscillation diagnosis procedure presented in Fig. 8.3

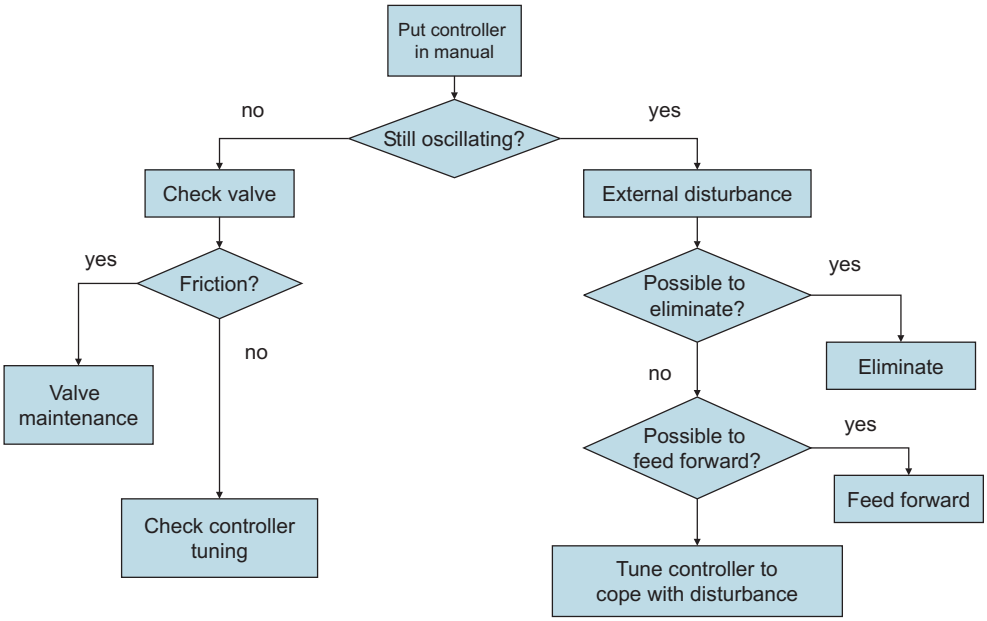

Figure 8.3: Hägglund's method for manual oscillation diagnosis.

The main problem with this procedure is the assumption that if the oscillation (in the controlled variable) stops when the controller in a particular loop is put in manual, then the oscillation is caused by that loop. Often, oscillations arise from multivariable interactions between loops, and the oscillation will then stop when any one of these loops are put in manual. The first loop to be put in manual will then receive the "blame" for the oscillations, and will consequently be detuned (made slower). Therefore, the results of this procedure will depend on the order in which the loops are examined. If several loops show a similar oscillation pattern, one should therefore first examine the loop for which slow control is more acceptable.

The procedure is also a little short on examining other instrumentation problems than valve friction (stiction), e.g., valve hysteresis, measurement problems, etc. Furthermore, the procedure gives no proposals for how to eliminate external disturbances. Clearly, the solution will be very dependent on the particular process, but typically it will involve modifying the process or the control in other parts of the process.

Additional flowcharts for oscillation diagnosis are presented in [98]. Some of those flowcharts do not require putting the controller in manual. They also show how useful diagnostic information can be derived from plotting the controlled variable (pv) vs. the setpoint (sp). Idealized plots for actuators with deadband, static friction in actuator, oversized valve as manipulated variable, and a linear loop with phase lag

are shown. The use of such sp-pv plots is clearly limited to loops with frequent setpoint changes, otherwise setpoint changes have to be introduced purely for diagnostic purposes (i.e., the plant has to be disturbed).

Thornhill and Hägglund [98] also address nonlinearity detection (without further classifying the non-linearity) using the regularity index and the power spectrum for the controlled variable.

8.3.2 Detecting and diagnosing valve stiction

A commonly occuring problem with valves is that they can have a tendency to stick due to stiction (short for 'static friction'). Once the controller applies sufficient force to overcome the stiction and move the valve, the friction force drops dramatically (since the 'dynamic' friction is much smaller than the static friction). This results in a large net force acting on the valve stem, causing a sudden move of it. It is well known that such stiction can cause oscillations.

Using the cross-correlation function to detect valve stiction

Horch [39] have developed a method for detecting stiction, based on measurements of the controlled variable and the controller output. The method assumes that the controller has integral action. The integral action will steadly increase the controller output, until the valve suddenly "jumps" to a new position. Persistent oscillations often result when the valve jumps too far, so that the controller has to stop the valve movement and move it in the opposite direction. Stopping the valve causes it to stick again, causing the sequence of events to repeat.

When there are problems with valve stiction, the controller output signal typically has a sawtooth shape. The controlled variable is typically almost like a square wave, especially if the dominant time constant of the process (in open loop) is much shorter than the period of oscillation.

Horch found that the cross-correlation function between controller output and controlled variable typically is an odd function[1] for a system oscillating due to stiction. On the other hand, if the oscillation is due to external disturbances, the cross-correlation function is normally close to an even function. Unstable loops oscillating with constant amplitude (due to input saturation) also have an even cross-correlation function.

For a data set with N data points, the cross-correlation function between u and y for lag τ (where τ is an integer) is given by

$$r_{uy}(\tau) = \frac{\sum_{k=k_0}^{k_1} u(k)y(k+\tau)}{\sum_{k=1}^{N} u(k)y(k)} \tag{8.4}$$

[1]Reflecting the 90° phase shift due to the interal action in the controller.

where

$$\begin{aligned} k_0 &= 1 \text{ for } \tau \geq 0 \\ k_0 &= \tau + 1 \text{ for } \tau < 0 \\ k_1 &= N - \tau \text{ for } \tau \geq 0 \\ k_1 &= N \text{ for } \tau < 0 \end{aligned}$$

Note that the denominator in Eq. (8.4) is merely a normalization, giving $r_{uy}(0) = 1$. It is not necessary for the stiction detection method.

Horch' stiction detection method has been found to work well in most cases. However, it fails to detect stiction in cases where the dominant time constant of the (open loop) process is large compared to the observed period of oscillation. In such cases the cross-correlation function will be approximately even also for cases with stiction. This problem is most common with integrating processes (e.g., level control loops), but may also occur for other processes with slow dynamics.

Histograms for detecting valve stiction

Horch [40, 41] has recently proposed an alternative method for stiction detection for integrating processes. Industrial experience with this alternative method is not known. This method is patented by ABB. The alternative method works by looking for abrupt changes in the process output, by twice differentiating the measured process output. This is illustrated in Fig. (8.4).

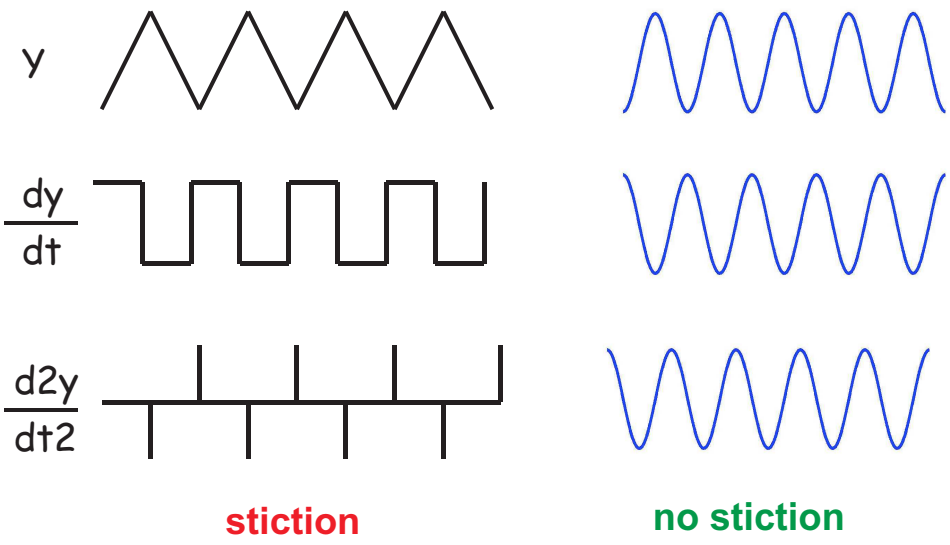

Figure 8.4: Stiction detection by twice differentiating the process output.

It can be seen from the figure that twise differentiation a sinusoidal signal (without any abrupt changes), results in a sinusoid. On the left of Fig. (8.4) is the output

of a pure integrator with a square-wave input, i.e., the typical input shape for a sticking control valve. Twice differentiating this signal gives an output that is zero except for periodic spikes of alternating sign. The stiction detection method for integrating processes is therefore based on a histogram showing the relative frequency of occurence of the various values for the twice-differentiated measurement signal. This is illustrated in Fig. 8.5. Although the difference between the two histograms in Fig. 8.5 become less distinct in the presence of measurement noise, this method claimed to work well also in the presence of measurement noise.

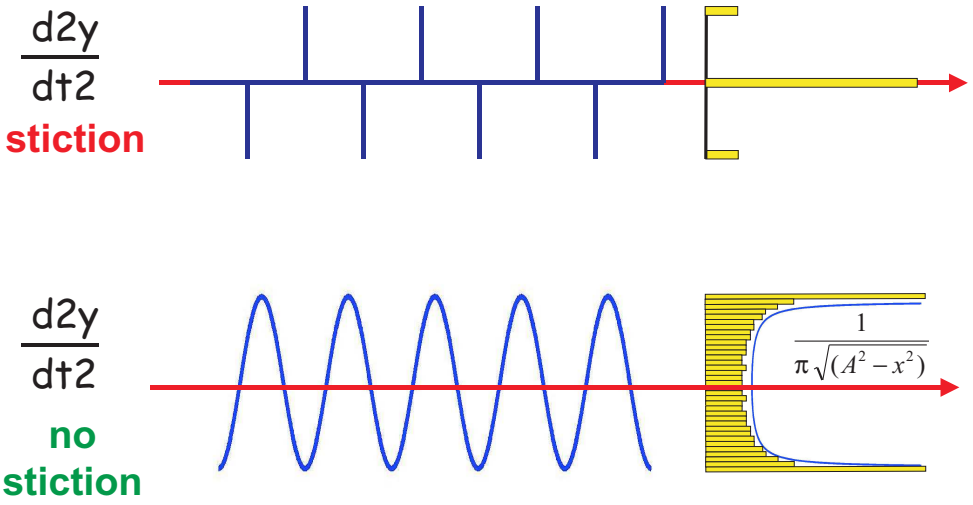

Figure 8.5: Histograms for detecting stiction in integrating processes.

The same method of detecting stiction is also proposed also for asymptotically stable plants [41] (for which the cross-correlation based stiction detection should work well). In this case, the measurement signal should be more like a square wave if the oscillations are caused by stiction, and the measurement signal is only differentiated once prior to obtaining the histograms.

Stiction detection using an x-y plot

The method involves plotting the controller output (manipulated variable) vs. the controlled variable. If these two variables tend to move in a closed path around an area where the curve seldom enters, this is a sign of an oscillating control loop, where there is a phase lag (different from $n \cdot 180°$) between input and output. If the x-y plot shows sharp 'corners', this is considered to be a sign of significant stiction. Without the sharp corners, there is no cause for suspecting non-linearity (i.e., stiction) to be the cause of the oscillations, since they may just as well be caused by poor tuning and random noise or oscillating disturbances. The use of an x-y plot is illustrated in Fig. 8.6, where the blue curve shows a case with stiction, and the red curve shows the same system without stiction. The use of this method is apparently widespread in

industrial practice, although its origin is not known to this author. In the example illustrated in Fig. 8.6, this method would correctly identify stiction in a case with some measurement noise.

However, numerical experience and intuition would suggest that this method may fail in cases with severe measurement noise, especially when there is a phase difference of close to $n \cdot 180^\circ$ at the dominant frequency of oscillation. Filtering may reduce the sensitivity to noise, but may also reduce the sharp corners in the x-y curve that are necessary to distinguish stiction from other causes of oscillation (which may occur also for linear systems).

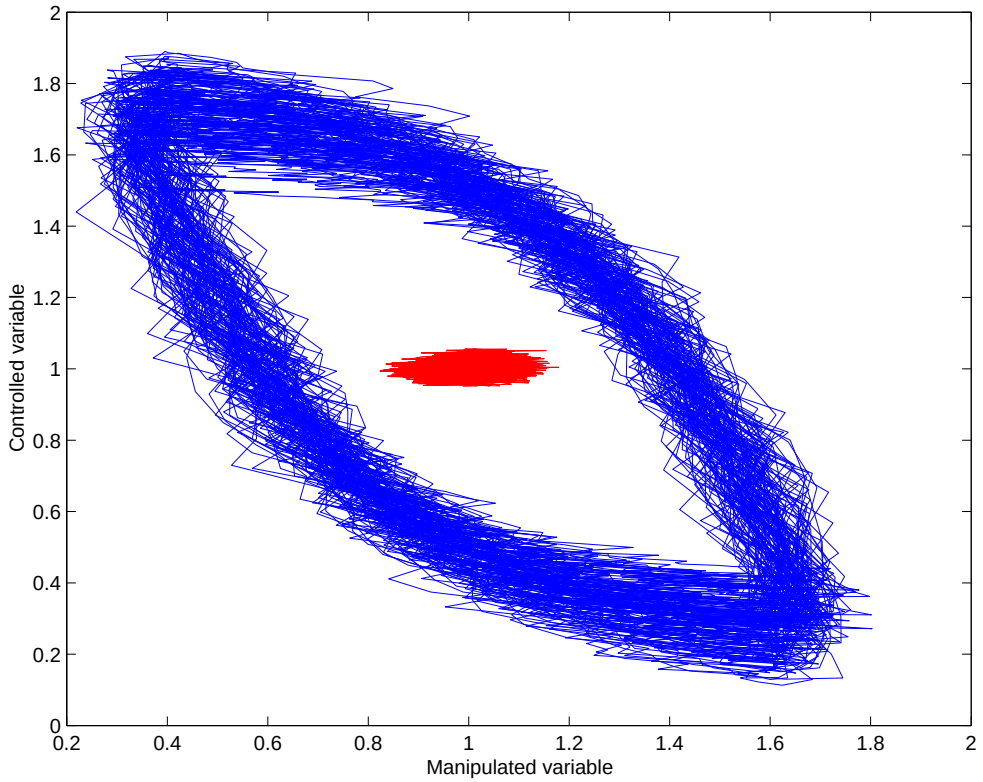

Figure 8.6: Use of xy-plot to detect stiction. The blue curve shows a system with stiction, the red curve shows the same system without stiction.

Comparison of stiction detection measures

To this authors knowledge, there is no available comparison of the three stiction detection methods described above in the open literature - although it is known that the cross-correlation method fails for integrating processes. There is thus a need for comparing these methods, both on industrial and simulated data. The cross-correlation and histogram methods are easily formulated in a form suitable for automatic analysis. Although visual stiction detection is easy using the x-y plot, a

formulation suitable for automatic analysis is not known. However, this should not be an insurmountable challenge.

8.3.3 Stiction compensation

There are a number of papers looking at using the controller to compensate for stiction, not only in process control, but also in other areas like robotics. There are many models for stiction - that all share the common trait that none of them can be expected to be a perfect representation of the phenomenon.

The compensation schemes are typically rather complex, finely tuned to the specifics of the stiction model used, and not very surprisingly they often work well for the same stiction model. What is lacking is the demonstration of any sort of robustness for the compensation scheme. In a simulation study one could at least use a different model for the 'system' than the stiction model used in designing the controller. The practical usefulness of such stiction compensation schemes are therefore at best not proven.

Industrial practitioners report that use of derivative action often has some positive effect on stiction. However, derivative control action may not be suitable for all control loops, and there is also the question whether it should be placed in the main controller or in the valve positioner. Some further work in this area may therefore be warranted.

Other practical approaches to managing control problems due to stiction, include changing the controller to a pure P controller, or introducing a deadband in the integrating term (only integrate when the offset is larger than the deadband). This may reduce or remove the oscillations, but have their own detrimental effects on control performance. These approaches are therefore mainly short-term modifications until valve maintenance can be performed.

8.3.4 Detection of backlash

Backlash is a particular type of hysteresis that occurs when the direction of movement changes for the input. The input then has to travel through the deadband before any change is detected at the output[2].

In a recent paper, Hägglund[30] proposes a method for on-line estimation of the deadband. Using describing function analysis, it is shown that an integrating system controlled with an integrating controller will exhibit oscillations in the presence of backlash. These oscillations are typically quite fast and of significant amplitude, and will therefore be detected by an appropriate oscillation detection method.

Asymptotically stable processes with integrating controllers, on the other hand, will typically not show pronounced oscillations, but rather drift relatively slowly around the setpoint. This results in slow, low amplitude oscillations that often will not be detected by oscillation detection methods. Hägglund's deadband estimation method is developed for this kind of systems. It uses the control loop measurement,

[2]Sometimes the words *backlash* and *deadband* are used as synonyms. Here the *deadband* refers to the width of the backlash.

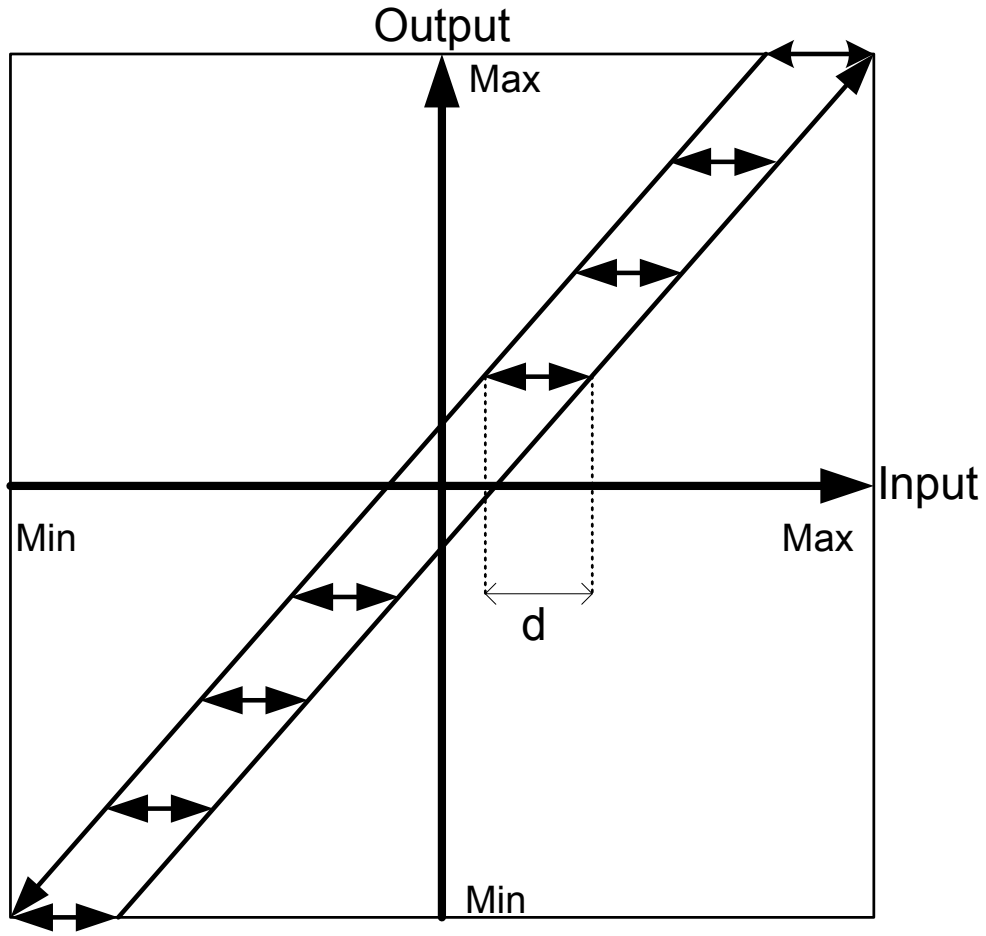

Figure 8.7: Illustration of backlash with deadband of width d.

filtered by a second order low pass filter to reduce the effect of measurement noise. The filtered loop measurement is denoted y_f. The slow oscillations are typically at a frequency lower than the plant dynamics, and hence the plant model is represented by the steady state gain K_p. The controller is assumed to be a PI controller with proportional gain K and integral time T_i. The plant gain K_p and the controller gain K are assumed to be given in compatible units (such that their product is 1 and dimensionless).

The filtered control error is given as $e = y_{sp} - y_f$, where y_{sp} is the setpoint (or reference) for the control loop. Let t_i be the times when the filtered control error e changes sign. Correspondingly, $\Delta t = t_{i+1} - t_i$ denotes the time between successive zero crossings of the filtered control error. The deadband estimation is executed only when the time between these zero crossings is large, i.e., when $\Delta t \geq 5T_i$. We also define

$$\Delta y = \int_{t_i}^{t_{i+1}} |e| dt / \Delta t \tag{8.5}$$

Deltay may thus be seen as the 'average' control error between the zero crossings.

The deadband is then estimated as

$$\hat{d} = K\left(\frac{\Delta t}{T_i} - \frac{1}{KK_p}\right)\Delta y \tag{8.6}$$

This deadband estimation suffers from the fact that the steady state gain needs to be known. In many cases this will be available (although not necessarily easily available) from steady state plant simulations - even if dynamic simulation models are not available. Instead, Hägglund takes a more practical approach and argue that the deadband estimate is relatively insensitive to the value of K_p for the majority of plants. This stems from the fact that the estimation is performed only when $\Delta t \geq 5T_i$, and the observation that the product KK_p is normally larger than 0.5 (assuming a reasonable controller tuning in the absence of backlash, and that the controller tuning is not dominated by pure time delay).

For more details of implementation of the deadband estimation, the reader is referred to the original publication by Hägglund[30].

8.3.5 Backlash compensation

8.3.6 Simultaneous stiction and backlash detection

8.3.7 Discriminating between external and internally generated oscillations

8.3.8 Detecting and diagnosing other non-linearities

In his thesis, Horch [40] found no systematic method for diagnosing oscillations due to other typical non-linearities than stiction. In particular, he considered dead-band and quantization effects, but found that they had similar effects on the cross-correlation function as external disturbances. However, the observation that non-linear effects are frequent causes for poor control performance in general, and oscillations in particular, leads to the conclusion that it is valuable to detect non-linear behaviour, even if one is not able to diagnose the type or cause of the non-linearity. This is the approach taken by Thornhill and coworkers [100, 15]. In [100], two measures are used to quantify non-linearity, a distortion factor D and a measure N based on non-linear time series analysis.

The distortion factor D compares the total power in the fundamental oscillation frequency *and* the harmonics to the power in the fundamental frequency alone. The calculation of D requires manual inspection of the power spectrum to determine the appropriate frequency range for the fundamental oscillation. Note that if several variables in a plant oscillate due to a common cause, the fundamental oscillating frequency will be the same for all these variables. The selection of an appropriate frequency range for evaluating D is therefore not an onerous task. D cannot be determined in cases with no well-defined oscillation and no spectral peak.

The measure N based on non-linear time-series analysis is based on the observation that the statistical description of the output of a linear system affected by normally distributed input signals is fully defined by its first and second order statistics (i.e., mean, variance, autocorrelation function). The idea is therefore to generate time series that could be the output of a linear system, but with the same first and second order statistics as the signal in question. Such time series are called *surrogate time series*, see the paper by Schreiber and Schmitz [84] for details. Next, one has to select a measure of non-linearity and calculate this measure for both the signal in question and the surrogates. Finally, hypothesis testing is used to assess whether the signal in question is significantly different from the surrogates (which would be an indication of non-linearity). Thornhill et al. [100] measured non-linearity in terms of the '*r.m.s. value of the error from zero-order non-linear prediction using matching of nearest neighbors in an m-dimensional phase space*'. This error is expected to be lower for a non-linear signal than for an arbitrary linear signal with the same first and second order statistics. A '*zero-order non-linear prediction using matching of nearest neighbors in an m-dimensional phase space*' essentially means the following:

1. A $m-$dimensional 'phase space' for the signal is established, where each point in that space are defined by the most recent and $(m-1)$ earlier observations of the signal. These m observations should be evenly spaced in time, but they do not necessarily have to be consequtive.

2. The time series is searched for neighboring points in this phase space.

3. The zero-order prediction of the next signal value is simply the mean of the next signal value for the neighboring points.

Tuning variables in for this non-linearity measure will be the dimension m of the phase space and the number of nearest neighbors to use (or, alternatively, the distacne from the present point within which the neighbors must lie).

In [100], it is shown that both D and N can be used successfully to detect non-linearity. N appears to be more reliable than D, but is also significanlty more computationally expensive.

In [15], the *bispectrum* is used to detect non-linearity. The bispectrum measures interaction between two frequencies, and is expected to be flat for a linear signal. Significant peaks and troughs in the bispectrum is therefore an indication of non-linearity. Whether commonly occuring non-linearities have easily identifiable bispecral shapes has apparently not been addressed, and would be of interest to analyze further.

8.4 Root-cause analysis for distributed oscillations

Thornhill et al. [100] demonstrate how detection of non-linearity can be used to locate the origin of an oscillation that affects multiple variables in a process plant, without necessarily diagnosing the cause or nature of the non-linearity. The basic idea is that

most units in a process plant has a low-pass characteristic, and will therefore tend to filter higher harmonics more than the fundamental oscillation frequency. Variables that are located far from the origin of the oscillations are therefore likely to appear 'more linear' than variables close to the origin of the oscillations. Root-cause analysis (or, rather, 'locating') then consists of first identifying groups of signals that oscillate with similar patterns, and then assessing the degree of non-linearity for the various signals within each group. Oscillations are then thought to arise at the location of the most non-linear signal.

Thornhill et al. [100] used Principal Component Analysis of the signal spectra to establish groups of signals with similar oscillation patterns. The measures D and N described in the previous section were then used as measures of the degree of non-linearity.

Whereas the ideas presented in [100] are very interesting, there seems to me little industrial experience on which to base the choice of the non-linearity measure. Although the use of surrogate data seems to be a popular (and statistically well-founded) method for detecting non-linearity, there are alternatives to the measures N and D (which does not require surrogate data) used by Thornhill et al. to quantify non-linearity. Schreiber and Schmitz [84] describe the measure

$$\phi^{rev}(\tau) = \frac{1}{N-\tau}\sum_{n=\tau+1}^{N}(s_n - s_{n-\tau})^3 \tag{8.7}$$

as particularly useful (where N is here the number of observations in the time series, and s_n is the measurement at time n), whereas Kantz and Schreiber [55] mention

$$\phi = \frac{1}{N-1}\sum_{n=1}^{N-1}(s_n s_{n+1}^2 - s_n^2 s_{n+1}) \tag{8.8}$$

Each of these measures would be less computationally intensive than the zero-order nonlinear prediction described above. In [100], the selection of the particular non-linearity measure for use with the surrogate data is not motivated, and there seems to be limited industrial experience justifying the selection of a particular non-linearity measure. There should be room for further research here, and in particular the higher order autocorrelations in Eqs. (8.5) and (8.6) should be compared to the measure N proposed by Thornhill et al.

One should also keep in mind that even linear systems can be unacceptably oscillatory, and therefore looking for non-linearity need not be a successful approach for locating the origin of oscillations in a plant. This problem is particularly difficult in multivariable systems, since the individual loops may function fine, while the oscillations are caused by interactions between the loops. This issue is also discussed in 8.3.1 above.

8.5 Control loop performance monitoring

Traditionally control loop monitoring has not received much attention, often being limited to logging whether a control loop is in automatic of manual, and logging alarms for the controlled variable in the loop. Although logging such variables and events can give valuable information about the control system, they hardly provide any diagnostic information or any 'standard' against which the actual behaviour of the control system can be measured. Autocorrelation functions and signal power spectra can also give valuable information. However, their evaluation require significant process understanding, and they therefore are not applicable for automatic performance monitoring.

8.5.1 The Harris Index

The most popular index for monitoring controller performance has been named after T. J. Harris. Control loop performance monitoring has received much attention since his publication of an influential paper on the subject [34], although similar ideas have been proposed earlier, by e.g., Fjeld [22]. The Harris' index simply compares the observed variance in the controlled variable with that theoretically could be obtained with a minimum variance controller (MVC)[3]. The observed variance is easily calculated from on-line data. The beauty of the method lies in that only modestly restrictive assumptions about the process are necessary in order to estimate the achievable variance under MVC control from available on-line data.

The necessary assumptions are:

1. The deadtime from manipulated variable u to controlled variable y must be known or estimated.

2. The process is asymptotically stable.

3. The process does not have any inverse response[4].

Assumptions 2 and 3 above may be relaxed, if a sufficiently accurate process model is available, see Tyler and Morari [101].

When assumptions 1-3 are fulfilled, a minimum variance controller may be used, and as the name says, this controller would achieve the minimum variance in the output. The minimum variance controller will not be derived here, but it is described in many textbooks on stochastic control theory. All we need is the following observations:

- No control action can influence the controlled variable before at least one deadtime has passed.

[3] As implied by its name, the minimum variance controller minimizes the variance in the controlled variable for a linear system, and hence gives a lower bound on the variance in the controlled variable.

[4] In terms of systems theory, the (discrete time) process should not have any zeros on or outside the unit disk. This corresponds to zeros in the right half plane for continuous-time systems.

- The minimum variance controller will remove all autocorrelation in the controlled variable for time lags greater than the deadtime.

Thus, if we have an impulse response model for the effect of the (unknown) disturbance on the controlled variable with the existing controller

$$y_k = \sum_{i \geq 0} h_i d_{k-i}$$

we know that h_i is unaffected by feedback for $i < \delta$, where δ is the deadtime (in number of sample intervals), whereas the minimum variance controller would achieve $h_i = 0$ for $i \geq \delta$. Thus, the minimum achievable variance in y is

$$\sigma^2_{y,mv} = (1 + h_1^2 + h_2^2 + \cdots + h^2_{\delta-1})\sigma_d^2 \tag{8.9}$$

where we have selected $h_0 = 1$, since this is equivalent to scaling the disturbance variance σ_d^2.

The Harris index provides a quantitative measure of control performance, relative to a well-defined idealized performance, while requireing a minimum of process information. The analysis is easily automated, and may be claimed to capture a significant part of the information a competent engineer could derive from the autocorrelation function. All commercial tools for control performance analysis therefore use the Harris index (or one simple modification thereof) as one of the main indicators of control performance.

8.5.2 Obtaining the impulse response model

In order to identify a model for the effect of the unknown disturbance on the controlled variable, we must first select a model structure. We will use an autoregressive (AR) model, where we assume that the disturbance d is a zero mean white noise:

$$y_k + a_1 y_{k-1} + a_2 y_{k-2} + \cdots = d_k$$

or, in terms of the *backwards shift operator* z^{-1}:

$$(1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + \cdots) y_k = A(z^{-1}) y_k = d_k$$

Now, the AR model is very simple, and one may therefore need a high order for the polynomial $A(z^{-1})$in order to obtain a reasonably good model. One therefore runs the risk of "fitting the noise" instead of modelling system dynamics. It is therefore necessary to use a data set that is much longer than the order of the polynomial $A(z^{-1})$. However, if a sufficiently large data set is used (in which there is significant variations in the controlled variable y), industrial experience indicate that acceptable models for the purpose of control loop performance monitoring is often obtained when the order of the polynomial $A(z^{-1})$ is 15-20. The AR model has the advantage that a simple least squares calculation is all that is required for finding the model,

and this calculation may even be performed recursively, i.e., it is applicable for on-line implementation. We will here only consider off-line model identification. The expected value of the disturbance d is zero, and thus we have for a polynomial $A(z^{-1})$ of order p and a data set of length N with index k denoting the most recent sample

$$\begin{bmatrix} y_{k-1} & y_{k-2} & \cdots & y_{k-p+1} & y_{k-p} \\ y_{k-2} & y_{k-3} & \cdots & y_{k-p} & y_{k-p-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ y_{k-N+p} & y_{k-N-1+p} & \cdots & y_{k-N+2} & y_{k-N+1} \\ y_{k-N-1+p} & y_{k-N-2+p} & \cdots & y_{k-N+1} & y_{k-N} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix}$$

$$= -\begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-N+p+1} \\ y_{k-N+p} \end{bmatrix} + \begin{bmatrix} d_k \\ d_{k-1} \\ \vdots \\ d_{k-N+p+1} \\ d_{k-N+p} \end{bmatrix}$$

$$\Updownarrow$$

$$Y\underline{a} = -\underline{y} + \underline{d}$$

where the underbars are used to distinguish vector-valued variables from scalar elements. The expected value of the disturbance d is zero, and thus the model is found from a least squares solution after setting $\underline{d} = 0$:

$$\underline{a} = -(Y^T Y)^{-1} Y^T \underline{y}$$

After finding $\underline{a}$, an estimate of the noise sequence is simply found from $\underline{d} = Y\underline{a} + \underline{y}$, from which an estimate of the disturbance variance σ_d^2 can be found. Having found the polynomial $A(z^{-1})$, the impulse response coefficients h_i are found from

$$y_k = \frac{1}{A(z^{-1})} d_k = H(z^{-1}) d_k$$

using polynomial long division. Here $H(z^{-1}) = 1 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} + \cdots$.

8.5.3 Calculating the Harris index

The Harris index is the ratio of the observed variance to the variance that would be obtained by MVC. The minimum achievable variance can be calculated from Eq. (8.7) above, using the identified impulse response coefficients and the estimated disturbance variance

$$\sigma_d^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left(d_i - \overline{d}\right)^2$$

where $\overline{d}$ is the mean value of the estimated disturbance, which is zero by construction.

The observed variance of the controlled variable can be computed similarly. However, if there is a persistent offset in the control loop, i.e., if the mean value of the controlled variable deviates from the reference, this should also be reflected in a measure of control quality. Hence, a modified variance should be used which accounts for this persistent offset

$$\sigma_{y,o}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - y_{ref})^2$$

If there is a persistent offset from the reference, the modified variance $\sigma_{y,o}^2$ will always be larger than the true variance σ_y^2, and the Harris index becomes

$$H_I = \frac{\sigma_{y,o}^2}{\sigma_{y,mv}^2}$$

8.5.4 Estimating the deadtime

A reasonably accurate estimate of the process deadtime is clearly a prerequisite for obtaining meaningful information from the Harris index. Sometimes such estimates may be available a priori, based on physical understanding of the process. In other cases, the deadtime must be extracted from process data. Clearly, the deadtime can be obtained from identification experiments. However, with the exception of quite small plants, the number of identification experiments required would be prohibitive, due both to an unacceptable workload and excessive process excitation. Instead, Isaksson et al. [53] propose to estimate the deadtime from closed loop data, based on data collected around the time of significant setpoint changes. Their method consists of:

1. First detect whether the control loop is close to a steady state (for details, see [53] and references therein).

2. If the loop is approximately at steady state, and a setpoint change significantly larger than the noise level of the output occurs, start collecting input-output data until the loop reaches a new steady state.

3. Fit a Laguerre filter model to the collected data.

4. Factorize the resulting model into a minimum phase and an all-pass part, where the all-pass part will contain all non-invertible zeros of the model, i.e. $G(z) = G_{mp}(z)G_{ap}(z)$.

5. The deadtime is estimated from

$$T_d = lim_{\omega \to 0} \left(-\frac{\angle G_{ap}(\omega)}{\omega} \right) \tag{8.10}$$

8.5.5 Modifications to the Harris index

Despite the theoretical elegance of the derivation of the minimum variance controller, the minimum variance controller is generally not a realistic choice for a controller in a real application. This is because it is sensitive to model errors, and may use excessive moves in the manipulated variable. It *does* provide an absolute lower bound on the theoretically achievable variance, but it is nevertheless of interest to have a control quality measure which compares the actual performance to something (hopefully) more realistic.

A simple modification to the Harris index is to simply use a too high value for the time delay, thus increasing the 'minimum' variance. This is discussed in Thornhill et al. [99] The resulting performance index will then no longer compare actual performance with a theoretically optimal performance. In [99], typical choices for the 'prediction horizons' are discussed for common control loop types in refineries (e.g., pressure control, flow control, etc.)

Another modification is to assume that the 'ideal' controller does not totally remove the effect of disturbances after one deadtime has passed, but rather that the effect of the disturbance decays as a first order function after the deadtime has passed. If we assume that this decay is described by the parameter μ ($0 < \mu < 1$), so that the ideal response to disturbances against which performance is measured would be

$$y_{k,mod} = \sum_{i=0}^{\delta-1} h_i d_{k-i} + \sum_{i=\delta}^{\infty} h_{\delta-1} \mu^{i-\delta+1} d_{k-i}$$

which results in a modified 'benchmark variance'

$$\sigma^2_{y,mod} = \sigma^2_{y,mv} + \frac{\mu^2}{1-\mu^2} \sigma^2_d$$

The modified control performance index then simply becomes

$$H_{I,mod} = \frac{\sigma^2_{y,o}}{\sigma^2_{y,mod}}$$

This modified Harris index is proposed by Horch and Isaksson [42] and Kozub [63]. Horch and Isaksson also provide some guidelines for how to specify the tuning factor μ. They find that if one wishes to account for a possible error in the estimated deadtime of ± 1 sample interval, and still require a gain margin of 2 for the 'ideal closed loop', this corresponds to choosing $\mu > 0.5$. It is also recommended to have a realistic attitude to how much the dynamics of the closed loop system can be speeded up, compared to the dynamics of the open loop process. Horch and Isaksson argue

that it is unrealistic to speed up the system by a factor of more than 2-4[5]. If we denote the open loop dominant time constant τ_{ol}, and the desired closed loop time constant is τ_{ol}/v, then the parameter μ should be chosen as

$$\mu = \exp\left(-\frac{vT_s}{\tau_{ol}}\right)$$

where T_s is the sampling interval for the control system.

8.5.6 Assessing feedforward control

The time series analysis behind the Harris index can also be extended to cases with feedforward control from measured disturbances. In cases where disturbances are measurable, but not used for feedforward control, the analysis can be used to quantify the potential benefit (in terms of variance reduction) from implementing a feedforward controller. This is described by Desborough and Harris in [18] The analysis requires knowledge of the deadtimes from measured disturbances to controlled variable in addition to the deadtime from the manipulated variable to the controlled variable[6]. Their analysis results in an Analysis of Variance table, which shows how much of the observed variance is due to the unavoidable minimum variance, and what fractions of the excess variance is affected by feedback control alone, how much is affected by feedforward control alone, and how much is affected by both feedback and feedforward control.

In a related paper, Stanfelj et al. [96] address the analysis of the cause for poor performance, and show how to determine whether it is due to poor feedforward or feedback control. If the cause is poor feedback control, it is sometimes possible to determine whether it is due to poor tuning, or due to errors in the process model. This obvioulsy requires that a (nominal) process model is available, in contrast with the analysis of Desborough and Harris which only requires the knowledge of deadtimes. Reliable model quality assessment also requires some external excitation of the control loop, typically via controller setpoint changes.

8.5.7 Comments on the use of the Harris index

[5]While this argument is reasonable for many control loops, it is obviously incorrect for integrating processes (e.g., levle control), where the open loop time constant is infinite. Ideally, one should base an estimate of the achievable bandwidth on more fundamental system properties like time delays, inverse response, or limitations in the manipulated variables.

[6]The deadtime from measured disturbances to the controlled variables should be possible to identify from closed loop data, given a data segment with significant variations in the measured disturbance. If the identified deadtime is equal to or higher than the time delay from manipulated to controlled variable, the measured disturbance does not contribute to variance in the controlled variable under minimum variance control.

Before screening for poorly performing loops using the Harris index (or preferably the modified version presented above), one should first remove any persistently oscillationg loops, as these will certainly require attention.

It is important to understand that an underlying assumption when using the Harris index is that small variance of the controlled variable is actually desired. Whereas this is normally the case, it is not always so. For example, for buffer tanks used to filter liquid flowrate disturbances, one actually desires the control to be as slow as possible. This means that the control should stabilize the liquid level and keep the tank from overflowing or emptying, but otherwise change the outlet flowrate as slowly as possible. Perfect level control would require the outlet flowrate to equal the inlet flowrate, and thus no flowrate filtering would be obtained.

Furthermore, onse should realize that the Harris index is a *relative* measure of control quality. Thus, if a process is modified to improve controllability, e.g., by installing a new measurement with less deadtime, the Harris index may well get worse even if the actual performance improves significantly. This is of course because the observed variances before and after the process modifications are not compared against the same minimum variance.

The Harris index is applicable to systems where the deadtime is the main factor limiting bandwidth and control performance. It was mentioned earlier that there are available modifications which allow consistent assessment of loops controlling an unstable process, or processes with inverse response (zero outside the unit disc). However, these modifications require much more detailed process knowledge than the basic Harris index. Similarly, the Harris index is not appliccable to control loops where the manipulated variable is in saturation much of the time, since no controller could then reduce variance in the controlled variable (i.e., comparison with a MVC controller becomes meaningless). Consistently saturating inputs would have to be resolved by other means, e.g.

- Resolving conflicts between control loops by changing control structures.
- Modifying the process to reduce the size of disturbances.
- Installing manipulated variables with a larger operating range.

Despite these limitations, the Harris index is applicable to many control loops in most chemical processes. Deterministic performance indeces may in some cases be desireable alternatives to the Harris index for performance assessment. In particular, measures like rise time or settling time may be easier to discuss with operators than a more complicated concept like variance. Some such measures may easily be derived from the autocorrelation function or the cross-correlation between reference and control error[7]. However, although actual performance may be assessed, it seems harder to assess how to correct for unacceptable performance, and to define an ideal

[7]To calculate this cross-correlation, it is of course a requirement that the there are significant changes in the reference in the period under investigation, i.e., the process has to be excited. The Harris index, on then other hand, can be evaluated from data obtained from passively observing process operation.

performance benchmark when using deterministic performance measures. Many control loops in the processing industries are used for regulatory purposes. Their main objective is to attenuate disturbances rather than quickly follow setpoints. For such loops a stochastic performance measure may be more relevant than a measure like rise time, which focuses on response to setpoint changes.

It appears that the Harris index, or some modification thereof, is an important part of any industrial CPM application, whereas deterministic measures play a much smaller part in industrial practice.

8.5.8 Performance monitoring for PI controllers

The minimum variance controller gives an absolute lower bound on the achievable performance. However, this performance may not be achievable with a particular controller type. In particular, most controllers in industry are of the PI or PID type. Ko and Edgar [59] has investigated the achievable performance (in terms of variance in the controlled variable) for PI controllers.

First, time series analysis of the closed loop is used to obtain the first few impulse response coefficients from (unmeasured) disturbance to the controlled variable, since the impulse response coefficients for times shorter than the process time delay will be invariant to feedback control. A low order ARIMA model is then fitted to these impulse response coefficients. Finally, the variance of the controlled variable is minimized with respect to the PI controller gain and integral time by solving an optimization problem.

Their method suffers from the fact that not only the process time delay is required to be known, but the entire plant transfer function. Furthermore, little is known about the properties of the optimization problem that is solved to find the minimum variance under PI control. Thus, there is no guarantee that the optimization problem is (quasi-)convex, or that a global minimum is found.

8.5.9 Performance monitoring for cascaded control loops

Ko and Edgar have developed a method for assessing performance of cascaded control loops. In [60] they develop expressions for minimum variance cascaded controller, considering the variance in the primary controlled variable (i.e., the outer loop) only. However, their development appears to be flawed. In example 1 in [60], they design minimum variance cascaded controllers. That design preserves input-output stability, but the control scheme is *not* internally stable; The system is unstable from the reference (or output disturbance) to the controller output of the primary controller. Although the proposed control may be feasible with a centralized implementation of the cascaded controller as a single 2-input 1-output controller, it is certainly not feasible with the conventional implementation as two stand-alone SISO controllers connected in cascade. In a CPM application, it is therefore not clear whether the minimum variance cascade controller derived by Ko and Edgar, and hence the benchmark variance used for performance assessment, will be realizable using a conventional

implementation of cascaded controllers. The resulting performance assessment may therefore be flawed, and performance assessment for cascaded control loops therefore does not appear to be completely solved.

8.6 Multivariable control performance monitoring

The concept of comparing the observed variance to the minimum variance can be extended to multivariable systems, see e.g., [33]. A complicating factor is that the minimum variance in general can not be determined based only on knowledge of the time delays in all transfer function elements, even in the absence of poles outside the unit disk or (finite) zeros outside the unit disk. Instead, the knowledge of the so-called '*interactor matrix*' is required, which contains all plant zeros at infinity (i.e., complete knowledge of the delay structure of the plant). Thus, the transfer function matrix $G(z^{-1})$ has to be factorized as

$$E(z)G(z^{-1}) = \widetilde{G}(z^{-1})$$

where $\widetilde{G}(z^{-1})$ is a delay-free matrix, containing only finite zeros, such that

$$\lim_{z^{-1}\to 0} \widetilde{G} = K$$

where K is a full rank, constant matrix. The interactor matrix $E(z)$ is a polynomial matrix such that $\det(E) = z^r$, where r is the number of infinite zeros of $G(z^{-1})$. The interactor matrix is not uniquely defined, and Huang et al. [51] observe that the optimal form of the interactor matrix depend on the application. A common form is a lower triangular interactor matrix. The use of such an interactor matrix for designing a minimum variance controller, would lead to minimum variance in the first output, whereas the variance in the second output is minimzed *subject to minimizing the variance in the first output*, etc. For multivariate performance assessment, such an ordering of the outputs according to priority appear misplaced, and Huang et al. instead proposes the use of a unitary interactor matrix. Filtering by a unitary interactor matrix leaves the spectrum of the original signal unchanged, i.e., no particular order of priority is imposed on the outputs. A weighted unitary interactor matrix can be used to give different weight to different outputs.

The determination of the interactor matrix has traditionally required the knowledge of the entire transfer function matrix. Huang et al. describe how it can be found from knowledge of the first few Markov parameter matrices of the plant. Since the delay structure is invariant under feedback control, closed loop identification can be used to determine the interactor matrix. However, even in closed loop the system has to be excited to perform the identification.

In cases where the plant has other non-invertible zeros (i.e., finite zeros outside the unit disk), Huang [49] has shown how a generalized interactor matrix can be defined and used for multivariable performance assessment. In the same way as for

monovariable performance monitoring, such non-invertible zeros need to be known a priori.

8.6.1 Assessing feedforward control in multivariable control

In a development similar to that in [18] for SISO systems, Huang et al. [52] have extended multivariable performance assessment to also account for feedforward control, or to assess the potential benefit of feedforward control when measurable disturbaces are not used for feedforward control.

8.6.2 Performance monitoring for MPC controllers

Ko and Edgar [61] try to address performance assessment for constrained model predictive control systems. However, their results are relatively trivial, since they assume that the model update in MPC is simply a conventional 'bias update'. The authors compare this to the improvement that can be obtained by using an optimal prediction of the model prediction error. Naturally, this optimal prediction will depend on the dynamics of any unmeasured disturbances. It is well known that the bias update is poorly suited for processes with slow disturbance dynamics, and the results in [61] are in this respect rather trivial - modern MPC algorithms will anyway have the capability of using more advanced model updating schemes than the 'bias update' (admittedly, many commercial MPC algorithmes use rather crude model updating techniques).

MPC controllers minimize a performance criterion online. The relative success at minimizing this criterion is therefore probably the best possible measure of MPC performance. This is actually a more complex problem to analyze than to assess whether a (multivariable) controller is close to minimzing some weighted variance in the outputs. Many MPC controllers place little importance on controlling variables that are within an acceptable operating range. Only variables that are at or outside their operational limits are actively controlled. This means that the 'weight' of the different variables are in a sense time-varying (or state dependent). This feature is quite easily captured in the formulation of the optimization problem in an MPC controller, but would vastly complicate any minimum variance based controller assessment.

Apart from the well-known problems of inappropriate model updating, which in an obscure way is re-visited in [61], a key issue is the accuracy of the model used by the MPC controller. This is essentially the issue of 'model (in)validation'. There is a substantial literature on model (in)validation, and no attempt have been made at reviewing this literature systematically. In a control performance monitoring framework, we would like to assess model quality in a passive sense, without exciting the system. This could appear to contradict the assumptions of most model (in)validation techniques, which require that the process is excited (in essence, that either an open loop or closed loop experiment is carried out).

Kammer et al. [54] propose a model invalidation technique based on the spectrum of the model prediction error. This method only requires the controller to be temporarily switched off, but beyond that no upset to the process is required. If the

process model is perfect, the spectrum of the prediction error should not change as the MPC controller is switched off (put in manual). Their approach is interesting, and require modest effort and process excitation. However, the issue of on-line model updating is not addressed in their paper, and hoping that the disturbances are well described as a stationary stochastic process both when collecting closed-loop and open loop data may be much to ask for.

Many MPC controllers have a separate 'optimization' layer, which optimizes the target values (setpoints for controlled variables, and/or 'ideal resting values for the manipulated variables) for the MPC controller. Often this optimization layer is executed at the same sampling frequency as the control layer. A reasonable conjecture is that when the model is poor, the optimization layer will send significant changes in target values to the control layer. Whether these target changes could function as 'external excitations' is an issue which should be studied further. Clearly, one cannot expect the target changes to be completely independent from the (measured or unmeasured) disturbances, but if the model is poor there may be a significant component of the target changes that is independent from the disturbances.

8.7 Some issues in the implementation of Control Performance Monitoring

There are several issues that need to be addressed when designing and/or implementing a control performance monitoring system. These include:

- *Structural issues.* For example, should the system be implemented centrally or in a decentralized manner? Some aspects of control performance monitoring, like oscillation/stiction detection, calculating the Harris index, etc., can be performed locally. While this will put higher demands on local computing power and data storage, it will reduce the requirement for transferring data over the network. On the other hand, inherently multivariable aspects like root cause analysis of distributed oscillations can only be performed centrally. Software updates are also simpler to handle with a centralized implementation. It appears that a centralized implementation is common in industrial practice. Honeywell has taken this position 'to the extreme', data is only collected locally, and is then encrypted before being trasnsmitted over the Internet to a central server for analysis.

- *Data quality.* From what source is process data obtained, and how often is it logged? Some guidelines can be found in e.g. [67] and [99]. Many process data historians use infrequent sampling and/or use irreversible data compression algorithms. This will permanently alter the statistical properties of the data, and can be very detrimental to control performance monitoring. The above references also contain recommendations for typical logging frequencies for various control loops (pressure control, flow control, etc.). On the one hand one would like frequent logging to be certain to capture all relevant process dynamics -

and possibly also allow some filtering of high-frequency noise without affecting the process dynamics. On the other hand, frequent logging - in particular when applied to hundreds or thousands of control loops - will cause high loads on the data communication network.

- *Integration into normal operating practice.* A control performance monitoring system can only succeed if it clearly contributes to making normal operation and maintenance of the plant simpler. If system configuration is complex, or significant effort is required to extract information from the CPM system, it is bound to fail in practice. Reports from the CPM system should be prepared regularly (e.g., once every day or week) and automatically, contain a prioritized list of problem areas and recommended corrective actions, and the report should automatically be sent to the responsible plant engineer.

8.8 Discussion

Common sense and reports from industry seem to agree that Control Performance Monitoring can make maintenance and optimization of process control systems much more effective. However, there are many aspects within this area for which there are few reports in the open literature of comparisons between alternative methods using real industrial data. This is the case for both stiction detection and measures of non-linearity used to locate the origin of distributed oscillations.

A relevant measure of control performance for surge-attenuating controllers (e.g., level controllers in buffer tanks) is not available. For such controllers a minimum variance based benchmark will be absurd, and there is a need for an alternative measure.

The research on root cause detection for distributed oscillations have focused on non-linearity as a cause for oscillations. There is certainly more work to be done in this area. It would be of interest to be able to diagnose other types of commonly occuring (and un-intended) non-linearities than stiction from operational data. Detecting and diagnosing valve hysteresis would appear to be of particular interest. However, inappropriate control structures, leading to severe interactions between control loops, can also cause oscillations - even if each loop works fine in on its own. Many inappropriate control structures can be identified from physical understanding of the process, if such understanding is backed up by a proper understanding of control. Automated detection and diagnosis of conflicting controls has received little attention, and it is not clear what can be found from operational data, and what requires other types of process information.

The minimum variance benchmark is often criticized, since it may not be achievable with a particular controller type, e.g., a PI controller. However, it seems unavoidable that any exact analysis based on a particular controller type (other than the minimum variance controller) will require a much more detailed process model. The modification of the Harris index described above should also make the index much more realistic. Whether process deadtime is the only factor which limits achievable

variance is another issue. Therefore, it is of interest to investigate whether event-triggered identification can be used to identify open-loop poles or zeros outside the unit disk from closed loop data, in addition to identifying the deadtime.

For multivariable systems, a minimum variance benchmark seems most appropriate for multi-loop control (i.e., decentralized control, using multiple single-loop controllers). In such a setting, the minimum variance benchmark may serve to illustrate the tradeoffs between control quality for different outputs, although the issue of restrictions in controller type becomes even more acute in such a setting.

Most multivariable controllers in the process industries are of the MPC type, for which the minimum variance benchmark will often be inappropriate, as discussed above. Model quality and model (in)validation, preferably based on closed loop data, appear to be of more relevance.

Chapter 9

Linear regression techniques applied in process control

Linear regression techniques are often applied in process control, to infer the value of an unmeasured variable from a number of available measurements. This is often termed *inferential control*. Most often this occurs when compositions are inferred from a spectrum of light absorbance data at different wavelengths (IR or NIR spectroscopy). Other examples include photo-spectroscopic measurement of quality parameters in the food industry. In such applications, there is a direct and immediate relationship between the measurements and the variable of interest, and there is no need for accounting for system dynamics, using, e.g., a Kalman filter.

In some other cases, the available measurements have dynamics that are closely linked to the dynamics of the variable of primary interest. Also in such a case it may be acceptable to ignore dynamic effects. For example, the use of temperature measurements to infer product compositions has been shown to be effective for high purity binary distillation columns. (ref. Mejdell)

However, in general one should exercise some care when using the linear regression techniques below for inferential control of dynamical systems. When it is necessary to account for dynamics, and no dynamical model is available, more conventional dynamical system identification (outside the scope of this note) would seem preferable to someone with a control background. The identified model would subsequently be used for control, typically involving state and/or parameter estimation.

Nevertheless, the cases mentioned above should illustrate that these techniques do have a natural place in process control, and a brief introduction is given here. The reader should be aware that there are numerous books devoted to this area, and the following treatment is therefore lacking both in depth and coverage. More information can be found in, e.g., ref MartensNaes, ref Braatz.

9.1 Data pre-treatment and organization

In a typical case, we wish to estimate a (difficult to measure) primary variable $\hat{y}$ using a linear combination of easily measurable secondary variables $\hat{w}$. That is, we want to

determine the terms k and k_0 in the relationship

$$\hat{y} = k^T \hat{w}^T + k_0 \tag{9.1}$$

Here k and $\hat{w}$ are transposed, because it is customary in the chemometrics literature to arrange the data from the same sample time in a row vector, and the parameter vector (here denoted k) in a column vector.

First, calibration data has to be obtained. Thus, a series of experiments are performed, and values for the secondary variables $\hat{w}$ and the primary variable $\hat{y}$ are obtained. In a process plant setting, the primary variable may be obtained from some laboratory measurement. In other cases, samples of known composition may be used to correlate absorbance spectra with composition. Experimental design should also be given some consideration, it is important that the variability in the experimental data cover the variability that can be expected in actual operation. Experimental design is another topic that is not covered by this note, there many books covering this topic (e.g., ...citation...)

The results of the experiments are arranged in a column vector for the primary variable[1], and in a matrix for the secondary variables, with one row per experiment. Thus, we end up with the data matrices

$$\hat{Y} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} \quad ; \quad \hat{W} = \begin{bmatrix} \hat{w}_{1,1} & \hat{w}_{1,2} & \cdots & \hat{w}_{1,m} \\ \hat{w}_{2,1} & \hat{w}_{2,2} & \cdots & \hat{w}_{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ \hat{w}_{n,1} & \hat{w}_{n,2} & \cdots & \hat{w}_{n,m} \end{bmatrix}$$

where the first subscript for $\hat{w}$ identifies the experiment, and the second index identifies the variable in question.

Next, the data is *centered*, by subtracting the mean value of a column from all elements in that column. The mean of $\hat{y}$ is denoted $\bar{y}$, and the mean values for $\hat{w}$ is denoted $\bar{w}$. The columns $\hat{w}$ may also be scaled, e.g., by the inverse of the standard deviation of the column, or to reflect different levels of measurement noise in different measurements. The resulting (centered and scaled) data matrices are denoted Y and W.

9.2 Ordinary Multivariable Linear Regression

We want to determine the vector k in

$$\hat{y} - \bar{y} = k^T (\hat{w}^T - \bar{w}^T) \tag{9.2}$$

allowing us to predict the value of $\hat{y}$ from available on-line measurements $\hat{w}$. Note that when the means are subtracted in (9.2), determination of k also allows us to calculate k_0 in (9.1) - if this is desired.

[1] Assuming that only a single variable is to be inferred, if several variables are to be inferred, the primary variables are arranged in a matrix, with one row per experiment.

Using the data matrices from the previous section, we want to find the vector k from

$$Y = Wk \tag{9.3}$$

It is well known that in order for (9.3) to be solvable, W must be invertible, and thus the number of samples must equal the number of measurements in w (i.e., $n = m$ above). In any case, the solution to (9.3) may be expressed as

$$k = W^{\dagger}Y \tag{9.4}$$

where the † symbol represents the *pseudo-inverse* - which is equal to the inverse of the matrix if the matrix is invertible. If $n < m$ the problem is under-determined, and the pseudo-inverse minimizes the size of k. The more common case is that $n > m$, and in this case the pseudo-inverse minimizes the error $\|Y - Wk\|_2$ [2]. For W of full column rank, the pseudo-inverse is given by

$$V^{\dagger} = (W^T W)^{-1} W^T$$

Multivariable Linear Regression does minimize the norm of the error *for the data set used in the regression*, as noted above. However, for that very same reason - it uses *all* the information available in the calibration data - it is also rather sensitive to noise in the calibration data. If there are combinations of inputs x are poorly excited in the data set, this problem may be particularly severe. The result may therefore be that the predictions obtained when using MLR are quite poor when applied to *new* data. In such cases, Principal Component Regression or Partial Least Squares may be good alternatives.

9.3 Principal Component Regression

The basic idea behind Principal Component Regression is to invert only the 'strong' data directions of W in (9.3), since inverting in the 'weak' directions is particularly sensitive to noise or errors in the data.

Here, the PCR will be presented using the Singular Value Decomposition (SVD), although alternative approaches exist which allow extracting individual factors one at the time.

Thus, the principal component regression is obtained by defining a 'pseudo-inverse' which uses only the p larger singular values of W. Let W have the SVD $W = U\Sigma V^H$, and let U_p and V_p consist of the p leading columns of U and V, respectively. Similarly, Σ_p is the $p \times p$ leading principal submatrix of Σ. This results in the following (approximate) solution to (9.3)

$$k = V_p \Sigma_p^{-1} U_p^H Y \tag{9.5}$$

Naturally, the result from PCR depends on the number of factors p. This should be chosen to include all strong directions in the data set W, but ignore weak directions

[2] Provided W has full column rank.

(identified by small singular values) which are dominated by noise and measurement error. It may be prudent to use an independent data set to test how prediction quality depends on the value of p.

In PCR or PCA (Principal Component Analysis), the columns of $T = U_p\Sigma_p$ are frequently referred to as *score vectors*, whereas the columns of $P = V^H$ are termed *loading vectors*.

9.4 Partial Least Squares

Although PCR overcomes the problem of co-linearity in the input data W, the selection of the directions used for the regression (the leading columns of U and V) does not consider whether variations in the input data W affects the output data Y. Partial Least Squares (PLS) tries to rectify this, and account for covariance between W and Y when selecting the directions used for regression. This requires a more complex procedure, and more involved notation.

There are several ways to calculate the PLS parameters. Here, the NIPALS (non-iterative partial least squares) algorithm will be briefly described. It repeatedly extracts regression vectors from the data set, and thereafter the data explained/used by these regression vectors are subtracted from the data set. Hence we will use a subscript to indicate the number of regression vectors that have already been found.

The NIPALS PLS algorithm.

1. Initialize with $Y_0 = Y$ and $W_0 = W$.

2. Select a vector in the output range (left range space) of Y_{i-1} to initialize the calculation of factors i. Often the column of Y_{i-1} with the larger variance is chosen. For a problem with a single output this gives $z_i = Y_{i-1}$.

3. Iterate until convergence:

$$\hat{m}_i = \frac{W_{i-1}^T z_i}{\|W_{i-1}^T z_i\|} \tag{9.6}$$

$$\hat{t}_i = W_{i-1}\hat{m}_i \tag{9.7}$$

$$\hat{q}_i = \frac{Y_{i-1}^T \hat{t}_i}{\|Y_{i-1}^T \hat{t}_i\|} \tag{9.8}$$

$$z_i = Y_{i-1} q_i \tag{9.9}$$

The iteration has converged when $\hat{t}_i$ does not change significantly between iterations. If Y contains only a single variable, $q_i = 1$, and no iteration is necessary.

4. Calculate loadings, and rescale to get orthonormal loading vectors

$$\begin{aligned} \hat{p}_i &= \frac{W_{i-1}^T \hat{t}_i}{\|\hat{t}_i^T \hat{t}_i\|} && (9.10)\\ p_i &= \frac{\hat{p}_i}{\|\hat{p}_i\|} && (9.11)\\ t_i &= \hat{t}_i \cdot \|\hat{p}_i\| && (9.12)\\ m_i &= \hat{m}_i \cdot \|\hat{p}_i\| && (9.13)\\ & && (9.14) \end{aligned}$$

5. Find the regression coefficient b_i for the relationship between z_i and t_i:

$$b_i = \frac{z_i^T t_i}{t_i^T t_i} \tag{9.15}$$

6. Before proceeding to the calculations for the next factor (or 'latent variable'), the data sets must be adjusted for the correlations in the data that are explained by the current factor:

$$\begin{aligned} W_i &= W_{i-1} - t_i p_i^T && (9.16)\\ Y_i &= Y_{i-1} - b_i t_i q_i && (9.17) \end{aligned}$$

7. Repeat from Step 2 if an additional factor needs to be calculated.

The approximate inverse formed by the PLS algorithm can be expressed as

$$W^\dagger = M(P^T M)^{-1} BQ \tag{9.18}$$

where the columns of M, P, and Q are calculated as explained above, and $B = \mathrm{diag}\{b_i\}$.

A point to note is that although the PLS also calculates *loadings* and *scores*, these are different from the PCR loadings and scores.

The calculations for PLS are more involved than those for PCR, but since the regression directions are based on the directions in W for which there is covariance with Y, a PLS model will often give improved predictions for the same number of factors.

9.5 Detecting anomalies using Principal Component Analysis

In addition to their use in regression, for the purpose of predicting hard-to-measure variables, principal components are also used to detect anomalies. In the context of this note, we are primarily interested in anomalies in process plant behaviour.

The PCR essentially splits the data matrix W into two parts

$$W = \tilde{W} + E \tag{9.19}$$

where $\tilde{W} = U_p \Sigma_p V_p^H$ is believed to to contain the dominant and reliable information in the data, whereas E is more unreliable and possibly dominated by noise (and is hopefully small).

In Principal Component Analysis, the analysis is performed by projecting new sample data onto the loading vectors t_i of the principal components. In this case, predicting the hard-to-measure output is not the focus, and the score vectors are therefore not used.

The PCA method can give two different indications indications of anomalous behaviour:

1. The variation *within the subspace covered by the principal components* may be abnormally large, i.e., the projections onto the loading vectors show large values. This is measured by Hotelling's T^2 statistic.

2. The variation *outside the subspace covered by the principal components* may be abnormally large, i.e., much of the data in the new sample is not covered by the principal component model. This is measured using what is known as the Q statistic.

Confidence limits can be calculated for both the T^2 and Q statistics, and these can be used to define threshold values for indicating anomalous behaviour. The interested reader should consult more specialized literature, e.g., ...cite Braatz?, PLS Toolbox. One should bear in mind, however, that the standard confidence limit calculations assume the samples to be independent. When the measurements that make up a sample are actually process plant measurements, the samples may be strongly correlated due to the plant dynamics. This should be considered when collecting data to construct the PCA model. In addition to *detecting* anomalous behaviour, inspection of the loadings may also assist in *diagnosis* of the anomaly, if understanding of the plant is combined with the knowledge of how the individual measurements contribute to the different loading vectors. Again, more specialized literature should be consulted.

Acknowledgement

The author is grateful to Alf Isaksson and Alexander Horch of ABB for being allowed to use some of their figures in this report.

Appendices

A1 Fourier-Motzkin elimination

Fourier-Motzkin elimination is a method for eliminating variables from sets of linear inequalities, or, equivalently, to project polyhedra described by linear inequalities onto lower dimensional subspaces. It has been described as 'Gauss-Jordan elimination applied to inequalities', and although less well known is equally simple. For illustration consider the two inequalities

$$a_{11}x_1 + a_{12}x_2 \leq b_1 \tag{A1.1}$$
$$a_{21}x_1 + a_{22}x_2 \leq b_2 \tag{A1.2}$$

Multiplying (A1.1) and (A1.2) with non-negative constants λ_1 and λ_2, respectively, and adding the resulting inequalities, results in the inequality

$$(\lambda_1 a_{11} + \lambda_2 a_{21})x_1 + (\lambda_1 a_{12} + \lambda_2 a_{22})x_2 \leq \lambda_1 b_1 + \lambda_2 b_2 \tag{A1.3}$$

Clearly, (A1.3) is also a valid inequality. If, a_{11} and a_{21} have opposite signs, λ_1 and λ_2 can be chosen to eliminate x_1 from (A1.3). Next, the procedure above will be generalized to arbitrary dimensions.

Problem A1.1. Consider the set Ξ_1 described by the linear inequalities

$$A_1 x_1 + A_2 x_2 \leq b \tag{A1.4}$$

where $x_1 \in R$, $x_2 \in R^r$, $b \in R^q$, and A_1 and A_2 are of consistent dimensions. Find the corresponding set

$$\Xi_2 = \{\mathcal{A}_\xi x_2 \leq b_\xi\}$$

such that $\forall x_2 \in \Xi_2 \exists x_1 \in R$ for which (A1.4) is fulfilled.

Algorithm A1.2. Solution to Problem A1.1: The Fourier-Motzkin elimination procedure.

1. Group the inequalities in (A1.4) in three subsets

   s^0: Inequalities for which the corresponding element of A_1 in (A1.4) is zero.

   s^1: Inequalities for which the corresponding element of A_1 in (A1.4) is positive.

   s^2: Inequalities for which the corresponding element of A_1 in (A1.4) is negative.

2. Form the set of inequalities s^{12} as follows:

   i) Take one inequality from s^1 and one inequality from s^2.

   ii) Multiply the two inequalities by appropriate positive constants, and add the results to form a new inequality in which x_1 does not appear.

iii) Include this new inequality in the set s^{12}.

iv) Repeat *i* - *iii* above for all possible pairs of inequalities from s^1 and s^2.

3. The set Ξ_2 is then described by the inequalities in s^0 and s^{12}.

Eliminating more than one variable from the inequalities is done by repeated application of Algorithm A.1.2. Note that many of the resulting inequalities describing the set Ξ_2 may be redundant. If a minimal set of inequalities is desired, removing redundant constraints will therefore be necessary.

Example A1.1.
Consider the set of linear inequalities

$$\begin{aligned} x_1 &\leq 2 && \text{(A1.5)} \\ -x_1 &\leq 2 && \text{(A1.6)} \\ x_2 &\leq 2 && \text{(A1.7)} \\ -x_2 &\leq 2 && \text{(A1.8)} \\ x_1 - x_2 &\leq \frac{3}{4} && \text{(A1.9)} \\ -\frac{1}{3}x_1 + x_2 &\leq 2 && \text{(A1.10)} \\ -3x1 - x2 &\leq 3 && \text{(A1.11)} \end{aligned}$$

what are the values of x_1 for which there exists a feasible x_2? These inequalities are illustrated in Fig. A1, from which it is simple to identify the range of values for x_1 for which a feasible value of x_2 can be found. Clearly, $s^0 \in \{(A1.5), (A1.6)\}$, $s^1 \in \{(A1.7), (A1.10)\}$, and $s^2 \in \{(A1.8), (A1.9), (A1.11)\}$. Forming the set s^{12} as described above, we get the following set of inequalities:

Combining (A1.7) and (A1.9): $x_1 \leq \frac{11}{3}$

Combining (A1.7) and (A1.11): $-3x_1 \leq 5$

Combining (A1.10) and (A1.8): $-\frac{1}{3}x_1 \leq 4$

Combining (A1.10) and (A1.9): $\frac{2}{3}x_1 \leq \frac{11}{3}$

Combining (A1.10) and (A1.11): $-\frac{10}{3}x_1 \leq 5$

The combination of inequalities (A1.7) and (A1.8) resiults in $0 \leq 0$, which is trivially always fulfilled. Forming the set Ξ_2 from s^0 and s^{12}, and removing redundant constraints, we find that the feasible region for x_1 is given by $-\frac{3}{2} \leq x_1 \leq 2$, which agrees with what we find from Fig. A1.

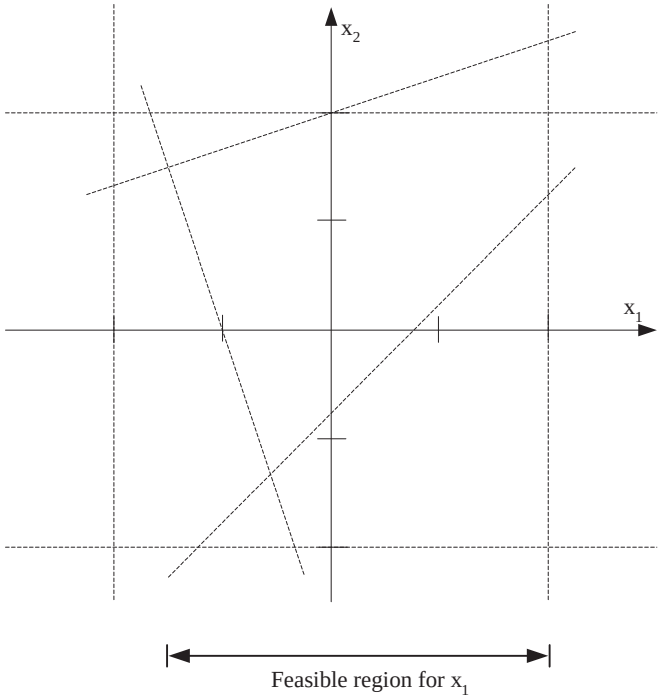

Figure A1: Feasible region for x_1 for Example 1.

A2 Removal of redundant constraints

The Fourier-Motzkin elimination described above results in many redundant constraints, which are superfluous in any application. Likewise, calculation of the maximal output admissible set also requires checking whether constraints are redundant.

We will here use an adaptation of the procedure proposed in [72], due to its conceptual and mathematical simplicity. No claims are made about the computational efficiency of the procedure. However, in an MPC setting, checking for constraint redundancy is generally done at the design stage, i.e., *offline*, when there is typically no strict limitations on available computation time.

We start from a bounded convex polyhedron described by q linear inequalities

$$\Xi = \{x_k | \mathcal{A}x_k \leq b\} \tag{A2.1}$$

A new constraint $A_c x_k \leq b_c$ is redundant if it can be added to the original set of constraints without altering the set, that is

$$\mathcal{A}x_k \leq b \Rightarrow A_c x_k \leq b_c \tag{A2.2}$$

This is checked with a simple LP:

$$m = \max_{x_k} A_c x_k \tag{A2.3}$$

$$s.t. \quad \mathcal{A}x_k \leq b \tag{A2.4}$$

If $m \leq b_c$ then the constraint $A_c x_k \leq b_c$ is redundant.

Applying the above method for redundancy checking does not necessarily guarantee that the final set of constraints is minimal (i.e., does not contain any redundant constraints), since adding new constraints may make some of the original constraints redundant. However, in applications where it is important to minimize the number of constraints in the optimization formulation, it is trivial to modify the redundancy checking method above to identify any redundant constraints.

A3 The Singular Value Decomposition

The SVD allows any matrix A of dimension $r \times c$ to be decomposed into three matrices

$$A = U\Sigma V^H \tag{A2.5}$$

where U has dimension $r \times r$, Σ is of dimension $r \times c$, V is of dimension $c \times c$, and the superscript H denotes the complex conjugate transpose (which is the same as the transpose for real valued matrices).

The matrices U and V are orthonormal, i.e., $U^H U = UU^H = I$, $V^H V = VV^H = I$, whereas Σ is real valued with non-zero elements only on the main diagonal. By convention, the elements on the diagonal of Σ are arranged in descending order. These diagonal elements of Σ are termed *singular values*, singular value number i is commonly denoted σ_i, and the largest and smallest singular value are denoted $\bar{\sigma}$ and $\underline{\sigma}$, respectively.

The fact that U and V are orthonormal, have a few immediate consequences:

- The determinant and singular values of a matrix are related through

$$|\det(A)| = \prod_i \sigma_i(A) \tag{A2.6}$$

- The SVD provides an expression for any matrix as a sum of rank-one matrices

$$A = \sum_i u_i \sigma_i v_i^H \tag{A2.7}$$

  where u_i and v_i are column vectors equal to column i of U and V, respectively.

- An expression for the inverse for the matrix (and the SVD of the inverse) is easily obtained

$$A^{-1} = V\Sigma^{-1}U^H \tag{A2.8}$$

  For rank defect matrices, the pseudo-inverse is similarly obtained by inverting only the non-zero singular values in Σ.

Bibliography

[1] J. A. E. Bryson and Y.-C. Ho. *Applied Optimal Control*. Blaisdell Publishing Company, Waltham, Massachusetts, USA, 1969.

[2] V. Alstad. *Studies on selection of controlled variables*. PhD thesis, Norwegian University of Science and Technology, Dept. of Chemical Engineering, March 2005.

[3] V. Alstad and S. Skogestad. Null space method for selecting measurement combinations as controlled variables. *Ind. Eng. Chem. Res.*, 46:846–853, 2007.

[4] V. Alstad, S. Skogestad, and E. S. Hori. Optimal measurement combinations as controlled variables. *Journal of Process Control*, 19:138–148, 2009.

[5] E. M. B. Aske. *Design of plantwide control systems with focus on maximising throughputs*. PhD thesis, Norwegian University of Science and Technology, Dept. of Chemical Engineering, March 2009.

[6] K. Åström and T. Hägglund. Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica*, 20(5):645–651, 1984.

[7] K. J. Åström and t. . C. p. . P. a. . E. B. Wittenmark, year = 1984.

[8] K. J. Åström and T. Hägglund. *Automatic Tuning of PID Controllers*. ISA, Research Triangle Park, NC, USA, 1988.

[9] K. J. Åström, C. C. Hang, P. Persson, and W. K. Ho. Towards intelligent PID control. *Automatica*, 28(1):1–9, 1992.

[10] J. G. Balchen. The stability of 2x2 multivariable control systems. *Modeling, Identification and Control*, 11:97–108, 1990.

[11] J. G. Balchen and K. I. Mummé. *Process Control. Structures and Applications.* Blackie Academic & Professional, Glasgow, Scotland, 1988.

[12] H. W. Bode. *Network Analysis and Feedback Amplifier Design*. Van Nostrand, Princeton, New Jersey, USA., 1945.

[13] R. D. Braatz and M. Morari. Minimizing the euclidian condition number. *SIAM Journal on Control and Optimization*, 32(6):1763–1768, 1994.

[14] E. H. Bristol. On a new measure of interactions for multivariable process control. *IEEE Transactions on Automatic Control*, AC-11:133–134, 1966.

[15] M. A. A. S. Choudhury, S. L. Shah, and N. F. Thornhill. Detection and diagnosis of system nonlinearities using higher order statistics. In *Accepted for IFAC World Congress*, 2002.

[16] G. Cohen and G. Coon. Theoretical considerations of retarded control. *Trans. ASME*, 75:827–834, 1953.

[17] N. M. C. de Olivieira and L. T. Biegler. Constraint handling and stability properties of model-predictive control. *AIChE Journal*, 40(7):1138–1155, July 1994.

[18] L. Desborough and T. Harris. Performance assessment measures for univariate Feedforward/Feedback control. *Canadian J. of Chemical Engineering*, pages 605–616, 1993.

[19] L. Desborough, P. Nordh, and R. Miller. Process out of control. *INTECH*, (No. 8):52–55, 2001.

[20] G. Evensen. Sequential data assimilation with nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99:143–162, 1994.

[21] G. Evensen. The ensemble kalman filter: Theoretical formulation and practical implementation. *Ocean Dynamics*, 53:343–367, 2003.

[22] M. Fjeld. On-line modelling of the performance of control systems. Annual Pittsburgh conference on modeling and simulation, 1981.

[23] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 2nd edition, 1987.

[24] K. Forsman and A. Stattin. A new criterion for detecting oscillations in control loops. In *Proceedings of the European Control Conference*, 1999.

[25] J. S. Freudenberg and D. P. Looze. *Frequency Domain Properties of Scalar and Multivariable Feedback Systems.* Springer-Verlag, Berlin, Germany, 1988.

[26] C. E. Garcia and A. M. Morshedi. Quadratic programming solution of dynamic matrix control (QDMC). *Chem. Eng. Commun.*, pages 73–87, 1986.

[27] E. Gilbert and K. Tan. Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Trans. Autom. Contr.*, 36:1008–1020, 1991.

[28] P. Grosdidier, M. Morari, and B. R. Holt. Closed-loop properties from steady-state gain information. *Industrial and Engineering Chemistry Process Design and Development*, 24:221–235, 1985.

[29] T. Hägglund. A control-loop performance monitor. *Control Eng. Practice*, 3(11):1543–1551, 1995.

[30] T. Hägglund. Automatic on-line estimation of backlash in control loops. *Journal of Process Control*, 17:489–499, 2007.

[31] I. J. Halvorsen, S. Skogestad, M. J. C., and A. V. Optimal selection of controlled variables. *Ind. Eng. Chem. Res.*, pages 3273–3284, 2003.

[32] R. Hanus, M. Kinnaert, and J.-L. Henrotte. Conditioning technique, a general anti-windup and bumpless transfer method. *Automatica*, 23(6):729–739, 1987.

[33] T. Harris, F. Boudreau, and J. F. MacGregor. Performance assessment of multivariable feedback controllers. *Automatica*, pages 1505–1518, 1996.

[34] T. J. Harris. Assessment of control loop performance. *Can. J. Chem. Eng.*, 67:856–861, October 1989.

[35] T. J. Harris, C. T. Seppala, and L. D. Desborough. A review of performance monitoring and assessment techniques for univariate and multivariate control systems. *J. of Process Control*, pages 1–17, 1999.

[36] K. Havre. *Studies on Controllability Analysis and Control Structure Design.* PhD thesis, Norwegian University of Science and Technology, 1998.

[37] B. R. Holt and M. Morari. Design of resilient processing plants v - the effect of deadtime on dynamic resilience. *Chemical Engineering Science*, 40:1229–1237, 1985.

[38] B. R. Holt and M. Morari. Design of resilient processing plants VI - the effect of right half plane zeros on dynamic resilience. *Chemical Engineering Science*, 40:59–74, 1985.

[39] A. Horch. A simple method for oscillation diagnosis in process control loops. *Control Engineering Practice*, pages 1221–1231, 1999.

[40] A. Horch. *Condition Monitoring of Control Loops.* PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2000.

[41] A. Horch and A. Isaksson. Detection of stiction in integrating processes. In *Proceedings of the European Control Conference*, pages 1327–1332, 2001.

[42] A. Horch and A. J. Isaksson. A modified index for control performance assessment. In *Proceedings of the American Control Conference*, pages 3430–3434, 1998.

[43] M. Hovd. *Studies on Control Structure Selection and Design of Robust Decentralized and SVD Controllers.* PhD thesis, Department of Chemical Engineering, University of Trondheim-NTH, 1992.

[44] M. Hovd. Feasible model predictive control with bounded disturbances. In *Preprints IFAC Symposium ADCHEM 2006*, pages 117–122, 2006.

[45] M. Hovd and R. D. Braatz. Handling state and output constraints in MPC using time-dependent weights. In *Proceedings of the American Control Conference*, 2001.

[46] M. Hovd, J. H. Lee, and M. Morari. Truncated step response models for model predictive control. *J. Proc. Cont.*, 3(2):67–73, 1993.

[47] M. Hovd and S. Skogestad. Improved independent design of robust decentralized controllers. *J. Proc. Cont.*, 3(1):43–51, 1993.

[48] M. Hovd and S. Skogestad. Procedure for regulatory control structure selection with application to the fcc process. *AIChE Journal*, 39(12):1938–1953, 1993.

[49] B. Huang. *Multivariate Statistical Methods for Performance Assessment*. PhD thesis, Department of Chemical Engineering, University of Alberta, 1997.

[50] B. Huang and S. L. Shah. *Performance Assessment of Control Loops. Theory & Applications*. Springer, 1999.

[51] B. Huang, S. L. Shah, and H. Fujii. The unitary interactor matrix and its determination using closed-loop data. *J. of Process Control*, pages 195–207, 1997.

[52] B. Huang, S. L. Shah, and R. Miller. Feedforward plus feedback controller performance assessment of MIMO systems. *IEEE Transactions on Control Systems Technology*, pages 580–587, 2000.

[53] A. Isaksson, A. Horch, and G. A. Dumont. Event-triggered deadtime estimation from closed-loop data. In *Proceedings of the American Control Conference*, pages 3280–3285, 2001.

[54] L. Kammer, D. Gorinevsky, and G. A. Dumont. Semi-intrusive multivariable model invalidation. In *Proceedings of the European Control Conference*, pages 2487–2492, 2001.

[55] H. Kantz and T. Schreiber. *Nonlinear Time Series Analysis*. Cambridge Nonlinear Science Series. Cambridge University Press, 1997.

[56] V. Kariwala, Y. Cao, and S. Janardhanan. Local self-optimizing control with average loss minimization. *Ind. Eng. Chem. Res.*, 47:1150–1158, 2008.

[57] V. Kariwala and S. Skogestad. Branch and bound methods for control structure design. In *Proc. 16th European Symposium on Computer Aided Process Engineering (ESCAPE)*, Garmish-Partenkirchen, Germany, 2006.

[58] E. C. Kerrigan and J. M. Maciejowski. Soft constraints and exact penalty functions in model predictive control. In *Proceedings of the UKACC International Conference (CONTROL 2000)*, 2000.

[59] B.-S. Ko and T. F. Edgar. Assessment of achievable PI control performance for linear processes with dead time. In *Proceedings of the American Control Conference*, pages 1548–1552, 1998.

[60] B.-S. Ko and T. F. Edgar. Performance assessment for cascade control loops. *AIChE Journal*, pages 281–291, 2000.

[61] B.-S. Ko and T. F. Edgar. Performance assessment of constrained model predictive control systems. *AIChE Journal*, pages 1363–1371, 2001.

[62] I. Kolmanovsky and E. G. Gilbert. Maximal output admissible sets for discrete-time systems with disturbance inputs. In *Proceedings of the American Control Conference*, pages 1995–1999, 1995.

[63] D. J. Kozub. Controller performance monitoring and diagnosis: Experiences and challenges. In *Proceedings of 5th Chemical Process Control Conference*, pages 83–96, Tahoe City, California, 1996.

[64] D. J. Kozub and C. Seppala. Book review: Performance assessment of control loops. theory & applications. *J. of Process Control*, pages 441–442, 2001.

[65] T. Larsson and S. Skogestad. Plantwide control - a review and a new design procedure. *Modeling, Identification and Control*, 21:209–240, 2000.

[66] T. Miao and D. E. Seborg. Automatic detection of excessively oscillating feedback control loops. In *IEEE Conference on Control Applications - Proceedings, Vol. 1.*, pages 359–364, 1999.

[67] R. Miller and L. Desborough. Web-enabled control loop assessment. *Chemical Engineering*, pages 76–79, May 2000.

[68] M. Moiso and J. Piiponen. Control loop performance evaluation. In *Proceedings of Control Systems*, pages 251–258, 1998.

[69] M. Morari and E. Zafiriou. *Robust Process Control*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, USA., 1989.

[70] K. R. Muske. Steady-state target optimization in linear model predictive control. In *Proc. American Control Conference*, pages 3597–3601, 1997.

[71] K. R. Muske and T. A. Badgwell. Disturbance modeling for offset-free linear model predictive control. *Journal of Process Control*, 12:617–632, 2002.

[72] P. T. ndel. *Constrained Optimal Control via Multiparametric Quadratic Programming*. PhD thesis, Engineering Cybernetics Department, NTNU, 2003.

[73] C. N. Nett and V. Manousiouthakis. Euclidian condition and block relative gain - connections, conjectures and clarifications. *IEEE Transactions on Automatic Control*, AC-32(5):405–407, 1987.

[74] A. Niederlinski. A heuristic approach to the design of linear multivariable interacting control systems. *Automatica*, 7:691–701, 1971.

[75] J. G. Owen, D. Read, H. Blekkenhors, and A. A. Roche. A mill prototype for automatic monitoring of control loop performance. In *Control Systems '96*, pages 171–177, 1996.

[76] R. Price and C. Georgakis. Plantwide regulatory control design procedure using a tiered framework. *Industrial and Engineering Chemistry Research*, 32:2693–2705, 1993.

[77] R. Price, P. R. Lyman, and C. Georgakis. Throughput manipulation in plantwide control structures. *Industrial and Engineering Chemistry Research*, 33:1197–1207, 1994.

[78] S. J. Qin. Control performance monitoring - a review and assesment. *Computers and Chemical Engineering*, pages 173–186, 1998.

[79] C. V. Rao, J. B. Rawlings, and J. H. Lee. Constrained linear state estimation - a moving horizon approach. *Automatica*, 37:1619–1629, 2001.

[80] J. E. Rijnsdorp. Interaction in two-variable control systems for distillation columns-i. *Automatica*, 1:15–28, 1965.

[81] D. E. Rivera, M. Morari, and S. Skogestad. Internal model control. 4. PID controller design. *Ind. Eng. Chem. Process Des. Dev.*, 25:252–265, 1986.

[82] N. Sandell, P. Varaiya, M. Athans, and M. G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Trans. on Automatic Control*, AC-32(2):108–128, 1978.

[83] T. Schei. A method for closed-loop automatic tuning of PID controllers. *Automatica*, 28(3):587–591, 1992.

[84] T. Schreiber and A. Schmitz. Surrogate time series. *Physica D*, pages 346–382, 2000.

[85] P. O. M. Scokaert and J. B. Rawlings. Feasibility issues in linear model predictive control. *AIChE Journal*, 45(8):1649–1659, August 1999.

[86] S. L. Shah, R. Patwardhen, and B. Huang. Multivariate controller performance analysis: Methods, applications and challenges. In *Proceedings of 6th Chemical Process Control Conference*, 2001.

[87] D. Simon. *Optimal State Estimation*. John Wiley & Sons Inc., Hoboken, New Jersey, USA, 2006.

[88] S. Skogestad. Plantwide contro: The search for the self-optimizing control structure. *Journal of Process Control*, 10:487–507, 2000.

[89] S. Skogestad. Control structure design for complete chemical plants. *Computers and Chemical Engineering*, 28:219–234, 2004.

[90] S. Skogestad and M. Morari. Implications of large RGA elements on control performance. *Industrial and Engineering Chemistry Research*, 26:2323–2330, 1987.

[91] S. Skogestad and M. Morari. Robust performance of decentralized control systems by independent designs. *Automatica*, 25(1):119–125, 1989.

[92] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control. Analysis and Design.* John Wiley & Sons Ltd, Chichester, England, 1996.

[93] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control. Analysis and Design.* John Wiley & Sons Ltd, Chichester, England, 2005.

[94] D. Sourlas, T. Edgar, and V. Manousiouthakis. Best achievable low order decentralized performance. In *Proceedings of the American Control Conference*, Baltimore, Maryland, June 1994.

[95] D. D. Sourlas and V. Manousiouthakis. Best achievable decentralized performance. *IEEE Transactions on Automatic Control*, 40(11):1858–1871, 1995.

[96] N. Stanfelj, T. E. Marling, and J. F. MacGregor. Monitoring and diagnosing process control performance: The single-loop case. In *Proceedings of the American Control Conference*, pages 2886–2892, 1991.

[97] A. P. Swanda and D. E. Seborg. Evaluating the performance of PID-type feedback control loops using normalized settling time. In *Preprints ADCHEM*, pages 283–288, 1997.

[98] N. F. Thornhill and T. Hägglund. Detection and diagnosis of oscillation in control loops. *Control Eng. Practice*, pages 1343–1354, 1997.

[99] N. F. Thornhill, M. Oettinger, and P. Fedenczuk. Refinery-wide control loop performance assessment. *J. of Process Control*, pages 109–124, 1999.

[100] N. F. Thornhill, S. L. Shah, and B. Huang. Detection of distributed oscillations and root-cause diagnosis. In *Preprints 4th IFAC Workshop on On-Line Fault Detection and Supervision in the Chemical Process Industries*, pages 167–172, 2001.

[101] M. L. Tyler and M. Morari. Performance assessment for unstable and nonminimum-phase systems. Technical report, California Institute of Technology, 1995. IfA-Report no. 95-03.

[102] M. L. Tyler and M. Morari. Performance monitoring of control systems using likelihood methods. *Automatica*, pages 1145–1162, 1996.

[103] J. Vada. *Prioritized Infeasibility Handling in Linear Model Predictive Control: Optimality and Efficiency*. PhD thesis, Engineering Cybernetics Department, NTNU, 2000.

[104] E. Zafiriou and A. L. Marchal. Stability of SISO quadratic dynamic matrix control with hard output constraints. *AIChE Journal*, 37(10):1550–1560, October 1991.

[105] J. Ziegler and N. Nichols. Optimum settings for automatic controllers. *Trans. ASME*, 64:759–768, 1942.