Predictive Control

A Unified Approach

Predictive Control

A Unified Approach

Proefschrift

ter verkrijging van de graad van
doctor aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus,
prof. drs. P.A. Schenck,
in het openbaar te verdedigen
ten overstaan van een commissie aangewezen
door het College van Dekanen
op donderdag 22 november 1990 te 14.00 uur

door

Aloisius Ronald Marie Soeterboek,

geboren te Rotterdam,
Elektrotechnisch ingenieur

Dit proefschrift is goedgekeurd door de promotoren:
Prof. ir. H.B. Verbruggen
Prof. dr. ir. P.P.J. van den Bosch

Contents

Chapter 1

Introduction

The concept of predictive control was introduced simultaneously by Richalet [1] and Cutler and Ramaker [2] in the late seventies. Since then many papers have been written on the subject of predictive control. See, for example, [3, 4, 5, 6, 7, 8, 9, 10]. These papers not only describe theoretical results or simulations, but also applications of industrial processes.

Predictive controllers are based on *prediction* of the *future* behavior of the process to be controlled. These predictions are based on a model of the process that is assumed to be available. This is the reason why predictive controllers are sometimes denoted internal model controllers. In the many papers that have been written about predictive control it is shown that not only 'simple' processes (e.g. first or second order without time delay) but also 'difficult' processes (e.g. processes with a long time delay, non-minimum phase and unstable processes) can be controlled by

predictive controllers without the designer having to take much special precautions. Moreover, in contrast with other control methods, predictive controllers have shown themselves to be remarkably robust with respect to model mismatch (see, for example, [4, 11]). Further, it is claimed that predictive controllers are easy to tune, also by people who are not control engineers.

The predictive control concept is not restricted to linear single-input single-output (SISO) processes, papers have also been written on predictive control of linear multi-input multi-output (MIMO) processes (e.g. [12]) and predictive control of nonlinear SISO processes (e.g. [13]). The following section describes the predictive control concept.

1.1 The Predictive Control Concept

Predictive control is a discrete-time controller concept that can be used to calculate the controller output, assuming that a model of the process to be controlled is available. The way predictive controllers operate for a single-input single-output system is illustrated in Figure 1.1. Suppose the current time is $t = k$ and $u(k)$, $y(k)$ and $w(k)$

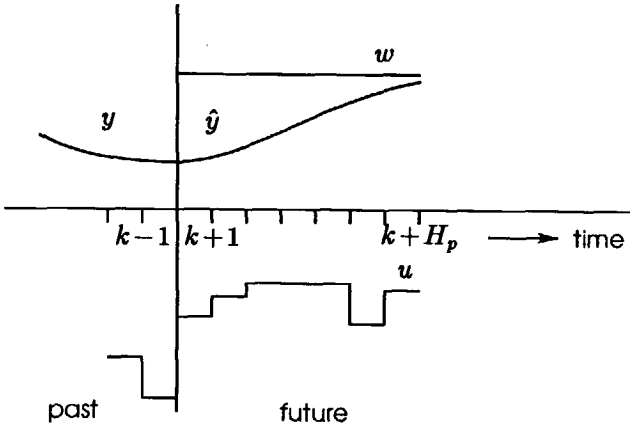

Figure 1.1: The predictive control concept.

denote the controller output, the process output and the desired process output at $t = k$, respectively. Further, define

$$\boldsymbol{u} = [u(k), \cdots, u(k + H_p - 1)]^T$$

$$\hat{\boldsymbol{y}} = [\hat{y}(k+1), \cdots, \hat{y}(k+H_p)]^T$$
$$\boldsymbol{w} = [w(k+1), \cdots, w(k+H_p)]^T$$

where H_p is called the prediction horizon and the symbol $\hat{}$ denotes estimation. Then, a predictive controller calculates such a future controller output sequence $\boldsymbol{u}$ that the predicted output of the process $\hat{\boldsymbol{y}}$ is 'close' to the desired process output $\boldsymbol{w}$. This desired process output is often called the reference trajectory and it can be an arbitrary sequence of points, despite the fact that a constant reference trajectory is shown in Figure 1.1. However, mostly the response of a first or second-order reference model is used. The first element of the controller output sequence determined in this way ($= u(k)$) is used to control the process. At the next sample, the whole procedure is repeated using the latest measured information. This is called the *receding horizon* principle [5].

The process output is predicted by using a model of the process to be controlled. Any model that describes the relationship between the input and the output of the process can be used. Hence, not only traditional ARMAX (Auto-Regressive Moving-Average eXogenous) models can be used, but also step-response models [2], state-space models [14] and nonlinear models [13]. Further, if the process is subject to disturbances, a disturbance or noise model can be added to the process model thus allowing the effect of disturbances on the predicted process output to be taken into account.

In order to define how well the predicted process output tracks the reference trajectory, a criterion function is used. Typically, such a criterion function is a function of $\hat{\boldsymbol{y}}$, $\boldsymbol{w}$ and $\boldsymbol{u}$. For example, a simple criterion function is:

$$J = \sum_{i=1}^{H_p} (\hat{y}(k+i) - w(k+i))^2$$

Now the controller output sequence $\boldsymbol{u}$ over the prediction horizon is obtained by minimization of J with respect to $\boldsymbol{u}$. Then, $\boldsymbol{u}$ is optimal with respect to the criterion that is minimized. As a result, the future tracking error is minimized and if the model is identical to the process, there are no disturbances and constraints, the process will track the reference trajectory exactly on the sampling instants. When the process has a time delay of, say, 2 samples, the approach is not different. Except that predicting $\hat{y}(k+1)$ and $\hat{y}(k+2)$ does not make sense because these values cannot be influenced by the control actions at $t = k$ and $t = k+1$. Then, the cost function can be changed into:

$$J = \sum_{i=3}^{H_p} (\hat{y}(k+i) - w(k+i))^2 \tag{1.1}$$

The optimal controller output sequence when there is time delay is then obtained by the minimization of (1.1) with respect to $\boldsymbol{u}$.

Clearly, calculating the controller output sequence is an optimization problem or, more specifically, a minimization problem. Usually, solving a minimization problem requires an iterative procedure. However, as will be shown in Chapter 2, an analytical solution is available when the criterion is quadratic, the model is linear and there are no constraints. Ideally, a criterion function is based on design specifications such as a desired settling time, overshoot, gain margin etc. However, by doing so, the resulting optimization problem is hard to solve [15]. This is the reason why in all predictive controllers a quadratic criterion is used. Now the design specifications must be translated into the parameters of the quadratic criterion function such that when this function is minimized, the original design specifications are satisfied. For this reason the parameters in the quadratic criterion function must be closely related to the traditional design specifications which makes the tuning of predictive controllers quite easy.

Note that the controller outputs calculated by minimization of a criterion function with respect to $\boldsymbol{u}$ are not structured. That is, they are not generated by means of some kind of control law as is the case in most other control strategies. For example, in LQ (Linear Quadratic) controllers, the controller output is usually calculated by using linear state feedback:

$$u(k) = -\boldsymbol{k}^T \boldsymbol{x} \tag{1.2}$$

where $\boldsymbol{k}$ is a vector containing the controller parameters and $\boldsymbol{x}$ represents the state vector of the process. The parameter vector $\boldsymbol{k}$ is obtained by the minimization of a quadratic criterion function with respect to $\boldsymbol{k}$. Not structuring the controller output and minimization with respect to the controller outputs has the advantage that constraints on the controller output can be taken into account in a natural way. For example, if the output of the controller is limited between two values (which is the case in most practical applications), the optimization problem can simply be formulated as: minimize J subject to

$$\underline{u} \leq u(k+i-1) \leq \overline{u} \qquad 1 \leq i \leq H_p$$

where $\underline{u}$ and $\overline{u}$ represent the lower and upper bound on the controller output, respectively. Now, the optimization problem is a constrained optimization problem for which an analytical solution is no longer available and, therefore, an iterative optimization method must be used. Chapter 4 discusses predictive control when there are level and rate constraints on the controller output. However, also other types of constraints on the controller output can occur in practice. For example, in most heating systems the controller output can have only two values, the heating device is either on or off. Predictive on-off controllers have been discussed in [16].

To here, the predictive control concept for SISO systems has been discussed. However, the predictive control concept is also applicable to multi-input multi-output (MIMO) processes. Although the computations become more complex, still an analytical solution to the optimization problem is available if the criterion is quadratic, the model is linear and there are no constraints [12].

1.2 Relationship to other Methods

As is shown in the previous section, predictive controllers are based on minimization of a criterion function. However, there are other controller design methods that are also based on the minimization of a criterion function. For example, H_∞ design methods are based on the minimization of the peak of a certain frequency-dependent criterion function (see, for example, [17]). This method, however, has little resemblance to predictive control. A class of controller design methods that is more closely related is LQ control.

LQ control

Because predictive controllers are discrete-time controllers, discrete LQ controllers are considered only. Further, because single-input single-output processes are considered only in this thesis, the relationship to LQ methods for SISO processes is discussed only.

Some LQ methods are based on linear state feedback while others are based on output feedback. The major differences between LQ methods and predictive control methods are:

- In LQ control the controller structure is usually determined a priori (see, for example, (1.2)). The criterion function used in LQ control is then minimized

with respect to the controller parameter vector $\boldsymbol{k}$. As a result, constraints on the controller output cannot be taken into account.

- LQ controller design considers linear(ized) processes only. Even in this case, an iterative method must be used to minimize the cost function used in LQ controllers, where in predictive controllers a straightforward analytical solution is available.

- In contrast to LQ control, predictive controllers employ the receding horizon strategy. However, if the process is linear, there are no constraints and the criterion function is quadratic, using the predictive control concept yields a fixed linear control law in spite of the fact that a receding horizon strategy is used (this is shown later in this thesis). In this case, there is a strong relationship between LQ control and predictive control as is shown in [18].

An important disadvantage of LQ control is that, in general, it is quite difficult to translate the design specifications into the criterion function that it minimizes because the states of a discrete-time process are usually artificial and hence they are not directly related to the true states of the process (see, [19], pp.267). Rule of thumb methods on how to choose the criterion parameters are not available. An important advantage of LQ controllers is that, under some quite general conditions, the closed-loop system is guaranteed to be stable. For predictive controllers this claim can usually not be achieved. A detailed discussion of linear optimal control can be found in [19], Chapter 11 and [20].

Pole-placement control

Another method that is frequently used to design a discrete controller is the pole-placement method (see, for example, [19], Chapter 10). In this method the controller output is generated by the linear control law:

$$\mathcal{R}u(k) = -\mathcal{S}y(k) + \mathcal{T}\ Sp \tag{1.3}$$

where $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ are the controller polynomials in the backward shift operator q^{-1} and Sp denotes the set point. The controller polynomials $\mathcal{R}$ and $\mathcal{S}$ are selected such that the closed-loop poles appear at the desired locations. $\mathcal{T}$ is selected such that the closed-loop system behaves as desired to set point changes. The major difficulty is to decide where to place the poles of the closed-loop system. That is, how to

translate the design specifications (which are usually defined in the time domain) into the location of the poles and zeros of the closed-loop system. Only in the case of low-order processes without zeros can this be done quite easily. Further, owing to the fact that the controller outputs are generated by (1.3), constraints on the controller output and an arbitrary reference trajectory cannot be included in the design.

At this point, there seems to be little resemblance to predictive controllers. However, it will be shown in Chapter 2 that if the model is linear, the criterion function is quadratic and if there are no constraints, then there is a close relationship between predictive and pole-placement controllers. Moreover, the predictive controller can be regarded as a pole-placement controller which places the closed-loop poles by the minimization of a criterion function. The required relationship between traditional design specifications and the location of the closed-loop poles is then established by the predictive controller. Further, it is shown in Chapter 2, that by choosing a special criterion function, the pole-placement controller coincides the predictive controller. The desired closed-loop poles then appear as parameters in the criterion function.

Adaptive control

Adaptive controllers are used if the parameters of the process are time variant and a fixed controller does not yield acceptable results. In an adaptive system, it is assumed that the controller parameters are adjusted all the time. This implies that the controller parameters are adapted as the process changes.

Basically, there are two types of adaptive controllers: Direct adaptive controllers and indirect adaptive controllers. In direct adaptive controllers, the controller is obtained without an explicit identification of the process. Most model reference adaptive controllers (see, for example, [21], Chapter 4 and [22]) are examples of direct adaptive controllers.

Indirect adaptive controllers identify the parameters of the process explicitly. A block diagram of an indirect adaptive controller is shown in Figure 1.2. The parameters of the process are estimated on line by using a suitable identification method. Based on the estimated model, a controller is designed on line assuming that the model is identical to the process. This is called the certainty equivalence principle. For identification of the process, a recursive least-squares method is often used. The on-line controller design which is based on the estimated model can be done by means of, for example, a pole-placement method. However, as is shown later in this thesis, predictive controllers are perfectly suited to solve the on-line design problem. Therefore, they can be used directly in an indirect adaptive control scheme

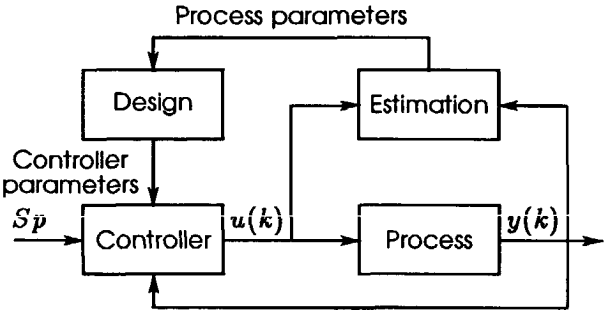

Figure 1.2: Block diagram of an indirect adaptive controller.

as depicted in Figure 1.2. For example, EPSAC (Extended Prediction Self-Adaptive Control) [7] and GPC (Generalized Predictive Control) [5] use the above-mentioned method to realize an adaptive controller. However, there are also adaptive predictive controllers that are based on the direct adaptive control scheme [23, 24].

In this thesis, adaptive control is not considered. Only the design of the control system when a model of the process is available is discussed. However, as is shown above, an adaptive control system can be obtained by using an estimator that provides the model for the predictive controller.

1.3 Motivation of the Thesis

Although all predictive controllers are based on the same concept, there are significant differences between predictive controllers in the literature (see, for example, [11, 25]). This is due to the fact that, within the concept of predictive control, many different approaches can be used to design a predictive controller. Different internal models, reference trajectories and criterion functions result in different predictive controllers and, therefore, it can be quite hard to select which controller to use in a particular situation.

Predictive controllers have remarkable properties, such as good robustness against model mismatch and the ability to control 'difficult' processes. However, the reason the controller has these properties is, in my opinion, not very well understood.

All predictive controllers proposed in the literature have a number of parameters to be selected. The influence of some parameters on the control system is

intuitively known. The effect of other parameters on the properties of the closed-loop system is much harder to predict and therefore less understood. Because of this it can be quite difficult to select such a parameter.

In many predictive controllers, heuristics play an important role in the design. Although these predictive controllers have been shown to operate well, it is preferable to have a theoretical background.

Although the predictive control concept allows the incorporation of constraints on the controller output, little attention has been paid in the literature to solving the resulting optimization problem.

These considerations are the motivation for writing a thesis on predictive control.

1.4 Outline and Scope of the Thesis

In order to get an insight into the different approaches to obtain a predictive controller encountered in literature, a unified approach is adopted: the most commonly used approaches to predictive controller design are united in the unified predictive controller called UPC. In Chapter 2, the design of the unified predictive controller for linear SISO processes is discussed together with some theoretical results concerning its design parameters. Relationships with other controller design methods such as minimum-variance, pole-placement and time-optimal control are also discussed in this chapter. Because of this unified approach, many well-known predictive controllers and other controllers, each with its own features, can be obtained simply by selecting the parameters of the UPC controller. Moreover, a number of extra features have been added to the UPC controller in order to overcome certain problems other predictive controllers cannot cope with.

In Chapter 3, the influence of the UPC design parameters on the performance and robustness of the closed-loop system is analyzed. This analysis is carried out by means of simulations using many different processes, and by theoretical means. In this chapter, the main goal is to obtain relations between traditional design specifications and the UPC design parameters. Further, rule of thumb methods on how to select these parameters are derived. Also in this chapter a comparison based on the results obtained in Chapters 2 and 3 is made between several well-known predictive controllers.

In Chapter 4, predictive control when there are constraints on the controller output is discussed. Level and rate constraints, which often appear in practice, are incorporated in the UPC controller. Much attention is paid to the performance of the

closed-loop system when there are constraints and the numerical complexity involved in solving the constrained optimization problem.

In order to show that UPC operates well not only in simulations, in Chapter 5, two industrial applications are discussed. One application shows that if an accurate model of the process and disturbances is available, a high-performance control system can easily be obtained. The second application shows that, even if a simple model of the process is used for predicting the process output, good results are still obtained. Both applications show that tuning the UPC controller is simple. Finally, in Chapter 6 some conclusions are drawn.

Chapter 2

Unified Predictive Controller Design

In this chapter the design of the Unified Predictive Controller is discussed. As shown in Chapter 1, any predictive controller is based on the predictive control concept depicted in Figure 1.1. As a result, any predictive controller has four major features in common:

1. A model of the process to be controlled. This model is used to predict the process output over the prediction horizon. Examples of such a model are the ARIMAX (Auto-Regressive Integrated Moving-Average eXogenous) model (used, for example, in GPC [5]) or the finite impulse response model (FIR). In section 2.1 prediction models are discussed.

2. The reference trajectory for the process output. In many controllers a first-order trajectory is used (e.g. in [1, 5, 26]). In this thesis an arbitrary reference trajectory is assumed. In section 2.4 several ways to generate the reference trajectory are discussed.

3. The criterion that is minimized in order to obtain the optimal controller output sequence over the prediction horizon. Most often, a quadratic criterion is used which weights tracking error and controller output. Section 2.2 discusses criterion functions.

4. The minimization procedure itself. If the criterion is quadratic, the model is linear and there are no constraints on the controller output, an analytical solution is readily available (see, section 2.3). If, on the other hand, there are constraints, the criterion must be minimized by using a nonlinear optimization method [27, 28]. Predictive control in the presence of constraints is discussed in Chapter 4.

In the following sections each of the above mentioned ingredients is discussed separately. In [29] a summary can be found with respect to the UPC design.

2.1 Prediction Models

In this section, prediction models are derived that predict the process output at $t = k + i$ using past information (thus, known information) and future controller outputs yet to be calculated. Such a model is called an *i-step-ahead predictor*.

An i-step-ahead predictor is based on a process model. A unified process model, unifying familiar process models such as ARX, ARMAX, ARIMAX, FIR and FSR (Finite Step Response) models is shown in (2.1).

$$y(k) = \frac{q^{-d}B(q^{-1})}{A(q^{-1})}u(k-1) + \underbrace{\frac{C(q^{-1})}{D(q^{-1})A(q^{-1})}}_{\text{noise model}}e(k) \tag{2.1}$$

in which q^{-1} is the backward shift operator, $y(k)$ is the process output, d is the time delay in samples ($d \geq 0$), $u(k)$ is the controller output and $e(k)$ is a discrete white

noise sequence with zero mean and variance σ_e^2. The polynomials A, B, C and D are given by:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \cdots + a_{n_A} q^{-n_A} \\ B(q^{-1}) &= b_0 + b_1 q^{-1} + \cdots + b_{n_B} q^{-n_B} \\ C(q^{-1}) &= 1 + c_1 q^{-1} + \cdots + c_{n_C} q^{-n_C} \\ D(q^{-1}) &= 1 + d_1 q^{-1} + \cdots + d_{n_D} q^{-n_D} \end{aligned}$$

where n_A, n_B, n_C and n_D are the degrees of the A, B, C and D polynomials respectively. Subsequently, the following notational convention is used with respect to polynomials: The jth element of the polynomial X is given by x_j and the degree of X is given by n_X. Further, a negative degree of a polynomial implies that the polynomial does not exist and may be replaced by zero. In Table 2.1 the settings for the unified process model are shown for 6 different process models. Δ is the

	n_A	n_B	n_C	n_D	D
ARX	n	$\leq n$	0	0	1
ARMAX	n	$\leq n$	> 0	0	1
ARIX	n	$\leq n$	0	1	Δ
ARIMAX	n	$\leq n$	> 0	1	Δ
FIR	0	$n_H - 1$	-	-	-
FSR	0	$n_S - 1$	-	-	-

Table 2.1: Process models and the settings of the unified process model.

differencing operator: $\Delta = 1 - q^{-1}$ and n is the order of the model. Further, the symbol '-' denotes that the FIR and FSR models do not consider a noise model. In the case of an FIR model, the output of the process is described by:

$$y(k) = \sum_{i=0}^{n_H - 1} h_i u(k - i - 1)$$

where n_H is the number of impulse response elements h_i that are taken into account. All other elements $h_{n_H}, \ldots$ are assumed to be equal to zero. An FSR model describes the process output by:

$$y(k) = \sum_{i=0}^{n_S-1} s_i \Delta u(k-i-1)$$

where n_S is the number of step response elements s_i that are taken into account. All other elements $s_{n_S}, \ldots$ are assumed to be constant. In an FIR model, the coefficients of the B polynomial are the impulse response elements: $b_i = h_i$. In an FSR model, the coefficients of the B polynomial are given by $b_0 = s_0$ and $b_i = s_i - s_{i-1}$ $\forall i \geq 1$. Note that both the FIR and the FSR models do not consider a noise model. However, as is shown later in this chapter, the i-step-ahead predictors based on an FIR and FSR model implicitly use a noise model with $C = 1$ and with D equal to 1 or Δ.

Note that the difference in the ARX, ARMAX, ARIX and ARIMAX models is the different modeling of the disturbance $\frac{C}{DA}e(k)$. Note also that in many noise models (also called disturbance models) Δ is a factor of D. It is shown in section 2.3.3 that by doing so integral action is introduced in the controller and hence offset does not result in steady-state errors. In section 2.1.5 it is shown that the correct modeling of offset is possible only if Δ is a factor of D. Hence, ARIX and ARIMAX models correctly model offset, while the ARX and ARMAX models do not.

The type of process model that should be used to derive the i-step-ahead predictor depends on the process that is to be controlled. For example, ARMAX models correctly describe processes with stationary disturbances while ARIMAX models correctly describe processes which are disturbed by Brownian motion (= random walk = integrated white noise) [21] or offset. Owing to the fact that disturbances like Brownian motion or offset often occur in real plants, ARIMAX models are to be preferred. For example, an ARIMAX model is the basis of the prediction model in GPC (Generalized Predictive Control [5]).

In order to examine and compare the influence of the C and D polynomials on the prediction models, first the case in which $C = D = 1$ is discussed. For the sake of simplicity, the time delay d is assumed to be zero. In section 2.1.6, prediction models are discussed with a time delay different from zero and a comparison is made with the well-known Smith predictor.

2.1.1 i-step-ahead Predictors in which $C = D = 1$

When $C = D = 1$ and $d = 0$, the unified process model (2.1) can be rewritten as:

$$A(q^{-1})y(k) = B(q^{-1})u(k-1) + e(k) \tag{2.2}$$

Note that this process model is valid for ARX and FIR process models with $C = D = 1$. Because there are significant differences in deriving the i-step-ahead predictor for $n_A > 0$ and for $n_A = 0$, these two cases are discussed separately.

i-step-ahead predictors when $n_A > 0$

The following Diophantine equation can be used to derive the i-step-ahead predictor for the given process model (for more details on Diophantine equations see section A.1 or [30]):

$$\frac{1}{A} = E_i + q^{-i}\frac{F_i}{A} \tag{2.3}$$

where E_i and F_i are polynomials of degree $i-1$ and $n_A - 1$ respectively (hence, $n_{E_i} = i-1$ and $n_{F_i} = n_A - 1$) and E_i is monic (because A is monic). The polynomials E_i and F_i can be obtained by using long division or the recursive relationship between two successive solutions of the Diophantine equation (see section A.2).

Substituting $k+i$ for k in (2.2) and multiplying by E_i yields:

$$E_iAy(k+i) = E_iBu(k+i-1) + E_ie(k+i) \tag{2.4}$$

Rewriting the Diophantine equation (2.3) yields:

$$E_iA = 1 - q^{-i}F_i \tag{2.5}$$

Substitution of the right-hand side of (2.5) for E_iA in (2.4) results in:

$$y(k+i) = E_iBu(k+i-1) + F_iy(k) + E_ie(k+i) \tag{2.6}$$

In order to split up the i-step-ahead predictor (2.6) in parts that are known at time $t = k$ and future signals, a second Diophantine equation is introduced:

$$E_iB = G_i + q^{-i}H_i \tag{2.7}$$

where G_i and H_i are polynomials of degree $i-1$ and n_B-1 respectively. In this case, solving (2.7) is quite simple: the coefficients of G_i are equal to the first i coefficients of E_iB while the coefficients of H_i are equal to the last n_B coefficients of E_iB.

Remarks:

- It is easy to show that G_i consists of the first i coefficients of the impulse response of the process that is controlled: Multiplying (2.3) by B yields:

$$\frac{B}{A} = E_iB + q^{-i}\frac{F_iB}{A} \tag{2.8}$$

  Substitution of (2.7) in (2.8) yields:

$$\frac{B}{A} = G_i + q^{-i}\left[\frac{F_iB}{A} + H_i\right] \tag{2.9}$$

  From (2.9) follows directly that the coefficients of G_i are equal to the first i coefficients of $\frac{B}{A}$ which describes the impulse response of the process that is controlled. This, of course, implies that $g_{i+1,j} = g_{i,j}$ making it unnecessary to distinguish separate coefficients for G_i. Therefore, G_i is simplified into:

$$G_i(q^{-1}) = g_0 + g_1q^{-1} + \cdots + g_{i-1}q^{-i+1}$$

- The solution if $n_B = 0$ is simply $G_i = E_iB$ and $H_i = 0$.

Using (2.7) the i-step-ahead predictor (2.6) becomes:

$$y(k+i) = \underbrace{G_iu(k+i-1)}_{\text{future}} + \underbrace{F_iy(k) + H_iu(k-1)}_{\text{past}} + \underbrace{E_ie(k+i)}_{\text{future}} \tag{2.10}$$

Because the last term involves future noise $(e(k+i), \cdots, e(k+1))$ only and because $e(k)$ is assumed to be discrete white noise with zero mean, the minimum variance (MV) i-step-ahead predictor can simply be obtained by eliminating this term from (2.10) (see section 2.1.5):

$$\hat{y}(k+i) = G_iu(k+i-1) + F_iy(k) + H_iu(k-1) \tag{2.11}$$

where the symbol $\hat{}$ in (2.11) indicates that $\hat{y}(k+i)$ is an estimate of $y(k+i)$.

Matrix notation

Collecting the i-step-ahead predictors for $i = 1, \ldots, H_p$ in a matrix notation yields:

$$\hat{\boldsymbol{y}} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} \tag{2.12}$$

where:

$$\begin{aligned}
\hat{\boldsymbol{y}} &= [\hat{y}(k+1), \cdots, \hat{y}(k+H_p)]^T \\
\boldsymbol{u} &= [u(k), \cdots, u(k+H_p-1)]^T \\
\boldsymbol{s} &= [y(k), \cdots, y(k-n_A+1), u(k-1), \cdots, u(k-n_B)]^T
\end{aligned}$$

and $[\hat{\boldsymbol{y}}] = H_p \times 1$, $[\boldsymbol{u}] = H_p \times 1$, $[\boldsymbol{s}] = (n_A + n_B) \times 1$. The matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$ are of dimension $H_p \times H_p$ and $H_p \times (n_A + n_B)$, respectively:

$$\boldsymbol{\Gamma} = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_{H_p-1} & \cdots & \cdots & g_0 \end{bmatrix}$$

$$\boldsymbol{\Psi} = \left[\begin{array}{ccc|ccc} f_{1,0} & \cdots & f_{1,n_A-1} & h_{1,0} & \cdots & h_{1,n_B-1} \\ \vdots & & \vdots & \vdots & & \vdots \\ f_{H_p,0} & \cdots & f_{H_p,n_A-1} & h_{H_p,0} & \cdots & h_{H_p,n_B-1} \end{array}\right]$$

i-step-ahead predictors if $n_A = 0$

If $n_A = 0$ (FIR or FSR process model), the derivation of the MV i-step-ahead predictor is slightly different. The derivation shown above is still valid except that now the solution of Diophantine equation (2.3) is: $E_i = 1$ and $F_i = 0$. Hence, the degree of E_i is now equal to zero. The degrees of G_i and H_i in (2.7) now become: $n_{G_i} = \min(i, n_H) - 1$ and $n_{H_i} = n_H - i - 1$.

Hence, the degree of H_i now depends on i; if $i \geq n_H$ then H_i is equal to zero (remember that n_H is the number of impulse response elements that is taken into account, see Table 2.1). The MV i-step-ahead predictor based on the FIR process model now becomes:

$$\hat{y}(k+i) = Bu(k+i-1) \tag{2.13}$$

or if (2.7) is used:

$$\hat{y}(k+i) = \underbrace{G_i u(k+i-1)}_{\text{future}} + \underbrace{H_i u(k-1)}_{\text{past}} \tag{2.14}$$

This i-step-ahead predictor is often used to make predictions of the process output if an impulse or step response of the process is available. Although a noise model is not explicitly considered when using an FIR or FSR model, implicitly $C = D = 1$ is used.

Again, the i-step-ahead predictors for $i = 1, \ldots, H_p$ can be collected in a matrix notation like (2.12). The degrees of the matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$ now become: $[\boldsymbol{\Gamma}] = H_p \times H_p$ and $[\boldsymbol{\Psi}] = H_p \times n_H - 1$.

2.1.2 The Unified i-step-ahead Predictor

In this section the MV i-step-ahead predictor based on the unified process model is derived. Further, the relationship between the MV i-step-ahead predictor with $C = D = 1$, as derived in the previous section, is derived and discussed. For sake of simplicity, the time delay is assumed to be equal to zero. In section 2.1.6, the MV i-step-ahead predictor when the time delay is different from zero is discussed. Further, the cases $n_A > 0$ and $n_A = 0$ will be discussed separately.

The unified i-step-ahead predictor if $n_A > 0$

In order to derive the MV i-step-ahead predictor based on the unified process model the following Diophantine equation is used:

$$\frac{C}{DA} = \overline{E}_i + q^{-i}\frac{\overline{F}_i}{DA} \tag{2.15}$$

where $\overline{E}_i$ and $\overline{F}_i$ are polynomials of degree $i-1$ and $\max(n_C - i, n_A + n_D - 1)$ respectively. Multiplying the unified process model (2.1) with $d = 0$ by $DA\overline{E}_i$ and substituting $k+i$ for k yields:

$$DA\overline{E}_i y(k+i) = \overline{E}_i BDu(k+i-1) + C\overline{E}_i e(k+i) \tag{2.16}$$

Rewriting (2.15) yields:

$$DA\overline{E}_i = C - q^{-i}\overline{F}_i \tag{2.17}$$

Substituting (2.17) in (2.16) results in:

$$y(k+i) = \frac{\overline{E}_i BD}{C}u(k+i-1) + \frac{\overline{F}_i}{C}y(k) + \overline{E}_i e(k+i)$$

Now, using the same considerations as discussed in the previous section, the MV i-step-ahead predictor based on the unified process model becomes:

$$\hat{y}(k+i) = \frac{\overline{E}_i BD}{C}u(k+i-1) + \frac{\overline{F}_i}{C}y(k) \tag{2.18}$$

Separation of (2.18) into future parts and past parts can be realized by using:

$$\frac{\overline{E}_i BD}{C} = \overline{G}_i + q^{-i}\frac{\overline{H}_i}{C} \tag{2.19}$$

where $\overline{G}_i$ and $\overline{H}_i$ are polynomials of degree $i-1$ and $\max(n_B + n_D, n_C) - 1$, respectively. Utilizing (2.19), the MV i-step-ahead predictor is given by:

$$\hat{y}(k+i) = \overline{G}_i u(k+i-1) + \frac{\overline{F}_i}{C}y(k) + \frac{\overline{H}_i}{C}u(k-1) \tag{2.20}$$

The relationship between the predictor in which $C = D = 1$ (2.11) and the one based on the unified process model (2.20) is not clear: It is hard to predict what the influence of C and D will be on the predictions. In order to make this relationship clear, the Diophantine equations (2.3) and (2.15) are compared. Multiplying (2.3) by C and (2.15) by D yields:

$$CE_i + q^{-i}\frac{F_i C}{A} = D\overline{E}_i + q^{-i}\frac{\overline{F}_i}{A} \tag{2.21}$$

In order to collect all coefficients up to $i + n_D - 1$, the following three Diophantine equations are introduced:

$$CE_i = M_i + q^{-i-n_D} N_i \quad n_{M_i} = i - 1 + \min(n_C, n_D) \quad n_{N_i} = n_C - n_D - 1 \tag{2.22}$$

$$\frac{\overline{F}_i}{A} = \overline{Q}_i + q^{-n_D} \frac{\overline{R}_i}{A} \quad n_{\overline{Q}_i} = n_D - 1 \tag{2.23}$$

$$\frac{F_i C}{A} = Q_i + q^{-n_D} \frac{R_i}{A} \quad n_{Q_i} = n_D - 1 \tag{2.24}$$

Using (2.22) $\cdots$ (2.24), (2.21) can be rewritten into:

$$M_i + q^{-i} Q_i + q^{-i-n_D} \left[N_i + \frac{R_i}{A} \right] = D\overline{E}_i + q^{-i} \overline{Q}_i + q^{-i-n_D} \frac{\overline{R}_i}{A} \tag{2.25}$$

Hence:

$$\begin{aligned} M_i + q^{-i} Q_i &= D\overline{E}_i + q^{-i} \overline{Q}_i \\ AN_i + R_i &= \overline{R}_i \end{aligned} \tag{2.26}$$

From (2.22) and (2.26) follows:

$$D\overline{E}_i = CE_i + q^{-i} \left[Q_i - \overline{Q}_i - q^{-n_D} N_i \right] \tag{2.27}$$

Now introduce a polynomial K_i defined by:

$$K_i = -[Q_i - \overline{Q}_i - q^{-n_D} N_i] \quad n_K = \max(n_C, n_D) - 1 \tag{2.28}$$

Using (2.27) and (2.28) the relationship between $\overline{E}_i$ and E_i becomes:

$$D\overline{E}_i = CE_i - q^{-i} K_i \tag{2.29}$$

The relationship between $\overline{F}_i$ and F_i is obtained by using (2.3), (2.15) and (2.29):

$$\overline{F}_i = F_i C + AK_i \tag{2.30}$$

Using the relationships (2.29) and (2.30) and replacing the polynomials A, B, C and D by their estimates $\hat{A}$, $\hat{B}$, $\hat{C}$ and $\hat{D}$ (= utilizing the certainty equivalence principle [21], pp.164), the MV i-step-ahead predictor (2.18) for the unified process model can be rewritten into:

$$\hat{y}(k+i) = E_i\hat{B}u(k+i-1) + F_iy(k) + \frac{K_i\hat{A}}{\hat{C}}[y(k) - \hat{y}(k)] \tag{2.31}$$

where E_i and F_i are calculated using (2.3) with $\hat{A}$ instead of A. Further, $\hat{y}(k)$ is given by:

$$\hat{y}(k) = \frac{\hat{B}}{\hat{A}}u(k-1)$$

Note that (2.31) involves, among other things, the estimations of the C and D (in K_i) polynomials. Because in practice C and D are difficult to estimate, in many predictive controllers (e.g. GPC [6]) *fixed* polynomials are used. Then, these polynomials become design polynomials where $\hat{C}$ is often denoted as T. Subsequently, T denotes $\hat{C}$ while $\hat{D}$ denotes the estimated D polynomial.

Using Diophantine equation (2.7) in order to separate future and past controller outputs, the unified MV i-step-ahead predictor becomes:

$$\hat{y}(k+i) = \underbrace{G_iu(k+i-1)}_{\text{future}} + \underbrace{F_iy(k) + H_iu(k-1)}_{\text{past}} + \underbrace{\frac{K_i\hat{A}}{T}[y(k) - \hat{y}(k)]}_{\text{additional term}} \tag{2.32}$$

Equation (2.32) shows that the MV i-step-ahead predictor (2.18) of the unified process model can be written as the MV i-step-ahead predictor based on the process model with $T = \hat{D} = 1$ extended by an additional term, depending on the way disturbances are modeled. Note that the additional term in (2.32) is fully determined at $t = k$. The advantage of writing the i-step-ahead predictor for the unified process model as in (2.32) is that now the prediction model is separated into two parts: one that is not affected by T and $\hat{D}$ and one that is. If either T or $\hat{D}$ is changed, only the additional term needs to be recalculated. Further, this way of writing the i-step-ahead predictor gives an insight into the influence of the T and $\hat{D}$ polynomials on the predictions and on the predictive controller based on this model.

For example, (2.32) shows that the differences in i-step-ahead predictors based

on ARX, ARMAX, ARIMAX or ARIX process descriptions can be expressed by a simple term which depends on the filtered difference between the *measured* process output and the *estimated* process output. The filter $\frac{K_i\hat{A}}{T}$ depends on the way disturbances are modeled. The calculation of K_i for several process models is discussed in section 2.1.3. Writing the MV i-step-ahead predictor based on the unified process model as in (2.32) results in the following theorem.

Theorem 2.1 *If $\hat{A} = A$, $\hat{B} = B$ and $\hat{d} = d$ (hence, the process is correctly estimated) and $e(k) = 0$ then the MV i-step-ahead predictor (2.11) based on a process model with $T = \hat{D} = 1$ and the one based on the unified process model (2.32) with $T \neq 1$ and/or $\hat{D} \neq 1$, are identical.*

Proof. If $\hat{A} = A$, $\hat{B} = B$, $\hat{d} = d$ and $e(k) = 0$ then $y(k) - \hat{y}(k) = 0$ and hence the additional term in (2.32) is equal to zero. This proves the theorem.

□

Hence, in this case, the design polynomials T and $\hat{D}$ do not influence the predictions and the closed-loop system. A detailed study of the influence of the T and $\hat{D}$ polynomials on the predictions and the closed-loop system can be found in sections 2.1.5 and 2.3.3 and in Chapter 3.

So far the situation with $n_A > 0$. In the following section, the situation with $n_A = 0$ is discussed.

The unified i-step-ahead predictor if $n_A = 0$

If $n_A = 0$, Diophantine equation (2.3) shows that:

$$E_i = 1 \quad \text{and} \quad F_i = 0$$

Further, utilizing (2.24) with $F_i = 0$ yields:

$$Q_i = 0$$

Using $E_i = 1$ in (2.22) yields for N_i:

$$N_i = t_{i-n_D} + t_{i-n_D+1}q^{-1} + \cdots + t_{n_T}q^{-n_T+n_D+i} \tag{2.33}$$

and $n_{N_i} = n_T - n_D - i$. Note that $N_i = 0$ if $i > n_T - n_D$ and hence $N_i = 0 \ \forall i \geq 1$ if $n_T \leq n_D$.

Finally, from (2.28) follows:

$$K_i = \overline{Q}_i + q^{-n_D}N_i \tag{2.34}$$

Now, two cases can be distinguished:

$n_D = 0$ In this case (2.23) shows that $\overline{Q}_i = 0$ and hence $K_i = N_i$ in which N_i is given by (2.33) with $n_D = 0$. Note that if $i > n_T$ then $N_i = 0$ and hence $K_i = 0$. Further, if $n_T = 0$ then $K_i = 0 \ \forall i \geq 1$.

$n_D > 0$ In this case (2.23) shows that $n_{\overline{F}_i} = \max(n_T - i, n_D - 1)$ and hence $\overline{Q}_i$ and K_i are different from zero $\forall i \geq 1$ and $\forall n_D, n_T \geq 0$.

Taking the above into account, the MV i-step-ahead predictor with $n_A = 0$ becomes:

$$\hat{y}(k+i) = \hat{B}u(k+i-1) + \frac{K_i}{T}[y(k) - \hat{y}(k)] \tag{2.35}$$

where:

$$\hat{y}(k) = \hat{B}u(k-1)$$

In some specific situation $K_i = 0$, namely if $i > n_T$ and $n_D = 0$. If $n_T = n_D = 0$, $K_i = 0 \ \forall i \geq 1$ and hence the MV i-step-ahead predictor becomes:

$$\hat{y}(k+i) = \hat{B}u(k+i-1) \tag{2.36}$$

Here the process output is assumed to be disturbed by $e(k)$ (hence, the noise model is equal to 1). Predictor (2.36) is the FIR process model used to predict $y(k+i)$ and is used, for example, in the MAC controller [3].

2.1.3 Calculation of K_i for Some Special Cases

In this section the calculation of K_i is discussed for some special choices of T and $\hat{D}$, more specifically, for the ARX, ARMAX, ARIX, ARIMAX, FIR with $T = 1$ and $\hat{D} = 1 - q^{-1}$ and the FSR process model with $T = 1$ and $\hat{D} = 1 - q^{-1}$.

The ARX Case

For the ARX model $T = 1$ and $\hat{D} = 1$ and hence $n_T = n_D = 0$. From (2.28) follows directly $K_i = 0$.

The ARMAX Case

For the ARMAX model $\hat{D} = 1$ and hence $n_D = 0$. From (2.23) and (2.24) follows that $Q_i = \overline{Q}_i = 0$. Hence, from (2.28) follows:

$$K_i = N_i$$

K_i can now be calculated using (2.22) and is in this case equal to the last n_T coefficients of the polynomial TE_i.

The ARIMAX Case

In the case of an ARIMAX model $\hat{D} = 1 - q^{-1}$ and hence $n_D = 1$. From (2.23) and (2.24) follows that Q_i and $\overline{Q}_i$ are scalars. Using (2.27) with $q = 1$ yields:

$$\overline{Q}_i = TE_i(1) - N_i(1) + Q_i \overset{(2.22)}{\Longrightarrow} \overline{Q}_i = M_i(1) + Q_i \tag{2.37}$$

From (2.28) follows using (2.37):

$$K_i = -[Q_i - M_i(1) - Q_i - q^{-1}N_i] = M_i(1) + q^{-1}N_i \tag{2.38}$$

N_i can easily be calculated using (2.22) and is in this case equal to the last $n_T - 1$ terms of TE_i.

The ARIX Case

In this case $T = 1$ and $\hat{D} = 1 - q^{-1}$ and thus $n_T = 0$ and $n_D = 1$. As with the ARIMAX case, both Q_i and $\overline{Q}_i$ are scalars. Now (2.27) can be rewritten into:

$$\Delta\overline{E}_i = E_i + q^{-i}[Q_i - \overline{Q}_i - q^{-1}N_i] \tag{2.39}$$

Further, using $n_T = 0$ in (2.22) yields $N_i = 0$. Posing, for example, $q = 1$ in (2.39) yields:

$$\overline{Q}_i = E_i(1) + Q_i \tag{2.40}$$

Using (2.28) and (2.40) yields for K_i:

$$K_i = E_i(1) \tag{2.41}$$

The FIR Case with $T = 1$ and $\hat{D} = 1 - q^{-1}$

In this case $n_A = n_T = 0$ and $n_D = 1$. From (2.28) follows that K_i is a scalar. Using $q = 1$ in (2.29) yields: $K_i = 1$ (remember that, in this case, $E_i = T = 1$). Hence, the MV i-step-ahead predictor becomes:

$$\hat{y}(k+i) = \hat{B}u(k+i-1) + [y(k) - \hat{y}(k)] \tag{2.42}$$

Predictor (2.42) is equal to the predictor that was used in the PCA controller [26]. The additional term $y(k) - \hat{y}(k)$ was added in an heuristic way to the predictions based on an FIR model in order to prevent steady-state errors. By doing so, the FIR process model was in fact extended by $\hat{D} = 1 - q^{-1}$ making the model correct also in the presence of offset (see section 2.1.5). The advantage of using the unified process model (2.1) is that it is now easily possible to extend the FIR process model by a more complex noise model (with $n_T \geq 1$ and $n_D \geq 1$). This results in a different K_i polynomial in (2.35).

The FSR Case with $T = 1$ and $\hat{D} = 1 - q^{-1}$

The FSR process model coincides with the FIR process model shown above. Hence, the MV i-step-ahead predictor is also given by (2.42). The only difference is that the elements of the $\hat{B}$ polynomial are given by $b_0 = s_0$ and $b_i = s_i - s_{i-1}$ $\forall i \geq 1$ in which s_i is the ith element of the process' step response. In the FIR process model $\hat{B}$ consists of the elements of the impulse response.

2.1.4 Matrix Notation of the Unified Prediction Model

It is convenient to put the unified prediction model (2.32) into a matrix notation. This is realized by collecting all MV i-step-ahead predictors for $i = 1, \ldots, H_p$. Because the unified MV i-step-ahead predictor can be written as the MV i-step-ahead predictor based on the unified process model with $T = \hat{D} = 1$ and an additional term, the matrix notation for the unified prediction model is equal to (2.12) extended by a vector $\boldsymbol{k}$ which represents the additional term in (2.32):

$$\hat{\boldsymbol{y}} = \underbrace{\boldsymbol{\Gamma u} + \boldsymbol{\Psi s}}_{(2.12)} + \boldsymbol{k} \tag{2.43}$$

Vector $\boldsymbol{k}$ is given by (when $n_A > 0$):

$$\boldsymbol{k} = \boldsymbol{K_c}\tilde{\boldsymbol{c}} \tag{2.44}$$

in which:

$$\tilde{\boldsymbol{c}} = [\tilde{c}(k), \cdots, \tilde{c}(k - n_K)]^T \quad [\tilde{\boldsymbol{c}}] = (n_K + 1) \times 1$$

and $\tilde{c}(k)$ is given by:

$$\tilde{c}(k) = \frac{\hat{A}y(k) - \hat{B}u(k-1)}{T} \tag{2.45}$$

The matrix $\boldsymbol{K_c}$ is built up of the coefficients of the polynomials K_i for $i = 1, \ldots, H_p$. Hence:

$$\boldsymbol{K_c} = \begin{bmatrix} k_{1,0} & \cdots & k_{1,n_K} \\ \vdots & & \vdots \\ k_{H_p,0} & \cdots & k_{H_p,n_K} \end{bmatrix} \quad [\boldsymbol{K_c}] = H_p \times (n_K + 1) \tag{2.46}$$

Matrix notation (2.43) is used to derive the control law in section 2.3.

If $n_A = 0$ and $n_D = 0$ then the last $H_p - n_T$ rows of $\boldsymbol{K}_c$ are equal to zero because $K_i = 0$ if $i > n_T$. Moreover, if $n_A = 0$, $n_T = 0$ and $n_D = 0$, then $\boldsymbol{K}_c = 0$ and hence $\boldsymbol{k} = 0$.

If, $n_A = 0$ and $n_D > 0$ then $\boldsymbol{k}$ is given by (2.44) and (2.45) with $n_A = 0$.

The matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$ if $n_A = 0$ (which correspond to the case in which $T = \hat{D} = 1$) are already discussed in section 2.1.1.

2.1.5 Properties of i-step-ahead Predictors

This section shows some of the properties of MV i-step-ahead predictors. One of the properties has already been discussed in section 2.1.2, Theorem 2.1. If there are no disturbances and the process is correctly estimated, then the MV i-step-ahead predictors do not depend on T and $\hat{D}$.

Another property involves the effect of the T polynomial on the predictions. The unified MV i-step-ahead predictor (2.18) with C replaced by T, D replaced by $\hat{D}$ and B replaced by $\hat{B}$ is:

$$\hat{y}(k+i) = \frac{\overline{E}_i \hat{B} \hat{D}}{T} u(k+i-1) + \frac{\overline{F}_i}{T} y(k) \tag{2.47}$$

Suppose that the process is given by:

$$y(k) = \frac{B}{A} u(k-1) + \xi(k)$$

in which $\xi(k)$ is a disturbance present on the output of the process. Now, using the fact that $\overline{F}_i = q^i(T - \hat{D}\hat{A}\overline{E}_i)$ (see (2.15) with C replaced by T, A replaced by $\hat{A}$ and D replaced by $\hat{D}$), (2.47) can be rewritten into (see also [31]):

$$\hat{y}(k+i) = y(k+i) + \frac{\overline{E}_i \hat{D} \hat{A}}{T} \left[\frac{\hat{B}}{\hat{A}} - \frac{B}{A} \right] u(k+i-1) - \frac{\overline{E}_i \hat{D} \hat{A}}{T} \xi(k+i) \tag{2.48}$$

from (2.48) follows directly that the prediction error $\epsilon(k+i)$ is given by:

$$\begin{aligned}\epsilon(k+i) &= y(k+i) - \hat{y}(k+i) = \\ &= \frac{\overline{E}_i \hat{D}\hat{A}}{T} \underbrace{\left[\frac{B}{A} - \frac{\hat{B}}{\hat{A}}\right]}_{\text{model error}} u(k+i-1) + \frac{\overline{E}_i \hat{D}\hat{A}}{T}\xi(k+i)\end{aligned} \tag{2.49}$$

From (2.49) follows that the effect of modeling errors and disturbances on the prediction error can be influenced by the polynomials T and $\hat{D}$. Because the predictive control concept relies on predictions of the process output, we can expect that both $\hat{D}$ and T affect the regulator behavior and the robustness of the closed-loop system with respect to model mismatch. A more detailed discussion on the effect of T and $\hat{D}$ on the closed-loop system can be found in sections 2.3.3 and 3.5.

In general, it is difficult to analyze what the effect of T and $\hat{D}$ will be on the prediction error. However, in some special situations this can be done quite easily.

- If $i = 1$ then $\overline{E}_i = 1$ and hence the disturbance $\xi(k)$ and the effects due to modeling errors are filtered by $\frac{\hat{D}\hat{A}}{T}$. By choosing T and $\hat{D}$ such that this filter is a band-pass filter, low frequency effects caused by an incorrectly estimated DC gain or constant disturbances are eliminated. Further, high-frequency effects due to an incorrect model for high frequencies or high-frequency disturbances on the output of the process (such as measurement noise), appear attenuated in the prediction error.

  In the case $i > 1$, this analysis is no longer straightforward. When $i \to \infty$, $\frac{\overline{E}_i \hat{D}\hat{A}}{T} \to 1$ because in this case $\overline{E}_i \to \frac{T}{\hat{D}\hat{A}}$ as is shown by (2.15).

- If $\xi(k) = \frac{C}{DA}e(k)$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$, $D = \hat{D}$ and $C = \hat{C} = T$ then the prediction error is equal to $\overline{E}_i e(k+i)$. Since, $e(k)$ was assumed to be a discrete white noise sequence with zero mean the variance of the prediction error is given by:

  $$E[\epsilon(k+i)^2] = \sigma_e^2$$

  Because this variance is equal to the variance of the noise signal $e(k)$, the i-step-ahead predictors (2.18) and (2.31) are *minimum-variance* predictors.

- If Δ is a factor of $\hat{D}$ and the disturbances and the controller output are constant (which will generally be the case if the reference trajectory is constant in the steady state), then the prediction error in the steady state is equal to zero independent of a mismatch between the process and model DC gain.

  Further, if Δ is not a factor of $\hat{D}$, A and $\hat{A}$, then correct predictions in the steady state are possible only if $K_{dc} = \hat{K}_{dc}$ and $\xi(k) = 0$.

  In section 2.3.3, a more detailed discussion can be found on the influence of $\hat{D}$ on the steady-state behavior of the closed-loop system.

2.1.6 i-step-ahead Predictors with Time Delay

In this section, i-step-ahead predictors for processes with time delay are discussed. Further, a comparison is made with the well-known Smith predictor.

In processes with time delay, an input $u(k)$ generated at $t = k$ has effect on $y(k+d+1)$, $y(k+d+2), \cdots$ only. The predictive control strategy as shown in Figure 1.1 becomes, in the presence of time delay, as shown in Figure 2.1. Now, a

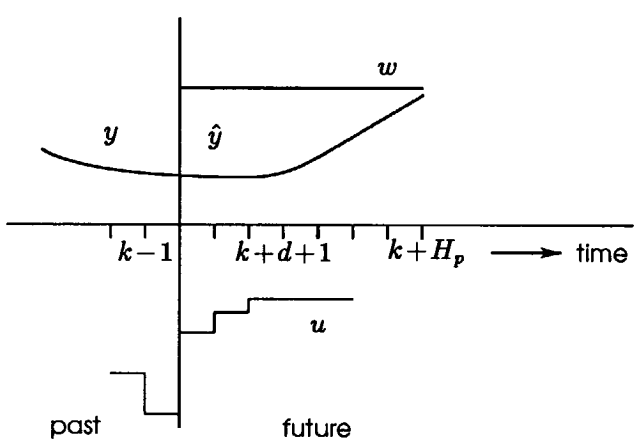

Figure 2.1: The predictive control strategy in the presence of time delay.

control sequence $u(k), \cdots, u(k+H_p-d-1)$ must be calculated which drives the predicted process outputs $\hat{y}(k+d+1), \cdots, \hat{y}(k+H_p)$ 'close' to the desired trajectory defined by $w(k+d+1), \cdots, w(k+H_p)$. Therefore, i-step-ahead predictors in the presence of time delay should predict $y(k+i)$ as a function of $u(k+i-d-1)$ for $i = d+1, \ldots, H_p$. Note, that the prediction horizon in the presence of time delay must satisfy the condition $H_p \geq d+1$.

i-step-ahead predictor with time delay and $T = \hat{D} = 1$

First, the MV i-step-ahead predictor with $T = \hat{D} = 1$ is derived in the presence of time delay and $n_A > 0$. Hence, (2.2) becomes:

$$A(q^{-1})y(k) = q^{-d}B(q^{-1})u(k-1) + e(k) \tag{2.50}$$

Substituting B by $q^{-d}B$ in (2.6) and eliminating the noise term yields the MV i-step-ahead predictor in the presence of time delay:

$$\hat{y}(k+i) = E_iBu(k+i-d-1) + F_iy(k) \qquad i \geq d+1$$

Separation of future and past can be realized by using the following Diophantine equation:

$$E_iB = G_{i-d} + q^{-i+d}H_{i-d} \tag{2.51}$$

The polynomials G_{i-d} and H_{i-d} are of degree $i-d-1$ and n_B+d-1 respectively. Note that because $i \geq d+1$ the degree of G_{i-d} is greater than or equal to zero and hence G_{i-d} always exists. The MV i-step-ahead predictor becomes using (2.51):

$$\hat{y}(k+i) = \underbrace{G_{i-d}u(k+i-d-1)}_{\text{future}} + \underbrace{F_iy(k) + H_{i-d}u(k-1)}_{\text{past}} \qquad i \geq d+1$$

Collecting the i-step-ahead predictors for $i = d+1, \ldots, H_p$ in a matrix notation yields:

$$\hat{\boldsymbol{y}} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s}$$

where:

$$\begin{aligned}
\hat{\boldsymbol{y}} &= [\hat{y}(k+d+1), \cdots, \hat{y}(k+H_p)]^T && [\hat{\boldsymbol{y}}] = (H_p-d)\times 1 \\
\boldsymbol{u} &= [u(k), \cdots, u(k+H_p-d-1)]^T && [\boldsymbol{u}] = (H_p-d)\times 1 \\
\boldsymbol{s} &= [y(k), \cdots, y(k-n_A+1), \\
&\qquad u(k-1), \cdots, u(k-n_B-d)]^T && [\boldsymbol{s}] = (n_A+n_B+d)\times 1
\end{aligned}$$

and Γ and Ψ are of dimension $(H_p - d) \times (H_p - d)$ and $(H_p - d) \times (n_A + n_B + d)$, respectively:

$$\Gamma = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_{H_p-d-1} & \cdots & \cdots & g_0 \end{bmatrix} \tag{2.52}$$

$$\Psi = \left[\begin{array}{ccc|ccc} f_{1,0} & \cdots & f_{1,n_A-1} & h_{1,0} & \cdots & h_{1,n_B+d-1} \\ \vdots & & \vdots & \vdots & & \vdots \\ f_{H_p-d,0} & \cdots & f_{H_p-d,n_A-1} & h_{H_p-d,0} & \cdots & h_{H_p-d,n_B+d-1} \end{array}\right] \tag{2.53}$$

If $n_A = 0$, then the MV i-step-ahead predictor is given by (2.14) with G_i and H_i replaced by G_{i-d} and H_{i-d} which are now of degree $\min(i-d-1, n_B)$ and $n_B - i + d$ respectively.

The unified i-step-ahead predictor with time delay

The unified MV i-step-ahead predictor in the presence of time delay is derived in the same way as described in section 2.1.2. In the presence of time delay and $n_A > 0$ the unified i-step-ahead predictor becomes:

$$\hat{y}(k+i) = \frac{\overline{E}_i \hat{B} \hat{D}}{T} u(k+i-d-1) + \frac{\overline{F}_i}{T} y(k) \qquad i \geq d+1 \tag{2.54}$$

Separation of (2.54) into future and past can be realized by using the Diophantine equation (2.55).

$$\frac{\overline{E}_i \hat{B} \hat{D}}{T} = \overline{G}_{i-d} + q^{-i+d} \frac{\overline{H}_{i-d}}{T} \tag{2.55}$$

where now $\overline{G}_{i-d}$ and $\overline{H}_{i-d}$ are polynomials of degree $i-d-1$ and $\max(n_B + n_D + d, n_T) - 1$, respectively. Utilizing (2.55), the MV i-step-ahead predictor is given by (2.20) with $\overline{G}_i$, $\overline{F}_i$ and $\overline{H}_i$ replaced by $\overline{G}_{i-d}$, $\overline{F}_{i-d}$ and $\overline{H}_{i-d}$.

The relationship between the unified i-step-ahead predictor and the i-step-ahead predictor with $T = \hat{D} = 1$ is, also when there is time delay, given by (2.29) and (2.30). Hence, the unified MV i-step-ahead predictor in the presence of time delay can be written as:

$$\hat{y}(k+i) = E_i\hat{B}u(k+i-d-1) + F_iy(k) + \frac{K_i\hat{A}}{T}[y(k) - \hat{y}(k)] \quad i \geq d+1$$

where E_i and F_i are calculated using (2.3) with A replaced by $\hat{A}$. $\hat{y}(k)$ is given by:

$$\hat{y}(k) = \frac{q^{-d}\hat{B}}{\hat{A}}u(k-1)$$

Using (2.51) in order to separate future and past, the MV i-step-ahead predictor for the unified process model with time delay becomes for $i \geq d+1$:

$$\hat{y}(k+i) = \underbrace{G_{i-d}u(k+i-d-1)}_{\text{future}} + \underbrace{F_iy(k) + H_{i-d}u(k-1)}_{\text{past}} + \underbrace{\frac{K_i\hat{A}}{T}[y(k) - \hat{y}(k)]}_{\text{additional term}} \tag{2.56}$$

Remark: The calculation of the polynomial K_i is independent of the time delay. Hence, the results shown in section 2.1.3 are valid also if the time delay differs from zero.

If $n_A = 0$ and a time delay different from zero, the unified MV i-step-ahead predictors are given by (2.36) or (2.35) (depending on the degrees of T and $\hat{D}$) with $\hat{B}$ replaced by $q^{-d}\hat{B}$.

Matrix notation of the unified prediction model with time delay

The matrix notation for the unified prediction model with time delay is slightly different from the one shown in section 2.1.4. The matrix equation (2.43) is still valid and again shown in (2.57).

$$\hat{\boldsymbol{y}} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} + \boldsymbol{k} \tag{2.57}$$

The first part of (2.57) ($= \boldsymbol{\Gamma u} + \boldsymbol{\Psi s}$) corresponds to the situation in which $T = \hat{D} = 1$ and is shown in section 2.1.6. Vector $\boldsymbol{k}$ is given by:

$$\boldsymbol{k} = \boldsymbol{K_c\tilde{e}} \tag{2.58}$$

in which:

$$\tilde{e} = [\tilde{e}(k), \cdots, \tilde{e}(k - n_K)]^T \qquad [\tilde{e}] = (n_K + 1) \times 1$$

$$\tilde{e}(k) = \frac{\hat{A}y(k) - \hat{B}u(k - 1 - d)}{T}$$

The matrix K_c is again given by (2.46).

i-step-ahead predictors if the time delay is not known

In general, the time delay of the process to be controlled is not exactly known. Assume that the time delay is known within two bounds:

$$0 \leq \underline{d} \leq d \leq \overline{d}$$

where $\underline{d}$ is the lower bound and $\overline{d}$ is the upper bound of the time delay. Figure 2.2 shows this situation. Figure 2.2 also shows that the prediction horizon H_p should

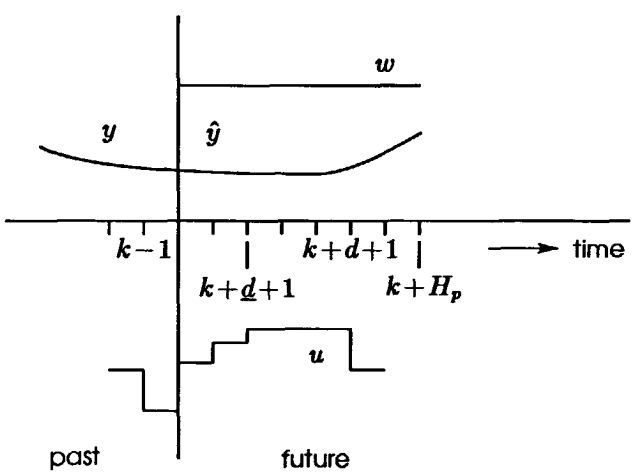

Figure 2.2: Predictive control strategy when there is an unknown time delay.

be greater than the upper bound on the time delay to make sure that the prediction horizon is always greater than the true time delay in the process. Hence, the following must hold:

$$0 \leq \underline{d} \leq d \leq \overline{d} < H_p \tag{2.59}$$

In contrast with the previous section, i-step-ahead predictors should now predict $y(k+i)$ as a function of $u(k+i-\underline{d}-1)$ for $i = \underline{d}+1, \ldots, H_p$.

An unknown time delay satisfying (2.59) can be tackled by modifying the unified prediction model in:

$$y(k) = q^{-\underline{d}}\frac{\tilde{B}(q^{-1})}{A(q^{-1})}u(k-1) + \frac{C(q^{-1})}{D(q^{-1})A(q^{-1})}e(k) \tag{2.60}$$

where:

$$\tilde{B}(q^{-1}) = \tilde{b}_0 + \tilde{b}_1 q^{-1} + \cdots + \tilde{b}_{n_B+\Delta d}q^{-n_B-\Delta d}$$

in which $\Delta d = \overline{d} - \underline{d}$ and the coefficients of $\tilde{B}$ are shown in Table 2.2. The

$\tilde{B} =$	$\tilde{b}_0$	$\cdots$	$\tilde{b}_{d-\underline{d}-1}$	$\tilde{b}_{d-\underline{d}}$	$\cdots$	$\tilde{b}_{n_B+d-\underline{d}}$	$\tilde{b}_{n_B+d-\underline{d}+1}$	$\cdots$	$\tilde{b}_{n_B+\overline{d}-\underline{d}}$
$=$	0	$\cdots$	0	b_0	$\cdots$	b_{n_B}	0	$\cdots$	0

Table 2.2: Coefficients of $\tilde{B}$.

differences between the unified process model with a known time delay discussed in section 2.1.6 and (2.60) are:

- The degree of $\tilde{B}$ is given by $n_{\tilde{B}} = n_B + \Delta d$ while the degree of B in (2.50) is given by n_B.
- The time delay in (2.60) is given by $\underline{d}$ instead of d in (2.50).

By using $\underline{d}$ instead of d and $n_B + \Delta d$ instead of n_B, the MV i-step-ahead predictor when there is an unknown time delay for the unified process model can easily be derived. The i-step-ahead predictor with $T = \hat{D} = 1$ becomes:

$$\hat{y}(k+i) = E_i\tilde{B}u(k+i-\underline{d}-1) + F_i y(k) \qquad i \geq \underline{d}+1 \tag{2.61}$$

Separation of future and past in (2.61) is realized by using (2.51) with $\underline{d}$ instead of d and $\tilde{B}$ instead of B:

$$E_i\tilde{B} = G_{i-\underline{d}} + q^{-i+\underline{d}}H_{i-\underline{d}} \tag{2.62}$$

The degrees of $G_{i-\underline{d}}$ and $H_{i-\underline{d}}$ are equal to $i-\underline{d}-1$ and $n_B+\overline{d}-1$ respectively. Note that if $d > \underline{d}$, then $G_1, \cdots, G_{d-\underline{d}}$ are equal to zero. The MV i-step-ahead predictor becomes utilizing (2.62):

$$\hat{y}(k+i) = G_{i-\underline{d}}u(k+i-\underline{d}-1) + F_iy(k) + H_{i-\underline{d}}u(k-1) \qquad i \geq \underline{d}+1$$

The unified MV i-step-ahead predictor is given by (2.56) with d replaced by $\underline{d}$. Again, the results in section 2.1.3 can be used to calculate K_i.

The matrix notation of the unified prediction model in the presence of an unknown time delay is given by (2.57) with:

$$\begin{aligned}
\hat{\boldsymbol{y}} &= [\hat{y}(k+\underline{d}+1), \cdots, \hat{y}(k+H_p)]^T && [\hat{\boldsymbol{y}}] = (H_p - \underline{d}) \times 1 \\
\boldsymbol{u} &= [u(k), \cdots, u(k+H_p-\underline{d}-1)]^T && [\boldsymbol{u}] = (H_p - \underline{d}) \times 1 \\
\boldsymbol{s} &= [y(k), \cdots, y(k-n_A+1), \\
&\qquad u(k-1), \cdots, u(k-n_B-\overline{d})]^T && [\boldsymbol{s}] = (n_A + n_B + \overline{d}) \times 1 \\
\tilde{c}(k) &= \frac{\hat{A}y(k) - \tilde{B}u(k-1-\underline{d})}{T}
\end{aligned} \tag{2.63}$$

and

$$\boldsymbol{\Gamma} = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ g_{H_p-\underline{d}-1} & \cdots & \cdots & g_0 \end{bmatrix}$$

$$\boldsymbol{\Psi} = \left[\begin{array}{ccc|ccc} f_{1,0} & \cdots & f_{1,n_A-1} & h_{1,0} & \cdots & h_{1,n_B+\overline{d}-1} \\ \vdots & & \vdots & \vdots & & \vdots \\ f_{H_p-\underline{d},0} & \cdots & f_{H_p-\underline{d},n_A-1} & h_{H_p-\underline{d},0} & \cdots & h_{H_p-\underline{d},n_B+\overline{d}-1} \end{array}\right]$$

Hence, the dimensions of the matrices become: $[\boldsymbol{\Gamma}] = (H_p - \underline{d}) \times (H_p - \underline{d})$, $[\boldsymbol{\Psi}] = (H_p - \underline{d}) \times (n_A + n_B + \overline{d})$ in the case $n_A > 0$.

If $n_A = 0$ the dimensions are given by: $[\boldsymbol{\Gamma}] = (H_p - \underline{d}) \times (H_p - \underline{d})$, $[\boldsymbol{\Psi}] = (H_p - \underline{d}) \times (n_B + \overline{d} - \underline{d})$. Note that in this case the last $H_p - n_B - \overline{d}$ rows of $\boldsymbol{\Psi}$ are equal to zero because $H_{i-\underline{d}} = 0$ for $i = n_B + \overline{d} + 1, \ldots, H_p$.

Remark: if $\hat{B}$ and $\hat{d}$ are known, then choosing $\underline{d}$ and $\overline{d}$ different from $\hat{d}$ does not make sense. However, if $\tilde{B}$ is estimated by, for example, a least-squares method, then it may be difficult to extract $\hat{d}$ from $\tilde{B}$ because the coefficients $\tilde{b}_0, \cdots, \tilde{b}_{\overline{d}-\underline{d}-1}$ shown in Table 2.2 will be different from zero and may even be of the same magnitude as the other coefficients of $\tilde{B}$.

In this approach, it is not required to extract $\hat{d}$ from $\tilde{B}$. The $\tilde{B}$ polynomial can be used directly to compute the predictions.

Comparison with the Smith predictor

In this section, a comparison with a predictor based on Diophantine equations and the Smith predictor [32, 33] is made. Suppose we have the closed-loop system as depicted in Figure 2.3. The relation between input and output is given by:

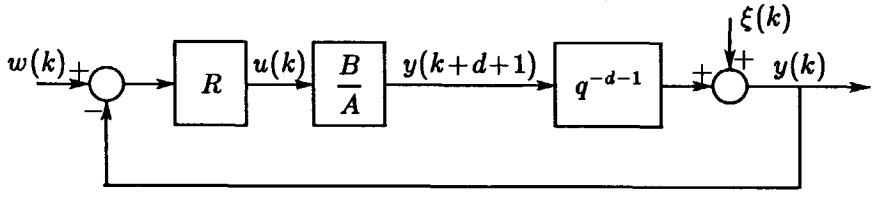

Figure 2.3: Controller structure using $y(k)$ for feedback.

$$y(k) = \frac{q^{-d-1}BR}{A + q^{-d-1}BR} w(k) \tag{2.64}$$

in which R is the transfer function of the controller.

Because the time delay of the process is present in the closed-loop characteristic equation, the closed-loop system easily becomes unstable (because of the large phase shifts caused by the time delay). In order to avoid such problems, the closed-loop structure as depicted in Figure 2.4 is preferable. Note that an algebraic loop is present in Figure 2.4 if the transfer function R does not contain at least a unit delay. Hence, calculation of $u(k)$ using the block diagram depicted in Figure 2.4 is not straightforward. However, this section is mainly concerned with the prediction of the process output rather than calculating the controller output $u(k)$. In section 2.3, the calculation of the controller output is discussed. Now the relation between input and output is given by:

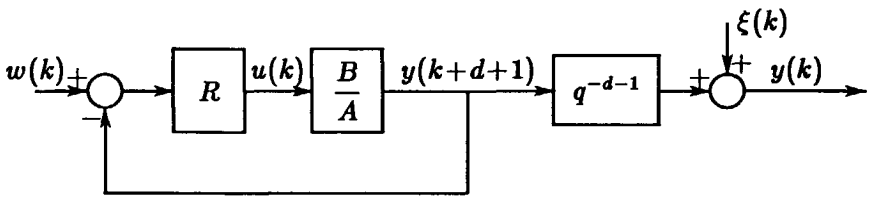

Figure 2.4: Controller structure using $y(k+d+1)$ for feedback.

$$y(k) = \frac{q^{-d-1}BR^*}{A + BR^*}w(k) \tag{2.65}$$

in which R^* is a transfer function different from R. With this closed-loop system, the time delay is no longer present in the closed-loop characteristic equation and hence phase shifts caused by the time delay no longer influence the stability of the closed-loop system. The signal that is used for feedback is $y(k+d+1)$. This signal is not measurable. Therefore, $y(k+d+1)$ has to be predicted.

Making the input-output relations (2.64) and (2.65) identical yields for R:

$$R = \frac{R^*}{1 + R^*\dfrac{B}{A}(1 - q^{-d-1})} \tag{2.66}$$

Using (2.66) the closed-loop system depicted in Figure 2.3 becomes as shown in Figure 2.5. The symbols $\hat{}$ indicate that now a model of the process is used to calculate $\hat{y}(k+d+1)$. The signal which is used for feedback is now a prediction of $y(k+d+1)$. If the process is correctly estimated and there are no disturbances ($\xi(k) = 0$), then the correction signal $c(k)$ is equal to zero. Thus, in this case $y(k)$ and $\hat{y}(k)$ are identical. If, on the other hand, disturbances are present or the process is not correctly estimated, $c(k) \neq 0$ and an extra feed-back loop is activated.

From Figure 2.5 it is easy to calculate $\hat{y}(k+d+1)$:

$$\hat{y}(k+d+1) = \frac{\hat{B}}{\hat{A}}u(k) + \left[y(k) - q^{-d}\frac{\hat{B}}{\hat{A}}u(k-1)\right] \tag{2.67}$$

The predictor (2.67) is called the Smith predictor [32]. Note that the Smith predictor allows correct predictions in the steady state also if the process is not correctly

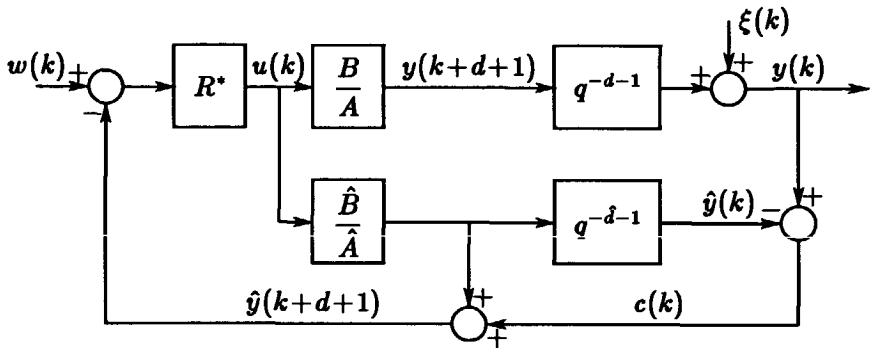

Figure 2.5: Controller structure using $\hat{y}(k+d+1)$ for feedback.

estimated (if the disturbances $\xi(k)$ and $w(k)$ are constant).

In order to compare the Smith predictor with the MV d+1-step-ahead predictor based on the unified process model recall (2.54) for $i = d+1$:

$$\hat{y}(k+d+1) = \frac{\overline{E}_{1+d}\hat{B}\hat{D}}{T}u(k) + \frac{\overline{F}_{1+d}}{T}y(k) \tag{2.68}$$

Recall also the Diophantine equation (2.15) for $i = d+1$ multiplied by $\hat{B}$:

$$\frac{\hat{B}T}{\hat{D}\hat{A}} = \overline{E}_{1+d}\hat{B} + q^{-d-1}\frac{\overline{F}_{1+d}\hat{B}}{\hat{D}\hat{A}} \Longrightarrow \overline{E}_{1+d}\hat{B} = \frac{\hat{B}T}{\hat{D}\hat{A}} - q^{-d-1}\frac{\overline{F}_{1+d}\hat{B}}{\hat{D}\hat{A}} \tag{2.69}$$

Using (2.69) yields for (2.68):

$$\hat{y}(k+d+1) = \frac{\hat{B}}{\hat{A}}u(k) + \frac{\overline{F}_{1+d}}{T}\left[y(k) - q^{-d}\frac{\hat{B}}{\hat{A}}u(k-1)\right]$$

In Figure 2.6 the MV d+1-step-ahead predictor is visualized. Comparing the Smith predictor (Figure 2.5) with the MV d+1-step-ahead predictor (Figure 2.6) shows that the latter filters the correction term $c(k)$ with the filter $\frac{\overline{F}_{1+d}}{T}$ while the Smith predictor does not filter the correction term $c(k)$. Hence, the Smith predictor differs from the MV d+1-step-ahead predictor.

The Smith predictor did not take disturbances into account. However, also

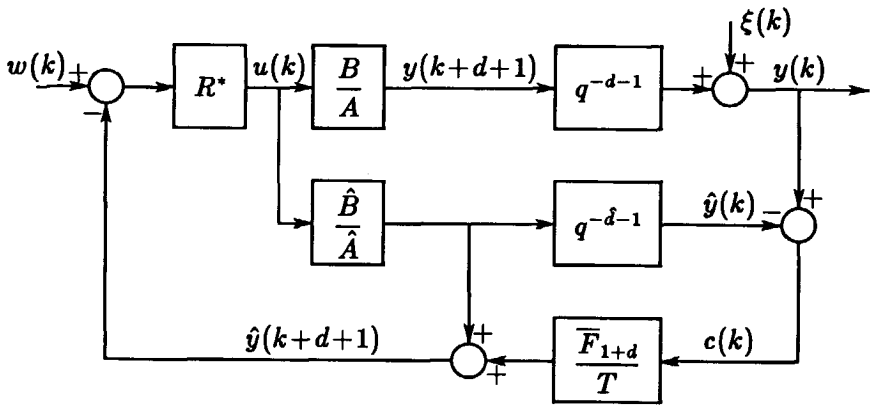

Figure 2.6: Controller structure using the MV d+1-step-ahead predictor for feedback.

if $T = \hat{D} = 1$ a filtering of the additional term remains necessary (with the filter $\overline{F}_{1+d}$).

As discussed above, the difference between the Smith predictor and the optimal predictor is the filter $\frac{\overline{F}_{1+d}}{T}$. However, we might wonder for what choice of T and $\hat{D}$ the MV d+1-step-ahead predictor coincides with the Smith predictor. Because the Smith predictor allows correct predictions in the steady state, the polynomial $\hat{D}$ must at least contain a factor Δ (see section 2.1.5). Let us use, for example, $\hat{D} = \Delta$. The Diophantine equation (2.15) that is used to derive the MV d+1-step-ahead predictor then becomes:

$$\frac{T}{\Delta\hat{A}} = \overline{E}_{1+d} + q^{-1-d}\frac{\overline{F}_{1+d}}{\Delta\hat{A}} \tag{2.70}$$

The condition which must be satisfied for the Smith predictor to coincide with the MV d+1-step-ahead predictor is:

$$\frac{\overline{F}_{1+d}}{T} = 1 \Longrightarrow \overline{F}_{1+d} = T$$

Now (2.70) becomes:

$$\frac{T}{\Delta\hat{A}} = \overline{E}_{1+d} + q^{-1-d}\frac{T}{\Delta\hat{A}} \tag{2.71}$$

It is easy to verify that for $T = \hat{A}$ (2.71) is satisfied. Then (2.71) becomes:

$$\frac{1}{\Delta} = \overline{E}_{1+d} + q^{-1-d}\frac{1}{\Delta}$$

and $\overline{E}_{1+d}$ is given by:

$$\overline{E}_{1+d} = 1 + q^{-1} + \cdots + q^{-d}$$

Hence, if $T = \hat{A}$ and $\hat{D} = \Delta$, the MV d+1-step-ahead predictor based on the unified process model coincides with the Smith predictor. This makes the Smith predictor the MV d+1-step-ahead-predictor if the process is described by:

$$y(k) = \frac{q^{-d}B(q^{-1})}{A(q^{-1})}u(k-1) + \frac{1}{\Delta}e(k) \tag{2.72}$$

Hence, the Smith predictor is identical to the MV d+1-step-ahead predictor if the disturbances can be described by Brownian motion. Because this description of disturbances was found to be a rather good one for many industrial processes, the Smith predictor is a good choice for predicting $y(k+d+1)$ and can be used in combination with, for example, a PID controller. A drawback is that in that case there is an algebraic loop (see Figure 2.5) and hence the calculation of $u(k)$ is not straightforward. This problem can be avoided by predicting $y(k+d)$ and using $\hat{y}(k+d)$ for feedback.

2.1.7 Conclusions

In this section a unified process model has been proposed. This model can be used to model processes described by ARX, ARMAX, ARIMAX, ARIX, FIR or FSR models. Based on this model, the MV i-step-ahead predictor has been derived. A comparison has been made between this predictor and the one based on the unified process model in which $T = \hat{D} = 1$. It has been shown that the unified MV i-step-ahead predictor can be written as that of the process model in which $T = \hat{D} = 1$ and an additional

term which depends on the difference between process and model output. This way of writing the MV i-step-ahead predictor clearly shows that disturbances due to model mismatch and other disturbances are filtered by a filter which depends on both T and $\hat{D}$. Therefore, it can be expected that T and $\hat{D}$ have an important influence on the regulator behavior and the robustness of the closed-loop system.

Finally, a comparison has been made between the Smith predictor and the MV i-step-ahead predictor based on the unified process model. It has been shown that both predictors are identical if the output of the process is disturbed by Brownian motion (hence, $T = A$ and $\hat{D} = \Delta$).

2.2 Criterion Functions

In this section various, often used, criterion functions are discussed. Because in predictive controllers, minimization of a criterion function yields the predictive control law, the choice of the criterion function is an important one.

Design objectives such as overshoot, rise time, settling time and damping ratio can easily be understood and specified. However, it is difficult to minimize criterion functions based on such objectives, because the relationship between the controller parameters and these criteria are in general highly nonlinear. Analytical solutions are seldom available. This is the reason why mathematically convenient criterion functions are often used. In Chapter 3, the relationships between these mathematical convenient criterion functions and the design objectives are discussed. In this section, a number of often-used criterion functions are discussed. At the end of this section, a unified criterion function is proposed which is used in the Unified Predictive Controller.

2.2.1 Single-step Criterion Functions

Criterion functions based on tracking error

A simple criterion function, taking into account the tracking error only is:

$$J = [\hat{y}(k+1) - w(k+1)]^2 \tag{2.73}$$

Because only one prediction is used (namely $\hat{y}(k+1)$), criterion function (2.73) is called a single-step criterion. In order to show what kind of controller is obtained when (2.73) is minimized with respect to $u(k)$, recall the unified process model (2.1) without time delay:

$$Ay(k) = Bu(k-1) + \frac{C}{D}e(k) \tag{2.74}$$

The MV 1-step-ahead predictor for this process model is (2.18) with $i = 1$ and with A, B, C and D replaced by their estimates:

$$\hat{y}(k+1) = \frac{\hat{B}\hat{D}}{T}u(k) + \frac{q(T - \hat{A}\hat{D})}{T}y(k)$$

By selecting $\hat{y}(k+1) = w(k+1)$ in (2.73), J has the minimal value. This yields:

$$u(k) = \frac{Tw(k+1) - q(T - \hat{A}\hat{D})y(k)}{\hat{B}\hat{D}} \tag{2.75}$$

In this section, a priori knowledge about the reference trajectory is assumed to be absent. Then, $w(k+1)$ represents the set point at $t = k$. In Figure 2.7 the closed-loop system is shown. Note that by introducing Δ as a factor of $\hat{D}$, integral action is introduced in the closed-loop system. Usually, discrete closed-loop systems are

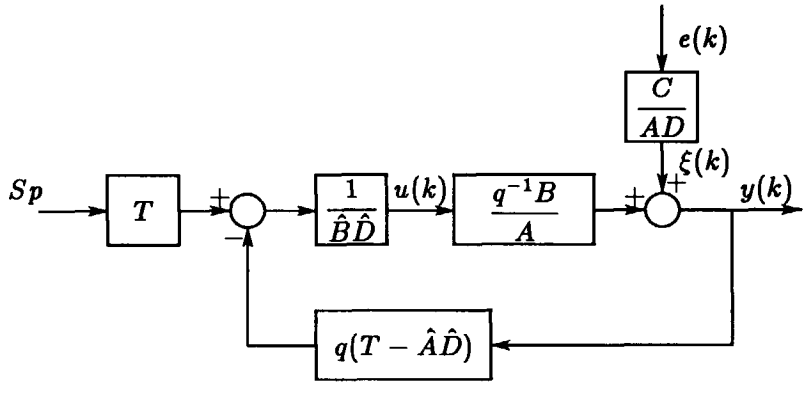

Figure 2.7: Closed-loop system of the minimum-variance controller.

depicted as in Figure 2.8 in which $\mathcal{R}, \mathcal{S}$ and $\mathcal{T}$ are polynomials in q^{-1}. Further, $\mathcal{R}$ is normalized such that $r_0 = 1$ (see e.g. [19], pp.225). From, Figure 2.8 follows:

$$y(k) = \frac{q^{-d-1}B\mathcal{T}}{A\mathcal{R} + q^{-d-1}B\mathcal{S}}Sp + \frac{\frac{C\mathcal{R}}{D}}{A\mathcal{R} + q^{-d-1}B\mathcal{S}}e(k) \tag{2.76}$$

$$u(k) = \frac{A\mathcal{T}}{A\mathcal{R} + q^{-d-1}B\mathcal{S}}Sp - \frac{\frac{C\mathcal{S}}{D}}{A\mathcal{R} + q^{-d-1}B\mathcal{S}}e(k) \tag{2.77}$$

The closed-loop poles can be found by solving $A\mathcal{R} + q^{-d-1}B\mathcal{S}$. Subsequently, the closed-loop systems in this thesis will be shown as in Figure 2.8.

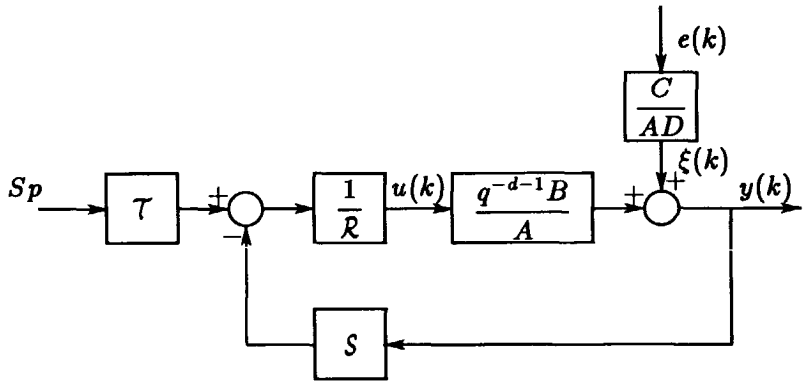

Figure 2.8: Standard visualization of a discrete closed-loop system.

Now, $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ become for the closed-loop system depicted in Figure 2.7:

$$\mathcal{R} = \frac{\hat{B}\hat{D}}{\hat{b}_0} \tag{2.78}$$

$$\mathcal{S} = \frac{q(T - \hat{A}\hat{D})}{\hat{b}_0} \tag{2.79}$$

$$\mathcal{T} = \frac{T}{\hat{b}_0} \tag{2.80}$$

and hence:

$$A\mathcal{R} + q^{-1}B\mathcal{S} = \frac{1}{\hat{b}_0}\left[BT + A\hat{B}\hat{D} - \hat{A}B\hat{D}\right] \tag{2.81}$$

Now, the transfer functions (2.76) and (2.77) become:

$$y(k) = \frac{q^{-1}BT}{BT + A\hat{B}\hat{D} - \hat{A}B\hat{D}}Sp + \frac{\hat{B}C\hat{D}}{D(BT + A\hat{B}\hat{D} - \hat{A}B\hat{D})}e(k) \tag{2.82}$$

$$u(k) = \frac{AT}{BT + A\hat{B}\hat{D} - \hat{A}B\hat{D}}Sp - \frac{qC(T - \hat{A}\hat{D})}{D(BT + A\hat{B}\hat{D} - \hat{A}B\hat{D})}e(k) \tag{2.83}$$

When the process is correctly estimated (hence, $A = \hat{A}$, $B = \hat{B}$, $C = \hat{C} = T$ and $D = \hat{D}$), the transfer functions become:

$$y(k) = \frac{q^{-1}BC}{BC}Sp + \frac{BCD}{BCD}e(k) = q^{-1}Sp + e(k) \tag{2.84}$$

$$u(k) = \frac{AC}{BC}Sp - \frac{qC(C - AD)}{BCD}e(k) = \frac{A}{B}Sp - \frac{q(C - AD)}{BD}e(k) \tag{2.85}$$

Further, if only the input-output behavior of the process is correctly estimated (hence, $A = \hat{A}$ and $B = \hat{B}$) then the process output is given by:

$$y(k) = q^{-1}Sp + \frac{\hat{A}\hat{D}}{T}\xi(k) \tag{2.86}$$

where $\xi(k)$ is a disturbance acting on the output of the process (see e.g. Figure 2.8). Considering the above, the following can be said with respect to the controller obtained when minimizing the single-step criterion (2.73):

- If the set point Sp is constant, equation (2.84) shows that the process output variance is equal to the noise variance $e(k)$. Because the noise is assumed to be a discrete white noise sequence with zero mean, this is the minimum variance that can be achieved. Hence, minimizing (2.73) with respect to $u(k)$ using the MV 1-step-ahead predictor yields the well-known minimum-variance controller (see e.g. [34], pp.166 and [21], pp.175-177).

- In the absence of noise, equation (2.84) shows that the process output $y(k)$ tracks the desired output Sp in minimum time and hence the controller obtained by minimizing (2.73) is identical to the minimum settling time (MSTC) controller [35].

- If the input-output behavior of the process is correctly estimated, then (2.86) shows that the T polynomial has no influence on the servo behavior (= behavior of the closed-loop system with $\xi(k) = 0$) of the closed-loop system. Then, T can be used to suppress disturbances on the output of the process. For example, high frequencies in $\xi(k)$ appear attenuated on the process output if the filter $\frac{\hat{A}\hat{D}}{T}$ is a low-pass filter.

  Further, (2.86) shows that including Δ as a factor of $\hat{D}$ eliminates constant disturbances from the process output. Combination of both properties can be obtained by making the filter $\frac{\hat{A}\hat{D}}{T}$ a band-pass filter.

- If the process has zeros outside the unit circle (= the process is inverse unstable), then (2.85) shows that the controller output is unbounded. Moreover, (2.81) shows that in this case the closed-loop system is unstable. The resulting response, however, is theoretically stable because the unstable poles are canceled by the zeros of B in (2.82).

  If the zeros of the process are inside the unit circle and on the negative real axis, then the controller has ringing poles which results in an oscillating controller output and intersampling ripple ([19], pp.295).

To conclude, the controller obtained by minimizing (2.73) cannot be used to control processes with an unstable inverse (such as non-minimum phase processes). Further, the performance of the closed-loop system can be unacceptable if the (discrete) process has zeros on the negative real axis within the unit circle, resulting in an oscillating controller output and intersampling ripple. Moreover, both the MV and the MSTC controllers are known for their large controller outputs.

Dead-beat control

One way to overcome problems with processes having badly situated zeros (= zeros that, if canceled, yield badly damped poles), is by excluding the zeros of the process from the closed-loop characteristic equation (2.81). For example, one can design $\mathcal{R}$ and S such that:

$$A\mathcal{R} + q^{-1}BS = 1 \tag{2.87}$$

In this case, the closed-loop poles are all chosen to be in the origin and a dead-beat controller is obtained (see [19], pp.203). One way to make sure that (2.87) holds, is by solving $\mathcal{R}$ and S from (2.87). This approach is called pole-placement design and is discussed in more detail in [19], Chapter 10. Note that a solution for $\mathcal{R}$ and S which satisfies (2.87) exists only if the following inequality is satisfied:

$$n_R + n_S + 1 \geq \max(n_A + n_R, n_B + n_S + 1)$$

In the case of equality, a unique solution for $\mathcal{R}$ and S exists.

Another way of realizing a dead-beat controller is by minimizing a multi-step criterion function as will be shown in section 2.2.2.

The transfer functions (2.76) and (2.77) for a dead-beat controller with $T = 1/B(1)$ (in order to prevent steady-state errors), $d = 0$ and $e(k) = 0$ are:

$$y(k) = \frac{q^{-1}B}{B(1)}Sp \tag{2.88}$$

$$u(k) = \frac{A}{B(1)}Sp \tag{2.89}$$

With respect to the dead-beat controller, the following can be said:

- Equations (2.88) and (2.89) show that the process and controller outputs are a moving average of the set point. In contrast with the minimum-variance controller, no problems arise in controlling inverse unstable processes.
- Equation (2.88) shows that the process output tracks the set point in $n_B + 1$ samples and hence the response is slower than that of the MSTC controller.
- Equation (2.89) shows that the number of different controller outputs that the controller must calculate in order to obtain this response is equal to $n_A + 1$.

Figure 2.9 illustrates the process and controller output for the dead-beat controller when the set point is constant and $n_A > n_B$.

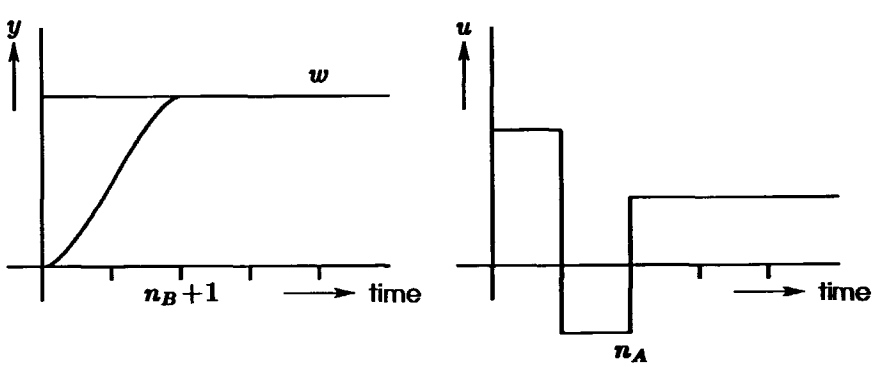

Figure 2.9: Typical dead-beat response.

Criterion functions based on tracking error and controller output

Another way of getting around the problems one has with controlling inverse unstable processes is to modify the criterion function in such a way that the controller output is taken into account too:

$$J = [\hat{y}(k+1) - w(k+1)]^2 + \rho u(k)^2 \tag{2.90}$$

where ρ is a weighting factor ≥ 0. Now two conflicting objectives arise: the minimization of the tracking error and the minimization of the controller output. The weighting factor ρ is introduced to make a tradeoff between these objectives. The minimal value of (2.90) with respect to $u(k)$ is obtained by setting the gradient $\frac{\partial J}{\partial u(k)}$ equal to zero:

$$u(k) = \frac{\hat{b}_0 T}{\hat{b}_0 \hat{B}\hat{D} + \rho T} Sp - \frac{q\hat{b}_0(T - \hat{A}\hat{D})}{\hat{b}_0 \hat{B}\hat{D} + \rho T} y(k) \tag{2.91}$$

where Sp is used instead of $w(k+1)$ as in the previous section. If (2.91) is written in the standard form depicted in Figure 2.8, the polynomials $\mathcal{R}$, S and $\mathcal{T}$ become:

$$\mathcal{R} = \frac{\hat{b}_0 \hat{B}\hat{D} + \rho T}{\hat{b}_0 + \rho} \tag{2.92}$$

$$S = \frac{q\hat{b}_0(T - \hat{A}\hat{D})}{\hat{b}_0 + \rho} \tag{2.93}$$

$$\mathcal{T} = \frac{\hat{b}_0 T}{\hat{b}_0 + \rho} \tag{2.94}$$

The closed-loop poles can be found by solving:

$$A\mathcal{R} + q^{-1}BS = \frac{1}{\hat{b}_0 + \rho}\left[T(\rho A + \hat{b}_0 B) + \hat{b}_0 A\hat{A}\hat{D}\left(\frac{\hat{B}}{\hat{A}} - \frac{B}{A}\right)\right] \tag{2.95}$$

The closed-loop transfer function, if the process is correctly estimated, is:

$$y(k) = \frac{q^{-1}}{1 + \frac{\rho A}{b_0 B}} Sp + \frac{1 + \frac{\rho C}{b_0 BD}}{1 + \frac{\rho A}{b_0 B}} e(k) \tag{2.96}$$

Equation (2.95) shows that with $\rho > 0$ the zeros of the process no longer appear as closed-loop poles. Hence, an inverse unstable process no longer yields an unstable controller for a well-chosen $\rho > 0$. Moreover, (2.96) shows that if $\rho > 0$ the process output variance is greater than the noise variance. Another disadvantage weighting the controller output is that steady-state errors can occur due to ρ. This is clearly demonstrated by (2.96). The steady-state error, for a constant set point Sp, is given by:

$$\epsilon_{ss} = Sp - y_{ss} = \frac{\rho Sp}{\rho + \frac{B(1)}{A(1)} b_0} = \frac{\rho Sp}{\rho + K_{dc} b_0}$$

where K_{dc} is the DC gain of the process. Only if $\rho = 0$ or the process contains at least one integrator (then $K_{dc} \to \infty$), is the steady-state error equal to zero. The larger the DC gain of the process, the smaller the steady-state error for a given value of ρ. The effect of ρ on the steady-state error is the reason why in many controllers the controller increments are weighted instead of the controller outputs:

$$J = [\hat{y}(k+1) - w(k+1)]^2 + \rho \Delta u(k)^2$$

In the steady state the controller increments are equal to zero if the reference signal and the disturbances are constant. Hence, in the steady state, ρ has no effect on the criterion function and thus none on the controller output.

The major disadvantage of using weighting factors in criterion functions in order to obtain the desired behavior of the closed-loop system is that it is difficult to select their values a priori. In order to overcome this problem, the following criterion function can be used:

$$J = [P\hat{y}(k+1) - Rw(k+1)]^2 + [Qu(k)]^2 \tag{2.97}$$

in which P, Q and R are polynomials and P is monic. Although this results in more parameters to tune, it is easier to obtain a desired closed-loop behavior as is shown next. Criterion function (2.97) is used in, for example, the generalized minimum-variance controller (GMV, [36]). Minimization of (2.97) with $Q = 0$ results in:

$$P\hat{y}(k+1) = Rw(k+1) \tag{2.98}$$

Now the criterion function is equal to zero. The MV 1-step-ahead predictor predicting $Py(k+1)$ can be derived by using the following Diophantine equation (see also section A.3):

$$\frac{PT}{\hat{A}\hat{D}} = E_1 + q^{-1}\frac{F_1}{\hat{A}\hat{D}} \implies E_1 = 1 \tag{2.99}$$

$$F_1 = q(PT - \hat{A}\hat{D})$$

Using (2.74) and (2.99) the MV 1-step-ahead predictor becomes:

$$P\hat{y}(k+1) = \frac{\hat{B}\hat{D}}{T}u(k) + \frac{q(PT - \hat{A}\hat{D})}{T}y(k) \tag{2.100}$$

From (2.98) and (2.100) follows the control law:

$$u(k) = \frac{RT}{\hat{B}\hat{D}}Sp - \frac{q(PT - \hat{A}\hat{D})}{\hat{B}\hat{D}}y(k) \tag{2.101}$$

where Sp is used instead of $w(k+1)$ as discussed before. If (2.101) is written in the standard form depicted in Figure 2.8, the polynomials $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ become:

$$\mathcal{R} = \frac{\hat{B}\hat{D}}{\hat{b}_0} \tag{2.102}$$

$$\mathcal{S} = \frac{q(PT - \hat{A}\hat{D})}{\hat{b}_0} \tag{2.103}$$

$$\mathcal{T} = \frac{RT}{\hat{b}_0} \tag{2.104}$$

The closed-loop poles can be found by solving:

$$A\mathcal{R} + q^{-1}B\mathcal{S} = \frac{1}{\hat{b}_0}\left[BPT + A\hat{A}\hat{D}\left(\frac{\hat{B}}{\hat{A}} - \frac{B}{A}\right)\right] \tag{2.105}$$

The closed-loop transfer function (2.76) becomes with $d = 0$:

$$y(k) = \frac{q^{-1}BTR}{BPT + A\hat{B}\hat{D} - \hat{A}B\hat{D}}Sp + \frac{\frac{C\hat{B}\hat{D}}{D}}{BPT + A\hat{B}\hat{D} - \hat{A}B\hat{D}}e(k) \qquad (2.106)$$

If the process is correctly estimated (2.106) becomes:

$$y(k) = \frac{q^{-1}R}{P}Sp + \frac{1}{P}e(k) \qquad (2.107)$$

and the closed-loop poles (2.105) are now given by:

$$A\mathcal{R} + q^{-1}B\mathcal{S} = BPT \qquad (2.108)$$

If only the input-output behavior of the process is correctly estimated (hence, $A = \hat{A}$ and $B = \hat{B}$), (2.106) becomes:

$$y(k) = \frac{q^{-1}R}{P}Sp + \frac{\hat{A}\hat{D}}{PT}\xi(k) \qquad (2.109)$$

Considering the above, the following conclusions can be drawn:

- The 'gains' of the polynomials P and R must be equal in order to prevent steady-state errors: $P(1) = R(1)$.
- By choosing P and R, the desired closed-loop transfer function can be defined if $A = \hat{A}$ and $B = \hat{B}$. Hence, one can say that minimization of (2.97) is another way of solving the pole-placement equation (2.108).
- The transfer function (2.109) shows clearly that P and T have the same effect on the regulator behavior of the control system. However, in contrast with T, the polynomial P affects also the servo behavior of the closed-loop system. This can be prevented by making P a factor of $\hat{D}$. Then, the servo behavior of the closed-loop system is defined by R and P while the regulator behavior is defined by $\hat{D}$ and T.
- The characteristic equation (2.108) shows that if the process is correctly estimated, the zeros of the process appear as closed-loop poles. Hence, as with the minimum-variance controller, problems arise when an inverse unstable process

must be controlled. A way to avoid this is by using controller output weighting as in (2.90) by making $Q \neq 0$. However, this also makes the closed-loop transfer function different from (2.106) and hence the direct connection between pole-placement controllers and controllers based on criterion functions no longer exists. Further, it may be quite difficult to choose Q.

As with the minimum-variance controller (2.75), a way to overcome problems when controlling inverse unstable processes is by selecting R and S such that the closed-loop poles are in PT (instead of BPT). Then (2.108) becomes:

$$AR + q^{-1}BS = PT \tag{2.110}$$

Together with $T = RT/B(1)$ the closed-loop transfer function (2.109) is:

$$y(k) = \frac{q^{-1}RB}{B(1)P}Sp + \frac{AR}{PT}\xi(k) \tag{2.111}$$

Now, the P polynomial can be used to define the desired closed-loop poles and consequently the desired servo behavior. Note that if $P = R = 1$, a dead-beat controller is obtained. Further, by choosing $P = B$ and $R = P(1)$, (2.111) shows that a minimum-variance controller is obtained. As has already been discussed, problems arise if badly situated process zeros are canceled. Therefore, the B polynomial is often factorized as:

$$B = B^+B^-$$

in which B^+ is a polynomial containing the zeros that, if cancelled, yield well-damped poles and B^- contains the badly situated zeros. For more details on how to choose B^+ and B^- see [19], pp.227.

By selecting $P = B^+$ and $R = P(1)$ in (2.111), only the well-situated zeros appear as poles in the closed-loop characteristic equation. The closed-loop transfer function now becomes:

$$y(k) = \frac{q^{-1}B^-}{B^-(1)}Sp + \frac{AR_1}{T}\xi(k) \tag{2.112}$$

where:

$$\mathcal{R} = B^+\mathcal{R}_1$$

because B^+ is a factor of both P and B in (2.110).

Equation (2.112) shows that $y(k)$ is a moving average of the set point. Further, if $T = 1$, $y(k)$ is a moving average of $\xi(k)$ too. Therefore, the controller obtained when choosing $P = B^+$ is called a *moving-average controller* [21], pp.175. Notice that the minimum-variance (for which $B^+ = B$) and dead-beat controllers (for which $B^+ = 1$) are special cases of the moving-average controller. The moving-average controller itself is a special case of the pole-placement controller.

The controller polynomials $\mathcal{R}$ and $\mathcal{S}$ must be solved from (2.110). However, as will be shown in section 2.2.2, the transfer function (2.111) can also be obtained by minimizing a multi-step criterion function.

Remark: usually R is chosen equal to $P(1)$ and is hence a scalar. Subsequently, in the thesis, R is assumed to be equal to $P(1)$ unless otherwise noted.

2.2.2 Multi-step Criterion Functions

If the prediction horizon is greater than one, a multi-step criterion function is required. A simple multi-step criterion function is given by:

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 \tag{2.113}$$

Criterion function (2.113) is the multi-step equivalent of the single-step criterion function (2.73). However, Theorem 2.2 shows that minimizing (2.113) yields the same controller as when (2.73) is minimized. Hence, there is no advantage to minimizing a multi-step criterion function in this case.

If the controller output is weighted too, the controller depends on the prediction horizon. An example of a multi-step criterion function weighting the controller output is the one used in the IMAC controller (Interlaced Multipredictor Adaptive Controller [37]):

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 + \rho u(k+i-1)^2$$

As with the single-step criterion function (2.90), weighting the controller output directly results in steady-state errors when the process does not contain one or more integrators. Again, this undesirable effect (for constant disturbances and reference trajectories) can be avoided by weighting the controller increments instead:

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 + \rho \Delta u(k+i-1)^2 \tag{2.114}$$

Together with a criterion function like (2.114), an additional constraint is often taken into account: in a certain interval, determined by the control horizon H_c and the prediction horizon, the controller increments are assumed to be zero:

$$\Delta u(k+i-1) = 0 \qquad 1 \leq H_c < i \leq H_p \tag{2.115}$$

Hence, $u(k) \ldots u(k+H_c-1)$ are calculated, all other controller outputs over the prediction horizon are assumed to be constant. The concept of using a control horizon was introduced by Cutler and Ramaker [2] and was used in their DMC (Dynamic Matrix Control) controller. The reason to introduce such an additional constraint is that the control horizon H_c is easy to select and in many situations makes the weighting of the controller output by means of ρ superfluous. Another useful extension which can be added to the criterion function (2.114) can be made by introducing an interval determined by H_s called the minimum cost horizon in which the (predicted) tracking errors are not weighted:

$$J = \sum_{i=H_s}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 + \sum_{i=1}^{H_p} \rho \Delta u(k+i-1)^2 \tag{2.116}$$

Now, the tracking errors for the first $H_s - 1$ samples are not taken into account in the optimization. The reason for doing this will be explained later in this section. Criterion function (2.116) in combination with (2.115) is the criterion that is used in the GPC controller [5].

The following sections show how several classical controllers can be obtained by making a proper selection of H_p, H_c, H_s and ρ.

Minimum-variance control

This section shows the relation between minimum-variance control and the criterion parameters H_p, H_c, H_s and ρ.

Theorem 2.2 *If $H_c = H_p$, $H_s = 1$ and $\rho = 0$, then, $\forall H_p \geq 1$, the controller obtained when minimizing (2.116) is the minimum-variance controller given by (2.75).*

Proof. In this case criterion function (2.116) is equal to (2.113). Further, because $H_c = H_p$, (2.115) is not in effect. The minimal value of (2.116) can be obtained by setting (then $J = 0$):

$$\hat{y}(k+i) = w(k+i) \quad 1 \leq i \leq H_p \tag{2.117}$$

Now, recall the prediction model (2.43):

$$\hat{\boldsymbol{y}} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} + \boldsymbol{k} \tag{2.118}$$

in which:

$$\begin{aligned} \hat{\boldsymbol{y}} &= [\hat{y}(k+1), \cdots, \hat{y}(k+H_p)]^T \\ \boldsymbol{u} &= [u(k), \cdots, u(k+H_p-1)]^T \end{aligned}$$

Rewriting (2.117) taking into account (2.118) yields:

$$\boldsymbol{\Gamma u} = \boldsymbol{w} - \boldsymbol{\Psi s} - \boldsymbol{k} \tag{2.119}$$

in which:

$$\boldsymbol{w} = [w(k+1), \cdots, w(k+H_p)]^T$$

Now $\boldsymbol{u}$ must be solved from (2.119). However, because $\boldsymbol{\Gamma}$ is lower triangular, the prediction horizon H_p does not have affect $u(k)$. Hence, the same $u(k)$ is found when (2.116) is minimized using $H_p = H_c = 1$. It was shown in section 2.2.1 that in this case the minimum-variance controller given by (2.75) is obtained.

□

When there is time delay, it is easy to show that minimum-variance control is obtained if $H_c = H_p - d$, $H_s = d + 1$, $\rho = 0$ and $H_p \geq d + 1$. Further, as was shown in section 2.2.1, the minimum-variance controller is identical to the MSTC controller.

Hence, the process output settles to a constant reference trajectory in $d+1$ samples. However, the MSTC controller cannot be used to control processes with an unstable inverse. Moreover, in general, the process output between the samples does not settle to a constant reference trajectory in $d+1$ samples (see section 2.2.1). In the following section, the relation between other time-optimal controllers not suffering from this problem and controllers based on minimization of a multi-step criterion function is discussed.

Dead-beat control

In this section, the relation between dead-beat controllers and controllers obtained by the minimization of the quadratic criterion function (2.116), is discussed. That there is such a relation is already recognized by Clarke *et al.* (see, for example, [18] and [31]). They, however, use the relation between LQ control and predictive control to show that there is a relation between dead-beat control and controllers based on a multi-step criterion function. In this section, another approach is used (see also [38]). The following theorem gives sufficient conditions for the criterion parameters H_p, H_c, H_s and ρ for which a controller based on minimization of a multi-step criterion function coincides to a dead-beat controller.

Theorem 2.3 *If* $H_p \geq n_A + n_B + 1$, $H_c = n_A + 1$, $H_s = n_B + 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d} = 0$ *and* $e(k) = 0$, *then minimization of (2.116) makes the process output settle to a constant reference trajectory in* $n_B + 1$ *samples.*

Proof. Consider the process (2.1) with $d = \hat{d} = 0$ and $e(k) = 0$:

$$
\begin{aligned}
Ay(k) &= Bu(k-1) \Longrightarrow \\
y(k) &= -a_1 y(k-1) - \cdots - a_{n_A} y(k-n_A) && (2.120) \\
&\quad + b_0 u(k-1) + \cdots + b_{n_B} u(k-n_B-1) && (2.121)
\end{aligned}
$$

Rewriting (2.121) with k substituted by $k + n_A + n_B + 2$ and both sides multiplied by Δ yields:

$$\begin{aligned}\Delta y(k + n_A + n_B + 2) = \\ -a_1 \Delta y(k + n_A + n_B + 1) - \cdots - a_{n_A} \Delta y(k + n_B + 2) \\ +b_0 \Delta u(k + n_A + n_B + 1) + \cdots + b_{n_B} \Delta u(k + n_A + 1)\end{aligned} \tag{2.122}$$

Now, the process output settles in $n_B + 1$ samples if the right-hand side of (2.122) is equal to zero and $y(k + n_B + 1)$ is equal to Sp, where Sp is the constant reference trajectory:

$$w(k + i) = Sp \qquad H_s \leq i \leq H_p \tag{2.123}$$

Hence, the following must hold:

$$\Delta u(k + i) = 0 \qquad n_A + 1 \leq i \leq n_A + n_B + 1 \tag{2.124}$$

$$\Delta y(k + j) = 0 \qquad n_B + 2 \leq j \leq n_A + n_B + 1 \tag{2.125}$$

$$y(k + n_B + 1) = Sp \tag{2.126}$$

The condition (2.124) is satisfied if $H_c = n_A + 1$ (remember that by definition (2.115), $\Delta u(k + i) = 0$ for $i \geq H_c$). The conditions (2.125) and (2.126) are satisfied if:

$$y(k + i) = Sp \qquad n_B + 1 \leq i \leq n_A + n_B + 1 \tag{2.127}$$

It is easy to show that (2.127) holds if $H_p = n_A + n_B + 1$, $H_s = n_B + 1$, $A = \hat{A}$, $B = \hat{B}$, $d = 0$ and $e(k) = 0$. Then, taking into account the assumption $\rho = 0$ and (2.123), criterion function (2.116) becomes:

$$J = \sum_{i=n_B+1}^{n_A+n_B+1} [\hat{y}(k + i) - Sp]^2 \tag{2.128}$$

The minimal value of (2.128) can be obtained by setting:

$$\hat{y}(k + i) = Sp \qquad n_B + 1 \leq i \leq n_A + n_B + 1 \tag{2.129}$$

Now one problem has to be solved. Is it possible to calculate such a controller output sequence that (2.129) can actually be obtained? In order to show that there is a controller output sequence over the control horizon that yields (2.129), (2.128) is rewritten in a matrix notation:

$$J = [\hat{\boldsymbol{y}} - \boldsymbol{w}]^T[\hat{\boldsymbol{y}} - \boldsymbol{w}]$$

where:

$$\begin{aligned} \hat{\boldsymbol{y}} &= [\hat{y}(k+n_B+1), \cdots, \hat{y}(k+n_A+n_B+1)]^T \quad [\hat{\boldsymbol{y}}] = n_A+1 \times 1 \\ \boldsymbol{w} &= [Sp, \cdots, Sp]^T \quad [\boldsymbol{w}] = n_A+1 \times 1 \end{aligned}$$

Further, (2.115) must be satisfied:

$$\Delta u(k+i) = 0 \qquad i \geq H_c = n_A + 1 \tag{2.130}$$

Now, recall the prediction model (2.43):

$$\hat{\boldsymbol{y}} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} + \boldsymbol{k} \tag{2.131}$$

where $[\boldsymbol{u}] = H_p \times 1 = (n_A + n_B + 1) \times 1$. Because (2.130) must be satisfied, the vector $\boldsymbol{u}$ in (2.131) becomes:

$$\boldsymbol{u} = [u(k), \cdots, u(k+n_A), u(k+n_A), \cdots, u(k+n_A)]^T$$

and $[\boldsymbol{u}] = n_A + n_B + 1 \times 1$. Assuming that (2.129) holds, (2.131) can be rewritten into:

$$\boldsymbol{w} = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} + \boldsymbol{k} \tag{2.132}$$

Now, the optimization problem is reduced to solving $n_A + 1$ unknowns $(u(k), \cdots, u(k+n_A))$ from $n_A + 1$ (= the dimension of $\boldsymbol{w}$) equations. If the matrix inverse that is involved in solving $u(k), \cdots, u(k+n_A)$ from (2.132) can be calculated, a unique solution exists and hence (2.129) is obtained when (2.128) is minimized.

By using the assumptions $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d} = 0$ and $e(k) = 0$, (2.129) becomes:

$$y(k+i) = Sp \qquad n_B + 1 \leq i \leq n_A + n_B + 1$$

and hence, the condition (2.127) is satisfied. Further, condition (2.127) is also satisfied if $H_p > n_A + n_B + 1$.

Note that by choosing $H_c = n_A + 1$, $n_A + 1$ distinct controller outputs are used to drive the process output in $n_B + 1$ samples to the constant reference trajectory Sp.

□

By using the result of Theorem 2.3, Theorem 2.4 can be stated:

Theorem 2.4 *If $H_p \geq n_A + n_B + 1$, $H_c = n_A + 1$, $H_s = n_B + 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d} = 0$, $e(k) = 0$ and if the reference trajectory is equal to the set point, then the servo behavior of the closed-loop system obtained by minimization of (2.116) coincides with that of the dead-beat controller given by (2.88) and (2.89).*

Proof. From (2.88) follows directly that the process output settles to the set point in $n_B + 1$ samples. Further, (2.89) shows that $n_A + 1$ distinct controller outputs are required to do so. Now, the theorem is proved by applying Theorem 2.3 and the fact that $n_A + 1$ distinct controller outputs are calculated under the assumptions mentioned in the theorem (see the proof of Theorem 2.3).

□

By choosing the criterion parameters as in Theorem 2.4, a dead-beat controller is obtained. Hence, processes with an unstable inverse or other badly situated zeros can be controlled without causing an unstable closed-loop system or intersampling ripple. Note that the dead-beat controller is the time-optimal controller under the condition that the process output between the samples also settles to the reference trajectory in $n_B + 1$ samples. Further, this time-optimal controller is known as the ripple-free-response controller (RFRC) [35].

The previous two theorems showed how to obtain a dead-beat controller for processes without time delay. The following theorem shows how to select H_p, H_s and H_c in (2.116) for processes with time delay.

Theorem 2.5 *If* $H_p \geq n_A + n_B + d + 1$, $H_c = n_A + 1$, $H_s = n_B + d + 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$ *and* $e(k) = 0$, *then minimization of (2.116) makes the process output settle to a constant reference trajectory in* $n_B + d + 1$ *samples.*

Proof. The proof is identical to the proof of Theorem 2.3 in which n_B is replaced by $n_B + d$.

□

Another interesting choice for the criterion parameters is $H_s = 1$, $H_c = n_A + 1$, $H_p \to \infty$ and $\rho = 0$. The major difference from the criterion parameters mentioned in Theorem 2.3 and 2.4 is that now the tracking errors for $i = 1, \ldots, H_s - 1$ are also taken into account in the minimization of (2.116). Obviously, large tracking errors in the first $H_s - 1$ samples are not allowed. It is shown in [35] (pp.280) that dead-beat control of a process with no zeros in the s-plane, does not result in overshoot and hence does not yield large values for the process output in the first $H_s - 1$ samples. Therefore, the controller obtained when minimizing (2.116) with $H_s = 1$, $H_c = n_A + 1$, $H_p \to \infty$ and $\rho = 0$ will be similar to the dead-beat controller. However, if, for example, the process is non-minimum phase, this is no longer the case. In general, time-optimal control of a non-minimum phase process results in a large negative process output for $t = k + 1$. When criterion function (2.116) is minimized with $H_s = 1$, $H_c = n_A + 1$, $H_p \to \infty$ and $\rho = 0$, this large process output will not occur because it would result in a large criterion value. Hence, a much slower response will result. The above-mentioned statements are illustrated by Figures 2.10 and 2.11. Figure 2.10 shows the response of a non-minimum phase process controlled by a time-optimal controller (i.e. $H_p = n_A + n_B + 1$, $H_s = n_B + 1$, $H_c = n_A + 1$ and $\rho = 0$). Figure 2.11 shows the response of the same process now controlled by a controller obtained when minimizing (2.116) with $H_p \gg n_A + n_B + 1$, $H_s = 1$, $H_c = n_A + 1$ and $\rho = 0$. The figures clearly show that in the latter case the response is much slower than the time-optimal response. If criterion function (2.116) with $H_p \gg n_A + n_B + 1$, $H_s = 1$ and $\rho = 0$ is calculated for both responses, the one corresponding to the response shown in Figure 2.11 is approximately 3 times smaller than the one corresponding to the response shown in Figure 2.10. This is obviously caused by the large negative process output at $t = 0.2$s which is in turn caused by the time-optimal control of the non-minimum phase process. To conclude, if the process is minimum phase, then the controller obtained when minimizing (2.116) with $H_p \gg n_A + n_B + 1$, $H_s = 1$, $H_c = n_A + 1$ and $\rho = 0$ approximates the time-optimal controller. When the process is non-minimum phase, the controller obtained

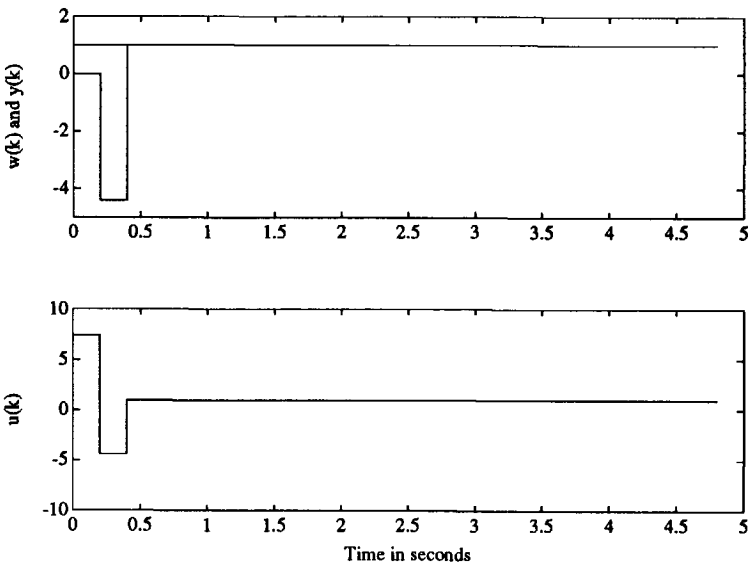

Figure 2.10: Time-optimal control of a non-minimum phase process: $H_p = n_A + n_B + 1$, $H_s = n_B + 1$, $H_c = n_A + 1$ *and* $\rho = 0$.

when minimizing (2.116) with $H_p \gg n_A + n_B + 1$, $H_s = 1$, $H_c = n_A + 1$ and $\rho = 0$ yields a much slower response than the time-optimal one.

Mean-level control

Making the control horizon smaller than $n_A + 1$, results in a less active controller output. When $H_c = 1$ only $u(k)$ is calculated, all other controller outputs over the prediction horizon are assumed to be equal to $u(k)$. When $H_p \to \infty$, $H_c = 1$ $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$ the closed-loop transfer functions are obviously given by:

$$\begin{aligned} u(k) &= \frac{A(1)}{B(1)} Sp \\ y(k) &= \frac{q^{-d-1} A(1) B(q^{-1})}{B(1) A(q^{-1})} Sp \end{aligned} \tag{2.133}$$

This controller is called the mean-level controller. The transfer function (2.133) shows that the closed-loop characteristic equation is equal to A. Further, as is shown

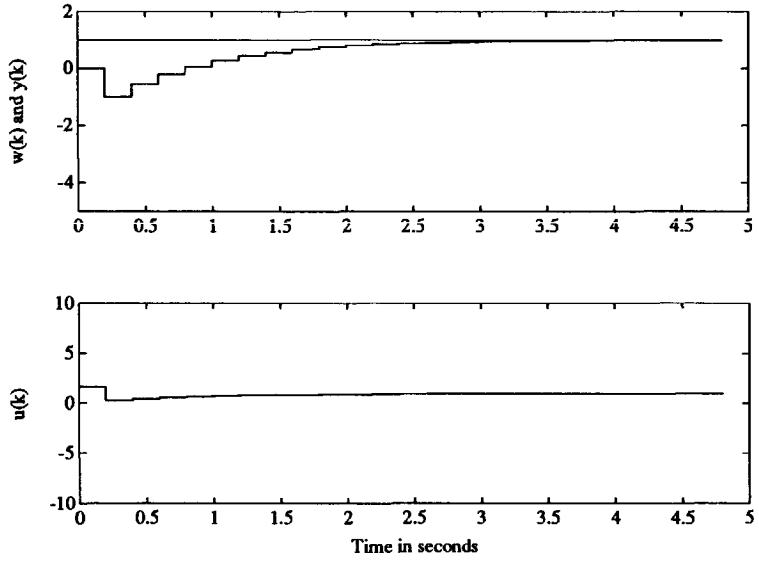

Figure 2.11: Time response of a non-minimum process with $H_p \gg n_A + n_B + 1$, $H_s = 1$, $H_c = n_A + 1$ *and* $\rho = 0$.

in section 2.3.3, T is a factor of the CLCE if the process is correctly estimated and hence:

$$AR + q^{-d-1}BS = AT \tag{2.134}$$

From (2.133) follows using (2.134) and (2.76) with $e(k) = 0$:

$$\mathcal{T} = \frac{A(1)T}{B(1)}$$

Now, the closed-loop transfer function (2.76) becomes with $e(k) \neq 0$ and $d = 0$:

$$y(k) = \frac{q^{-1}A(1)B}{B(1)A}Sp + \frac{R}{T}\xi(k) \tag{2.135}$$

where $\xi(k)$ is a disturbance acting on the output of the process. Equation (2.135) shows that apart from a gain factor, the servo behavior of the closed-loop system

is identical to that of the open-loop system. Therefore, the response to a set point change may be quite slow. However, the rejection of disturbances may be quite fast. If, for example, $T = 1$ and $\mathcal{R}$ contains a factor Δ then step wise disturbances are rejected in n_R samples. In section 2.3.3 it is shown how an arbitrary number of factors Δ can be introduced in $\mathcal{R}$. Further, T can be used to obtain the desired regulator behavior. Note that T has no effect on the servo behavior of the closed-loop system.

Remarks:

- If the process is unstable, the mean-level control strategy cannot be used because then the closed-loop system would be unstable too.
- If $T = 1$ and $\mathcal{R}$ and $\mathcal{S}$ are solved from (2.134) with minimum degree, then $\mathcal{R} = 1$ and $\mathcal{S} = 0$. Hence, feedback no longer exists and disturbances are no longer rejected.

Extensions to the criterion function

So far, the reference trajectory was assumed to be constant. However, one may wonder if, for example, ramp-wise reference trajectories have an effect on the criterion parameters. If the reference trajectory is not constant, steady-state errors can occur when using a control horizon smaller than H_p. A detailed discussion of steady-state errors caused by non-constant reference trajectories can be found in section 2.3.3. In order to prevent steady-state errors caused by the control horizon, the definition (2.115) must be extended:

$$N(q^{-1})\Delta^{\beta}u(k+i-1) = 0 \qquad 1 \leq H_c < i \leq H_p - \underline{d} \tag{2.136}$$

where N is a monic polynomial in q^{-1} and β is an integer ≥ 1. Note that if $\beta = 1$ and $N = 1$, the former definition of the control horizon (2.115) is obtained. Now, Theorem 2.5 can be modified such that a constant reference trajectory is no longer required to obtain a dead-beat controller. Consider a reference trajectory of type r (where $r = 1$ denotes a constant trajectory, $r = 2$ denotes a ramp-wise trajectory and so on), then the following theorem can be proved.

Theorem 2.6 *If $H_p \geq n_A + n_B + d + r$, $H_c = n_A + r$, $N = 1$, $\beta = r$, $H_s = n_B + d + 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$ and $e(k) = 0$, then minimization of (2.116) makes the process output settle to a reference trajectory of type r in $n_B + d + 1$ samples.*

Proof. The proof is similar to that of Theorem 2.3. Consider the process (2.1) without time delay and with $e(k) = 0$:

$$\begin{aligned} Ay(k) &= Bu(k-1) \Longrightarrow \\ y(k) &= -a_1 y(k-1) - \cdots - a_{n_A} y(k-n_A) \qquad (2.137) \\ &\quad +b_0 u(k-1) + \cdots + b_{n_B} u(k-n_B-1) \qquad (2.138) \end{aligned}$$

Rewriting (2.138) with k substituted by $k+n_A+n_B+1+r$ and both sides multiplied by Δ^β yields:

$$\begin{aligned} &\Delta^\beta y(k+n_A+n_B+1+r) = \\ &\quad -a_1 \Delta^\beta y(k+n_A+n_B+r) - \cdots - a_{n_A} \Delta^\beta y(k+n_B+1+r) \\ &\quad +b_0 \Delta^\beta u(k+n_A+n_B+r) + \cdots + b_{n_B} \Delta^\beta u(k+n_A+r) \qquad (2.139) \end{aligned}$$

Now, the process output settles in n_B+1 samples to the reference trajectory if the right-hand side of (2.139) is equal to zero and $y(k+n_B+1)$ is equal to $w(k+n_B+1)$. Hence, the following must hold:

$$\begin{aligned} \Delta^\beta u(k+i) &= 0 \quad n_A+r \le i \le n_A+n_B+r \qquad (2.140) \\ \Delta^\beta y(k+j) &= 0 \quad n_B+1+r \le j \le n_A+n_B+r \qquad (2.141) \\ y(k+n_B+1) &= w(k+n_B+1) \qquad (2.142) \end{aligned}$$

The condition (2.140) is satisfied if $H_c = n_A + r$, $N = 1$ and $\beta = r$ because, by definition (2.136), $N\Delta^\beta u(k+i) = 0$ for $i \ge H_c$. The conditions (2.141) and (2.142) are satisfied if:

$$y(k+i) = w(k+i) \quad n_B+1 \le i \le n_A+n_B+r \qquad (2.143)$$

and if the reference trajectory is of type r.

Now, it is easy to show that (2.143) holds if $H_p = n_A + n_B + r$ and $H_s = n_B+1$. Taking into account the assumption $\rho = 0$, criterion function (2.116) becomes:

$$J = \sum_{i=n_B+1}^{n_A+n_B+r} [\hat{y}(k+i) - w(k+i)]^2 \qquad (2.144)$$

Minimization of (2.144) yields (taking into account the same considerations as in the proof of Theorem 2.3):

$$\hat{y}(k+i) = w(k+i) \qquad n_B + 1 \leq i \leq n_A + n_B + r \tag{2.145}$$

By using the assumptions $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d} = 0$ and $e(k) = 0$, (2.145) coincides with (2.143). Further, condition (2.143) is also satisfied if $H_p > n_A + n_B + r$.

When the process involves time delay, the proof is identical with n_B replaced by $n_B + d$ (see also Theorem 2.5).

□

Remark: satisfying the conditions mentioned in the theorem ensures that the sampled process output tracks the reference trajectory in $n_B + d + 1$ samples. If the process output must also track the reference trajectory between the samples, the process must contain at least $r - 1$ integrators [35].

Another extension which is often made to the criterion function (2.116) is weighting $P\hat{y}(k+i)$ instead of weighting the predicted process output directly [36, 31]:

$$J = \sum_{i=H_s}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p} u(k+i-1)^2 \tag{2.146}$$

where P is a monic polynomial and R is a scalar equal to $P(1)$ (see the remark on Page 53). By minimizing J, the desired closed-loop transfer function can be defined by means of P and R. For example, in section 2.2.1 it was shown that in the case $H_p = 1$, $H_c = 1$, $H_s = 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$, the closed-loop transfer function is given by (2.107):

$$y(k) = \frac{q^{-1}R}{P} Sp + \frac{1}{P} e(k)$$

The closed-loop poles are given by BPT (see (2.108)). It was shown that because B is a factor of the closed-loop characteristic equation, problems arise in controlling inverse unstable processes. In order to overcome these problems, B must not be included in the closed-loop characteristic equation and hence (2.108) becomes:

$$AR + q^{-1}BS = PT \tag{2.147}$$

Together with $T = RT/B(1)$ the servo behavior of the closed-loop system, in the absence of modelling errors, is described by:

$$y(k) = \frac{q^{-1}RB}{B(1)P} Sp \tag{2.148}$$

In order to show how (2.148) can be obtained when minimizing (2.146), the following theorem is used.

Theorem 2.7 *If $H_p \geq n_A + n_B + d + r$, $H_c = n_A + r$, $N = P$, $R = P(1)$, $\beta = r$, $H_s = n_B + d + 1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$ and $e(k) = 0$, then minimization of (2.146) makes $Py(k)$ settle to a reference trajectory of type r in $n_B + d + 1$ samples.*

Proof. The proof is similar to that of Theorem 2.6. Multiplying (2.139) times P yields:

$$\begin{aligned} &\Delta^\beta Py(k + n_A + n_B + 1 + r) = \\ &\quad -a_1\Delta^\beta Py(k + n_A + n_B + r) - \cdots - a_{n_A}\Delta^\beta Py(k + n_B + 1 + r) \\ &\quad +b_0 P\Delta^\beta u(k + n_A + n_B + r) + \cdots + b_{n_B} P\Delta^\beta u(k + n_A + r) \end{aligned} \tag{2.149}$$

By using the same considerations as those used in the proof of Theorem 2.6, the following must hold:

$$P\Delta^\beta u(k+i) = 0 \qquad n_A + r \leq i \leq n_A + n_B + r \tag{2.150}$$

$$\Delta^\beta Py(k+j) = 0 \qquad n_B + 1 + r \leq j \leq n_A + n_B + r \tag{2.151}$$

$$Py(k + n_B + 1) = P(1)w(k + n_B + 1) \tag{2.152}$$

The condition (2.150) is satisfied if $H_c = n_A + r$, $N = P$ and $\beta = r$ because, by definition (2.136), $N\Delta^\beta u(k+i) = 0$ for $i \geq H_c$. The conditions (2.151) and (2.152) are satisfied if:

$$Py(k+i) = P(1)w(k+i) \qquad n_B + 1 \leq i \leq n_A + n_B + r$$

and if the reference trajectory is of type r.

Now, the proof is completed by replacing $\hat{y}(k+i)$ by $P\hat{y}(k+i)$ in (2.144) $\cdots$ (2.145), by replacing $w(k+i)$ by $P(1)w(k+i)$ and by using the same considerations.

Again, if the process involves time delay, the proof is identical with n_B replaced by $n_B + d$ (see also Theorem 2.5).

□

Now the following theorem can be stated:

Theorem 2.8 *If* $H_p \geq n_B+n_A+d+1$, $H_c = n_A+1$, $N = P$, $\beta = 1$, $H_s = n_B+d+1$, $\rho = 0$, $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$ *and if the reference trajectory is equal to the set point, then the servo behavior of the closed-loop system obtained when minimizing (2.146) is:*

$$y(k) = \frac{q^{-d-1}P(1)B}{B(1)P}Sp \tag{2.153}$$

Proof. Rewriting (2.153) yields:

$$Py(k) = \frac{q^{-d-1}P(1)B}{B(1)}Sp \tag{2.154}$$

In order to obtain (2.154), the signal $Py(k)$ must settle to the set point in $n_B + d + 1$ samples. Considering Theorem 2.7, this can be achieved by minimizing criterion function (2.146) with $H_p \geq n_B + n_A + d + 1$, $H_c = n_A + 1$, $N = P$, $\beta = 1$, $H_s = n_B + d + 1$, $\rho = 0$ and if the reference trajectory is equal to the set point.

□

Remarks:

- In this case, the closed-loop poles are equal to P and the zeros of B are not canceled (if B is not a factor of P) and inverse unstable processes can be controlled.
- Minimization of a multi-step criterion function with the settings as in Theorem 2.8 can be regarded as a different way of solving (2.147). Hence, a pole-placement controller is obtained when choosing the UPC design parameters as in Theorem 2.8. For particular settings of P, minimum-variance, dead-beat, and moving-average control can be obtained (see section 2.2.1, Page 53). Moreover, if $P = A$, mean-level control is obtained without the necessity to use an infinite prediction horizon.

Another extension to the criterion function which is used in, for example, EPSAC (Extended Prediction Self-Adaptive Control) [11] is weighting the tracking error instead of the controller output. For this purpose weighting factors on the tracking error have been introduced in the criterion function. This yields for (2.146) with $\rho = 0$:

$$J = \sum_{i=H_s}^{H_p} \gamma_i \left[P\hat{y}(k+i) - Rw(k+i)\right]^2 \tag{2.155}$$

where γ_i is the weighting factor on the tracking error at $t = k + i$. In EPSAC, a single parameter $\lambda > 0$ is used to determine the weighting factors γ_i:

$$\gamma_i = \lambda^{H_p - i} \qquad H_s \leq i \leq H_p$$

Note that if $\lambda = 1$, the tracking error is equally weighted over the prediction horizon. Then, criterion function (2.155) coincides (2.146) with $\rho = 0$.

By choosing $\lambda < 1$, the tracking error in the near future is less important than the tracking error in the remote future. This choice for λ is motivated by the fact that if $P = R = 1$, a real process cannot track a set point change in, for example, one sample. As a result, the tracking error (now the difference between the predicted process output and the set point) can be quite large in the first few samples (= the near future). By not including these (large) tracking errors in the criterion function that is minimized, the controller will not attempt to make them smaller. Hence, choosing $\lambda < 1$ in combination with $P = R = 1$ usually yields smooth control. However, for non-minimum phase processes choosing $\lambda < 1$ and $P = R = 1$ yields opposite results. This can be explained by the fact that choosing $\lambda < 1$ is similar to choosing $H_s > d + 1$. If $\lambda \to 0$, then the same results are obtained by choosing $H_s = H_p$. That choosing $H_s > d + 1$ results in a more active control in the case of a non-minimum phase process without delay is clearly illustrated by the Figures 2.11 ($H_s = 1$) and 2.10 ($H_s > 1$). Obviously, in this case tracking errors in the near future must be taken into account and hence λ must not be chosen too small.

In the case $P \neq 1$ and $w(k+i) = Sp$, choosing $\lambda < 1$ seems not to be useful. In this case the tracking error $\epsilon(k+i)$ can be regarded as the difference between the predicted process output and the desired process output:

$$\epsilon(k+i) = \hat{y}(k+i) - \frac{1}{P}Sp$$

If the P polynomial is chosen such that the process can track the filtered set point, there is no reason to choose $\lambda < 1$. Making the controller output less active can be achieved in a natural way by imposing weighting on the controller output as in (2.146).

So far the choice $\lambda < 1$. By choosing $\lambda > 1$, the tracking error in the near future is the most important yielding more active control actions. However, in

contrast with choosing $\lambda < 1$, there seems no motivation to choose $\lambda > 1$. Moreover, simulations have shown that in controlling a non-minimum phase process, choosing $\lambda > 1$ can result in an unstable closed-loop system.

To conclude: imposing weights on the tracking error as in EPSAC seems not be necessary. Similar results can be obtained by choosing $H_s > d+1$ in combination with controller output weighting by means of ρ. This is the reason why weighting on the tracking error is not included in the UPC design. As a result, EPSAC with $\lambda \neq 1$ cannot be obtained by choosing particular settings for the UPC design parameters. In EPSAC, however, weighting on the tracking error by means of λ is required because controller output weighting and a minimum cost horizon different from $d+1$ are not included into its design.

The unified criterion function

In order to be able to select and examine different criterion functions, the following unified criterion function is used in the thesis:

$$J = \sum_{i=H_s}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p} \left[\frac{Q_n}{Q_d} u(k+i-1)\right]^2 \qquad (2.156)$$

with constraints:

$$N\Delta^\beta u(k+i-1) = 0 \qquad 1 \leq H_c < i \leq H_p \qquad (2.157)$$

where Q_n and Q_d are monic polynomials with no common factors.

If the process to be controlled contains time delay, criterion function (2.156) can still be used. However, in this case $u(k) \ldots u(k+H_p-1)$ do not have effect on $\hat{y}(k+1) \ldots \hat{y}(k+\underline{d})$ and hence on the optimization problem. Thus there is no need to include these predicted process outputs in the criterion to be minimized. Hence, the minimum cost horizon H_s can be selected greater than or equal to $\underline{d}+1$ and criterion function (2.156) becomes:

$$J = \sum_{i=H_s}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p-\underline{d}} \left[\frac{Q_n}{Q_d} u(k+i-1)\right]^2 \qquad (2.158)$$

with

$$N\Delta^{\beta}u(k+i-1)=0 \qquad 1 \leq H_c < i \leq H_p - \underline{d} \tag{2.159}$$

and $H_s \geq \underline{d}+1$.

In Table 2.3 the settings of the parameters in criterion function (2.156) are shown for 7 well-known controllers. In Table 2.3 the symbol '$\surd$' means that the parameter is a design parameter of the UPC controller and the controller that is listed at the same row of the table. Further, the symbol '-' denotes that the corresponding UPC design parameter may have any value.

Because criterion function (2.158) is a unification of many well-known criteria, it is used, together with the unified prediction model (2.32), to derive the Unified Predictive Controller (UPC) in section 2.3.

Controller	H_p	H_s	H_c	N	β	P	R	ρ	Q_n	Q_d
MV	$d+1$	$d+1$	1	-	-	1	1	0	-	-
GMV	$d+1$	$d+1$	1	-	-	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
DMC	$\surd$	$d+1$	$\surd$	1	1	1	1	$\surd$	Δ	1
MAC	$\surd$	$d+1$	H_p-d	-	-	1	1	$\surd$	Δ	1
GPC	$\surd$	$\surd$	$\surd$	1	1	$\surd$	$\surd$	$\surd$	Δ	1
EPSAC[a]	$\surd$	$d+1$	1	1	1	$\surd$	$\surd$	0	-	-
EHAC[b]	$\surd$	H_p	1	1	1	1	1	0	-	-

Table 2.3: Criterion parameters for several well-known controllers.

[a]EPSAC is considered in the case $\lambda = 1$. Only in this case, the criterion that is minimized in EPSAC can be obtained by selecting particular parameter settings in the UPC criterion function.

[b]EHAC (Extended Horizon Adaptive Control) [10] is discussed in section 3.8.6. In this table one form of EHAC is considered only.

Summary of controller design

In this section, the results of the Theorems 2.2 $\cdots$ 2.8 are summarized in a table. Table 2.4 shows how the parameters of the criterion function (2.158) with (2.159) must be selected in order to obtain, for example, a minimum-variance or deadbeat controller. It is assumed that $\rho = 0$, $\hat{A} = A$, $\hat{B} = B$, $\hat{d} = d$ and the reference trajectory is constant. Further, Table 2.5 shows the closed-loop characteristic equation (CLCE) for each controller.

Controller	H_p	H_s	H_c	N	β	P
MV/MSTC	$\geq d+1$	$d+1$	H_p-d	N	≥ 1	1
Mean-level	$\rightarrow \infty$	≥ 1	1	1	1	1
Dead-beat/RFRC	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	1	1	1
Moving-average	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	B^+	1	B^+
Pole-placement	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	P	1	P

Table 2.4: Classical controllers as a function of the criterion parameters.

Controller	CLCE
MV/MSTC	BT
Mean-level	AT
Dead-beat/RFRC	T
Moving-average	B^+T
Pole-placement	PT

Table 2.5: Classical controllers and their closed-loop characteristic equations.

Notes:

- The MV controller can also be obtained by using a pole-placement controller with $P = B$. The method using $H_p = d + 1$, $H_s = d + 1$ and $H_c = 1$ is preferable: It is the simplest and the fastest (the matrix to be inverted is a scalar, see section 2.3.1).
- The mean-level controller can also be obtained by using a pole-placement controller with $P = A$. In comparison with the method outlined in row 2 of Table 2.4, this method has the advantage that H_p is small. Its disadvantage is, however, that the matrix to be inverted is now of dimension $n_A + 1 \times n_A + 1$ instead of a scalar (see section 2.3.1).
- The T polynomial is present in the closed-loop characteristic equation for all controllers shown in Table 2.4. In section 2.3.3 it is shown that this is always the case if the process is correctly estimated (i.e. $\hat{A} = A$, $\hat{B} = B$ and $\hat{d} = d$).
- All controllers mentioned in Table 2.4 can be obtained by using the pole-placement controller with particular settings for P.

For all controllers mentioned in Table 2.4, the closed-loop transfer function (CLTF)

from the set point and the disturbance to the output is given by:

$$y(k) = \frac{q^{-d-1}B(q^{-1})P(1)}{P(q^{-1})B(1)}Sp + \frac{A(q^{-1})\mathcal{R}(q^{-1})}{P(q^{-1})T(q^{-1})}\xi(k) \tag{2.160}$$

Remarks:

- If $PT = A$ or $PT = 1$, then $y(k)$ is a moving average of the disturbance $\xi(k)$. Hence, the disturbance is rejected in $n_R + 1$ or $n_R + n_A + 1$ samples, respectively.
- If $P = B^+P_1$, where $B = B^+B^-$, then the CLTF is:

$$y(k) = \frac{q^{-d-1}B^-(q^{-1})P_1(1)}{P_1(q^{-1})B^-(1)}Sp + \frac{A(q^{-1})\mathcal{R}_1(q^{-1})}{P_1(q^{-1})T(q^{-1})}\xi(k)$$

  where $\mathcal{R} = B^+\mathcal{R}_1$.

2.2.3 Criterion Functions when there are Constraints

In many practical applications there are constraints on the controller output because D/A converters or actuators have a limited range. For example, when a controller is used for the course keeping of a ship, the controller output is supplied to the steering machine which controls the rudder. Such a steering machine has two major constraints: the rudder angle is limited to, for example, between plus or minus 30° and the speed of the rudder is limited to, for example, between plus of minus $10°/s$. The first constraint is called a level constraint while the second one is called a rate constraint. Rate constraints especially can have a disastrous effect on the stability of the closed-loop system owing to the fact that this type of constraint yields large phase shifts when it is active.

A natural way of taking into account constraints on the controller output is to minimize (2.158) in the presence of these constraints :

$$J = \sum_{i=H_e}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p-d} \left[\frac{Q_n}{Q_d}u(k+i-1)\right]^2 \tag{2.161}$$

and

$$N\Delta^{\beta}u(k+i-1)=0 \qquad 1\leq H_c<i\leq H_p-\underline{d} \tag{2.162}$$

$$\underline{u}\leq u(k+i-1)\leq \overline{u} \qquad 1\leq i\leq H_p-\underline{d} \tag{2.163}$$

$$\underline{\Delta u}\leq \Delta u(k+i-1)\leq \overline{\Delta u} \quad 1\leq i\leq H_p-\underline{d} \tag{2.164}$$

where $\underline{u}$ and $\overline{u}$ are the lower and the upper limits of the level constraint respectively; $\underline{\Delta u}$ and $\overline{\Delta u}$ are the lower and the upper limits of the rate constraint, respectively.

Because minimization of (2.161) with respect to the controller output sequence and the constraints (2.163) and (2.164) cannot be performed analytically, a nonlinear minimization method is required. Chapter 4 is devoted to the optimization problem when there are constraints on the controller output. In section 2.3, the optimal controller is derived without taking into account the constraints (2.163) and (2.164). It is shown that in this case, the optimal solution of (2.161) with (2.162) can be calculated analytically.

2.2.4 Conclusions

In this section a number of criterion functions and their properties have been discussed. It has been shown that by using multi-step criterion functions, several classical controllers, among which minimum-variance, dead-beat, moving-average and mean-level controllers can be obtained simply by selecting the parameters of the criterion function. Moreover, it was shown that for a particular setting of the parameters of the multi-step criterion function, minimization of this criterion function yields a pole-placement controller. Hence, all kinds of processes, such as inverse unstable processes, processes with time delay and unstable processes can be controlled by selecting appropriate parameters in the criterion function. Further, theorems have been provided on how to select the criterion parameters to obtain the controllers mentioned above. Moreover, the relationship between the criterion function and the resulting closed-loop system has been shown for some special choices for the parameters in the criterion function. Finally, a unified multi-step criterion function is proposed which is a unification of all the criterion functions discussed in this section. Based on this unified criterion function and the MV unified i-step-ahead predictor discussed in section 2.1, the Unified Predictive Controller is derived in the following section.

2.3 The Predictive Control Law

The optimal control law can be derived by the minimization of criterion function (2.161) subject to (2.162) with respect to the controller output sequence over the control horizon H_c: $u(k), \ldots, u(k + H_c - 1)$. In some predictive controllers, for example in the MAC controller [1] or the ECM controller [39], the minimization of the criterion function is performed by using an iterative method. However, when the process model is linear, the criterion function is quadratic and there are no constraints on the controller output, the criterion function can be minimized analytically. In this section, it is assumed that there are no constraints. In Chapter 4, the criterion function is minimized taking into account the constraints.

2.3.1 Derivation of the Unified Predictive Control Law

If the criterion function J is optimized with respect to the vector $\boldsymbol{u}$ then any local optimum $\boldsymbol{u}$ satisfies:

$$\boldsymbol{g} = \frac{\partial J}{\partial \boldsymbol{u}} = 0$$

where $\boldsymbol{g}$ denotes the gradient and the symbol ∂ denotes the partial derivative. Moreover, if the Hessian is positive definite $\forall \boldsymbol{u}$ then any local optimum is the global minimum. The Hessian $\boldsymbol{H}$ is given by:

$$\boldsymbol{H} = \frac{\partial^2 J}{\partial \boldsymbol{u}^2}$$

In order to calculate the gradient of (2.158) with respect to $\boldsymbol{u}$, the criterion function (2.158) is rewritten into a matrix notation:

$$J = (\hat{\boldsymbol{y}}^* - \boldsymbol{w}^*)^T(\hat{\boldsymbol{y}}^* - \boldsymbol{w}^*) + \rho \boldsymbol{u}^{*T}\boldsymbol{u}^* \tag{2.165}$$

where:

$$\begin{aligned}
\boldsymbol{w}^* &= [Rw(k + H_s), \ldots, Rw(k + H_p)]^T \\
\hat{\boldsymbol{y}}^* &= [P\hat{y}(k + H_s), \ldots, P\hat{y}(k + H_p)]^T \\
\boldsymbol{u}^* &= [u^*(k), \ldots, u^*(k + H_p - \underline{d} - 1)]^T \\
u^*(k) &= \frac{Q_n}{Q_d} u(k)
\end{aligned}$$

Further, introduce the vector $\overline{u}$:

$$\overline{u} = [u(k), \ldots, u(k + H_c - 1)]^T \tag{2.166}$$

Note that $\overline{u}$ contains only those elements of the controller output sequence that must be calculated. The other elements over the prediction horizon must satisfy (2.159).

The gradient of (2.165) with respect to $\overline{u}$ becomes:

$$\frac{\partial J}{\partial \overline{u}} = 2\frac{\partial \hat{y}^*}{\partial \overline{u}}(\hat{y}^* - w) + 2\rho\frac{\partial u^*}{\partial \overline{u}}u^* \tag{2.167}$$

Equation (2.167) shows that the partial derivatives $\frac{\partial \hat{y}^*}{\partial \overline{u}}$ and $\frac{\partial u^*}{\partial \overline{u}}$ are required.

Relationship between u^* and $\overline{u}$

The relationship between u and $\overline{u}$ can be derived by solving $u(k + H_c), \ldots, u(k + H_p - \underline{d} - 1)$ from (2.159):

$$N\Delta^\beta u(k + i - 1) = 0 \qquad 1 \leq H_c < i \leq H_p - \underline{d} \tag{2.168}$$

The required relationship can be obtained by using the following Diophantine equation:

$$\frac{1}{N\Delta^\beta} = E_{i-H_c} + q^{-i+H_c}\frac{F_{i-H_c}}{N\Delta^\beta} \Longrightarrow N\Delta^\beta E_{i-H_c} = 1 - q^{-i+H_c}F_{i-H_c} \tag{2.169}$$

where the degree of F_{i-H_c} is given by $\beta + n_N - 1$. From (2.169) follows using (2.168):

$$u(k + i - 1) = F_{i-H_c}u(k + H_c - 1) \qquad 1 \leq H_c < i \leq H_p - \underline{d} \tag{2.170}$$

Separation of future and past terms is realized by using:

$$F_{i-H_c} = G_{i-H_c} + q^{-H_c}H_{i-H_c} \tag{2.171}$$

in which the degrees of G_{i-H_c} and H_{i-H_c} are given by $n_G = \min(H_c, \beta + n_N) - 1$ and $n_H = \beta + n_N - H_c - 1$ respectively. Using (2.171), (2.170) becomes:

$$u(k+i-1) = G_{i-H_c}u(k+H_c-1) + H_{i-H_c}u(k-1)$$

with $1 \leq H_c < i \leq H_p - \underline{d}$. Note that if $\beta = 1$ and $N = 1$ (hence, controller outputs are assumed to be constant for $i = H_c+1, \ldots, H_p - \underline{d}$), then $G_{i-H_c} = 1$ and $H_{i-H_c} = 0$.

Now, the relationship between $\boldsymbol{u}$ and $\overline{\boldsymbol{u}}$ becomes in a matrix notation:

$$\boldsymbol{u} = \boldsymbol{M}\overline{\boldsymbol{u}} + \boldsymbol{N}\check{\boldsymbol{u}} \tag{2.172}$$

in which $\boldsymbol{M}$ is a matrix of dimension $H_p - \underline{d} \times H_c$:

$$\boldsymbol{M} = \left[\begin{array}{cccccc} 1 & 0 & & \cdots & & 0 \\ 0 & 1 & & & & \\ & & & & & \vdots \\ \vdots & & \ddots & & \ddots & \\ & & & & & 0 \\ 0 & & \cdots & & 0 & 1 \\ \hline 0 & \cdots & 0 & g_{1,n_G} & \cdots & g_{1,0} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & g_{j,n_G} & \cdots & g_{j,0} \end{array}\right] \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \\ \\ \end{array}\right\} H_c \\ \left.\begin{array}{l} \\ \\ \end{array}\right\} H_p - H_c - \underline{d} \end{array} \tag{2.173}$$

$\boldsymbol{N}$ is a matrix of dimension $(H_p - \underline{d}) \times (\beta + n_N - H_c)$:

$$\boldsymbol{N} = \left[\begin{array}{ccc} 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \\ \hline h_{1,0} & \cdots & h_{1,n_H} \\ \vdots & & \vdots \\ h_{j,0} & \cdots & h_{j,n_H} \end{array}\right] \begin{array}{l} \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} H_c \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} H_p - H_c - \underline{d} \end{array} \tag{2.174}$$

where $j = H_p - H_c - \underline{d}$ and $\check{\boldsymbol{u}}$ is given by:

$$\check{\boldsymbol{u}} = [u(k-1), \ldots, u(k+H_c-\beta-n_N)]^T \tag{2.175}$$

Note that if $H_c = H_p - \underline{d}$ then $\boldsymbol{M} = \boldsymbol{I}$ and $\boldsymbol{N} = \boldsymbol{0}$. Further, the relation between $\boldsymbol{u}$ and $\boldsymbol{u}^*$ is required:

$$u^*(k+i-1) = \frac{Q_n}{Q_d} u(k+i-1) \qquad 1 \leq i \leq H_p - \underline{d} \tag{2.176}$$

Separation of future and past elements is realized by using:

$$\frac{Q_n}{Q_d} = \Phi_i + q^{-i}\frac{\Omega_i}{Q_d} \tag{2.177}$$

where Φ_i and Ω_i are polynomials of degree $i-1$ and $\max(n_{Q_n} - i, n_{Q_d} - 1)$ if $n_{Q_d} > 0$. If $n_{Q_d} = 0$, the degrees of Φ_i and Ω_i are given by $\min(i-1, n_{Q_n})$ and $n_{Q_n} - i$ respectively. Note that because Q_n and Q_d are monic, Φ_i is monic too. Using (2.177), (2.176) becomes:

$$u^*(k+i-1) = \Phi_i u(k+i-1) + \Omega_i \frac{u(k-1)}{Q_d}$$

Collecting $u^*(k+i-1)$ for $i = 1, \ldots, H_p - \underline{d}$ in a matrix notation yields:

$$\boldsymbol{u}^* = \boldsymbol{\Phi}\boldsymbol{u} + \boldsymbol{\Omega}\tilde{\boldsymbol{u}} \tag{2.178}$$

where $\boldsymbol{\Phi}$ is a lower triangular matrix of dimension $(H_p - \underline{d}) \times (H_p - \underline{d})$ and $\boldsymbol{\Omega}$ is a matrix of dimension $(H_p - \underline{d}) \times n_\Omega$ with $n_\Omega = \max(n_{Q_n}, n_{Q_d})$. The vector $\tilde{\boldsymbol{u}}$ is given by:

$$\tilde{\boldsymbol{u}} = \left[\frac{u(k-1)}{Q_d}, \ldots, \frac{u(k-n_\Omega)}{Q_d}\right]^T$$

Using (2.172) and (2.178), the relation between $\overline{\boldsymbol{u}}$ and $\boldsymbol{u}^*$ is given by:

$$\boldsymbol{u}^* = \boldsymbol{\Phi}\boldsymbol{M}\overline{\boldsymbol{u}} + \boldsymbol{\Omega}\tilde{\boldsymbol{u}} + \boldsymbol{\Phi}\boldsymbol{N}\check{\boldsymbol{u}} \tag{2.179}$$

The partial derivative $\dfrac{\partial \boldsymbol{u}^*}{\partial \overline{\boldsymbol{u}}}$ now becomes:

$$\frac{\partial \boldsymbol{u}^*}{\partial \overline{\boldsymbol{u}}} = \boldsymbol{M}^T\boldsymbol{\Phi}^T$$

Relationship between y^* and $\overline{u}$

The partial derivative $\frac{\partial \hat{y}^*}{\partial \overline{u}}$ can be calculated by using the unified prediction model (A.32) as derived in section A.3:

$$\hat{y}^* = \Gamma u + \Psi s + k \tag{2.180}$$

in which:

$$\begin{aligned}
\hat{y}^* &= [P\hat{y}(k+H_s), \cdots, P\hat{y}(k+H_p)]^T & [\hat{y}] &= H_p - H_s + 1 \times 1 \\
u &= [u(k), \cdots, u(k+H_p-1-\underline{d})]^T & [u] &= H_p - \underline{d} \times 1 \\
s &= [y(k), \cdots, y(k-n_F), \\
&\qquad u(k-1), \cdots, u(k-n_B-\overline{d})]^T & [s] &= (n_F + 1 + n_B + \overline{d}) \times 1
\end{aligned}$$

where $n_F = \max(n_P - H_s, n_A - 1)$ and the matrices Γ and Ψ are of dimension $(H_p - H_s + 1) \times (H_p - \underline{d})$ and $(H_p - H_s + 1) \times (\max(n_P - H_s + 1, n_A) + n_B + \overline{d})$, respectively. The vector k is given by:

$$\begin{aligned}
k &= K_c \tilde{c} \\
\tilde{c} &= [\tilde{c}(k), \cdots, \tilde{c}(k-n_K)]^T \\
\tilde{c}(k) &= \frac{\hat{A}y(k) - \hat{\tilde{B}}u(k-1-\underline{d})}{T}
\end{aligned}$$

The matrix K_c is built up of the coefficients of the polynomials K_i for $i = H_s, \ldots, H_p$:

$$K_c = \begin{bmatrix} k_{H_s,0} & \cdots & k_{H_s,n_K} \\ \vdots & & \vdots \\ k_{H_p,0} & \cdots & k_{H_p,n_K} \end{bmatrix} \quad [K_c] = (H_p - H_s + 1) \times (n_K + 1)$$

where $n_K = \max(n_T, n_D) - 1$.

Remark: The above dimensions are valid only if $n_A > 0$. The dimensions of the vectors and matrices if $n_A = 0$ are discussed in section A.3.

Utilizing (2.180) and (2.172), the relation between $\hat{y}^*$ and $\overline{u}$ is given by:

$$\hat{y}^* = \Gamma M \overline{u} + \Psi s + k + \Gamma N \check{u} \tag{2.181}$$

and hence:

$$\frac{\partial \hat{y}^*}{\partial \overline{u}} = M^T \Gamma^T$$

The predictive control law

The gradient (2.167) becomes:

$$\frac{\partial J}{\partial \overline{u}} = 2M^T \left(\Gamma^T\Gamma + \rho\Phi^T\Phi\right) M\overline{u} +$$

$$+2M^T \left[\Gamma^T \left(\Psi s + k + \Gamma N\check{u} - w^*\right) + \rho\Phi^T(\Omega\tilde{u} + \Phi N\check{u})\right] \qquad (2.182)$$

and the Hessian H is:

$$H = 2M^T \left(\Gamma^T\Gamma + \rho\Phi^T\Phi\right) M \qquad (2.183)$$

Note that the Hessian is independent of $\overline{u}$ and positive definite (remember that $\rho \geq 0$). Hence, a global minimum of J with respect to $\overline{u}$ can be obtained by setting the gradient (2.182) equal to zero and solving for $\overline{u}$:

$$\overline{u} = \left[M^T \left(\Gamma^T\Gamma + \rho\Phi^T\Phi\right) M\right]^{-1} M^T \left[\Gamma^T(w^* - \Psi s -\right.$$

$$\left. k - \Gamma N\check{u}) - \rho\Phi^T(\Omega\tilde{u} + \Phi N\check{u})\right] \qquad (2.184)$$

Because this control law is based on the unified process model and the unified criterion function, this control law is called the unified predictive control law.

Note that the matrix to be inverted is of dimension $H_c \times H_c$. Hence, a small control horizon is preferable for numerical reasons and in order to save computation time. In many practical situations the control horizon can be chosen small ($H_c \leq n_A+1$, see section 2.2.2) and hence the inverse can be calculated quite easily for low-order processes. The first element of $\overline{u}$ (= $u(k)$) is used to control the process. All other elements are not used and need not be calculated. Then the unified predictive control law (2.184) can be reduced to:

$$u(k) = v^T w^* - \lambda^T\tilde{c} - l^T s - v_2^T\check{u} - \rho z^T\tilde{u} - \rho z_2^T\check{u} \qquad (2.185)$$

where:

$$
\begin{aligned}
\boldsymbol{v}^T &= \boldsymbol{x}^T\boldsymbol{\Gamma}^T && 1\times(H_p-H_s+1)\\
\boldsymbol{\lambda}^T &= \boldsymbol{v}^T\boldsymbol{K}_c && 1\times(n_K+1)\\
\boldsymbol{v}_2^T &= \boldsymbol{v}^T\boldsymbol{\Gamma}\boldsymbol{N} && 1\times(\beta+n_N-H_c)\\
\boldsymbol{x}^T &= \boldsymbol{e}_1^T\boldsymbol{R}_v{}^{-1}\boldsymbol{M}^T && 1\times(H_p-\underline{d})\\
\boldsymbol{R}_v &= \boldsymbol{M}^T\left(\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}+\rho\boldsymbol{\Phi}^T\boldsymbol{\Phi}\right)\boldsymbol{M} && H_c\times H_c\\
\boldsymbol{e}_1^T &= [1,0,\ldots,0]^T && 1\times H_c\\
\boldsymbol{l}^T &= \boldsymbol{v}^T\boldsymbol{\Psi} && 1\times[s]\\
\boldsymbol{z}^T &= \boldsymbol{x}^T\boldsymbol{\Phi}^T\boldsymbol{\Omega} && 1\times n_\Omega\\
\boldsymbol{z}_2^T &= \boldsymbol{x}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{N} && 1\times(\beta+n_N-H_c)
\end{aligned}
$$

The control law (2.185) can easily be implemented in a computer. Once the controller vectors $\boldsymbol{v}, \boldsymbol{\lambda}, \boldsymbol{l}, \boldsymbol{v}_2, \boldsymbol{z}$ and $\boldsymbol{z}_2$ are calculated, the calculation of $u(k)$ is simple and fast. Note that when $Q_n = 1$ and $Q_d = 1$ then $\boldsymbol{z} = 0$ and $\boldsymbol{\Phi}^T\boldsymbol{\Phi} = \boldsymbol{I}$. If $Q_n = \Delta$ and $Q_d = 1$ then $\boldsymbol{z} = -r_{11}$ where r_{11} is the first element of $\boldsymbol{R}_v{}^{-1}$.

Feasibility of the predictive control law

A problem that can occur when solving $\overline{\boldsymbol{u}}$ from (2.184) is that the matrix $\boldsymbol{R}_v$ may be ill conditioned or even singular. For example, if $H_s = \underline{d} + 1$ and $\rho = 0$ problems with minimization of (2.158) arise if $d > \underline{d}$. Then, $u(k), \cdots, u(k+d-1)$ do not have effect on $y(k+1), \cdots, y(k+d)$ and hence $d - \underline{d}$ rows of $\boldsymbol{\Gamma}$ will be equal to zero making $\boldsymbol{\Gamma}^T\boldsymbol{\Gamma}$ singular. The control horizon H_c can be used in order to prevent $\boldsymbol{R}_v$ from being singular. It is easy to show that now H_c must be chosen such that $H_c \leq H_p - d$.

In general, $\boldsymbol{R}_v$ is singular if there is no unique solution to the optimization problem. Such a situation can easily occur. Consider the criterion (2.158) with (2.159) and $\rho = 0$. Then H_c different controller outputs must be determined such that the predicted process outputs $P\hat{y}(k+H_s), \cdots, P\hat{y}(k+H_p)$ minimize (2.158). Obviously, if $H_c > H_p - H_s + 1$, more than one solution exists and hence $\boldsymbol{R}_v$ is singular.

To conclude, we can state that if $\rho = 0$, at least the condition $H_c \leq H_p - \max(H_s - 1, d)$ must be satisfied in order to avoid singularity of $\boldsymbol{R}_v$. Usually, the time delay of the process d is not exactly known. Then, at least, H_c must be chosen such that: $H_c \leq H_p - \max(H_s - 1, \overline{d})$.

Also in some other situations the matrix $\boldsymbol{R}_u$ can be singular or badly conditioned. A simple solution to this problem is to use a small value for ρ (e.g. $\rho = 10^{-6}$).

2.3.2 The Polynomial Approach

In this section it is shown that the unified predictive control law (2.185) can be written as:

$$\mathcal{R}u(k) = -\mathcal{S}y(k) + \mathcal{T}w(k + H_p) \tag{2.186}$$

The control law (2.186) is the one used in many pole-placement controllers [21] and can also be used to describe the standard PID controller. In these cases $w(k + H_p)$ is equal to the set point. The purpose of writing the predictive control law in a polynomial form is for analysis purposes. The implementation of the predictive control law is realized by using (2.185).

Rewriting (2.185) in a polynomial form yields:

$$u(k) = \tilde{V}w^*(k + H_p) - \tilde{L}_1 y(k) - \tilde{L}_2 u(k-1) - \tilde{V}_2 u(k-1) -$$

$$\rho\tilde{Z}_2 u(k-1) - \rho\tilde{Z}\frac{u(k-1)}{Q_d} - \Lambda\left[\frac{\hat{A}y(k) - q^{-d}\hat{\tilde{B}}u(k-1)}{T}\right] \tag{2.187}$$

in which:

$$\begin{aligned}
\tilde{\Lambda} &= P\{\lambda^T\} \\
\tilde{V} &= v_{H_p-H_c+1} + v_{H_p-H_c}q^{-1} + \cdots + v_1 q^{-H_p+H_c} \\
\tilde{V}_2 &= P\{v_2^T\} \\
\tilde{L}_1 &= P\{l_1^T\} \\
\tilde{L}_2 &= P\{l_2^T\} \\
\tilde{Z} &= P\{z^T\} \\
\tilde{Z}_2 &= P\{z_2^T\}
\end{aligned}$$

where $\mathcal{P}$ is an operator that translates a vector into a polynomial (see the list of symbols). Further, $\boldsymbol{l}_1$ is made up of the first $n_F + 1$ elements of $\boldsymbol{l}$ while $\boldsymbol{l}_2$ consists of the last $n_B + \overline{d}$ elements of $\boldsymbol{l}$.

Rewriting (2.187) results in:

$$\left[TQ_d + q^{-1}\left(\tilde{L}_2 TQ_d + \tilde{V}_2 TQ_d + \rho\tilde{Z}T + \rho\tilde{Z}_2 TQ_d - \Lambda Q_d q^{-\underline{d}}\hat{\tilde{B}}\right)\right] u(k) =$$

$$TQ_d\tilde{V}w^*(k+H_p) - \left(\tilde{L}_1 TQ_d + \Lambda Q_d\hat{A}\right) y(k) \qquad (2.188)$$

From (2.188) directly follows:

$$\mathcal{R} = TQ_d + q^{-1}\left(\tilde{L}_2 TQ_d + \tilde{V}_2 TQ_d + \rho\tilde{Z}T + \right.$$
$$\left. + \rho\tilde{Z}_2 TQ_d - \Lambda Q_d q^{-\underline{d}}\hat{\tilde{B}}\right) \qquad (2.189)$$

$$\mathcal{S} = \left(\tilde{L}_1 T + \Lambda\hat{A}\right) Q_d \qquad (2.190)$$

$$\mathcal{T} = TQ_d\tilde{V}R \qquad (2.191)$$

and the degrees of $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ when $n_A > 0$ are given by:

$$n_R = \max\left(n_T + n_{Q_d}, 1 + \max\left(n_B + \overline{d} + n_T + n_{Q_d} - 1,\right.\right.$$
$$\max(n_{Q_n}, n_{Q_d}) + n_T, \beta + n_N - H_c - 1 + n_T + n_{Q_d},$$
$$\left.\left. n_{Q_d} + n_B + \overline{d} + \max(n_T, n_D) - 1\right)\right)$$
$$n_S = n_{Q_d} + \max\left(n_T + \max(n_P - H_s, n_A - 1), n_A + \max(n_T, n_D) - 1\right)$$
$$n_T = n_T + n_{Q_d} + H_p - H_s$$

Note that in the case $T = 1$, $\hat{D} = 1$, $Q_n = 1$ and $Q_d = 1$, the controller polynomials $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ can be simplified into:

$$\mathcal{R} = 1 + q^{-1}\left(\tilde{L}_2 + \tilde{V}_2 + \rho\tilde{Z}_2\right)$$
$$\mathcal{S} = \tilde{L}_1$$
$$\mathcal{T} = \tilde{V}R$$

Further, if $n_A = 0$ (as with the FIR process model), $T = 1$ and $\hat{D} = 1$ then $\mathcal{S} = 0$ (because then $\tilde{L}_1 = \Lambda = 0$). Hence, in this situation feedback no longer exists.

However, if $T \neq 1$ or $\hat{D} \neq 1$, feedback is guaranteed by Λ being different from zero in (2.190). In predictive controllers which are based on FIR process models, feedback is obtained by a special way of generating the reference trajectory (see section 2.4 and [3]) or by using a simple noise model (see section 2.1.3 and [26]).

Figure (2.12) shows the control law (2.186) in a block diagram. From Figure

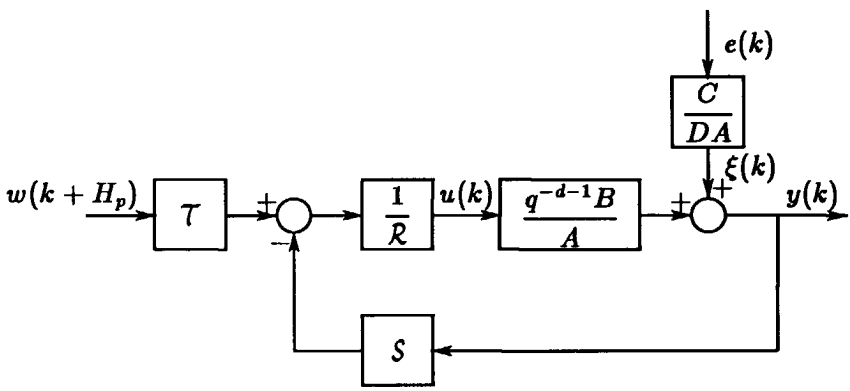

Figure 2.12: The closed-loop system

2.12 the following transfer functions can be calculated:

$$y(k) = \frac{q^{-d-1}BTw(k+H_p) + \frac{C\mathcal{R}}{D}e(k)}{A\mathcal{R} + q^{-d-1}BS} \tag{2.192}$$

$$u(k) = \frac{ATw(k+H_p) + \frac{CS}{D}e(k)}{A\mathcal{R} + q^{-d-1}BS} \tag{2.193}$$

2.3.3 Properties of the Predictive Control Law

In this section, some properties of the predictive control law (2.192) are derived and discussed. First, it is shown for which choices of Q_n, $\hat{D}$ and β, there are no steady-state errors. Then, the link between these polynomials and the controller polynomial $\mathcal{R}$ is made. Finally, the effect of the T polynomial on the closed-loop behavior is discussed.

Steady-state performance

Suppose that the reference trajectory is given by:

$$W(z^{-1}) = \frac{K_w}{(1-z^{-1})^r} = \frac{K_w}{\Delta^r} \tag{2.194}$$

where r denotes the type of the reference trajectory (i.e. $r = 1$, denotes a constant trajectory, $r = 2$ denotes a triangular reference trajectory and so on) and K_w is the magnitude of the reference trajectory. Let the disturbance $\xi(z^{-1})$, acting on the output of the process, be given by:

$$\xi(z^{-1}) = \frac{K_\xi}{\Delta^p}$$

where p denotes the type of the disturbance and K_ξ is the magnitude of the disturbance. Further, assume that the process and model contain n integrators. Now, the process output is given by:

$$Y(z^{-1}) = \frac{B(z^{-1})}{\tilde{A}(z^{-1})\Delta^n} U(z^{-1}) + \xi(z^{-1}) \tag{2.195}$$

where $A(z^{-1}) = \tilde{A}(z^{-1})\Delta^n$. Now introduce the following variables:

δ = the number of factors Δ in $\hat{D}$. Hence, $\hat{D} = \Delta^\delta \tilde{\hat{D}}$.

γ = the number of factors Δ in Q_n. Hence, $Q_n = \Delta^\gamma \tilde{Q}_n$.

Now the following theorem gives sufficient conditions for δ, γ and β such that steady-state errors do not occur.

Theorem 2.9 *If $\delta \geq \max(p,r) - n$, $\gamma \geq \max(p,r) - n$ and $\beta \geq \max(p,r) - n$ then steady-state errors do not occur if the reference trajectory is of type r, the disturbance is of type p and the process and model contain n integrators.*

Proof. Recall the i-step-ahead predictor for the unified process model (2.18) in which B, C and D are replaced by their estimates $\hat{B}$, T and $\hat{D}$, respectively:

$$\hat{y}(k+i) = \frac{\overline{E}_i \hat{B}\hat{D}}{T} u(k+i-1) + \frac{\overline{F}_i}{T} y(k) \tag{2.196}$$

Equation (2.196) can also be written as:

$$\hat{Y}(z^{-1}) = \frac{\overline{E}_i(z^{-1})\hat{B}(z^{-1})\hat{D}(z^{-1})}{T(z^{-1})} z^{-1} U(z^{-1}) + \frac{\overline{F}_i(z^{-1})}{T(z^{-1})} z^{-i} Y(z^{-1}) \tag{2.197}$$

By using Diophantine equation (2.15) with A, C and D replaced by their estimates $\hat{A}$, T and $\hat{D}$, respectively, (2.197) becomes:

$$\hat{Y}(z^{-1}) = \frac{\overline{E}_i(z^{-1})\hat{B}(z^{-1})\hat{D}(z^{-1})}{T(z^{-1})} z^{-1} U(z^{-1}) +$$

$$Y(z^{-1}) - \frac{\hat{A}(z^{-1})\hat{D}(z^{-1})\overline{E}_i(z^{-1})}{T(z^{-1})} Y(z^{-1})$$

By using the final-value theorem (see, for example [19], pp.53), the predicted process output in the steady state becomes:

$$\hat{y}_{ss} = K_1 \lim_{z \to 1}(1 - z^{-1}) \left\{ \Delta^{\delta} U(z^{-1}) \right\} + \lim_{z \to 1}(1 - z^{-1}) Y(z^{-1}) -$$

$$K_2 \lim_{z \to 1}(1 - z^{-1}) \left\{ \Delta^{\delta+n} Y(z^{-1}) \right\} \tag{2.198}$$

in which:

$$K_1 = \frac{\overline{E}_i(1)\hat{B}(1)\tilde{\hat{D}}(1)}{T(1)}$$

$$K_2 = \frac{\tilde{\hat{A}}(1)\tilde{\hat{D}}(1)\overline{E}_i(1)}{T(1)}$$

Substitution of (2.195) in (2.198) yields:

$$\hat{y}_{ss} = \lim_{z \to 1}(1 - z^{-1}) \left\{ \left(K_1 \Delta^{\delta} + \frac{\tilde{K}_{dc}}{\Delta^n} - K_2 \tilde{K}_{dc} \Delta^{\delta} \right) U(z^{-1}) + \right.$$

$$\left. \frac{K_{\xi}}{\Delta^p} - K_{\xi} K_2 \Delta^{\delta+n-p} \right\} \tag{2.199}$$

where:

$$\tilde{K}_{dc} = \frac{B(1)}{\tilde{A}(1)}$$

The criterion function that is minimized (remember, $R = P(1)$) is given by (2.161) and (2.162). The steady-state value of this criterion is given by:

$$J_{ss} = \lim_{z \to 1}(1 - z^{-1}) \left\{ P(1)^2 [\hat{Y}(z^{-1}) - W(z^{-1})]^2 + \rho K_q^2 [\Delta^\gamma U(z^{-1})]^2 \right\} \quad (2.200)$$

where:

$$K_q = \frac{\tilde{Q}_n(1)}{Q_d(1)}$$

Further, in the steady state, (2.162) becomes:

$$N(1) \lim_{z \to 1}(1 - z^{-1}) \Delta^\beta U(z^{-1}) = 0 \quad (2.201)$$

Using (2.194), (2.195) and (2.199), equation (2.200) can be written as:

$$J_{ss} = P(1)^2 \lim_{z \to 1}(1 - z^{-1}) \left\{ \left(K_1 \Delta^\delta + \frac{\tilde{K}_{dc}}{\Delta^n} - \tilde{K}_{dc} K_2 \Delta^\delta \right) U(z^{-1}) + \frac{K_\xi}{\Delta^p} - \right.$$

$$\left. K_\xi K_2 \Delta^{\delta+n-p} - \frac{K_w}{\Delta^r} \right\}^2 + \rho K_q^2 \lim_{z \to 1}(1 - z^{-1}) \left\{ \Delta^\gamma U(z^{-1}) \right\}^2 \quad (2.202)$$

Minimization of (2.202) with respect to $U(z^{-1})$ yields the optimal controller output in the steady state. Now, it can easily be verified that if the conditions

$$\delta + n \geq \max(p, r) \quad (2.203)$$

$$\gamma + n \geq \max(p, r) \quad (2.204)$$

$$\beta + n \geq \max(p, r) \quad (2.205)$$

are satisfied, then the optimal controller output in the steady state is given by:

$$U(z^{-1}) = \frac{K_w \Delta^n}{\tilde{K}_{dc} \Delta^r} - \frac{K_\xi \Delta^n}{\tilde{K}_{dc} \Delta^p} \quad (2.206)$$

Then criterion function (2.202) is equal to zero and (2.201) is satisfied. Substituting (2.206) in (2.195) yields:

$$Y(z^{-1}) = \frac{B(z^{-1})}{\tilde{A}(z^{-1})\Delta^n}\left(\frac{K_w\Delta^n}{\tilde{K}_{dc}\Delta^r} - \frac{K_\xi\Delta^n}{\tilde{K}_{dc}\Delta^p}\right) + \frac{K_\xi}{\Delta^p}$$

Now, the process output in the steady state is given by:

$$y_{ss} = \lim_{z\to 1}(1 - z^{-1})Y(z^{-1}) = \lim_{z\to 1}(1 - z^{-1})\frac{K_w}{\Delta^r} = w_{ss} \qquad (2.207)$$

Equation (2.207) shows clearly that if (2.203), (2.204) and (2.205) are satisfied, steady-state errors do not occur.

□

Remarks:

- In the case $\rho = 0$, condition (2.204) need not be satisfied.
- If $H_c = H_p - \underline{d}$, condition (2.205) need not be satisfied. Then, (2.201) is not in effect and hence β does not have effect.
- Theorem 2.9 gives sufficient conditions for which the steady-state error is zero at the sampling instants. The steady-state error between the samples is zero only if the conditions (2.203), $\cdots$, (2.205) are satisfied and if the process contains at least $\max(p, r) - 1$ integrators (hence, if $n \geq \max(p, r) - 1$) [35].

It is well known (e.g. [40], pp.122) that in order to prevent steady-state errors, the closed loop should at least contain p integrators if there is a disturbance and/or reference signal of type p. The same result can be achieved by satisfying the conditions (2.203), $\cdots$, (2.205). From Figure 2.12 and (2.192) follows that in this case $\mathcal{R}$ contains $p - n$ factors Δ and $\mathcal{S}(1) = \mathcal{T}(1)$. Hence, in general, we can say that including x factors Δ in Q_n and $\hat{D}$ and by choosing $\beta = x$ results in x factors Δ in $\mathcal{R}$ and hence in x integrators in the closed-loop system.

The effect of T and $\hat{D}$ on the closed-loop system

When the process is correctly estimated (hence, $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$), the effect of T and $\hat{D}$ on the closed loop can easily be shown.

Theorem 2.10 *If $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$ then T is a factor of the closed-loop characteristic equation. Further, $\hat{D}$ does affect the closed-loop characteristic equation.*

Proof. Consider the CLCE and substitute $\mathcal{R}$ and S by (2.189) and (2.190):

$$\begin{aligned} A\mathcal{R} + q^{-d-1}BS = \\ A\left(TQ_d + q^{-1}\left(\tilde{L}_2TQ_d + \tilde{V}_2TQ_d + \rho\tilde{Z}T + \rho\tilde{Z}_2TQ_d\right)\right) + \\ q^{-d-1}BQ_d\tilde{L}_1T - q^{-d-1}\hat{\tilde{B}}Q_d\Lambda A + q^{-d-1}BQ_d\Lambda\hat{A} \end{aligned} \tag{2.208}$$

Taking into account $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$, then (2.208) can be rewritten into:

$$\begin{aligned} A\mathcal{R} + q^{-d-1}BS = \\ T\left\{A\left(Q_d + q^{-1}\left(\tilde{L}_2Q_d + \tilde{V}_2Q_d + \rho\tilde{Z} + \rho\tilde{Z}_2Q_d\right)\right) + \right. \\ \left. + q^{-d-1}BQ_d\tilde{L}_1\right\} \end{aligned} \tag{2.209}$$

Equation (2.209) shows that T is a factor of the CLCE.

It was shown in section 2.1.2 that $\hat{D}$ affects $\boldsymbol{K}_c$ only. Further, (2.185) and (2.187) show that $\boldsymbol{K}_c$ affects Λ only and hence $\hat{D}$ affects Λ only. Now, (2.209) shows that Λ is not present in the CLCE and hence $\hat{D}$ does not affect the CLCE under the assumptions mentioned in the theorem.

□

By using Theorem 2.10, the following theorem can be stated.

Theorem 2.11 *If $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$ then T and $\hat{D}$ do not affect the servo behavior of the closed-loop system.*

Proof. In order to prove the theorem, consider the closed-loop transfer function (2.192):

$$y(k) = \frac{q^{-d-1}B\mathcal{T}w(k+H_p) + \dfrac{C\mathcal{R}}{D}e(k)}{A\mathcal{R} + q^{-d-1}B\mathcal{S}} \tag{2.210}$$

The servo behavior of the closed-loop system is described by:

$$y(k) = \frac{q^{-d-1}B\mathcal{T}}{A\mathcal{R} + q^{-d-1}B\mathcal{S}}w(k+H_p) \tag{2.211}$$

The proof is completed by applying Theorem 2.10 and by using the fact that T is a factor of $\mathcal{T}$ while $\hat{D}$ does not affect $\mathcal{T}$ (see (2.191)).

□

However, T and $\hat{D}$ do affect the regulator behavior of the closed-loop system. See, for example, the closed-loop transfer functions for the minimum-variance (2.86) or dead-beat controller (2.111) in which $R = P = 1$. To conclude, T and $\hat{D}$ can be used to tune the regulator behavior without affecting the servo behavior of the closed-loop system if the process is correctly estimated. In practice, the process will never be correctly estimated. Then, the theorems mentioned above can no longer be applied. Consequently, T and $\hat{D}$ affect both the servo and the regulator behavior of the closed-loop system. However, simulations showed (see, section 3.5) that if the 'difference' between model and process is small, then the effect of T and $\hat{D}$ on the servo behavior is small too. Further, in section 3.5, the effect of T and $\hat{D}$ on the robustness and performance of the closed-loop system is discussed.

2.4 The Reference Trajectory

In this section the reference trajectory is discussed. For the sake of convenience, time delay is assumed to be absent and hence, $d = \hat{d} = \underline{d} = \overline{d} = 0$.

Often a reference trajectory is used to define the behavior of the process output in going from one set point to another. In generating a reference trajectory there are two distinct approaches:

- Set point changes are known a priori. Then, a set point change at $t = k + H_p$ is already known at $t = k$ and the controller can respond to it by calculating the proper $u(k)$. Figures 2.13a and 2.13b illustrate this approach when the reference trajectory is equal to the set point. Figure 2.13a shows the situation

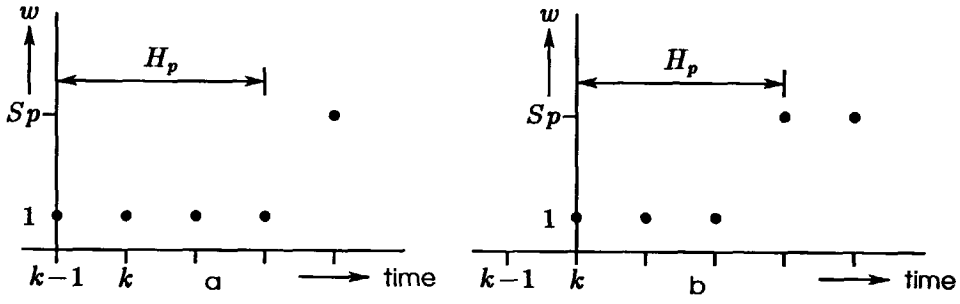

Figure 2.13: Set point changes known a priori.

at $t = k - 1$. Now the set point change at $t = k + H_p$ is beyond the prediction horizon and hence not included in the reference trajectory:

$$\boldsymbol{w} = [1, \cdots, 1]^T \tag{2.212}$$

Figure 2.13b shows the situation at $t = k$. Now, the set point change is detected and $\boldsymbol{w}$ becomes:

$$\boldsymbol{w} = [1, \cdots, 1, Sp]^T$$

Obviously, by generating $\boldsymbol{w}$ as described above, the relation $w(k + i - 1) = q^{-1}w(k + i)$ is valid for all i. Further, in this case, $\mathcal{T}$ in (2.186) is given by $\tilde{V}TQ_dR$ with $w(k + H_p)$ equal to the set point. However, the polynomial

$\tilde{V}$ can have roots outside the unit circle which can result in a non-minimum phase closed-loop system. Although the controller is optimal with respect to the criterion function that is minimized, the behavior of the closed loop may not be desirable. An example of this phenomenon is given in section 3.6.4.

- Set point changes are **not** known a priori. If, for example, an operator changes the set point at $t = k$, this set point can be fed to the controller by making $\boldsymbol{w}$ equal to:

$$\boldsymbol{w} = [Sp, \cdots, Sp]^T$$

Note that at $t = k - 1$, the reference trajectory $\boldsymbol{w}$ is still given by (2.212). The Figures 2.14a and 2.14b illustrate this approach. Again Figure 2.14a shows

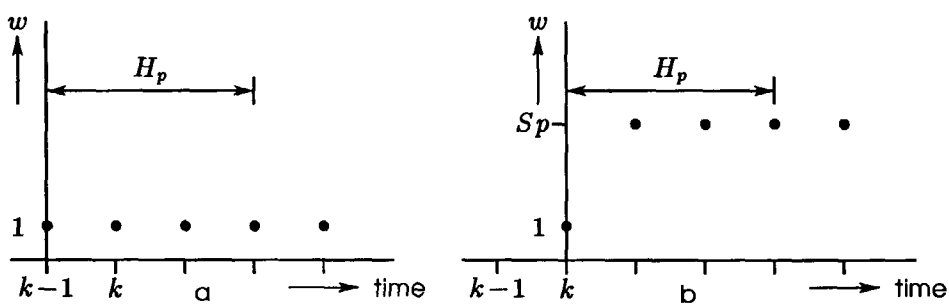

Figure 2.14: Set point changes not known a priori.

the reference trajectory at $t = k - 1$ while Figure 2.14b shows the reference trajectory at $t = k$.

Remark: In this situation the relation $w(k + i - 1) = q^{-1}w(k + i)$ is violated and hence (2.191) is no longer valid. Because, now $w(k + i) = Sp$ for $i = 1, \ldots, H_p$, T is given by:

$$T = \tilde{V}(1)TQ_dR \tag{2.213}$$

and hence problems with roots of $\tilde{V}$ outside the unit circle no longer occur.

This approach is used in Chapter 3 unless otherwise noted.

Apart from the fact of whether or not changes in the set point are known a priori, the way the trajectory is generated plays an important role. The trajectory may be an

arbitrary sequence of points describing a path that, for example, a robot must track, or a paraboloid which is, for example, well suited to a number of servo systems. However, for many processes a simple first-order trajectory can be used. Such a trajectory is generated by:

$$w(k+i) = (1-\alpha)Sp + \alpha w(k+i-1) \qquad 1 \leq i \leq H_p \wedge 0 \leq \alpha < 1 \quad (2.214)$$

One question has not yet been answered: how is the reference trajectory initiated (= how to choose $w(k)$ in (2.214))? Two possibilities are:

1. Use $w(k) = y(k)$ as in many predictive controllers (e.g. [3, 26, 5]).
2. Use $w(k)$ as the initial value of the reference trajectory.

Both possibilities are shown in Figure 2.15 where Figure 2.15a shows a first-order trajectory initiated at $y(k)$ and Figure 2.15b shows a first-order trajectory initiated at $w(k)$. By using the first alternative, an extra feed-back loop is activated: Rewriting

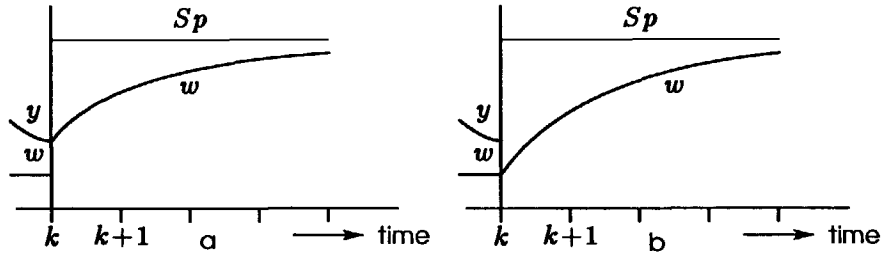

Figure 2.15: Two ways of generating a first-order reference trajectory.

(2.214) yields:

$$w(k+i) = (1-\alpha^i)Sp + \alpha^i y(k) \quad (2.215)$$

In a vector notation (2.215) becomes:

$$\boldsymbol{w} = \boldsymbol{\pi} Sp + \boldsymbol{\nu} y(k) \quad (2.216)$$

in which π and ν are vectors of dimension $H_p \times 1$.

$$\pi = [(1-\alpha),(1-\alpha^2),\cdots,(1-\alpha^{H_p})]^T \tag{2.217}$$

$$\nu = [\alpha,\alpha^2,\cdots,\alpha^{H_p}]^T \tag{2.218}$$

Substituting (2.216) in (2.185) and using $P = R = 1$ yields (now $w^* = w$):

$$u(k) = v^T(\pi Sp - k) + v^T\nu y(k) - l^T s - v_2^T\check{u} - \rho z^T\tilde{u} - \rho z_2^T\check{u} \tag{2.219}$$

From (2.219) follows directly that S in (2.186) becomes:

$$S = S - v^T\nu$$

Because $v^T\nu$ is a scalar, the first coefficient of S is modified only. Note that if $n_A = 0$, $T = 1$ and $\hat{D} = 1$ (like in the MAC controller [3]) then feedback is realized only by the way the reference trajectory is generated:

$$S = -v^T\nu$$

Choosing a reference trajectory is a way of defining the desired closed-loop response. However, the same goal can be achieved by defining P and R in the criterion function that is minimized (see section 2.2.2). Hence, there is a strong relationship between generating the reference trajectory as in (2.214) and by choosing $P = 1 - \alpha q^{-1}$ and $R = 1 - \alpha$. If $H_p = H_s = 1$, the two ways of defining the closed-loop behavior are identical. Then (2.214) becomes:

$$w(k+1) = (1-\alpha)Sp + \alpha y(k) \tag{2.220}$$

Substituting (2.220) in criterion function (2.161) with $P = R = 1$ yields:

$$J = \left[(1-\alpha q^{-1})\hat{y}(k+1) - (1-\alpha)Sp\right]^2 + \rho\left[\frac{Q_n}{Q_d}u(k)\right]^2 \tag{2.221}$$

From (2.221) follows directly that the same result could be achieved by using $w(k+1) = Sp$ and choosing P and R in (2.161) as:

$$P = 1 - \alpha q^{-1}$$

$$R = 1 - \alpha$$

If $H_p > 1$, the criterion function (2.161) with $P = 1 - \alpha q^{-1}$ and $R = 1 - \alpha$ can be rewritten into:

$$J = \sum_{i=H_s}^{H_p} [\hat{y}(k+i) - [(1-\alpha)Sp + \alpha\hat{y}(k+i-1)]]^2 +$$

$$\rho \sum_{i=1}^{H_p} \left[\frac{Q_n}{Q_d} u(k+i-1)\right]^2$$

Hence, minimization of (2.161) with $P = 1 - \alpha q^{-1}$ and $R = 1 - \alpha$ is equivalent to minimization (2.161) with $P = R = 1$ and with the reference trajectory generated by:

$$w(k+i) = (1-\alpha)Sp + \alpha\hat{y}(k+i-1) \tag{2.222}$$

If $\hat{y}(k+i-1) = w(k+i-1)$ then (2.214) and (2.222) are identical. In general this will not be the case. However, if the predicted process output tracks the reference signal closely, then both methods yield approximately the same results. If, for example, $H_c = H_p$, $H_s = 1$ and $\rho = 0$, then both ways of generating the reference trajectory are identical because then the controller minimizing (2.161) does not depend on H_p (see Theorem 2.2).

2.5 Overview of Design Parameters

In this section, an overview of all design parameters of the UPC controller is given. In Table 2.6 the parameters are shown together with their range and type (polynomial, integer, real). In Chapter 3, each group of design parameters is discussed separately.

2.6 Conclusions

In this chapter, the four basic ingredients of predictive controllers were discussed. After defining a unified process model which can be used to model ARX, ARIX, ARMAX, ARIMAX, FIR and FSR process models, its minimum-variance i-step-ahead predictor was derived.

It was shown that this predictor can be rewritten in a part that does not depend

	description	type	range
H_p	prediction horizon	integer	$\geq H_s$
H_s	minimum cost horizon	integer	$\underline{d}+1 \leq H_s \leq H_p$
H_c	control horizon	integer	$1 \leq H_c \leq H_p - \underline{d}$
β	type of control horizon	integer	≥ 1
N	with β and H_c: control horizon	polynomial	-
P	with R: filter on tracking error	polynomial	-
R	with P: filter on tracking error	real	$= P(1)$
ρ	weighting factor	real	≥ 0
Q_n	with Q_d: filter on controller output	polynomial	-
Q_d	with Q_n: filter on controller output	polynomial	-
$\underline{d}$	lower bound time delay	integer	$0 \leq \underline{d} \leq d$
$\overline{d}$	upper bound time delay	integer	$d \leq \overline{d} < H_p$
T	with $\hat{A}$ and $\hat{D}$: noise model	polynomial	-
$\hat{D}$	with $\hat{A}$ and $\hat{T}$: noise model	polynomial	-

Table 2.6: UPC design parameters.

on the noise model and in a part that does. The latter consists of the filtered difference between measured process output and estimated process output. The filter depends, among other things, on the noise model. Because the noise model is, in practice, difficult to estimate, a fixed model is used making T and $\hat{D}$ design polynomials.

Writing the unified MV i-step-ahead predictor in such a way showed that deviations between process and model, caused by model mismatch or disturbances, are filtered before they are added to the predictions. Because the controller output directly depends on these predictions, the effect of model mismatch and disturbances on the controller output can be suppressed by choosing an appropriate noise model. Further, this predictor was used to show that if the A and B polynomials of the process are correctly estimated, then the servo behavior of the closed loop system does not depend on the noise model. Moreover, it was shown that in this case the roots of T appear as poles of the closed-loop system.

A unified multi-step criterion function unifies the criteria utilized in well-known predictive controllers such as GPC, MAC, IMAC and GMV. The definition of the control horizon used in DMC and GPC is extended. This made it possible to derive sufficient conditions to prevent steady-state errors for a given type of reference trajectory or disturbance. Further, it was shown that by selecting the criterion function parameters of the unified criterion function, a number of well-known classical controllers, such

as minimum-variance and dead-beat controllers, can be obtained.

Also, it was shown that the unified criterion function can be minimized analytically yielding the unified predictive control law. This control law can also be written in the standard form $\mathcal{R}u(k) = -\mathcal{S}y(k) + \mathcal{T}w(k + H_p)$. Because UPC is a unification of other predictive controllers, this holds true for those too.

It was shown that generating the reference trajectory by filtering the set point by using a first-order filter initiated at $y(k)$ is closely related to choosing the P and R polynomials in the criterion function according to this filter and using a step as the reference trajectory.

Chapter 3

Analysis of Design Parameters

In the previous chapter it was shown that the UPC controller is capable of controlling a wide variety of processes among which processes with time delay, unstable processes and processes with an unstable inverse if a model of the process to be controlled is available and if its design parameters are properly selected. Theorems are provided of how to select the parameters of the controller if a model of the process is available. However, in many theorems the assumption was made that the process is correctly estimated. In general, this will not be the case and hence those theorems can no longer be applied. In this respect the robustness of the closed-loop system with respect to modeling errors becomes an important issue.

The purpose of this chapter is to gain an insight into the influence of the UPC parameters on the performance and robustness of the closed-loop system. Further,

some rules of thumb on how to select these parameters, given a model of the process and the desired performance and robustness, are formulated. At the end of this chapter, a review of several well-known predictive controllers is given, based on the results obtained in Chapter 2 and in this chapter.

In order to define performance and robustness, recall block diagram 2.12. The transfer function from $w(k+H_p)$ and $\xi(k)$ to $y(k)$ is:

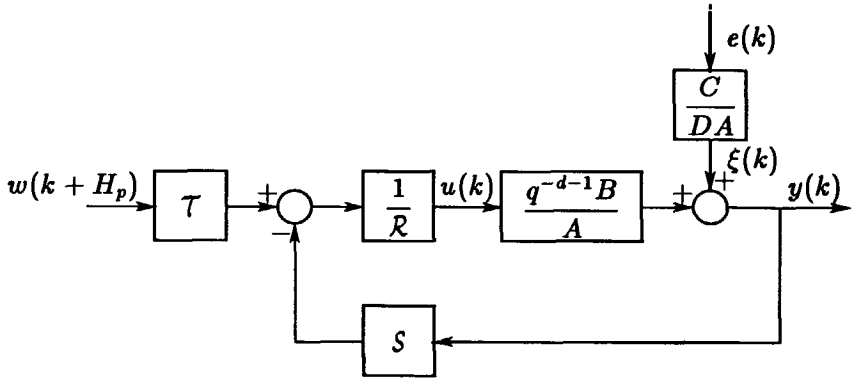

Figure 3.1: Block diagram of the closed-loop system.

$$\begin{aligned} y(k) &= \frac{q^{-d-1}BT}{AR+q^{-d-1}BS}w(k+H_p) + \frac{AR}{AR+q^{-d-1}BS}\xi(k) \\ &= \underbrace{\frac{\mathcal{F}H_L}{1+H_L}w(k+H_p)}_{\text{servo}} + \underbrace{\frac{1}{1+H_L}\xi(k)}_{\text{regulator}} \end{aligned} \tag{3.1}$$

in which:

$$\begin{aligned} H_L &= \frac{q^{-d-1}BS}{AR} \\ \mathcal{F} &= \frac{T}{S} \end{aligned}$$

H_L is called the loop transfer function. Note that $\mathcal{F}H_L$ denotes the open-loop transfer function.

3.1 Performance

The servo behavior of the closed-loop system is determined by the first part of (3.1) while the regulator behavior is determined by the second part of (3.1). Now, two types of performance can be distinguished: servo and regulator performance.

3.1.1 Servo performance

The servo behavior of the closed-loop system is measured by using a unit step as the reference trajectory (hence, $w(k+1) = w(k+2) = \cdots = w(k+H_p) = 1$) and with $\xi(k) = 0$. The resulting response is used to define the servo performance of the closed-loop system. Classical criteria like rise time (t_r), peak time (t_p), settling time (t_s), overshoot (O) and steady-state error (ϵ_{ss}) are used for this purpose.

The rise time is defined as the time required for the system to reach 90% of the input amplitude (= 1), the peak time is defined as the time in which the response reaches its maximum value and the settling time is defined as the time required for the system to settle within 2% of its steady-state value:

$$\begin{aligned} t_r &= \min_t\{t/y(t) \geq 0.9\} && [s] \\ t_p &= \arg\max_t\{y(t)\} && [s] \\ t_s &= \min_t\{t/|y(t) - y_{ss}| < 0.02|y_{ss}|\} && [s] \end{aligned}$$

The overshoot and steady-state error are defined as:

$$O = (y(t_p) - y_{ss}) \times 100 \quad [\%] \tag{3.2}$$

$$\epsilon_{ss} = |(1 - y_{ss})| \times 100 \quad [\%] \tag{3.3}$$

where y_{ss} is the steady-state value of $y(k)$. The above-mentioned criteria are illustrated in Figure 3.2.

Remarks:

- If the closed-loop system is unstable, y_{ss} is not defined and hence ϵ_{ss}, O, t_p and t_s are not defined. Further, t_r is not defined if $y_{ss} < 0.9$ and t_p, O are not defined if $y(t) \leq y_{ss}\ \forall t \geq 0$.

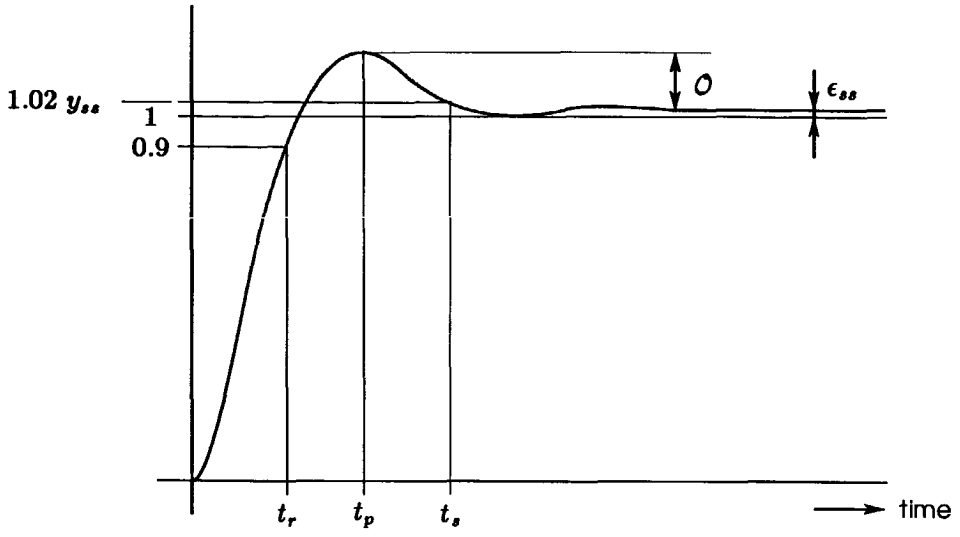

Figure 3.2: Servo performance criteria.

- Every continuous process that is used in the simulations described in this chapter is simulated by using its discrete time equivalent. As a result, information about the process output is available only at $t = kT_s$ where k is an integer ≥ 0. Now, t_{rk}, t_{pk} and t_{sk} are given by:

$$\begin{aligned} t_{rk} &= T_s \min_k\{k/y(kT_s) \geq 0.9\} && [s] \\ t_{pk} &= T_s \arg\max_k\{y(kT_s)\} && [s] \\ t_{sk} &= T_s \min_k\{k/|y(kT_s) - y_{ss}| < 0.02|y_{ss}|\} && [s] \end{aligned}$$

where the subscript k denotes that the criterion is based on discrete time information. The overshoot and the steady-state error are still given by (3.2) and (3.3), respectively. However, the overshoot is now denoted as O_k. The accuracy of the servo performance criteria t_{rk}, t_{pk}, t_{sk} and O_k depends on the sampling period that is used. The relations between t_{rk} and t_{sk} and between t_r and t_s are:

$$\begin{aligned} t_{rk} - T_s < \; t_r \; \leq t_{rk} \\ t_{sk} - T_s < \; t_s \; \leq t_{sk} \end{aligned}$$

The relations between t_{pk} and O_k and between t_p and O are not known. How-

ever, the smaller the sampling period, the more accurate the servo performance criteria based on discrete time information will be.

In this chapter, the accuracy of t_r, t_p, t_s and O is not of paramount importance because of the fact that they are used only to show trends. The sampling periods chosen in the examples are such that the accuracy of the servo performance criteria is sufficient. Subsequently, the index k is dropped yielding: t_r, t_p, t_s and O.

3.1.2 Regulator performance

The regulator behavior is measured by using $\xi(k) = \frac{C}{DA}e(k)$ with $\sigma_e^2 \neq 0$ and the reference trajectory equal to zero. The following two criteria are used to judge the regulator performance of the closed-loop system:

$$\sigma_{yN}^2 = \frac{1}{N+1}\sum_{k=0}^{N} y(k)^2$$

$$\sigma_{uN}^2 = \frac{1}{N+1}\sum_{k=0}^{N} u(k)^2$$

in which $N = 1000$. Then, the above mentioned criteria can be regarded as approximations of the process output variance and the controller output variance, respectively.

Another way of inspecting the regulator behavior is by simulating a load change:

$$\xi(k) = \frac{A(1)}{A}l(k)$$

in which $l(k)$ is a unit step. In all simulations, the controller parameters are chosen such that steady-state errors, due to the load change, do not occur. Then, the maximum deviation of the response from the reference trajectory (which was taken equal to zero) and the settling time, now defined as the time required for the system to settle within 2% of its maximum deviation, are used to judge the regulator performance with respect to a load change:

$$Max = \max_k\{y(kT_s)\}$$

$$t_s = T_s \min_k\{k/|y(kT_s)| < 0.02|Max|\}$$

Remark: all simulations in this chapter are carried out without constraints being present on the controller output.

3.2 Robustness

It is highly unrealistic to assume that the process model, which is used to derive the controller, is equal to the real process. Therefore, it is important to investigate the influence of modeling errors on the properties of the closed-loop system. A closed-loop system is called robust if it maintains certain properties in spite of process variations [17]. Two major properties of a closed-loop system are stability and performance. Now, two forms of robustness can be distinguished: stability robustness and performance robustness. Stability robustness means that the closed-loop system remains stable in spite of process changes. Performance robustness ensures that the performance of the closed-loop system, as defined in the previous sections, remains within specified bounds if the true process is different from its model [17].

3.2.1 Stability robustness

In [41] and [17] sufficient conditions are derived that, if satisfied, ensure stability of the closed-loop system in spite of process changes. If the loop transfer functions of the nominal system (= the closed-loop system based on the model) and the true system are given by $\hat{H}_L$ and H_L, respectively, then stability of the closed-loop system is guaranteed if the nominal system and the process are stable and if the following condition is satisfied [17]:

$$\frac{|H_L - \hat{H}_L|}{|\hat{H}_L|} \frac{|\hat{H}_L|}{|1 + \hat{H}_L|} < 1 \qquad \forall \omega \geq 0 \tag{3.4}$$

where the first part of (3.4) represents the relative change of the open-loop frequency response while the second part of (3.4) is the frequency response of the complementary sensitivity function $\frac{\hat{H}_L}{1 + \hat{H}_L}$. By making the second part of (3.4) small for a certain frequency, the first part may be large without violating the stability condition (3.4). Hence, for those frequencies where $|\hat{H}_L|$ is small, large deviations between process and model are possible without making the closed-loop system instable: the

stability robustness is large.

As a measure for stability robustness one can, for example, use the peak value of $\frac{|\hat{H}_L|}{|1+\hat{H}_L|}$. The smaller this value, the larger the stability robustness of the closed-loop system. However, it is hard to give a physical interpretation to this value. Therefore, in this thesis, the classical stability robustness criteria gain and phase margin are used (see, for example, [40], pp.272). Further, the phase margin is translated into a time delay margin. The gain and time delay margin can be calculated from the Nyquist figure of $\hat{H}_L$. Then, the closed-loop system is stable if the process DC gain satisfies the following condition:

$$\underline{gm} < \frac{K_{dc}}{\hat{K}_{dc}} < \overline{gm}$$

where $\underline{gm}$ and $\overline{gm}$ are called the lower and upper gain margin and $\hat{K}_{dc}$ is the DC gain of the model. Usually, the lower gain margin is equal to zero. However, as will be shown in the simulations, $\underline{gm}$ can be different from zero for processes with time delay. Further, if $\underline{gm} > 1$ or $\overline{gm} < 1$, the closed-loop system is unstable if $K_{dc} = \hat{K}_{dc}$. Hence, a general measure of the stability robustness of the closed-loop system with respect to changes in the process DC gain is:

$$gm = \min(\overline{gm}, 1/\underline{gm})$$

In the thesis the variable gm is called the gain margin (in contrast to the upper and lower gain margin).

Further, stability margins with respect to changes in the time delay of the process can be defined. If the closed-loop system is stable for $T_d = \hat{T}_d$, then the closed-loop system remains stable if:

$$\underline{dm} < T_d - \hat{T}_d < \overline{dm}$$

in which $\underline{dm}$ (< 0), $\overline{dm}$ (> 0), T_d and $\hat{T}_d$ are the lower and upper delay margins (in seconds) and the time delays of the process and model (in seconds), respectively. A general measure of the stability robustness of the closed-loop system with respect to changes in the time delay of the process is:

$$dm = \min(\overline{dm}, -\underline{dm}) \quad [s]$$

In the thesis the variable $\underline{dm}$ is called the time delay margin. If the closed-loop system is unstable, dm, $\overline{dm}$ and $\underline{dm}$ are not defined.

Further, root loci are used to get an insight into the stability of the closed-loop system as a function of one of the design parameters of the UPC controller. In general, these root loci cannot be described by:

$$-\frac{1}{K} = H \tag{3.5}$$

in which H is a transfer function and K is the parameter of the root locus. Only in the case $H_c = 1$ and with ρ as root-locus parameter can the above-mentioned form be used because in this situation the matrix R_v in (2.185) is a scalar. Hence, only in this case can the root loci be drawn by using the classical design rules [40], pp.165-214.

3.2.2 Performance robustness

In the previous chapter, criteria were defined to measure the stability robustness of the closed-loop system. But what about the performance robustness? According to Kwakernaak [17], performance robustness can be defined as the sensitivity of the performance of the closed-loop system to process changes. Hence, the less sensitive the performance of the closed-loop system is with respect to process changes, the larger is the performance robustness.

Performance robustness can be analyzed by using the transfer function (3.1). It shows that the regulator performance can be made high if $|H_L|$ is made large. The servo performance is also affected by $|H_L|$ but, in contrast to the regulator behavior, can be made as large as desired by adjusting the feed-forward transfer function $\mathcal{F}$. The relative change Δ_s of the transfer function describing the servo behavior with respect to process changes is given by [17]:

$$\Delta_s = \frac{\dfrac{\mathcal{F}H_L}{1+H_L} - \dfrac{\mathcal{F}\hat{H}_L}{1+\hat{H}_L}}{\dfrac{\mathcal{F}H_L}{1+H_L}} = \frac{H_L - \hat{H}_L}{H_L}\frac{1}{1+\hat{H}_L} = \Delta_p \frac{1}{1+\hat{H}_L}$$

where Δ_p is the relative change of the open-loop transfer function. The relative change Δ_r of the transfer function describing the regulator behavior is given by:

$$\Delta_r = \frac{\frac{1}{1+H_L} - \frac{1}{1+\hat{H}_L}}{\frac{1}{1+H_L}} = \frac{\hat{H}_L - H_L}{\hat{H}_L} \frac{\hat{H}_L}{1+\hat{H}_L} = \Delta_p^* \frac{\hat{H}_L}{1+\hat{H}_L}$$

where Δ_p^* describes the relative change of the open-loop transfer function similar to Δ_p. Then, the sensitivity of the transfer function describing the servo behavior with respect to changes in the open loop transfer function becomes:

$$\frac{\Delta_s}{\Delta_p} = \frac{1}{1+\hat{H}_L} \tag{3.6}$$

The function on the right-hand side of (3.6) is called the sensitivity function [17].

The sensitivity of the transfer function describing the regulator behavior with respect to changes in the open-loop transfer function becomes:

$$\frac{\Delta_r}{\Delta_p^*} = \frac{\hat{H}_L}{1+\hat{H}_L} \tag{3.7}$$

The function on the right-hand side of (3.7) is called the complementary sensitivity function [17].

Now, (3.6) shows that the servo-performance robustness can be made large by making $|\hat{H}_L|$ large while in order to obtain a large regulator-performance robustness $|\hat{H}_L|$ must be made small.

In this chapter, the performance robustness is not explicitly examined in contrast to the stability robustness. However, there is a close relationship between both forms of robustness because the sum of the sensitivity and the complementary sensitivity function is equal to 1 $\forall\omega$. Hence, a large stability robustness will result in a large regulator-performance robustness, a small servo-performance robustness and a small regulator performance. Vice versa, a small stability robustness will result in a small regulator-performance robustness, a large servo-performance robustness and a large regulator performance. The servo performance can be made as large as desired by adjusting the feed-forward transfer function $\mathcal{F}$.

The above-mentioned results are summarized in Table 3.1 where + denotes large, − denotes small and 'x' denotes that, independent of $|\hat{H}_L|$, the servo performance can be made arbitrarily high by adjusting the feed-forward transfer function $\mathcal{F}$. In the thesis the term robustness will be used to denote stability robustness unless otherwise noted. Further, whether or not large is sufficiently large and small is sufficiently small, depends on the specifications the closed-loop system must satisfy.

	$\lvert\hat{H}_L\rvert \gg 1$	$\lvert\hat{H}_L\rvert \ll 1$
stability robustness	−	+
regulator-perf. robustness	−	+
servo-perf. robustness	+	−
regulator performance	+	−
servo performance	x	x

Table 3.1: Robustness and performance as a function of $|\hat{H}_L|$.

3.3 Controller Output Weighting

As the criterion (2.146) shows, the parameters ρ, Q_n and Q_d realize controller output weighting. First, the influence of the weighting factor ρ on the stability of the closed-loop system is examined. For this purpose two choices for Q_n are considered: $Q_n = 1$ and $Q_n = 1 - q^{-1}$. Q_d is assumed to be equal to 1. The reason for choosing these values for Q_n and Q_d is that they are often used in controllers known in the literature (see Table 2.3). The following table shows the settings used in the first example.

Example 3.1 Influence of ρ and H_p on the stability of the closed-loop system.

Settings: $H_s = 1,\ H_c = 1,\ \beta = 1,\ N = 1,\ P = 1,\ R = 1,\ T = 1,\ \hat{D} = 1,$ $\underline{d} = 0,\ \overline{d} = 0,\ Q_n = 1 - q^{-1},\ Q_d = 1$

Process:
$$H(s) = \frac{5}{(4s+1)(5s+1)} \quad \overset{T_s = 0.5s}{\Longrightarrow}$$
$$H(z^{-1}) = \frac{0.029z^{-1}(1 + 0.928z^{-1})}{(1 - 0.882z^{-1})(1 - 0.905z^{-1})} \tag{3.8}$$

Model: identical to process

Parameters: ρ: $0 \leq \rho < \infty$, H_p: 1, 2, 10

Note that the process that is used in this example is a simple second-order process with real poles. Figures 3.3a, 3.3b and 3.3c show root loci as a function of ρ with H_p equal to 1, 2 and 10, respectively. In these root loci, and in all other root loci shown in this thesis, the symbols 'x' indicate the closed-loop poles for $K = K_{min}$ and the symbols 'o' indicate the closed-loop poles for $K = K_{max}$, if the root-locus parameter K is varied between K_{min} and K_{max}. Further, the dotted curve indicates pole locations for which the relative damping is 0.7.

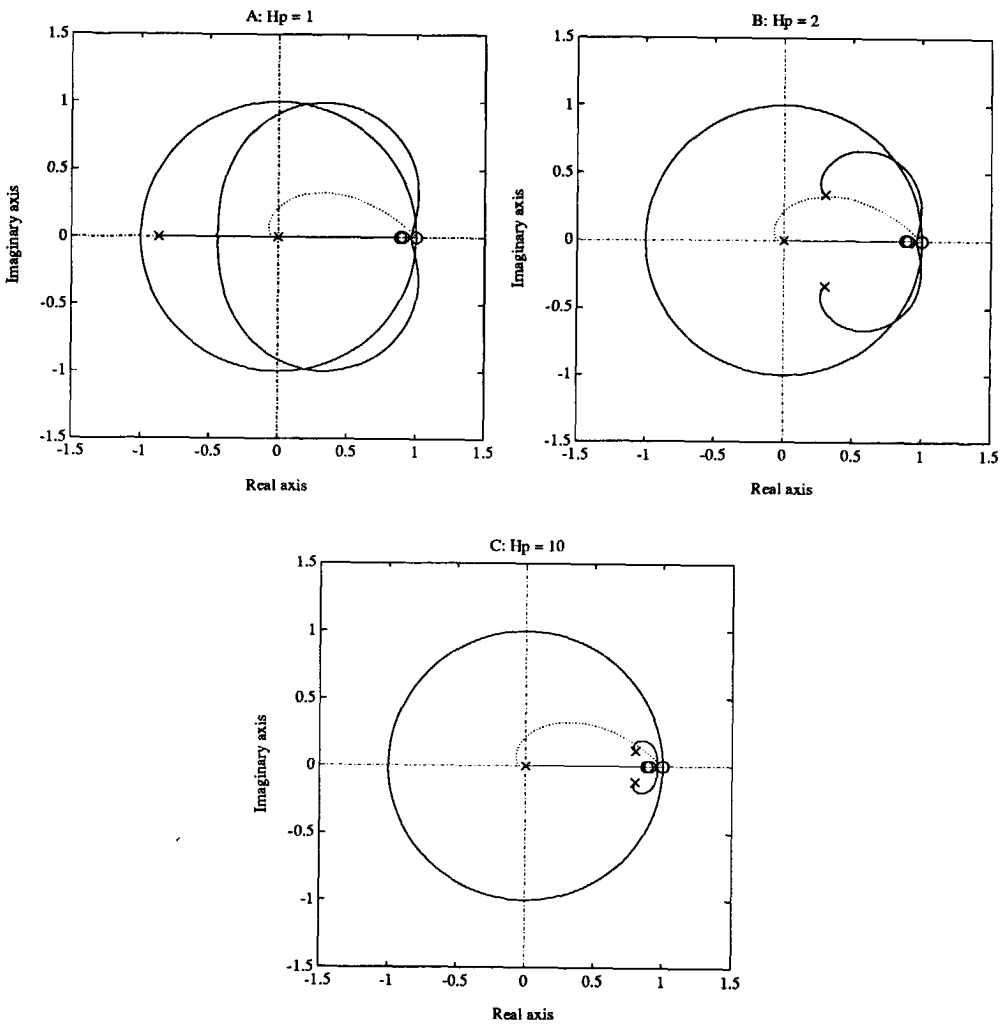

Figure 3.3: Root loci as a function of ρ where $Q_n = \Delta$ for $H_p = 1, 2$ and 10.

Figure 3.3a shows that the closed-loop system becomes unstable for a certain range of ρ: $7.7 \cdot 10^{-4} < \rho < 0.36$. This is a rather peculiar result because the controller is still an optimal controller **with respect to the criterion that is minimized!** This is exactly what the problem is: The criterion is minimized over a finite number of samples (in this case one sample only is taken into account) and weights tracking error and controller increments. Hence, low frequency oscillations result in small values for the controller increments and hence the corresponding part of the criterion remains small. If the criterion is minimized over a larger horizon the situation is improved (see Figure 3.3b). Now, the closed-loop system is unstable for $0.04 < \rho < 1.6$. Further increasing the prediction horizon keeps improving the situation. Figure 3.3c shows a root-locus plot with $H_p = 10$ which is a more realistic choice for the prediction horizon (see section 3.4). In this case the closed-loop system remains stable for all ρ. However, in a certain range of ρ the closed-loop system is badly damped. This is clearly shown in Figure 3.4 which shows the overshoot as a function of ρ for $H_p = 10$. In a certain range of ρ the overshoot is quite large ($> 20\%$) which is caused by the closed-loop poles being close to the unit circle. Hence, even when using a simple second-order process, using ρ together

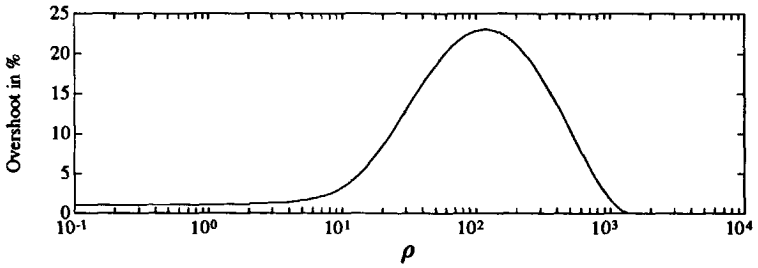

Figure 3.4: Overshoot in % as a function of ρ for $H_p = 10$.

with $Q_n = \Delta$ and $Q_d = 1$ can result in an unstable closed-loop system or at least badly damped closed-loop poles. The reason that $Q_n = \Delta$ is used in many predictive controllers is that in this case the weighting factor does not influence the steady-state behavior if the reference trajectory and the disturbances are constant (see section 2.3.3). However, as is shown, the stability of the system can be in danger. In the following example the stability of the closed-loop system when $Q_n = 1$ and $Q_d = 1$ is examined.

Example 3.2	Influence of ρ and H_p on the stability of the closed-loop system.
Settings:	same as in Example 3.1 except $Q_n = 1$
Process:	same as in Example 3.1
Model:	same as in Example 3.1
Parameters:	ρ: $0 \leq \rho < \infty$, H_p: 1, 2, 10

Now, the weighting factor ρ does influence the steady-state error because condition (2.204) is not satisfied. This effect is clearly demonstrated by Figure 3.5 which shows the steady-state error as a function of ρ for $H_p = 10$. Hence, the steady-state

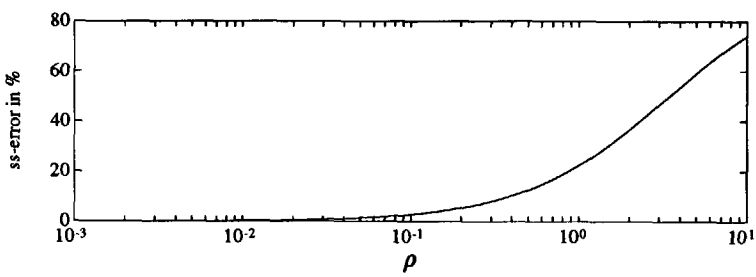

Figure 3.5: Steady-state error in % as a function of ρ for $H_p = 10$.

performance is decreased by using $Q_n = 1$. But what about the dynamic behavior of the closed-loop system? In order to compare the dynamic behavior as a function of ρ using $Q_n = 1$ with that of using $Q_n = \Delta$, root loci are calculated with the same parameters as the root loci in Figure 3.3. Figures 3.6a, 3.6b and 3.6c show these root loci. They clearly show that in this case the closed-loop poles remain stable for all ρ. Moreover, they are better damped. In the case $H_p = 10$, the overshoot is smaller than 1% $\forall \rho \geq 0$.

When weighting the controller output directly the closed-loop system remains stable. This can be explained by the fact that now both parts of the criterion (2.146) go to infinity if the closed-loop system is unstable. This is certainly not the optimal solution. Because making $u(k+i)$ smaller, for example zero, will result in a much better (smaller) criterion. Moreover, the following theorem concerning the stability of the closed-loop system as a function of ρ can be stated.

Theorem 3.1 *If the correctly estimated stable second-order process has a positive DC gain, no time delay and a stable inverse and if $Q_n = 1$, $Q_d = 1$, $H_p = H_s = H_c = 1$ and the roots of T are inside the unit circle, then the closed-loop system is stable for all $\rho \geq 0$.*

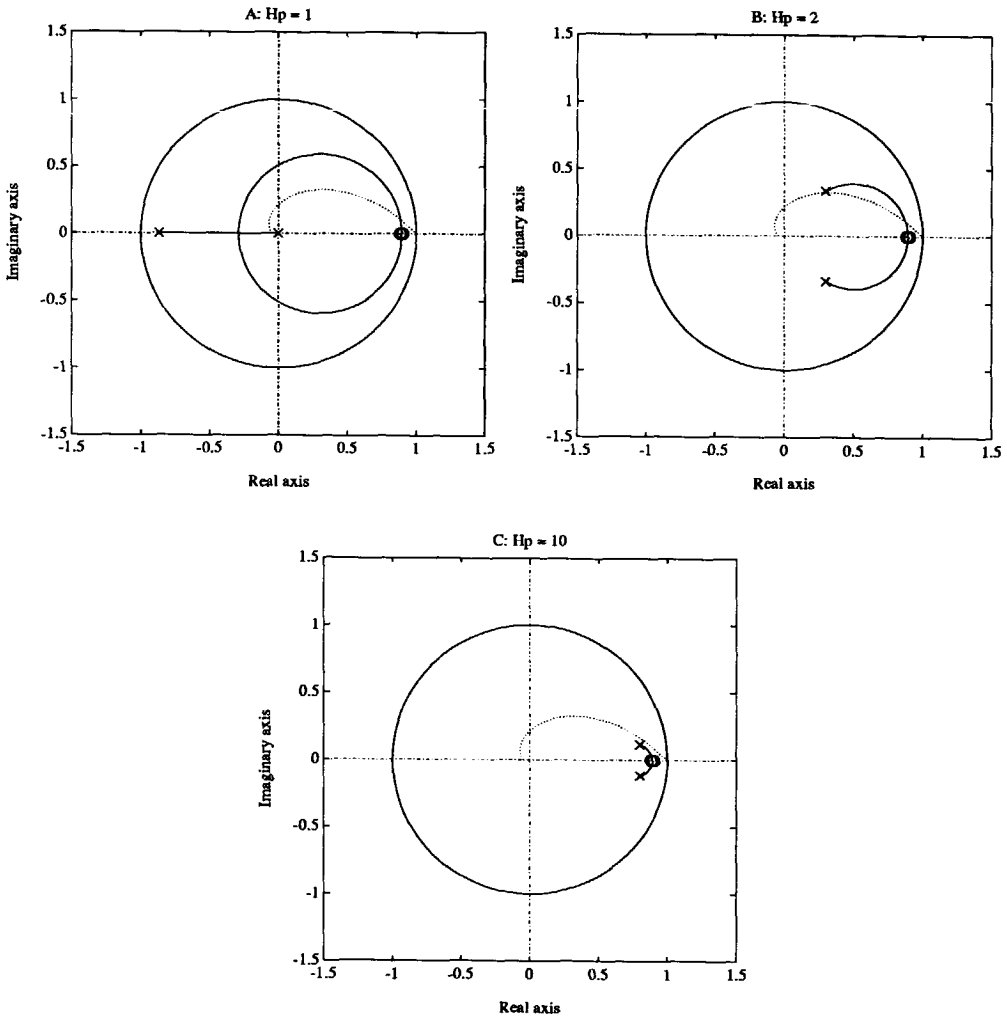

Figure 3.6: Root loci as a function of ρ where $Q_n = 1$ for $H_p = 1, 2$ and 10.

Proof. The case in which $Q_n = 1$, $Q_d = 1$, $H_p = H_s = H_c = 1$ was already discussed in section 2.2.1. It was shown that the CLCE is given by (2.95) which, if the process is correctly estimated, becomes:

$$A\mathcal{R} + q^{-1}BS = \frac{1}{b_0 + \rho}[T(\rho A + b_0 B)]$$

Now, using the assumption that the roots of T are inside the unit circle, it remains to prove that the roots of C are inside the unit circle for all $\rho \geq 0$ where C is given by:

$$C = \rho A + b_0 B \tag{3.9}$$

If the process is of the second order, (3.9) becomes:

$$C = (b_0^2 + \rho) + (b_0 b_1 + \rho a_1)q^{-1} + \rho a_2 q^{-2} \tag{3.10}$$

According to the stability criterion of Jury [42], the following conditions must be satisfied $\forall \rho \geq 0$ in order to guarantee that the closed-loop system is stable $\forall \rho \geq 0$.

1. $b_0^2 + \rho > 0$
2. $b_0(b_0 + b_1) + \rho(1 + a_1 + a_2) > 0$
3. $b_0(b_0 - b_1) + \rho(1 - a_1 + a_2) > 0$
4. $|\rho a_2| < |b_0^2 + \rho|$

Taking into account the assumptions that the roots of A and B are inside the unit circle and that the DC gain is positive, the following inequalities are valid:

$$1 + a_1 + a_2 > 0 \tag{3.11}$$
$$1 - a_1 + a_2 > 0 \tag{3.12}$$
$$|a_2| < 1 \tag{3.13}$$
$$b_0 > 0 \tag{3.14}$$
$$b_0 + b_1 > 0 \tag{3.15}$$
$$b_0 - b_1 > 0 \tag{3.16}$$

Taking these inequalities into account, it is easy to show that the conditions are satisfied for all $\rho \geq 0$:

Ad.1 This condition is satisfied $\forall \rho \geq 0$.

Ad.2 Using (3.11), (3.14), (3.15) and the fact that $\rho \geq 0$ shows that condition 2 is satisfied $\forall \rho \geq 0$.

Ad.3 Using (3.12), (3.14), (3.16) and the fact that $\rho \geq 0$ shows that condition 3 is satisfied $\forall \rho \geq 0$.

Ad.4 Rewriting condition 4 yields:

$$|a_2| < \frac{b_0^2}{\rho} + 1$$

Taking into account (3.13) and the fact that $\rho \geq 0$ shows that condition 4 is satisfied $\forall \rho \geq 0$ too.

The conditions 1, ..., 4 are satisfied $\forall \rho \geq 0$ and hence the closed-loop system is stable $\forall \rho \geq 0$ under the assumptions mentioned in the theorem.

□

The closed-loop poles for $\rho = 0$ and $\rho \to \infty$ (the starting and ending points of the root locus) can be determined by utilizing the following theorems.

Theorem 3.2 *If $\rho = 0$, $H_p = H_c$, $H_s = 1$, $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$ (the model is identical to the process), then the closed-loop poles are determined by PTQ_dB, while the closed-loop zeros are determined by BTQ_d.*

Proof. If $\rho = 0$ and $H_p = H_c$, then criterion (2.146) becomes:

$$J = \sum_{i=1}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 \tag{3.17}$$

Further, it was shown in section 2.2.2 (see Theorem 2.2) that the controller obtained by minimization of (3.17) is independent of H_p. Hence, we might as well minimize:

$$J = [P\hat{y}(k+1) - Rw(k+1)]^2 \tag{3.18}$$

It was shown in section 2.2.1, that if (3.18) is minimized with respect to $u(k)$, the closed-loop transfer function is given by (2.107):

$$y(k) = \frac{q^{-1}R}{P}w(k+1) + \frac{1}{P}e(k) \tag{3.19}$$

From (2.213) and (2.192) follows that the zeros of the transfer function from $w(k+1)$ to $y(k)$ are determined by BTQ_d. Because the transfer function from $w(k+1)$ to $y(k)$ is equal to $\frac{q^{-1}R}{P}$ (see (3.19)), the poles of the transfer function are determined by PTQ_dB.

□

Note that time delay is not considered in the proof above. However, it can easily be included. Further, note that the numerator and the denominator of the transfer function from $w(k+1)$ to $y(k)$ have the factor BTQ_d in common. This is not the case if $A \neq \hat{A} \vee B \neq \hat{B} \vee \rho \neq 0$.

Theorem 3.3 *If $\rho \to \infty$, then $S \to 0$ (feedback is no longer present).*

Proof. If $\rho \to \infty$ then $R_v^{-1} \to 0$ (2.185). Further, $x^T, v^T, v_2^T, l^T, \lambda^T \to 0$. From (2.190) follows that $S \to 0$.

□

Corollary 3.1 *If $\rho \to \infty$, then A is a factor of the closed-loop characteristic equation.*

Proof. If $\rho \to \infty$ then $S \to 0$ (Theorem 3.3) and hence feedback has vanished. Obviously, the poles of the process appear as closed-loop poles.

□

Hence, if the process is stable, the closed-loop system is also stable if $\rho \to \infty$.

Corollary 3.2 *If $\rho \to \infty$ then $u(k) = 0 \; \forall k$.*

Proof. If $\rho \to \infty$ then $\boldsymbol{R}_v^{-1}, \boldsymbol{v}^T \to 0$ (2.185) and hence $\tilde{V} \to 0$. From (2.191) follows that $\mathcal{T} \to 0$. Taking into account $S \to 0$ (Theorem 3.3), (2.193) shows that $u(k) \to 0\ \forall k$.

□

Theorem 3.4 *If $\rho \to \infty$ and $H_p = H_c$, then the closed-loop poles are determined by ATQ_n.*

Proof. If $H_p = H_c$ then $\boldsymbol{N} = \boldsymbol{0}$ (2.174) $\Longrightarrow \rho \boldsymbol{z}_2^T = 0$ (2.185), $\rho \tilde{Z}_2 = 0$ (2.187) and $\boldsymbol{M} = \boldsymbol{I}$ (2.173). Now, $\rho \boldsymbol{z}^T$ in (2.185) becomes for $\rho \to \infty$:

$$\rho \boldsymbol{z}^T = \rho \boldsymbol{e}_1^T \boldsymbol{R}_v^{-1} \boldsymbol{\Phi}^T \boldsymbol{\Omega} = \boldsymbol{e}_1^T \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^T \boldsymbol{\Omega} = \boldsymbol{e}_1^T \boldsymbol{\Phi}^{-1} \boldsymbol{\Omega} \tag{3.20}$$

$\boldsymbol{\Phi}$ is a lower triangular Toeplitz matrix with ones on its diagonal (see (2.177) and (2.178)) and hence its inverse is also a lower triangular matrix with ones on its diagonal. Now, (3.20) becomes: $\rho \boldsymbol{z}^T = \boldsymbol{\Omega}_1$. In which $\boldsymbol{\Omega}_1$ is the first row of $\boldsymbol{\Omega}$. Its elements are given by the solution of (2.177) for $i = 1$:

$$\begin{aligned} \Phi_1 &= 1 \\ \Omega_1 &= q(Q_n - Q_d) \end{aligned}$$

Taking into account $\boldsymbol{x}^T, \boldsymbol{v}^T, \boldsymbol{v}_2^T, \boldsymbol{l}^T, \boldsymbol{\lambda}^T \to 0$ if $\rho \to \infty$ (see the proof of Theorem 3.3), $\mathcal{R}$ (2.189) becomes:

$$\mathcal{R} = TQ_d + q^{-1} \left(\rho \tilde{Z} T \right) = TQ_n$$

By using Theorem 3.3, the closed-loop transfer function (2.192) is:

$$y(k) = \frac{q^{-d-1} B \mathcal{T} w(k + H_p) + \dfrac{CTQ_n}{D} e(k)}{ATQ_n} \tag{3.21}$$

From (3.21) directly follows that the closed-loop poles are given by ATQ_n.

□

The above-mentioned theorems are illustrated by the root loci as a function of ρ with $Q_n = 1$ and $Q_n = \Delta$ (Figures 3.3 and 3.6). Both figures show that for $\rho = 0$, one of the closed-loop poles cancels the zero of the process (= −0.928, see (3.8)). Further, if $\rho \to \infty$, the closed-loop poles are equal to the poles of the process (= 0.882 and 0.905, see also (3.8)) and the roots of Q_n.

Comparing the results obtained when $Q_n = 1$ and $Q_n = \Delta$ (Examples 3.1 and 3.2) one can say that weighting the controller increments ($Q_n = \Delta$) can be dangerous. The closed-loop system can become badly damped or even unstable. This effect did not occur when the controller output was weighted directly. Of course, one process has been examined only. However, one can expect the same behavior for other processes as well. This can be explained by the fact that when $\rho \to \infty$ the closed-loop poles are determined by ATQ_n (Theorem 3.4). Hence, the roots of Q_n appear as closed-loop poles for $\rho \to \infty$. Compared to the situation in which $Q_n = 1$, using $Q_n = \Delta$ results in an extra end point for the root locus at $z = 1$. Therefore, a branch of the root locus can easily approach the unit circle making the closed-loop system badly damped or even unstable. This is more likely to occur if the process contains poles which are also close to $z = 1$ (for example, processes with an integrator).

So far two possible choices for Q_n have been discussed. The first one ($Q_n = \Delta$) makes sure that the steady-state error is not affected by the weighting factor ρ. However, the dynamic behavior of the closed loop system can be bad. The second one ($Q_n = 1$) results in a much better dynamic behavior of the closed-loop system. However, the steady-state error depends on ρ.

Another point of view is that when $Q_n = \Delta$ is used, the higher frequencies in $u(k)$ are heavily weighted, while the lower frequencies are less weighted. The frequency 0 is not weighted at all. When $Q_n = 1$ all frequencies in $u(k)$ are equally weighted. Thus, what we are really looking for is a filter $\frac{Q_n}{Q_d}$ which does not pass frequency zero but passes all other frequencies. Of course, such a filter is hard to realize. However, a compromise can be made by using:

$$\frac{Q_n}{Q_d} = \frac{1 - q^{-1}}{1 - \mu q^{-1}} \tag{3.22}$$

with $\mu \to 1$. The frequency response of this filter with $\mu = 0.95$ is shown in Figure 3.7 and shows that it approximates the desired response: The frequency 0 is not passed at all (hence, the steady-state performance is not affected by ρ) while the higher frequencies are all passed. Subsequently, $\mu = 0.95$ is used. In order to compare the results obtained using $Q_n = 1$ and $Q_n = \Delta$ with those obtained

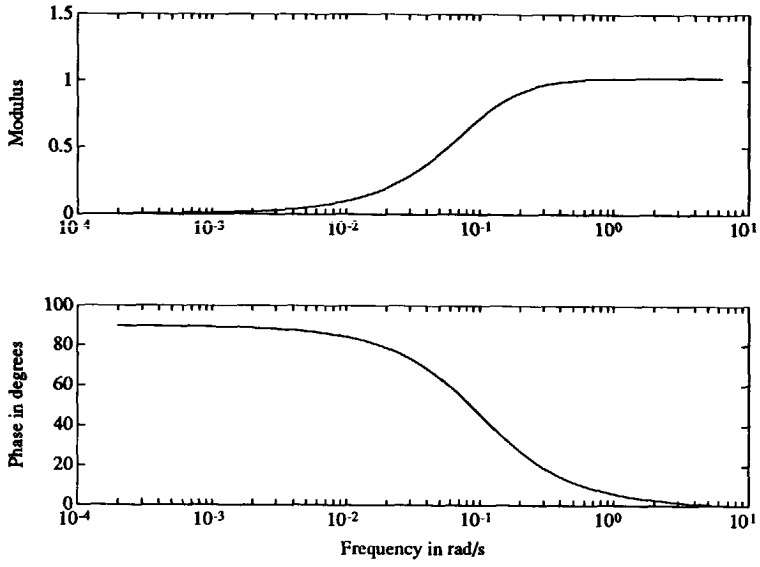

Figure 3.7: Frequency response of the approximated filter.

by using the filter (3.22), the simulations described above are repeated using the proposed filter.

Example 3.3	Influence of ρ and H_p on the stability of the closed-loop system.
Settings:	same as in Example 3.1 except $Q_n = \Delta$ and $Q_d = 1 - 0.95q^{-1}$
Process:	same as in Example 3.1
Model:	same as in Example 3.1
Parameters:	ρ: $0 \leq \rho < \infty$, H_p: 1, 2, 10

The Figures 3.8a, 3.8b and 3.8c show again root loci as a function of ρ. The root loci shown in these figures approximate those obtained with $Q_n = 1$ and $Q_d = 1$ and hence the closed-loop system remains stable. Further, if $H_p = 10$, the closed-loop poles are well damped ($O < 1\% \ \forall \rho \geq 0$).

Remark: Compared to the root loci shown in Figure 3.6, the root loci shown in Figure 3.8 show extra branches starting at $z = 0.95$ and ending at $z = 1$.

To conclude, we can say that when Q_n should contain a factor Δ in order to prevent steady-state errors, one should make $1 - 0.95q^{-1}$ a factor of Q_d. Or, more in general, when Q_n should contain x factors Δ, Q_d should contain x factors $1 - 0.95q^{-1}$.

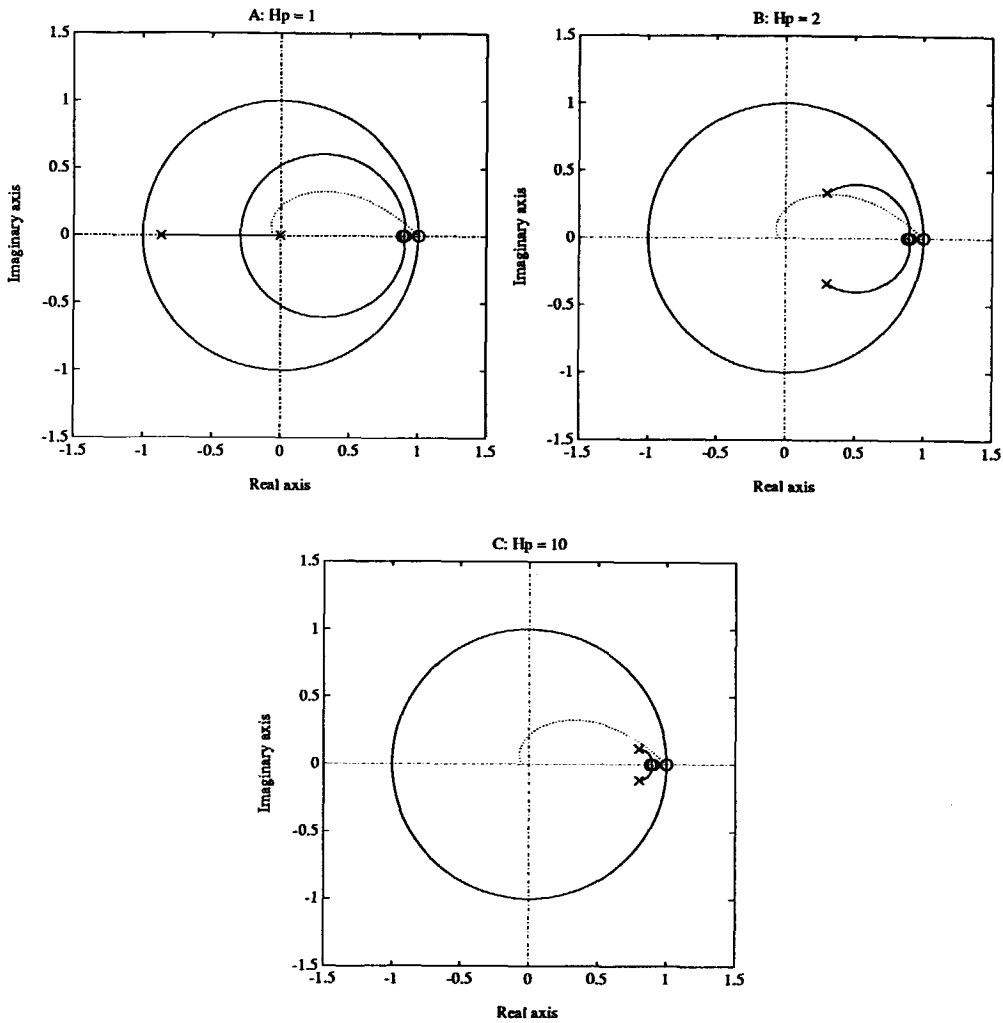

Figure 3.8: Root loci as a function of ρ where $Q_n = \Delta$ and $Q_d = 1 - 0.95q^{-1}$ for $H_p = 1, 2$ and 10.

Of course, when a specific spectrum in the controller output must be enhanced or attenuated, Q_d should be chosen in a different way. Section 3.3.2 gives an example of how to choose Q_n and Q_d in such a case.

3.3.1 Controller output weighting versus robustness

In this section the robustness of the closed-loop system with respect to variations in the process DC gain and time delay is addressed. For this purpose the robustness criteria gain and delay margin are used as discussed in section 3. In many cases increasing the weighting factor ρ makes the closed-loop system slower. If $\rho \rightarrow \infty$, feedback disappears (Theorem 3.3) and the poles of the process appear as closed-loop poles. Hence, in this case the closed-loop system is stable if the process is stable and hence the stability robustness of the closed-loop system is infinite. Of course, this is not a realistic situation because the controller output is zero (see Corollary 3.2). However, this special choice for ρ shows that when ρ is increased, an increase of robustness and a decrease of servo and regulator performance can be expected for a stable process. If the process is unstable, a decrease of robustness can be expected if ρ is increased beyond a certain value.

The above-mentioned statements are illustrated by two examples. First, a stable process is analyzed with respect to its robustness and performance as a function of ρ.

Example 3.4 Influence of ρ on the robustness and performance of the closed-loop system for a stable process.

Settings:	$H_p = 10,\ H_s = 1,\ H_c = 2,\ \beta = 1,\ P = N = 1,\ R = 1,\ Q_n = \Delta,$ $Q_d = 1 - 0.95q^{-1},\ T = 1,\ \hat{D} = 1,\ \underline{d} = 0,\ \overline{d} = 0$
Process:	$H(s) = \dfrac{4}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$ $H(z^{-1}) = \dfrac{0.0712z^{-1}(1 + 0.898z^{-1})}{1 - 1.591z^{-1} + 0.726z^{-2}}$
Model:	identical to process
Parameters:	ρ

The process that is used in this example is a second-order process with complex poles. The prediction and control horizon are chosen such that the closed-loop system has a high performance and low robustness for $\rho = 0$. The upper gain margin and the time delay margin as a function of ρ are shown in Figure 3.9. The dotted line in

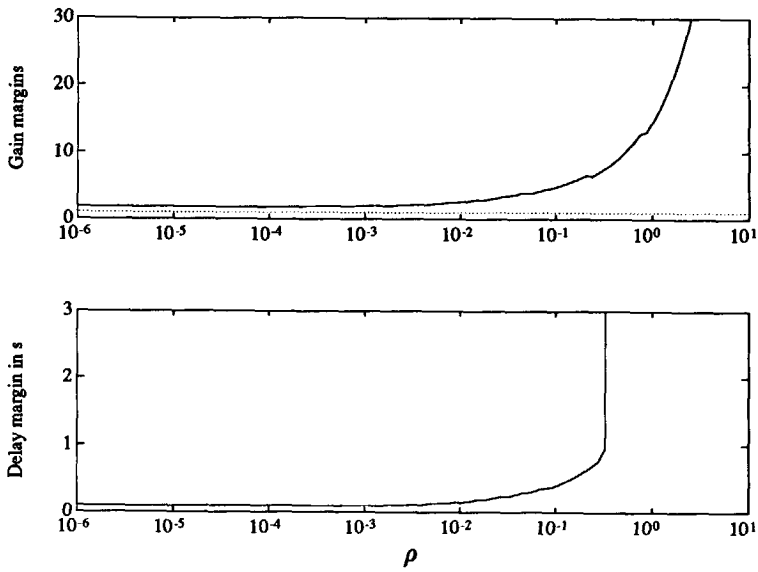

Figure 3.9: Gain and delay margins as a function of ρ.

Figure 3.9 and in all other figures in this chapter showing gain margins denotes a gain margin equal to 1. Hence, the closed-loop system is stable for $K_{dc} = \hat{K}_{dc}$ if the curve of the upper gain margin ($\overline{gm}$) is above the dotted line and the curve of the lower gain margin ($\underline{gm}$) is below the dotted line.

Figure 3.9 shows that increasing ρ increases the gain margin $\overline{gm}$ (the lower gain margin $\underline{gm}$ is equal to zero $\forall \rho \geq 0$). Also, the delay margin increases when ρ is increased. Moreover, if $\rho > 0.15$ the delay margin is infinite. Hence, as expected, the overall robustness of the closed-loop system increases with ρ.

But what about the performance of the closed-loop system? In order to examine the performance of the system, in Figure 3.10 the rise and settling time are shown as a function of ρ. As expected, the increase of robustness results in a slower system.

So far, a stable process. In the following, an unstable process is controlled and the effect of the weighting factor ρ on the robustness and performance is examined.

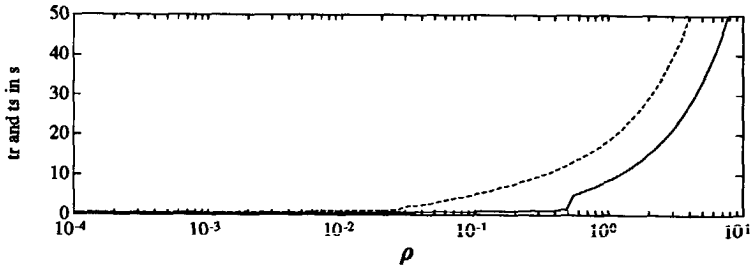

Figure 3.10: Rise time (solid line) and settling time (dashed line) as a function of ρ.

Example 3.5 Influence of ρ on the robustness of the closed-loop system for an unstable process.

Settings: same as in Example 3.4

Process:
$$H(s) = \frac{-0.5}{s^2 + 0.5s - 0.5} \quad T_s = 0.5s \Longrightarrow$$
$$H(z^{-1}) = \frac{-0.058z^{-1}(1 + 0.92z^{-1})}{(1 - 0.61z^{-1})(1 - 1.28z^{-1})} \tag{3.23}$$

Model: identical to process

Parameters: ρ

In Figure 3.11 the gain and delay margins are shown as a function of ρ. The upper gain margin (solid line) increases with ρ, in a way similar to the stable process mentioned in Example 3.4. However, for this process the lower gain margin (dashed line) is different from zero for $\rho > 11$ and increases when ρ increases. For $\rho > 25$ the lower gain margin becomes even larger than 1, resulting in an unstable closed-loop system. The delay margin increases when ρ increases making the closed-loop system more robust with respect to variations in the time delay. The closed-loop becoming unstable is also illustrated by a root-locus plot as a function of ρ. This root-locus is shown in Figure 3.12. The closed-loop poles for $\rho \to \infty$ are given by ATQ_n (see Theorem 3.4). These poles are shown in Figure 3.12a by the circles at $z = 0.61$ and $z = 1.28$ (poles of the process) and the circle at $z = 1$ (root of Q_n). Further, Figure 3.12b, which is a magnification of Figure 3.12a around $z = 1.1$, shows that the pole at $z = 0.95$ (for $\rho = 0$) first moves to the right, then moves to the left, becomes imaginary and, finally, as $\rho \to \infty$ approaches $z = 1.28$.

Remark: This behavior of closed-loop poles cannot occur if the root-locus equation could be written as in (3.5).

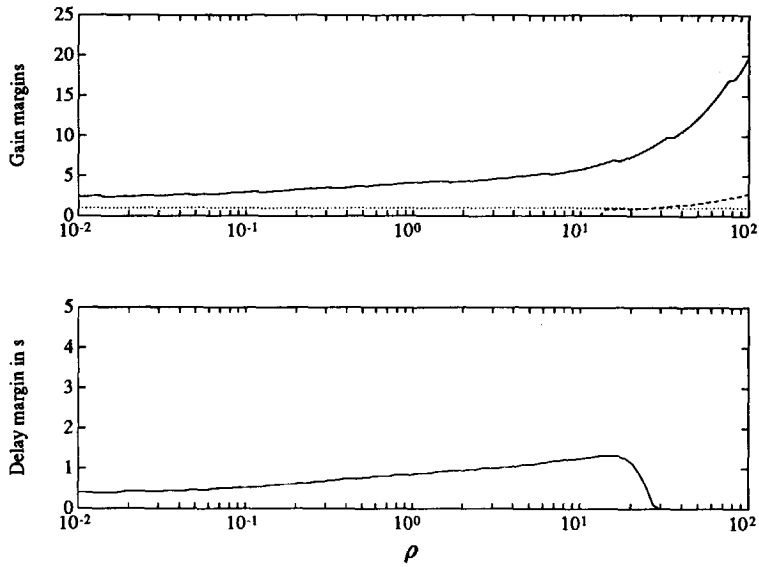

Figure 3.11: Gain and delay margins as a function of ρ for an unstable process.

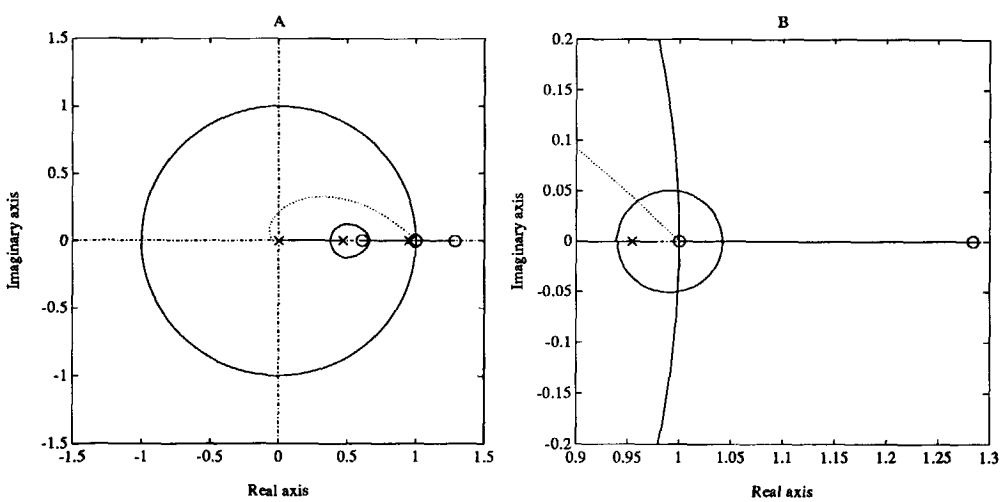

Figure 3.12: Root-locus plots as a function of ρ for an unstable process. Figure B is a magnification of Figure A around $z = 1.1$.

3.3.2 When do we use controller output weighting?

In the previous section it was shown that controller output weighting can be useful in improving the robustness of the closed-loop system. But in what other situations can controller output weighting be useful? One feature of controller output weighting in combination with the filter $\frac{Q_n}{Q_d}$ is that certain frequencies in $u(k)$ can be enhanced or attenuated. This can be used, for instance, if the process to be controlled is a mechanical system which has one (or more) resonance frequencies. In that particular case one can design the filter Q such that these frequencies are more weighted than other frequencies. When one frequency in the controller output is to be attenuated, a notch filter (see e.g. [19], p.247) can be designed and used as Q. Such a filter has a frequency response as shown in Figure 3.13. Or, a filter can be designed such that

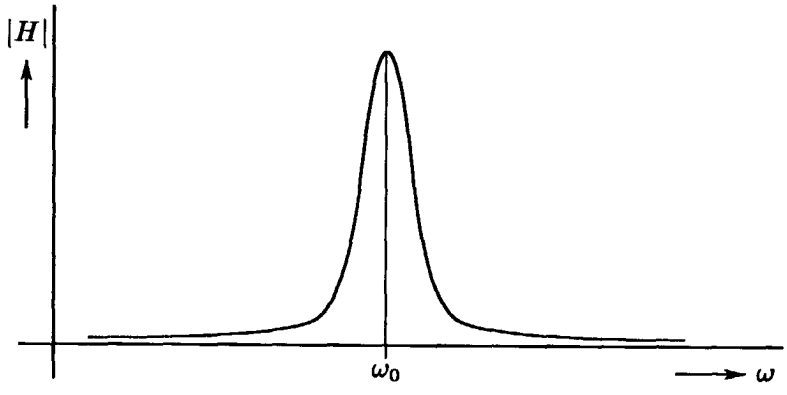

Figure 3.13: Frequency response of a notch filter.

only frequencies above a certain frequency are weighted. Then, high frequencies can be attenuated in the controller output. This can be useful if the process output is disturbed by measurement noise. For example, suppose that the process (3.8) is disturbed by measurement noise. Then the time response when $H_p = 10$, $H_c = 2$ and $\rho = 0$ is as shown in Figure 3.14. Figure 3.14 shows that the controller strongly reacts to the measurement noise. This can be explained by drawing a Bode diagram of the transfer function from $\xi(k)$ to $u(k)$. This diagram is depicted in Figure 3.15 (the dashed line) and shows that the higher frequencies in $\xi(k)$ are amplified, resulting in large (high frequency) control actions. In order to attenuate the amplification of the higher frequencies and to preserve the steady-state behavior (hence, the conditions

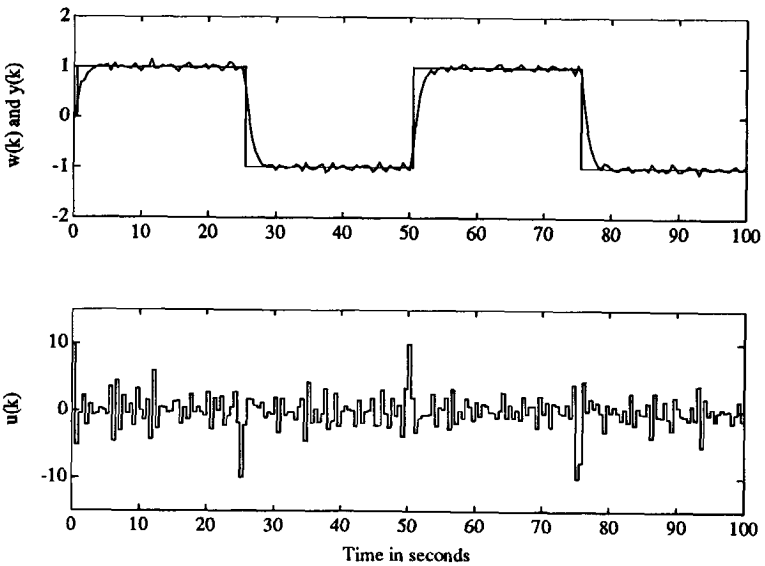

Figure 3.14: Time response when $\rho = 0$.

(2.203) ... (2.205) must be satisfied), the Q filter is selected as:

$$Q = \frac{Q_n}{Q_d} = \frac{(1 - q^{-1})^2}{(1 - 0.1q^{-1})^2}$$

Hence, a second-order filter with cross-over frequency at 2.7 rad/s. The frequency response of this filter is shown in Figure 3.16. In Figure 3.17 the simulation is repeated with $\rho = 0.02$ and the proposed Q-filter. Now, the controller output is much smoother while the servo behavior of the process is hardly affected (the response to a step change is slightly slower). The effect of the Q-filter is clearly visualised in the Bode diagram from $\xi(k)$ to $u(k)$ depicted in Figure 3.15 by the solid line. Frequencies over 2 rad/s are less amplified by the controller compared to those when using $\rho = 0$. Further, frequencies under 2 rad/s are amplified equally by both controllers.

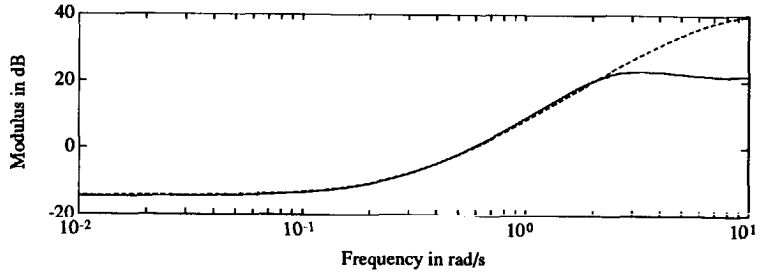

Figure 3.15: Frequency response of the transfer function from $\xi(k)$ to $u(k)$ with (solid line) and without (dashed line) controller output weighting.

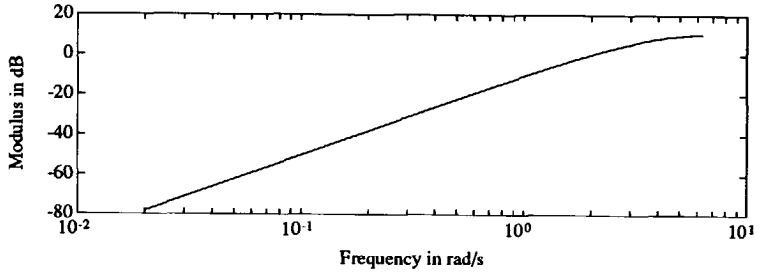

Figure 3.16: Frequency response of Q-filter.

3.3.3 Conclusions

In this section, the influence of the weighting factor ρ in combination with Q_n and Q_d on the robustness and performance of the closed-loop system is shown. It has been shown that using the weighting factor in combination with $Q_n = \Delta$ and $Q_d = 1$ can result in a badly damped or unstable closed-loop system. A better choice has shown to be $Q_n = \Delta$ and $Q_d = 1 - 0.95q^{-1}$. Then, still the steady-state requirements are fulfilled while the dynamic behavior is approximately equal to that when the controller output is weighted directly.

Further, weighting the controller output improves the robustness of the closed-loop system but also makes it slower. When the process is unstable, increasing the weighting factor results in an unstable closed-loop system, because for a large ρ the poles of the process appear as closed-loop poles. Finally, it was shown, by means of an example, that ρ in combination with Q_n and Q_d is a useful tool to attenuate unwanted frequencies in the controller output.

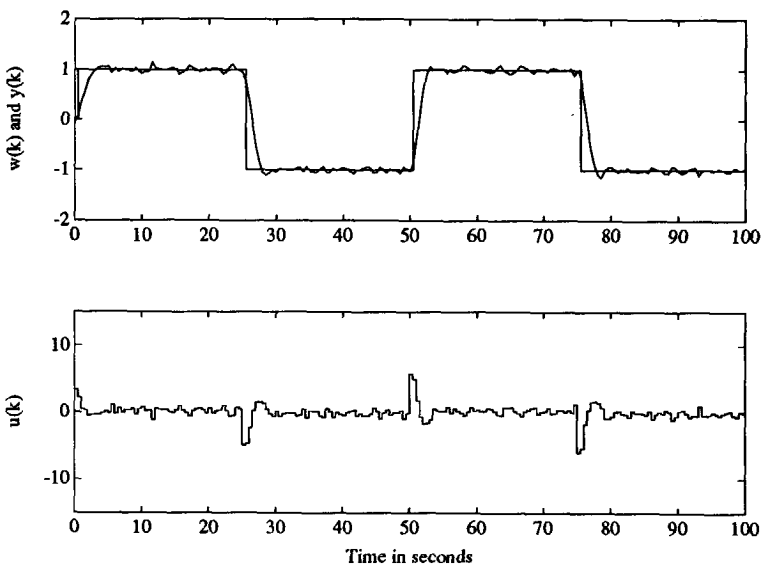

Figure 3.17: Time response when $\rho = 0.02$ and with Q-filter.

3.4 Prediction, Control and Minimum Cost Horizon

In this section the influence of the prediction horizon (H_p), control horizon (H_c) and minimum cost horizon (H_s) on the performance and robustness of the closed-loop system is examined. As discussed in section 2.2.2, H_p, H_c (in combination with N and β) and H_s can be used to select different well-known controllers. Table 2.4, which is repeated below (Table 3.2), shows how these parameters must be selected assuming that $\rho = 0$. Further, it was shown in Chapter 2 that N must be chosen equal to P in all cases examined. Subsequently, $N = P$ is used unless otherwise noted.

Controller	H_p	H_s	H_c	β	P	CLCE
MV	$\geq d+1$	$d+1$	H_p-d	≥ 1	1	BT
Mean-level	$\rightarrow \infty$	≥ 1	1	1	1	AT
Dead-beat	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	1	1	T
Moving-average	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	1	B^+	B^+T
Pole-placement	$\geq n_A+n_B+d+1$	n_B+d+1	n_A+1	1	P	PT

Table 3.2: Classical controllers as a function of the criterion parameters.

The following example illustrates the use of H_p, H_c, H_s and P in selecting MV, dead-beat, pole-placement and mean-level controllers.

Example 3.6 The use of H_p, H_c, H_s and P in selecting MV, dead-beat, pole-placement and mean-level controllers.

Settings: $\beta = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$

Process:
$$H(s) = \frac{5}{(4s+1)(5s+1)} \quad T_s = 0.5s \Longrightarrow$$

$$H(z^{-1}) = \frac{0.029z^{-1}(1+0.928z^{-1})}{(1-0.882z^{-1})(1-0.905z^{-1})} = \frac{0.029z^{-1}(1+0.928z^{-1})}{1-1.787z^{-1}+0.798z^{-2}} \tag{3.24}$$

Model: identical to process

Parameters: H_p, H_s, H_c and P

According to Table 3.2 minimum-variance control can be obtained by choosing $H_p = H_s = H_c = N = P = 1$. Figure 3.18a shows the resulting response. Due to the cancellation of the badly situated zero of the process (at $z = -0.928$) ringing in the controller output occurs. The process output tracks the set point in 1 sample as

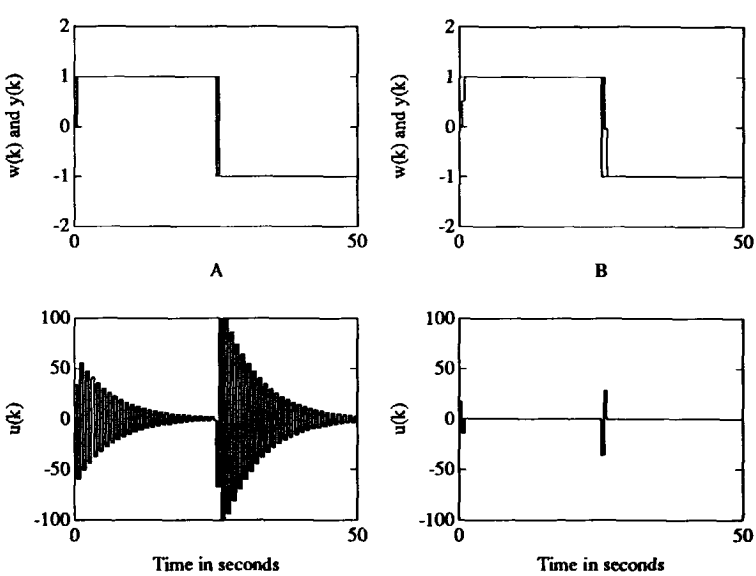

Figure 3.18: Minimum-variance (Figure A) and dead-beat control (Figure B).

expected.

Table 3.2 shows that dead-beat control can be obtained by choosing $H_p = 4$, $H_s = 2$, $H_c = 3$ and $N = P = 1$. Figure 3.18b shows the resulting response. Now that the process zero is no longer cancelled the ringing in the controller output disappears. The process tracks the set point in 2 ($= n_B + 1$) samples and 3 ($= n_A + 1$) distinct controller outputs are required. Pole-placement and mean-level control are

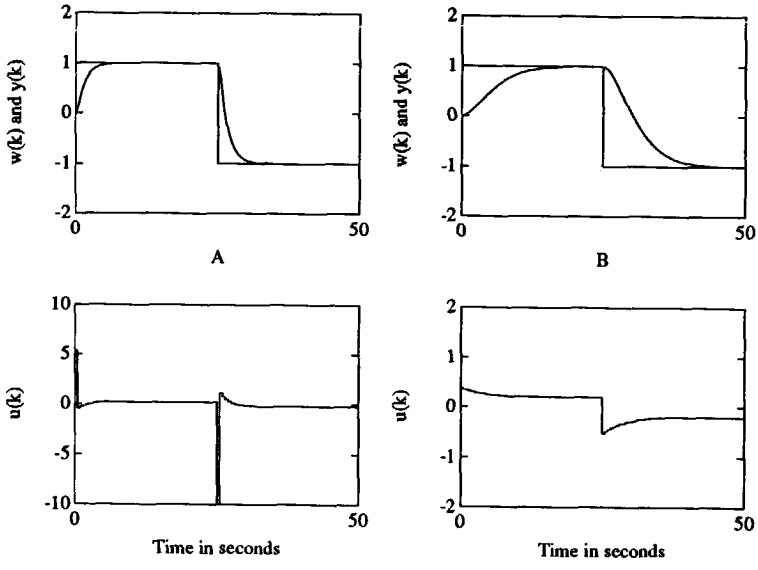

Figure 3.19: Pole-placement (Figure A) and mean-level control (Figure B).

illustrated by Figure 3.19. In Figure 3.19a a pole-placement controller is used where $H_p = 4$, $H_s = 2$, $H_c = 3$ and $N = P = (1 - 0.7q^{-1})$. Now, the closed-loop system behaves as a first-order process with a pole at 0.7 which corresponds to a time constant of 1.4s taking into account the sampling period of 0.5s. More on the use of the P polynomial in defining the servo behavior of the closed-loop system is given in section 3.6.

Finally, Figure 3.19b shows a response of the UPC controller with parameters $H_p = 25$ and $H_s = H_c = N = P = 1$. This response approximates that of the mean-level controller (for which H_p must be infinite). The closed-loop transfer function is given by:

$$H(z^{-1}) = \frac{1.047 \cdot 10^{-2}(1 + 0.928z^{-1})}{1 - 1.735z^{-1} + 0.755z^{-2}} \tag{3.25}$$

The closed-loop transfer function when using the mean-level controller should be equal to that of the process (3.24) corrected for the DC gain. Obviously, the transfer function (3.25) approximates the one obtained when using the mean-level controller.

In the next example, the closed-loop poles as a function of H_c are examined.

Example 3.7	The influence of H_c on the closed-loop poles.
Settings:	$H_p = 25$, $H_s = 1$, $\beta = 1$, $P = N = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	same as in Example 3.6
Model:	identical to process
Parameters:	$1 \leq H_c \leq 25$

The closed-loop poles as a function of H_c are shown in Figure 3.20. For $H_c = 1$ the closed-loop poles approximate the poles of the process (0.88 and 0.9, see (3.24)). Increasing the control horizon makes the closed-loop pole move to -0.928 (the other pole is for $H_c \geq 2$ situated in the origin and is not shown in Figure 3.20). For $H_c = 25$ the closed-loop pole cancels the zero at -0.928 and minimum-variance control is obtained. For $H_c = 3$ the closed-loop poles are in -0.14 and in the origin and hence a dead-beat controller is approximated. It was argued in section 2.2.2 (see page 60) that this will be the case if $H_c = n_A + 1$ and the process is minimum phase.

Another interesting aspect of finite horizon predictive controllers is that when $H_c = H_p$, $H_s = 1$ and $\rho \neq 0$, the controller no longer depends on H_p if H_p is sufficiently large (as with LQ controllers for which the horizon goes to infinity). Consequently, the controller for a finite prediction horizon coincides with the controller for an infinite prediction horizon. The following example illustrates this effect.

Example 3.8	The influence of H_c on the closed-loop poles in the case $H_p = H_c$ and $\rho \neq 0$.
Settings:	$H_s = 1$, $\beta = 1$, $P = N = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0.01$, $Q_n = 1$, $Q_d = 1$
Process:	same as in Example 3.6
Model:	identical to process
Parameters:	$1 \leq H_p \leq 25$ and $H_c = H_p$

Figure 3.21 shows a root-locus plot as a function of H_p with $H_c = H_p$ and $H_p = 1, \ldots, 25$. It clearly shows that for $H_p \geq 4$ the closed-loop poles remain almost constant and hence the controller obtained for $H_c = 4$ is almost equal to the controller

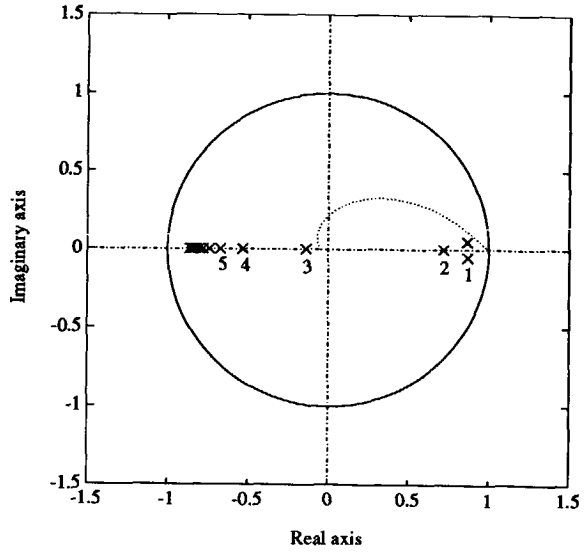

Figure 3.20: Root locus as a function of H_c where $H_p = 25$.

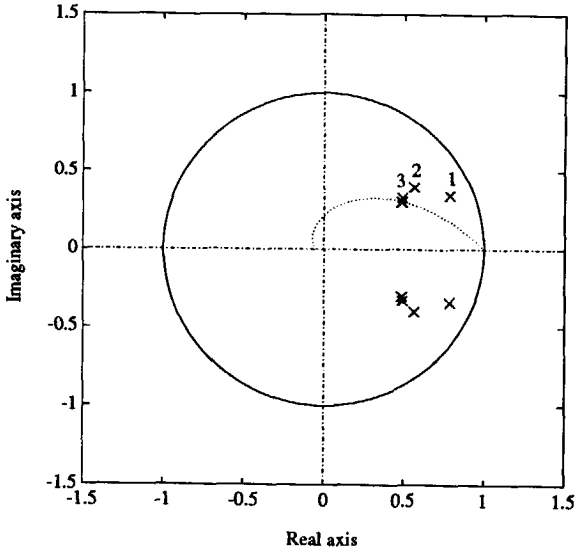

Figure 3.21: Root locus as a function of H_p ($H_p = 1, \ldots, 25$) where $H_c = H_p$ and $\rho = 0.01$.

obtained for $H_p \to \infty$.

This can be explained as follows. Suppose that t_{sk} is the settling time (in samples) of the (discrete) closed-loop system for $H_p \to \infty$. Then the closed-loop system is the same for all $H_p \geq t_{sk}$ because in this case the contribution of the predicted process output $\hat{y}(k+i)$ and the controller output $u(k+i-1)$ for $i = t_{sk}+1, \ldots, \infty$ to the criterion function is constant (and thus does not affect the gradient of the criterion function with respect to u). In the above-mentioned simulation, the settling time of the closed-loop system for $H_p \to \infty$ is approximately equal to 2s. Hence, for $H_p \geq 4$ (the sampling period is 0.5s) we can expect that the closed-loop system is independent of H_p. Example 3.8 illustrates this.

Hence, in general, it is sufficient to choose the prediction horizon according to the settling time of the closed-loop system. If the settling time of the closed-loop system when $H_p \to \infty$ is equal to $t_{s,\infty}$ (in seconds), then H_p can be chosen equal to $\text{int}(t_{s,\infty}/T_s)$, where int(.) is a function that converts a real value to an integer. A drawback is, however, that it can be quite difficult to determine $t_{s,\infty}$. Then, as a rule of thumb, H_p can be chosen equal to $\text{int}(t_s(5\%)/T_s)$ where $t_s(5\%)$ is the 5% settling time of the process to be controlled. Under the assumption that the closed-loop system settles faster than the open-loop system (which is usually the case), this rule of thumb yields sufficiently large values for H_p. However, for badly damped or unstable processes this rule of thumb cannot be used. Then, H_p can be related to the rise time of the process' step response in a way similar to the one described above. For example, H_p can be chosen equal to $\text{int}(3t_r/T_s)$ where t_r denotes the rise time of the process' step response. Simulations have shown that this rule of thumb usually yields sufficiently large prediction horizons.

Notes:

- If, as a rule of thumb, the sampling period is chosen $10-20$ times smaller than the settling time of the step response of the closed-loop system [19], H_p is $10-20$.
- By choosing H_p as described above, the sensitivity of the closed-loop system to changes in H_p is small. Therefore, H_p cannot be used as a tuning parameter. A more detailed discussion on how to tune the UPC controller can be found in section 3.9.

The role of β

So far, the control horizon was discussed with $\beta = 1$. β was introduced in section 2.2.2 in order to guarantee zero steady-state error if the disturbance and/or reference trajectory is not constant and the control horizon is different from the prediction horizon. Further, $\beta > 1$ is required when the process output must settle in a time-optimal way to a non-constant reference trajectory (see Theorem 2.6). However, β can also be used if the reference trajectory is constant. The effect of β on the closed-loop system in this situation is discussed in this section.

The use of the control horizon can be considered as a kind of controller output weighting just as is the use of ρ. Assume that the criterion to be minimized is given by:

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 \tag{3.26}$$

Further, assume that $H_c \neq H_p$, $N = 1$ and $H_s = 1$. Then (2.157) becomes:

$$\Delta^\beta u(k+i-1) = 0 \qquad H_c < i \leq H_p \tag{3.27}$$

Now consider the following criterion function:

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2 + \rho \sum_{i=H_c+1}^{H_p} \Delta^\beta u(k+i-1)^2 \tag{3.28}$$

where $\rho \to \infty$. Clearly, the optimal solution of (3.28) satisfies (3.27) and hence the optimal solution of (3.28) is the same solution as when (3.26) is minimized taking into account (3.27). Depending on the control horizon some controller increments are weighted with infinite weights while others are not weighted at all.

In section 3.3 it was shown that weighting the controller increments ($\beta = 1$) can result in an unstable closed-loop system. Therefore, one can expect that the same can happen when the control horizon is used, especially when the control horizon is small and the prediction horizon is large (in this case, many controller increments are assumed to be zero). Increasing β will make the situation even worse. This effect is illustrated in the following example for different values of H_c.

Example 3.9 The influence of β and H_c on the closed-loop poles.

Settings:	$H_p = 10$, $H_s = 1$, $N = 1$, $P = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	same as in Example 3.6
Model:	identical to process
Parameters:	$1 \leq \beta \leq 5$ and $H_c = 1, 2$ and 3

Figure 3.22a shows that if $H_c = 1$ the process becomes unstable as β increases. Increasing H_c (Figures 3.22b and 3.22c) improves the situation though still the closed-loop poles tend to go to the unit circle. It can be concluded that in order to preserve stability, β has to be kept small.

3.4.1 Prediction and control horizon versus robustness

The prediction and control horizon are closely related to the stability of the closed-loop system. In this section, the influence of H_p and H_c on the (stability) robustness of the system is examined for several processes. In the first example, the influence of H_p is examined when $H_c = 1$.

Example 3.10 The influence of H_p on the robustness of the closed-loop system when $H_c = 1$ and $\rho = 0$.

Settings:	$H_c = 1$, $H_s = 1$, $\beta = 1$, $P = N = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	$H(s) = \dfrac{4}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$
	$H(z^{-1}) = \dfrac{0.0712z^{-1}(1 + 0.898z^{-1})}{1 - 1.591z^{-1} + 0.726z^{-2}}$ (3.29)
Model:	identical to process
Parameters:	$1 \leq H_p \leq 25$

In this case, increasing the prediction horizon makes the controller move from a minimum-variance controller to a mean-level controller. Hence, an increase of the robustness of the closed-loop system can be expected if the process is stable. The process that is used in this example is a second-order process with complex poles. Its step response is shown in Figure 3.23.

Figure 3.24 shows the upper gain margin and the upper delay margin for

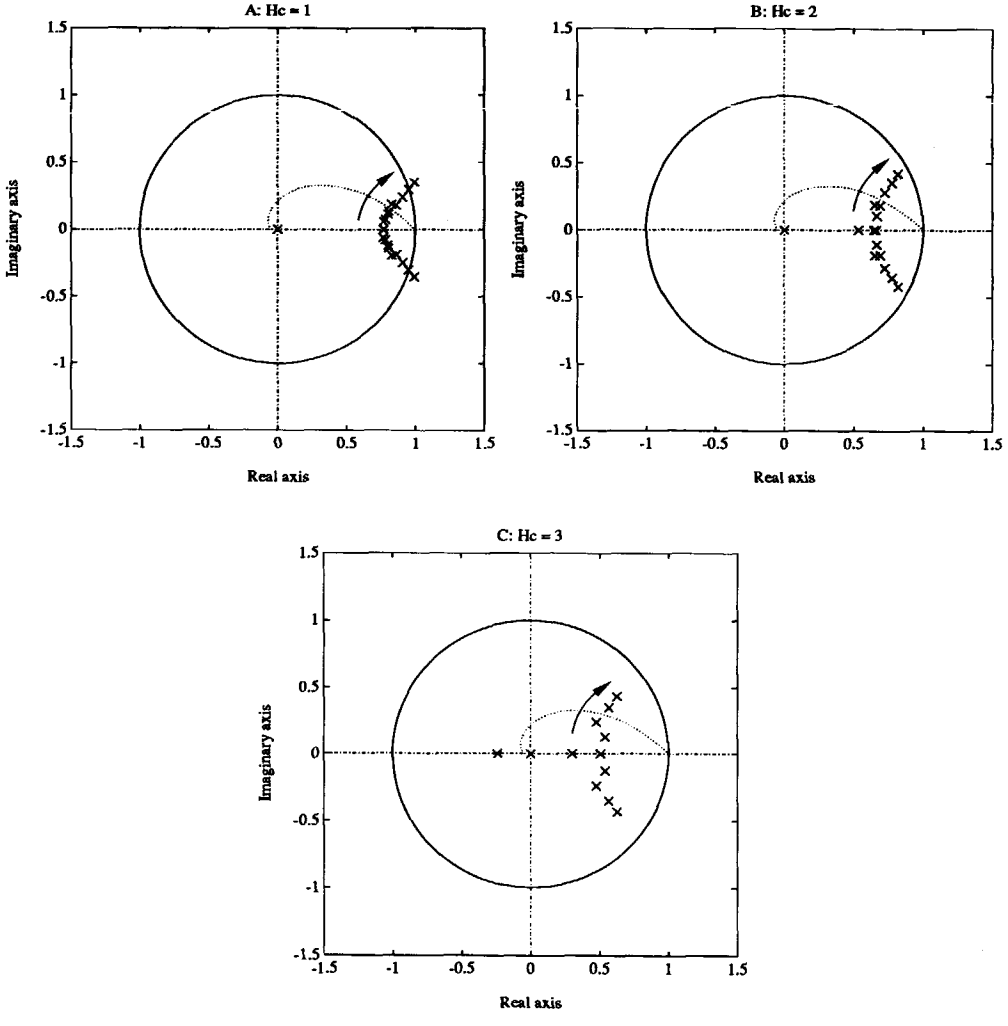

Figure 3.22: Root-locus plots as a function of β for $H_c = 1, 2$ and 3.

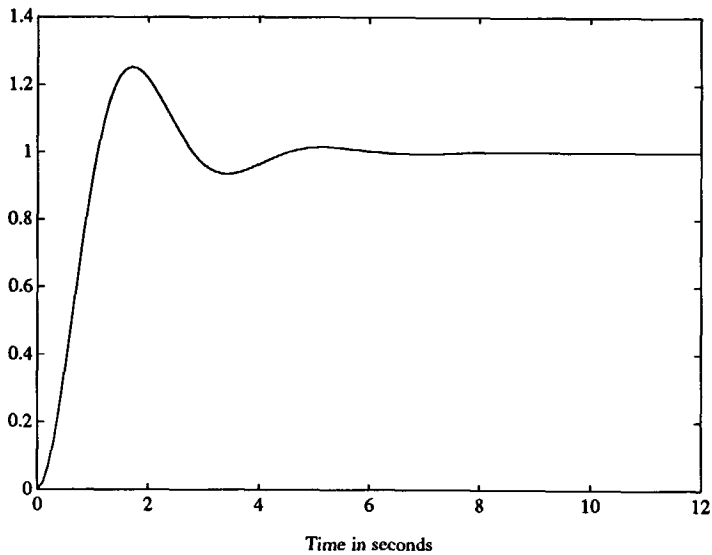

Figure 3.23: Step response of the second-order process with complex poles used in Example 3.10.

$H_p = 1, \ldots, 25$ and $H_c = 1$. The lower gain margin is equal to zero. The lower delay margin is not of interest because the process and model do not contain time delay. Figure 3.24 shows that for $H_p \geq 5$ the delay margin is infinite. Hence, the closed-loop system cannot become unstable due to delay mismatch. Further, the figure shows that increasing the prediction horizon increases the upper gain margin. However, increasing the prediction horizon from 8 to 9 makes the upper gain margin suddenly drop. Further increasing H_p makes the upper gain margin increase again. This sudden drop of the gain margin is caused by the fact that the process has complex poles. The step response of the process (see Figure 3.23) shows a peak at $t = 1.6$s (= 8 samples).

That a process with real poles only does not show a drop in gain margin as the prediction horizon increases, is shown by Figure 3.25. This figure shows the upper gain and delay margin when the process is described by (3.24).

The effect of H_p (with $H_c = 1$) on the performance of the system is shown in Figure 3.26. This figure shows the rise, peak and settling time as a function of H_p with $H_c = 1$. Increasing H_p obviously makes the closed-loop system respond slower to step-wise set point changes. This effect can easily be explained by the fact that the controller moves from a minimum-variance to a mean-level controller as H_p

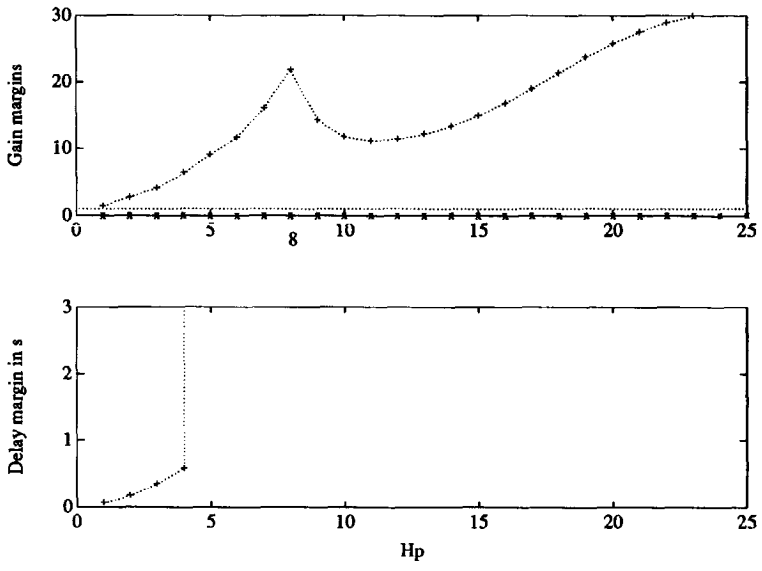

Figure 3.24: Upper gain and delay margin as a function of H_p where $H_c = 1$ and $\rho = 0$ in controlling a process with complex poles.

increases. When $H_p \rightarrow \infty$ the step response of the closed-loop system will be as slow as that of the open-loop system. This is also illustrated in Figure 3.26. When $H_p = 25$, the rise, peak and settling time approximate those of the process (for which $t_r = 1.2$s, $t_p = 1.8$s and $t_s = 4.4$s, see also Figure 3.23). So far, the situation with $H_c = 1$ has been discussed. In the next example, the situation where $H_c = 2$ and $H_c = 3$ is discussed.

Example 3.11 The influence of H_p and H_c on the robustness of the closed-loop system.	
Settings:	same as in Example 3.10 except that now $H_c = 2$ or $H_c = 3$
Process:	same as in Example 3.10
Model:	identical to process
Parameters:	$H_c = 2, 3$ and $H_c \leq H_p \leq 25$

The case where $H_c = 3$ is of interest because then $H_c = n_A + 1$ and with H_p large, a dead-beat controller is approximated. Remember that a dead-beat controller yields a fast closed-loop system and is capable of controlling processes with an unstable inverse. Further, if $H_c = 2$, a compromise between dead-beat and mean-level control

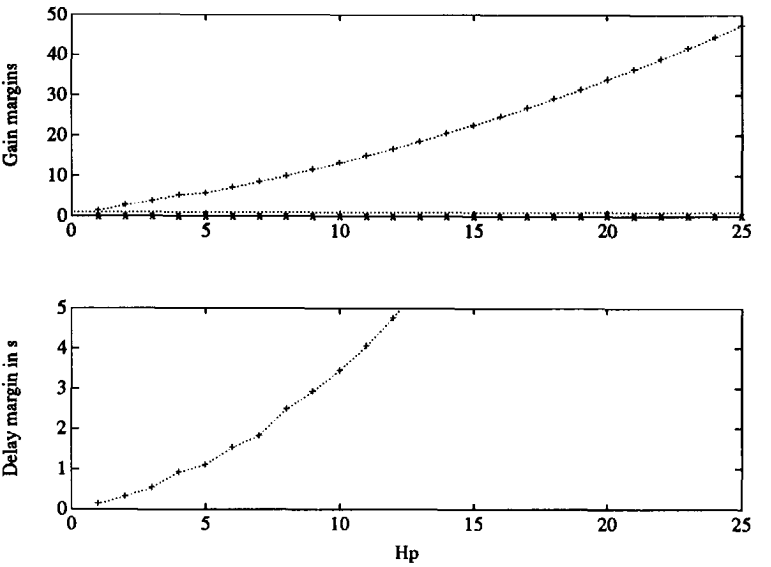

Figure 3.25: Upper gain and delay margin as a function of H_p where $H_c = 1$ and $\rho = 0$ in controlling a process with real poles.

is obtained.

The Figures 3.27 and 3.28 show the upper gain and delay margin as a function of H_p for $H_c = 2$ and $H_c = 3$, respectively. The lower gain margin is equal to zero for both settings of H_c. Further, the lower gain margin is not of interest (see also Example 3.10). The Figures 3.29 and 3.30 show the criteria t_r and t_s as a function of H_p for $H_c = 2$ and $H_c = 3$, respectively. The Figures 3.27 and 3.29 show that for $H_c = 2$ the robustness of the closed-loop system increases with H_p but settles at $\overline{gm} = 4$ and $\overline{dm} = 0.3$s. Increasing $H_p \geq 15$ no longer shows much effect. The (2%) settling time of the closed-loop system for $H_p \rightarrow \infty$ is approximately equal to 2s. The closed-loop system then should be constant for $H_p \geq 10$. However, the figure shows that for $H_p \geq 15$ the closed-loop system becomes constant. The difference can be explained by the fact that the 2% settling time is considered. Theoretically the system becomes independent of H_p if H_p is chosen beyond the 0% settling time of the step response of the closed-loop system (then the system is in the steady state). Using the 0.3% settling time (= 3s) yields $H_p \geq 15$ which is a more accurate estimate of the prediction horizon for which the closed-loop system becomes independent of H_p.

Compared to the case in which $H_c = 1$, the robustness of the process is

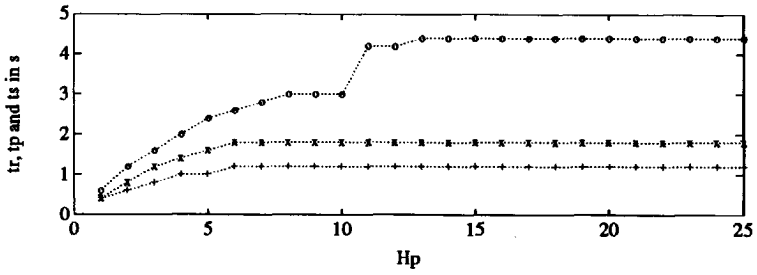

Figure 3.26: Rise ('+'), peak ('x') and settling time ('o') as a function of H_p where $H_c = 1$ in controlling a second-order process with complex poles.

smaller for a large H_p but may still be acceptable (the gain margin is quite large, however, the delay margin shows that adding two samples delay to the process makes the closed-loop system unstable). However, the closed-loop system settles faster compared to the case in which $H_c = 1$. Choosing, for example, $H_p = 15$ yields a settling time of 2s, while where $H_c = 1$ the settling time is equal to the settling time of the process: $t_s = 4.4$s.

When $H_c = 3$, the robustness is smaller than when $H_c = 2$. The time domain criteria (Figure 3.30) show that for $H_p > 9$ the closed-loop system has a rise time of 2 samples (= 0.4s) and settles in 3 samples (= 0.6s). Hence, dead-beat control is approximated. Further, note that by choosing $H_p > \text{int}(t_{s,\infty}/T_s) = 5$ makes the closed-loop system independent of H_p (for $t_{s,\infty}$ the 0.1% settling time (= 1.0s) has been used).

Compared to the case in which $H_c = 1$, the gain and delay margins and the time domain criteria obtained with $H_c = 2$ and 3 remain constant for H_p sufficiently large. By choosing H_p according to the 5% settling time of the process' step response, as discussed earlier in this section, the closed-loop system is barely sensitive to changes in H_p.

If $H_c = 1$, choosing the prediction horizon according to the 5% settling time of the process results in a closed-loop system which has almost the same dynamics as the open-loop system. The closed-loop poles and the time domain criteria are in this case hardly affected by the choice for H_p. However, because, for the settings used in this example, feedback no longer exists if $H_p \rightarrow \infty$ (see section 2.2.2), the robustness criteria keep improving when H_p is increased. For $H_p \rightarrow \infty$, these criteria are infinitely large. If $H_c = 1$ and H_p is significantly smaller than the 5% settling time of the process' step response, the closed-loop system is highly sensitive to changes H_p. Then, H_p can possibly be used as a tuning parameter.

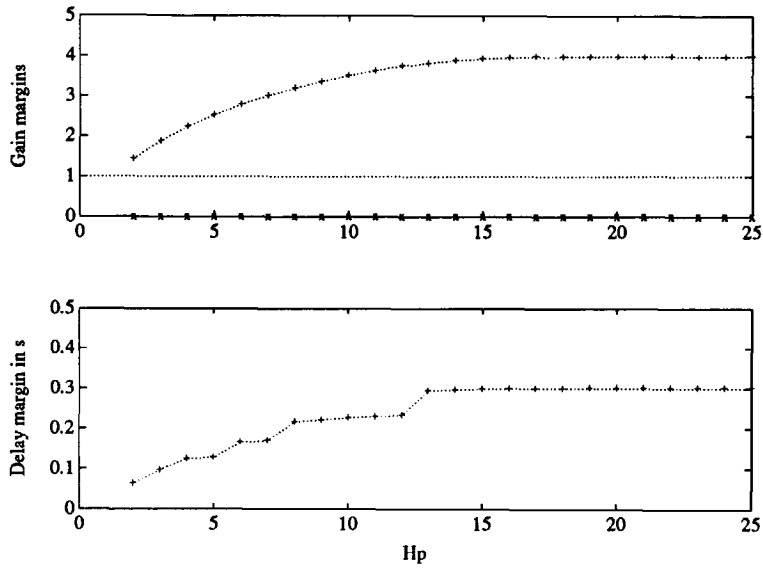

Figure 3.27: Upper gain and delay margin as a function of H_p where $H_c = 2$.

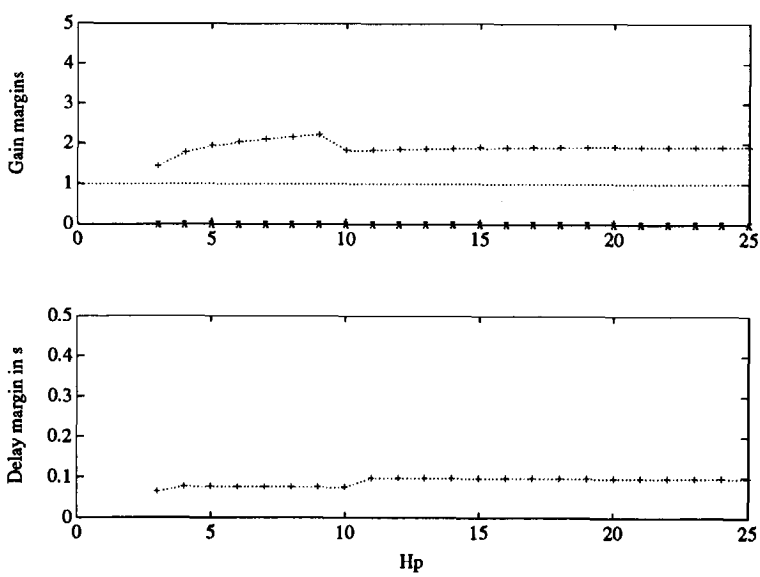

Figure 3.28: Gain and delay margins as a function of H_p where $H_c = 3$.

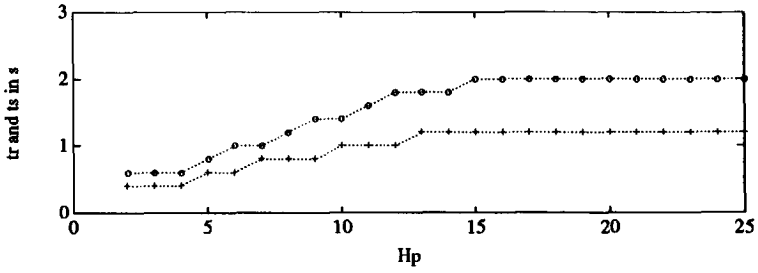

Figure 3.29: Rise ('+') and settling time ('o') as a function of H_p where $H_c = 2$.

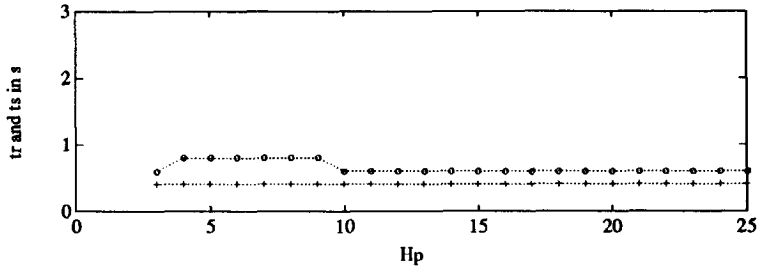

Figure 3.30: Rise ('+') and settling time ('o') as a function of H_p where $H_c = 3$.

For example, H_p could be chosen equal to 4. Now, the closed-loop system is quite fast: $t_r = 1s, t_p = 1.4s, t_s = 2$s while the robustness is given by: $\overline{gm} = 6.3$ and $\overline{dm} = 0.52$s.

Prediction and control horizon and non-minimum phase processes

To here, a simple second-order process with complex poles was used to examine the parameters H_p, H_s and H_c. It was shown that for H_p greater than or equal to $\text{int}(t_s(5\%)/T_s)$ and $H_c = 2$ or 3, a fast response and an acceptable robustness could be obtained. In order to show how the prediction and control horizon should be selected for a more complex process, a non-minimum phase process having the same complex poles as (3.29) is selected for the next example.

Example 3.12 The influence of H_p and H_c on the robustness of the closed-loop system in controlling a non-minimum phase process.

Settings: same as in Example 3.10 except that now $H_c = 1, 2, 3$

Process:
$$H(s) = \frac{-4(s-1)}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$$
$$H(z^{-1}) = \frac{-0.5954z^{-1}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}} \tag{3.30}$$

Model: identical to process

Parameters: $H_c = 1, 2, 3$ and $H_c \leq H_p \leq 25$

The step response of (3.30) is shown in Figure 3.31. Figure 3.32 shows the upper

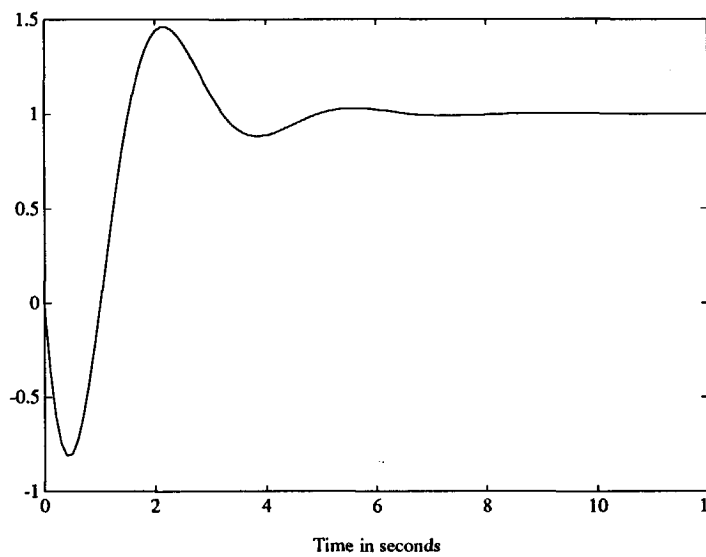

Figure 3.31: Step response of the non-minimum phase process used in Example 3.12.

and lower gain margin and the lower delay margin as a function of H_p with $H_c = 1$. The figure shows that the closed-loop system is stable for $H_p \geq 9$ (then, the lower gain margin is smaller than 1).

Remark: the delay margin is not shown for $H_p \leq 8$. Then the closed-loop system is unstable and the delay margin is not defined (see section 3.2). Increasing the prediction horizon makes the robustness increase as shown in Example 3.10. Again, as H_p increases, the closed-loop dynamics approximate those of the process.

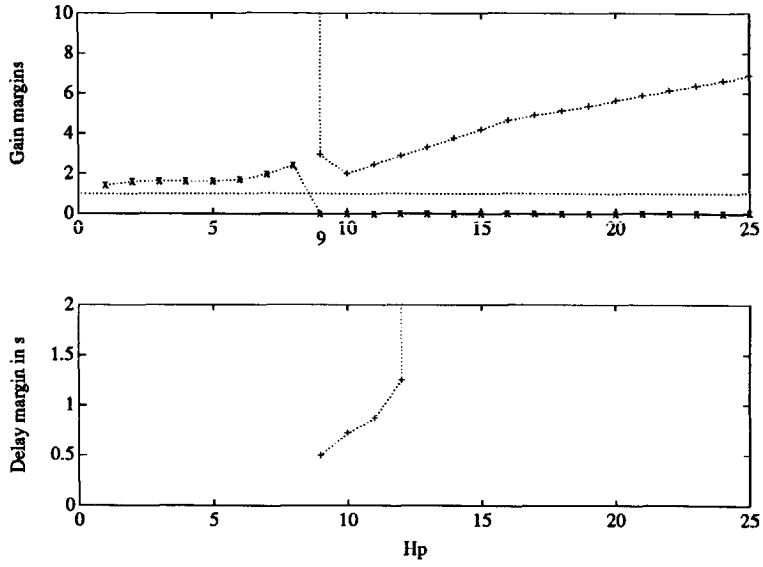

Figure 3.32: Upper ('+') and lower gain margin ('x') and upper delay margin as a function of H_p where $H_c = 1$.

Figure 3.33 shows the robustness criteria when $H_c = 2$. Now the closed-loop system is stable for $H_p \geq 7$. However, increasing the prediction horizon hardly affects the robustness.

When $H_c = 3$, the closed-loop system is stable if $H_p \geq 9$ and the robustness is hardly affected by H_p. The performance of the closed-loop system is illustrated by Figure 3.34. This figure shows a step response of the system where $H_p = 20$ and $H_c = 3$. This figure clearly shows that dead-beat control is not obtained when the option $H_p = 20$, $H_s = 1$ and $H_c = n_A + 1 = 3$ is chosen. This is caused by the fact that the process is non-minimum phase. This phenomenon is discussed in detail in section 2.2.2, page 60.

The prediction horizon and processes with time delay

In the previous sections it was shown that, as a rule of thumb, the prediction horizon can be taken approximately equal to $\text{int}(t_s(5\%)/T_s)$ if $H_c > 1$. Because adding time delay to a process yields a shifted step response only, we can expect that this rule of thumb remains valid. Moreover, we can expect that the prediction horizon must

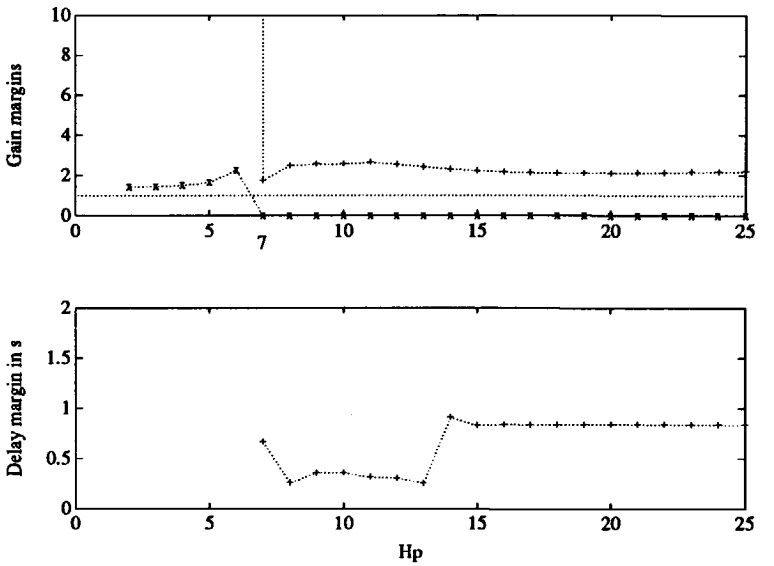

Figure 3.33: Upper ('+') and lower gain margin ('x') and upper delay margin as a function of H_p where $H_c = 2$.

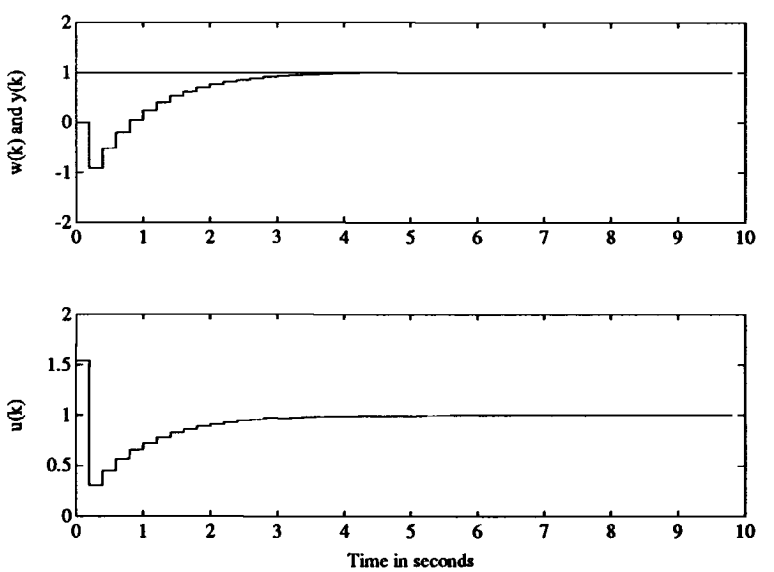

Figure 3.34: Step response of the closed-loop system where $H_p = 20$ and $H_c = 3$.

be increased by the amount of time delay (in samples) that is added to the process. In the following example the non-minimum phase process (3.30) is extended by 3 samples time delay. Note that H_s must be chosen equal to $\underline{d} + 1$.

Example 3.13 The influence of H_p on the robustness of the closed-loop system in controlling a non-minimum phase process with time delay.

Settings: same as in Example 3.12 but now $H_c = 2$, $H_s = 4$, $\underline{d} = 3$ and $\overline{d} = 3$

Process:
$$H(s) = \frac{-4(s-1)e^{-0.6s}}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$$
$$H(z^{-1}) = \frac{-0.5954z^{-4}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}}$$

Model: identical to process

Parameters: $5 \leq H_p \leq 25$

Figure 3.35 shows the gain and delay margins as a function of H_p in the case $H_c = 2$. Note that in this case H_p must be chosen ≥ 5. (see section 2.3.1, page 80).

The figure shows that the closed-loop system now becomes stable for $H_p \geq 10$ which is 3 samples larger than for the same process without time delay (see Example 3.12). This is caused by the time delay of 3 samples being present in the process. However, the robustness of the closed-loop system is quite poor. The gain margins may still be acceptable, the delay margins, however, are rather small: a mismatch of 0.05s in the time delay makes the closed-loop system unstable. The robustness of the closed-loop system can be increased by increasing the weighting factor ρ where $Q_n = \Delta$ and $Q_d = 1 - 0.95q^{-1}$ as discussed in section 3.3.1. Figure 3.36 illustrates the use of ρ in increasing the robustness of the closed-loop system when $H_p = 25$. Consequently, the closed-loop system becomes significantly slower as is shown in Figure 3.37. For example, for $\rho = 2$ the rise time is equal to 19s which is quite large compared to that of the process which has a rise time of 2.1s. However, the overshoot and steady-state error are equal to zero.

In section 3.5 another way of increasing the robustness of the closed-loop system is discussed. There, the noise model is used to increase the robustness of the closed-loop system without making the closed-loop respond more slowly to set point changes.

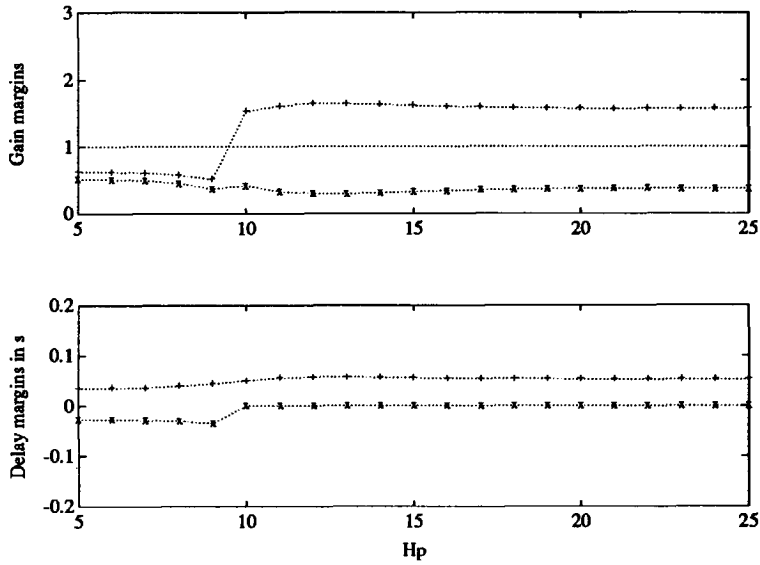

Figure 3.35: Upper gain and delay margin ('+') and lower gain and delay margin ('x') as a function of H_p where $H_c = 2$.

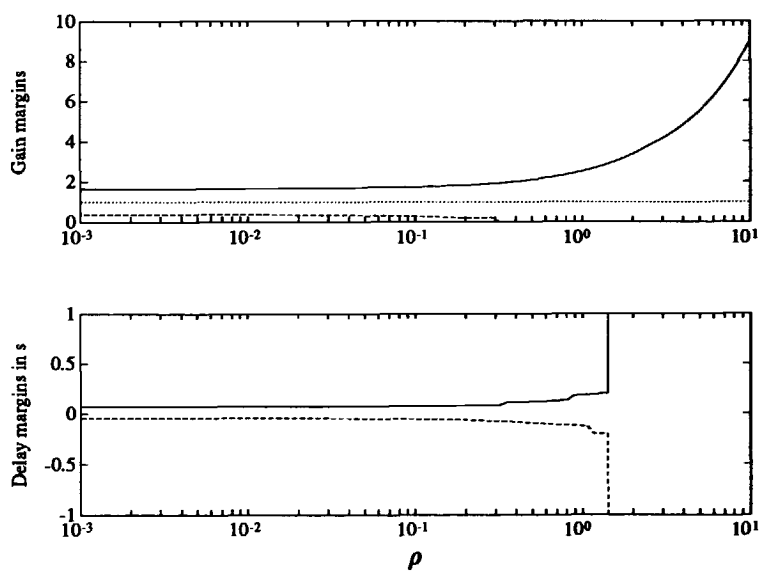

Figure 3.36: Upper gain and delay margin (solid line) and the lower gain and delay margin (dashed line) as a function of ρ where $H_p = 25$ and $H_c = 2$.

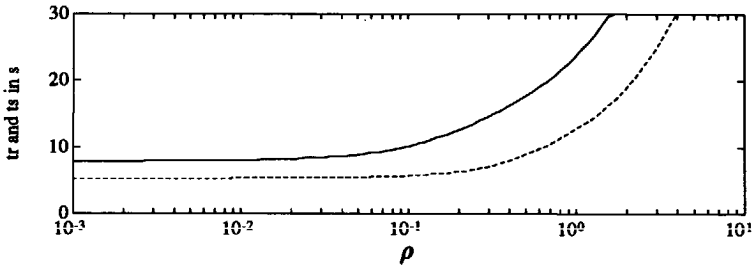

Figure 3.37: Rise (dashed line) and settling time (solid line) as a function of ρ where $H_p = 25$ and $H_c = 2$.

The prediction horizon and unstable processes

As a rule of thumb, the prediction horizon can be chosen according to the 5% settling time of the step response of the process. However, when the process is unstable this step response does not settle. Ideally, H_p must be related to the settling time of the step response of the closed-loop system for $H_p \rightarrow \infty$, as discussed earlier in this section. However, as a rule of thumb, it has been argued that, if the process is badly damped or unstable, H_p can also be related to the rise time of the process' step response: $H_p = \text{int}(3t_r/T_s)$.

The following example shows how the prediction horizon affects the robustness of the closed-loop system if the process has poles on the imaginary axis.

Example 3.14 The influence of H_p and H_c on the robustness of the closed-loop system in controlling process with poles on the imaginary axis.

Settings: same as in Example 3.10 but now $H_c = 1, 2, 3$

Process:
$$H(s) = \frac{4}{(s+1)(s^2+4)} \quad T_s = 0.2s \Longrightarrow$$
$$H(z^{-1}) = \frac{5.036 \cdot 10^{-3} z^{-1}(1+3.5207z^{-1})(1+0.2571z^{-1})}{1 - 2.6609z^{-1} + 2.5082z^{-2} - 0.8187z^{-3}} \quad (3.31)$$

Model: identical to process

Parameters: $H_c = 1, 2, 3$ and $H_c \leq H_p \leq 25$

The process 3.31 has two complex poles on the unit circle and an unstable inverse. Therefore, the prediction horizon should be chosen larger than the control horizon because otherwise minimum-variance control is obtained, resulting in an unstable closed-loop system. The step response of the process is shown in Figure 3.38. In order to examine the robustness of the closed-loop system, the gain and delay margins

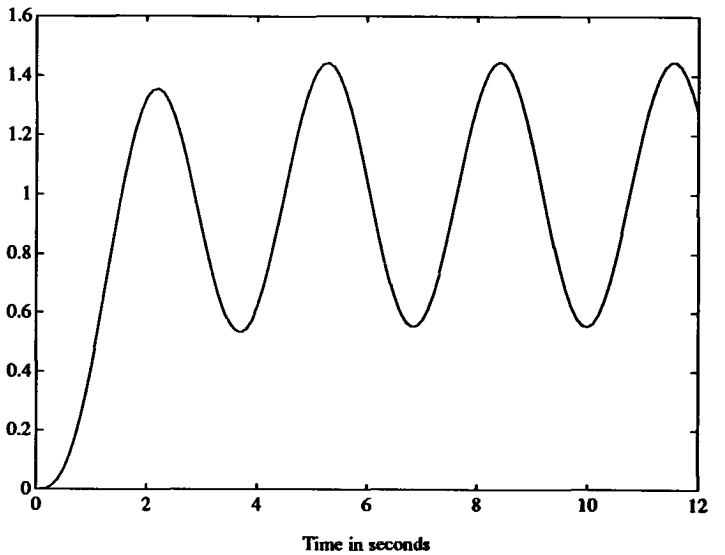

Figure 3.38: Step response of the process with poles on the imaginary axis.

are calculated as a function of H_p for $H_c = 1, 2$ and 3. The results are shown in the Figures 3.39, 3.40 and 3.41, respectively. These figures show clearly that the robustness criteria for $H_c = 2, 3$ settle for $H_p > 17$. By using the rule of thumb mentioned above, approximately the same result is obtained: $H_p = \text{int}(3t_r/T_s) = 20$. When $H_c = 1$, the robustness criteria show rather peculiar behavior. This can be explained by the fact that for $H_c = 1$ and H_p large, mean-level control is approximated which yields an unstable closed-loop system.

The next example shows what happens if the process to be controlled has poles in the right half plane.

Example 3.15 The influence of H_p and H_c on the robustness of the closed-loop system in controlling an unstable process.

Settings: same as in Example 3.14

Process:
$$H(s) = \frac{4.04}{(s+1)(s^2 - 0.4s + 4.04)} \quad \overset{T_s = 0.3s}{\Longrightarrow}$$
$$H(z^{-1}) = \frac{1.710 \cdot 10^{-3} z^{-1}(1 + 3.505z^{-1})(1 + 0.2611z^{-1})}{1 - 2.4936z^{-1} + 2.4260z^{-2} - 0.8353z^{-3}} \quad (3.32)$$

Model: identical to process

Parameters: $H_c = 1, 2, 3$ and $H_c \leq H_p \leq 25$

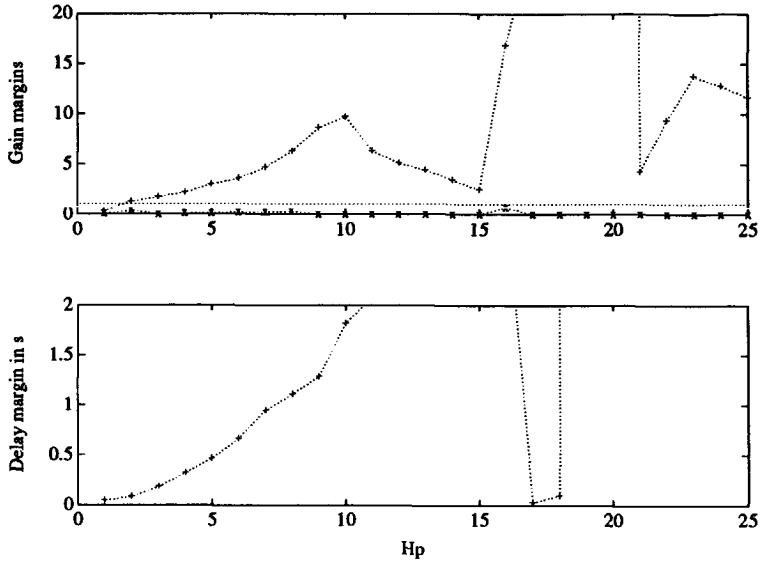

Figure 3.39: Gain and delay margins as a function of H_p where $H_c = 1$.

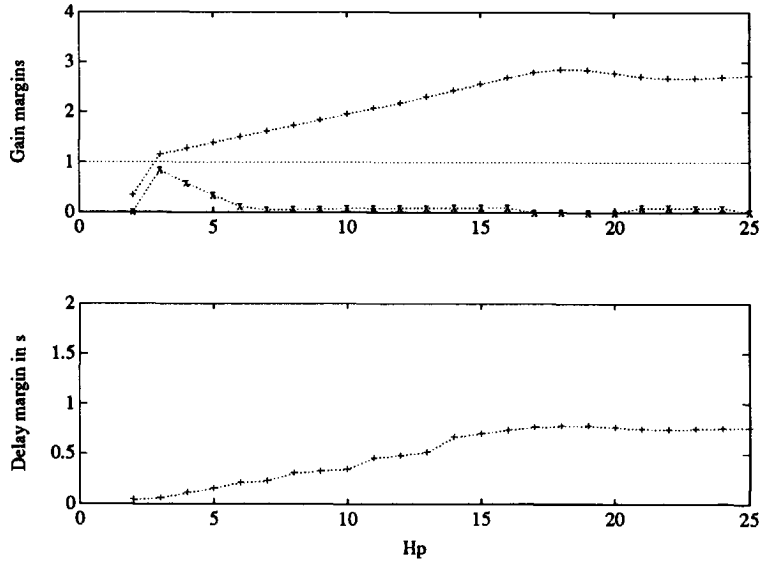

Figure 3.40: Gain and delay margins as a function of H_p where $H_c = 2$.

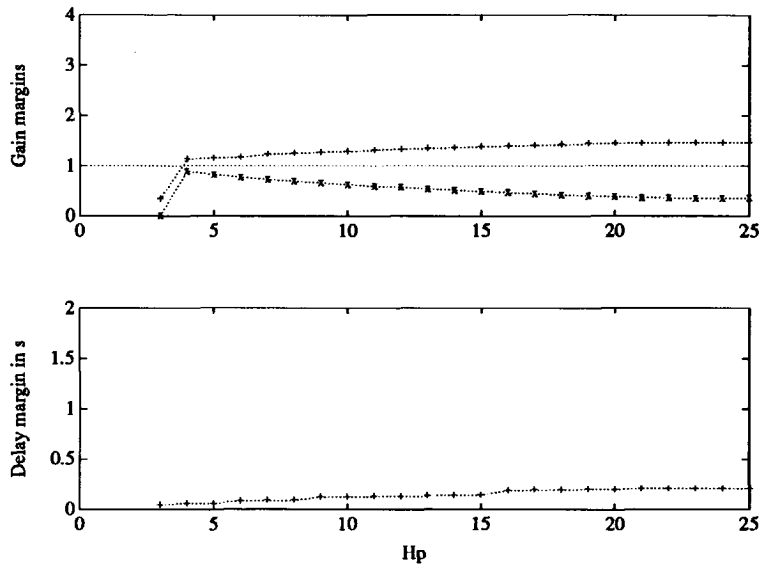

Figure 3.41: Gain and delay margins as a function of H_p where $H_c = 3$.

The continuous process has its poles at -1 and $0.2 \pm 2j$. Further, the discrete process has an unstable inverse. The step response of the process (3.32) is shown in Figure 3.42. Analysis of the robustness as a function of H_p with $H_c = 1$ showed that the closed-loop system is unstable for $H_p > 7$. Further, it is unstable if $H_p = 1$, owing to the fact that the process has an unstable inverse. The results for $H_c = 2$ and 3 are shown in the Figures 3.43 and 3.44, respectively. Figure 3.43 elucidates that for $H_c = 2$ the robustness criteria tend to settle for $H_p > 20$. For $H_p > 2$ the closed-loop system is stable. When $H_c = 3$, the robustness criteria settle for $H_p > 14$. However, the robustness is smaller than when $H_c = 2$. If the above-mentioned rule of thumb would have been used to select H_p, this would have resulted in $H_p = \text{int}(3t_r/T_s) = 14$.

Finally, a step response of the closed-loop system is shown in Figure 3.45. This step response is calculated using $H_p = 20$ and $H_c = 3$. In the above simulation, the robustness criteria are: $\underline{gm} = 0.32$, $\overline{gm} = 1.5$ and $\overline{dm} = 0.14$s. The rise and settling time are: $t_r = 2.4$s and $t_s = 3.6$s. The overshoot and steady-state error are zero. The robustness is rather poor. However in section 3.5 it is shown that the noise model can be used to improve the robustness without affecting the servo behavior of the closed-loop system.

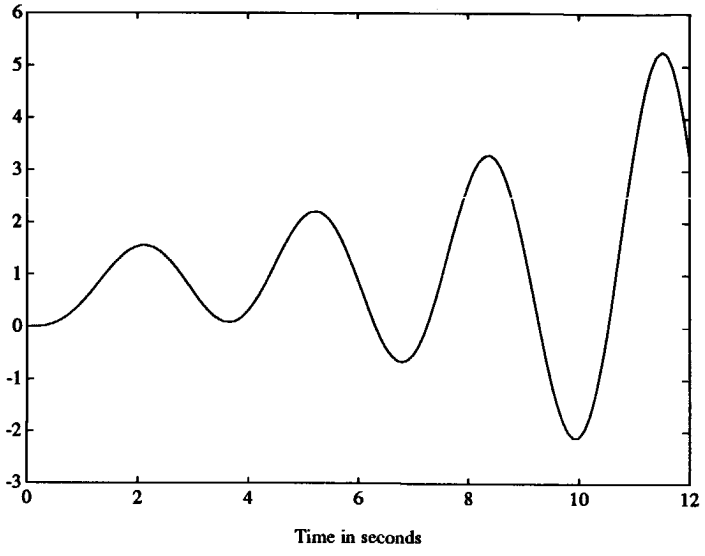

Figure 3.42: Step response of the unstable process (3.32).

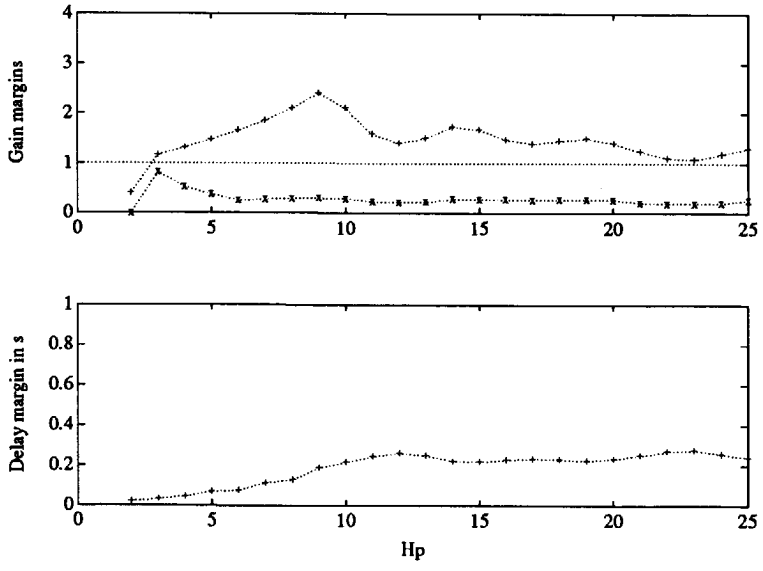

Figure 3.43: Gain margins and upper delay margin as a function of H_p where $H_c = 2$.

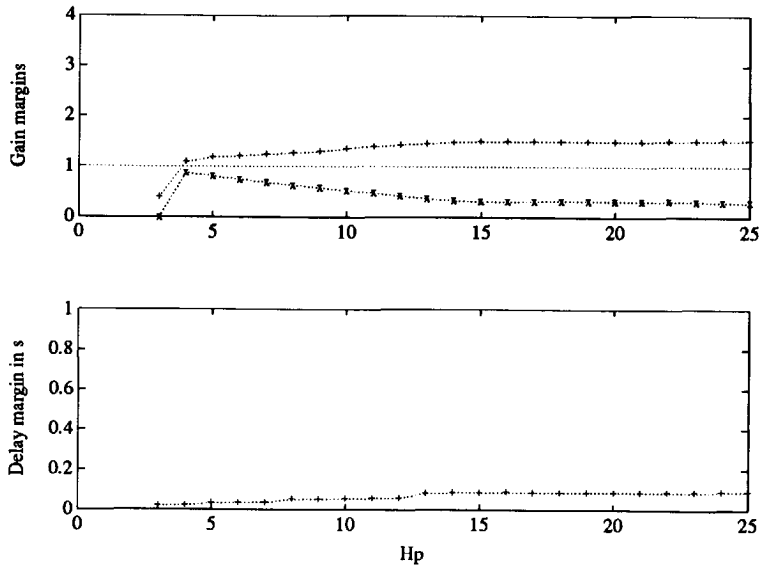

Figure 3.44: Gain margins and upper delay margin as a function of H_p where $H_c = 3$.

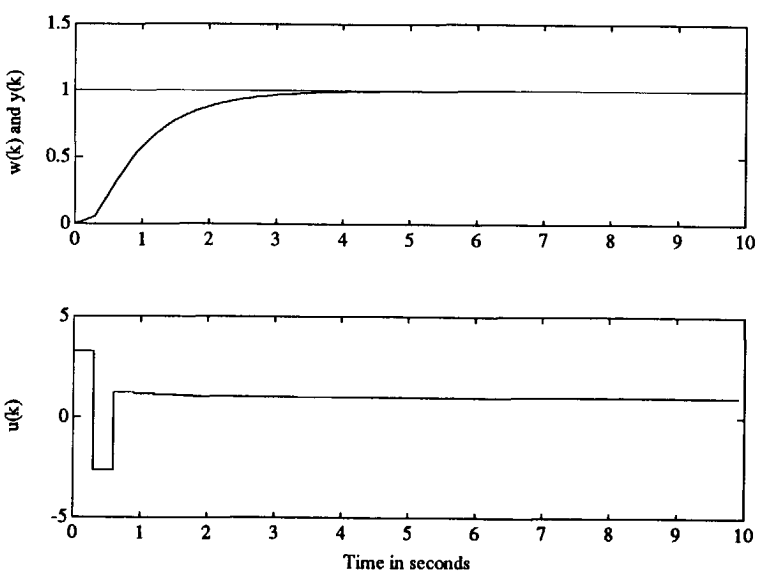

Figure 3.45: Step response of the closed-loop system with $H_p = 20$, $H_c = 3$ and an unstable process.

3.4.2 The Minimum Cost Horizon

As has already been discussed in section 2.2.2, increasing the minimum cost horizon in the case of a minimum phase process is expected to make the closed-loop system respond more slowly to set point changes. Because of this, also an increase of robustness can be expected. The following example shows that this is the case.

Example 3.16 The influence of H_s and H_c on the performance and robustness of the closed-loop system.

Settings: $H_p = 10$, $\beta = 1$, $N = P = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 10^{-4}$, $Q_n = \Delta$, $Q_d = 1$

Process: $H(s) = \dfrac{5}{(4s+1)(5s+1)}$ $\overset{T_s = 0.5s}{\Longrightarrow}$

$$H(z^{-1}) = \frac{0.029z^{-1}(1 + 0.928z^{-1})}{(1 - 0.882z^{-1})(1 - 0.905z^{-1})}$$

Model: identical to process

Parameters: $H_s = 1, \ldots, 10$ and $H_c = 1, 2, 3$

For $H_s = 1, \ldots, 9$ and $H_c = 1$, the performance criteria t_r and t_s and the robustness criteria gm and dm hardly change. Taking $H_s = 10 = H_p$ does show a small, though not spectacular, change in these criteria. The performance criteria t_r and t_s and the robustness criteria gm and dm are shown in Table 3.3. This table also shows the criteria when $H_c = 2$. Also in this case, the criteria for $H_s = 1, \ldots, 9$ hardly change. However, taking $H_s = 10 = H_p$ now makes the closed-loop system considerably

H_c	H_s	t_r in s	t_s in s	gm	dm in s
1	1	6.5	8	13.3	3.7
1	10	8	11	17.2	6.5
2	1	2.5	3.5	2.6	0.5
2	10	8	10.5	17.5	6.6

Table 3.3: Performance and robustness criteria as a function of H_s and H_c.

slower and simultaneously much more robust.

Note: if $H_c > 1$ a unique solution to the optimization problem is not obtained when $H_s > 1$ and $\rho = 0$. For this reason, a small value for ρ ($\rho = 10^{-4}$) is used in combination with $Q_n = \Delta$ in order to avoid steady-state errors.

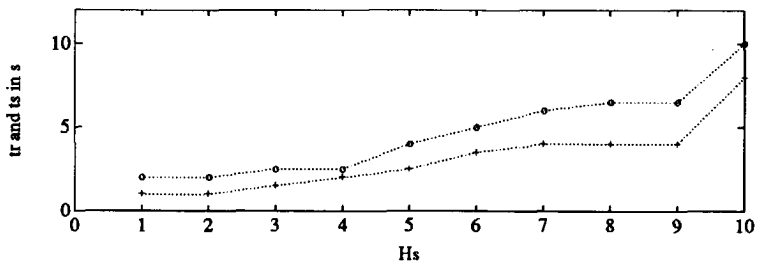

Figure 3.46: Rise ('+') and settling time ('o') as a function of H_s where $H_c = 3$.

As is shown by the Figures 3.46 and 3.47, the performance and robustness criteria increase continuously as H_s increases and $H_c = 3$.

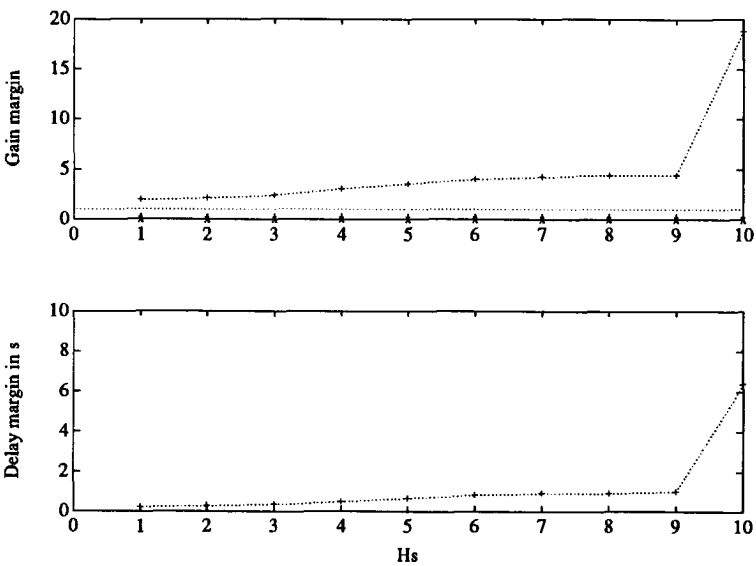

Figure 3.47: Gain and delay margin as a function of H_s where $H_c = 3$.

Thus far, the influence of H_s on the closed-loop system in the case of an overdamped minimum phase process has been examined. That, in the case of a non-minimum phase process and some particular settings for H_p and H_c, increasing H_s can make the system respond faster to set point changes has already been shown in section 2.2.2. The next example shows what the effect of H_s is in the case of an underdamped

non-minimum phase process for other values of H_c.

Example 3.17 The influence of H_s and H_c on the performance of the closed-loop system in the case of a non-minimum phase process.

Settings: $H_p = 25,\ \beta = 1,\ N = P = 1,\ R = 1,\ T = 1,\ \hat{D} = 1,\ \underline{d} = 0,\ \overline{d} = 0,\ \rho = 10^{-4},\ Q_n = \Delta,\ Q_d = 1$

Process: $H(s) = \dfrac{-4(s-1)}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$

$$H(z^{-1}) = \frac{-0.5954z^{-1}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}}$$

Model: identical to process

Parameters: $H_s = 1, \ldots, 25$ and $H_c = 1, 2$

When $H_c = 1$, simulations have shown that the closed-loop system is hardly affected by the choice of H_s. For all values for H_s, the controller is approximately equal to a mean-level controller as is shown in section 2.2.2, page 61. As a result, the step response of the closed-loop system is identical to that of the process (see Figure 3.31). The performance criteria as a function of H_s when $H_c = 2$ are shown in Figure 3.48. The figure shows the quite remarkable behavior of t_r and t_s. Increasing

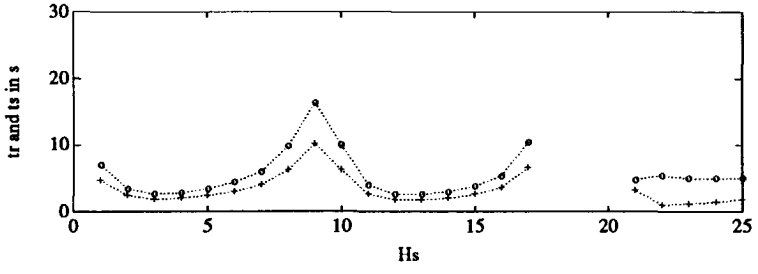

Figure 3.48: Rise ('+') and settling time ('o') as a function of H_s where $H_c = 2$.

H_s from 1 to 3, makes the system faster. Further increasing H_s makes the system respond slower to set point changes. When $H_s = 18, \ldots, 20$, t_r and t_s are not defined because for these values of H_s the closed-loop system is unstable. This is clearly illustrated in Figure 3.49 which shows a root-locus plot as a function of H_s. The root locus also shows that for $H_s = 22, \ldots, 25$, the closed-loop poles are badly damped resulting in a large overshoot ($> 30\%$). Only in the case $H_s = 21$ is a good response obtained as is shown in Figure 3.50. That the behavior of the closed-loop system is very sensitive to the choice of H_s is obvious. Choosing $H_s = 20$ yields

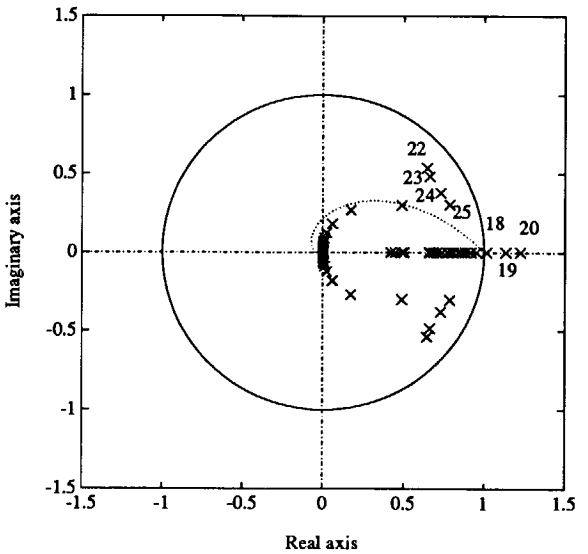

Figure 3.49: Root locus as a function of H_s where $H_s = 1, \ldots, 25$ and $H_c = 2$.

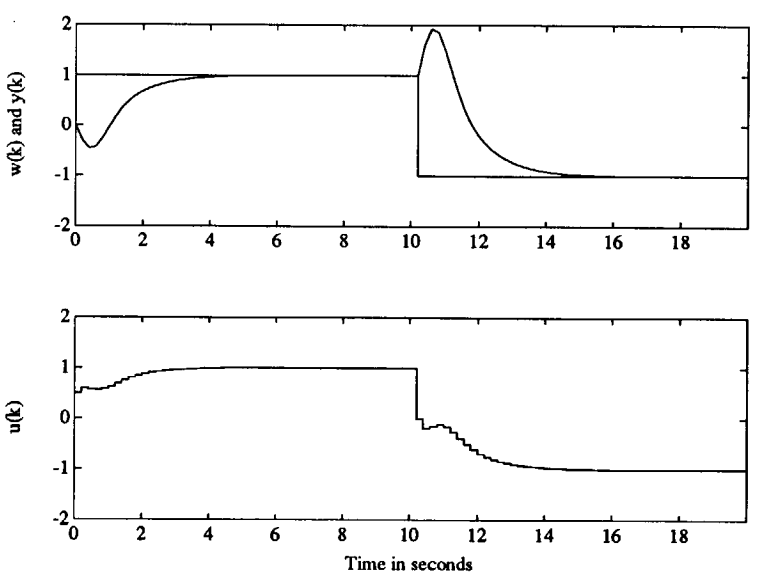

Figure 3.50: Response when $H_s = 21$ and $H_c = 2$.

an unstable closed-loop system. Choosing $H_s = 22$ yields a closed-loop system in which the step response shows an overshoot of 101%.

To conclude: By increasing H_s, the tracking error in the near future is not taken into account in the criterion function. Especially in the case of a non-minimum phase process this can result in unpredictable behavior of the closed-loop system when H_s is changed. A small change in H_s can even make the closed-loop system move from nicely damped to unstable or vice versa. Therefore, it is advised that H_s is not used as a tool to tune the closed-loop system and, therefore, H_s should be chosen equal to $\underline{d} + 1$ or according to the theorems stated in the previous chapter.

The prediction horizon when $H_s = H_p$

In this section, the effect of H_p in the case $H_s = H_p$ is discussed. This choice for H_p is used in the EHAC controller. Here the combination with $H_c = 1$ is discussed (one of the possible strategies to realize a unique solution to the optimization problem, see [11]). In the following example the effect of the prediction horizon in the absence of controller output weighting by means of ρ is shown.

Example 3.18 The influence of H_p on the performance and robustness of the closed-loop system when $H_s = H_p$.

Settings:	$H_c = 1$, $\beta = 1$, $N = P = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)} \quad T_s = 0.5s \Longrightarrow$ $H(z^{-1}) = \dfrac{0.029z^{-1}(1+0.928z^{-1})}{(1-0.882z^{-1})(1-0.905z^{-1})}$
Model:	identical to process
Parameters:	$H_p = 1, \ldots, 25$ and $H_s = H_p$

In this example a nicely damped minimum phase process is controlled. Figure 3.51 shows the rise and settling time as a function of H_p. The overshoot is for all H_p smaller than 4%. Obviously, increasing H_p makes the closed-loop system respond more slowly to set point changes. This is caused by the fact that if $H_p \rightarrow \infty$ and $H_c = 1$ a mean-level controller is obtained which makes the dynamics of the closed-loop system coincide with the dynamics of the process. The robustness of the system also increases as H_p increases.

In the case of a non-minimum phase process, the effect of H_p on the system is rather different as is shown in the following example.

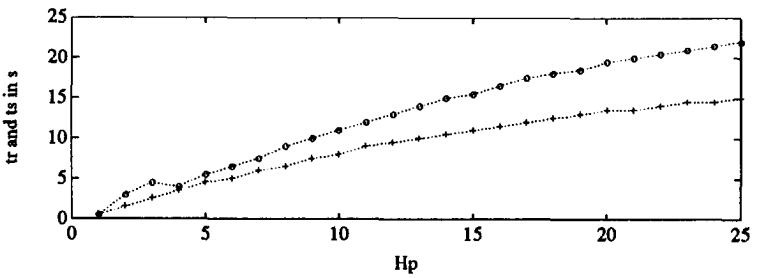

Figure 3.51: Rise ('+') and settling time ('o') as a function of H_p where $H_s = H_p$.

Example 3.19 The influence of H_p on the performance of the closed-loop system in the case of a non-minimum phase process and $H_s = H_p$.

Settings: $\beta = 1$, $N = P = 1$, $R = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$

Process: $H(s) = \dfrac{-4(s-1)}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$

$$H(z^{-1}) = \frac{-0.5954z^{-1}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}}$$

Model: identical to process

Parameters: $H_p = 1, \ldots, 25$ and $H_s = H_p$

The effect of H_p on the rise and settling time and the overshoot is shown in Figure 3.52. For $H_p \leq 5$, the closed-loop system is unstable. For $H_p \geq 6$, the closed-loop system is stable. However, the overshoot is rather large for $H_p \geq 9$. Only when $H_p = 6, 7$ or 8 are acceptable results obtained. This example shows that it is quite difficult to choose H_p if $H_s = H_p$ and $H_c = 1$ while this was not the case if $H_s = \underline{d} + 1$ as has been shown in the previous sections.

3.4.3 Conclusions

In this section it has been shown that the prediction, control and minimum cost horizon can be used to select a particular controller. Further, the influence of these parameters on the stability robustness and servo performance of the closed-loop system has been illustrated by a number of examples of several processes. Based on the theorems derived in Chapter 2 and the examples shown in this section, the

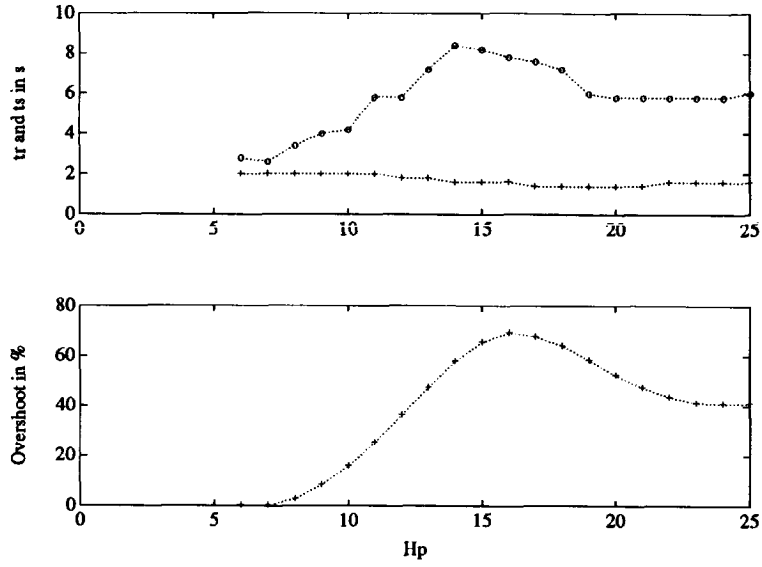

Figure 3.52: Rise ('+'), settling time ('o') and overshoot as a function of H_p where $H_s = H_p$.

following conclusions can be drawn:

- By choosing $H_p \geq n_A + n_B + d + 1$, $H_s = n_B + d + 1$, $H_c = n_A + 1$ and $\rho = 0$, a pole-placement controller is obtained. The desired pole locations are defined by P. By a proper selection of P, minimum-variance, dead-beat, moving-average and mean-level controllers can be selected (see Table 3.2). Further, for these particular settings for H_p, H_s, H_c and ρ, the closed-loop system is independent of H_p and hence the robustness and performance do not depend on H_p.

- If the process is stable, $H_s = \underline{d} + 1$ and $H_c = 1$; increasing the prediction horizon improves the robustness of the closed-loop system and simultaneously makes the system respond slower to set point changes. This behavior is caused by the fact that for $H_p \rightarrow \infty$, a mean-level controller is obtained.

- If the closed-loop system is stable, the prediction horizon can be chosen equal to $\text{int}(t_{s,\infty}/T_s)$ where $t_{s,\infty}$ is the settling time of the step response of the closed-loop system for $H_p \rightarrow \infty$. and int(.) is a function that converts a real value to an integer. If the process is stable and $H_c > 1$, a rule of thumb on how

to choose the prediction horizon is: $H_p = \text{int}(t_s(5\%)/T_s)$, where $t_s(5\%)$ is the 5% settling time of the process' step response. Thus, the closed-loop system is hardly sensitive to changes in H_p. Further, H_p depends on the sampling period. A short sampling period results in a large prediction horizon. However, if, as a rule of thumb, the sampling period is chosen $10 - 20$ times smaller than the settling time of the closed-loop system's step response, typical values for H_p are $10 - 20$.

- If the process is stable but has badly situated zeros, then H_c must be chosen $\leq n_A + 1$. The smaller H_c, the slower the closed-loop system's step response and the larger its robustness.

- If the process is badly damped or unstable, H_c should be chosen > 1. Further, as a rule of thumb, H_p can be chosen according to: $H_p = \text{int}(3t_r/T_s)$, where t_r is the rise time of the process.

  The resulting robustness is quite small compared to the robustness that can be obtained in controlling stable processes. Obviously, in controlling an unstable process an accurate model is required.

- The minimum cost horizon H_s can be chosen equal to $\underline{d} + 1$ or if a dead-beat, moving-average or pole-placement controller is required, equal to $n_B + d + 1$ (see Table 3.2). Choosing other values for H_s is not advised.

- The design parameter β must be chosen as small as possible in order to avoid stability problems. However, the condition (2.205) must always be satisfied if $H_c \neq H_p - \underline{d}$ in order to prevent steady-state errors due to H_c.

- In all examples, choosing $H_c = n_A$ has shown to yield a compromise between robustness and performance.

3.5 The Noise Model

3.5.1 The Noise Model and the Stability Robustness of the Closed-loop System - Theoretical Results

It was pointed out in section 2.1.5 that the effect of modeling errors and disturbances on the prediction error can be influenced by, among others, the polynomials T and $\hat{D}$ which realize, together with $\hat{A}$, the noise model. Therefore, it can be expected that these polynomials have a great influence on the robustness and regulator behavior of the closed-loop system.

In order to examine this influence, recall the stability result mentioned in section 3.2: If the nominal system is stable and the process is stable, then the true system is stable if:

$$|H_L - \hat{H}_L| < |1 + \hat{H}_L| \qquad \forall \omega \geq 0 \tag{3.33}$$

where H_L and $\hat{H}_L$ are the loop transfer functions of the true and the nominal system, respectively:

$$H_L = \frac{q^{-d-1}BS}{A\mathcal{R}} = H\frac{S}{\mathcal{R}} \tag{3.34}$$

$$\hat{H}_L = \frac{q^{-\hat{d}-1}\hat{B}S}{\hat{A}\mathcal{R}} = \hat{H}\frac{S}{\mathcal{R}} \tag{3.35}$$

where H and $\hat{H}$ are the transfer functions of the process and model, respectively.

In robustness analysis, often two types of modeling errors are considered: additive and multiplicative modeling errors. By definition, the following relation between process and model transfer function exists:

$$H = \hat{H} + \Delta_a = \hat{H}(1 + \Delta_m) \tag{3.36}$$

where Δ_a and Δ_m are transfer functions that represent the additive and multiplicative modeling errors, respectively. In this section, additive modeling errors are discussed only. However, the results can also be used for multiplicative modeling errors by using the relationship (3.36). Rewriting (3.33) using (3.34) and (3.35) yields:

$$|H - \hat{H}| < |1 + \hat{H}_L| \left| \frac{\mathcal{R}}{S} \right| \qquad \forall \omega \geq 0 \tag{3.37}$$

By using (3.36), equation (3.37) becomes:

$$|\Delta_a| < \left| \frac{\hat{A}\mathcal{R} + q^{-\hat{d}-1}\hat{B}S}{\hat{A}S} \right| = \frac{1}{|\hat{H}_{\xi u}|} \qquad \forall \omega \geq 0 \tag{3.38}$$

where $\hat{H}_{\xi u}$ is the transfer function from $\xi(k)$ to $u(k)$ for the nominal system.

Hence, if the nominal system is stable and the process is stable, then the closed-loop system is stable if the additive modeling errors satisfy (3.38). The right-hand side of (3.38) is fully determined by the model and the design parameters of the UPC controller. If the process is stable and if an upper bound on the modeling errors is known, hence if:

$$|\Delta_a| < |H_{max}| \quad \forall \omega \geq 0 \tag{3.39}$$

then, in order to guarantee stability of the closed-loop system, the UPC design parameters must be chosen such that:

$$|H_{max}| < \left| \frac{\hat{A}\mathcal{R} + q^{-\hat{d}-1}\hat{B}S}{\hat{A}S} \right| \qquad \forall \omega \geq 0 \tag{3.40}$$

Because the relation between the polynomials $\mathcal{R}$ and S and the UPC design parameters is quite complicated, it is almost never possible to calculate these parameters such that (3.40) is satisfied. Hence, when the design is done, one must verify whether or not (3.40) is satisfied. In robust control design methods (see e.g. [41, 17]), knowledge of $|H_{max}|$ is incorporated into the design of the controller. As a result, the true system is guaranteed to be stable if $|\Delta_a| < |H_{max}| \quad \forall \omega \geq 0$. However, in some special cases, knowledge about $|H_{max}|$ can be incorporated in the UPC controller design too.

Robustness of generalized minimum-variance controllers

In the case of generalized minimum-variance control, the polynomials $\mathcal{R}$ and S are given by (2.102) and (2.103):

$$\mathcal{R} = \frac{\hat{B}\hat{D}}{\hat{b}_0} \tag{3.41}$$

$$S = \frac{q(PT - \hat{A}\hat{D})}{\hat{b}_0} \tag{3.42}$$

and (3.38) becomes (remember that $\hat{d} = 0$):

$$|\Delta_a| < |\hat{H}| \frac{1}{\left|1 - \frac{\hat{A}\hat{D}}{PT}\right|} \qquad \forall\omega \geq 0 \tag{3.43}$$

Now, P, T and $\hat{D}$ can be used to shape the function on the right-hand side of (3.43). Note, that when $PT = \hat{A}\hat{D}$, the closed-loop system is stable for all modeling errors if the process is stable. This can be explained by the fact that there is now no feedback ($S = 0$, see Theorem 3.3). Then, the closed-loop system will always be stable if the process is stable. By choosing PT 'close' to $\hat{A}\hat{D}$ the stability robustness can be made large. Further, (3.43) shows that P has the same effect on the robustness of the closed-loop system as T has.

Robustness of pole-placement controllers

In the case of pole-placement controllers (see Table 3.2), the characteristic equation of the nominal system is given by:

$$\hat{A}R + q^{-\hat{d}-1}\hat{B}S = PT \tag{3.44}$$

Now, (3.38) can be written as:

$$|\Delta_a| < \left|\frac{PT}{\hat{A}S}\right| \qquad \forall\omega \geq 0 \tag{3.45}$$

S in (3.45) depends, among others, on $\hat{D}$, P and T and still knowledge of $|H_{max}|$ cannot be incorporated in the controller design. However, in the case of mean-level control and $\hat{D} = \Delta$, S can be written such that knowledge of $|H_{max}|$ can be incorporated into the design.

Robustness of mean-level controllers

When mean-level control is selected, $P = \hat{A}$ in (3.44) (see, for example, Table 3.2):

$$\hat{A}\mathcal{R} + q^{-\hat{d}-1}\hat{B}S = \hat{A}T \tag{3.46}$$

From (3.46) follows that $\hat{A}$ is a factor of S. Rewriting (3.46) yields:

$$\mathcal{R} + q^{-\hat{d}-1}\hat{B}S_1 = T \tag{3.47}$$

where: $S = \hat{A}S_1$. Further, it was shown in section 2.3.3 that if $\hat{D} = \Delta$, $\rho = 0$ and $\beta = 1$, Δ is a factor of $\mathcal{R}$. For mean-level control, $\rho = 0$ and $\beta = 1$ (see Table 3.2). Further, assume that $\hat{D} = \Delta$. Now, (3.47) can be rewritten into:

$$\Delta\mathcal{R}_1 + q^{-\hat{d}-1}\hat{B}S_1 = T \tag{3.48}$$

where: $\mathcal{R} = \Delta\mathcal{R}_1$. The minimum-degree solution to (3.48) yields $n_{R_1} = n_B + \hat{d}$ and $n_{S_1} = 0$ if $n_T \leq n_B + \hat{d} + 1$. Hence, S_1 is a scalar. Posing $q = 1$ in (3.48) yields:

$$S_1 = \frac{T(1)}{\hat{B}(1)}$$

and hence:

$$S = \frac{\hat{A}(q^{-1})T(1)}{\hat{B}(1)} \tag{3.49}$$

Using (3.49), equation (3.45) becomes:

$$|\Delta_a| < \left|\frac{\hat{B}(1)T(q^{-1})}{\hat{A}(q^{-1})T(1)}\right| = \hat{K}_{dc}\left|\frac{\hat{A}(1)T(q^{-1})}{\hat{A}(q^{-1})T(1)}\right| \qquad \forall\omega \geq 0 \tag{3.50}$$

Equation (3.50) shows clearly that if an upper bound on the additive modeling error is known, this knowledge can be used to select T such that the closed-loop system is stable for all modeling errors satisfying (3.39).

Remark: (3.50) is valid only if $n_T \leq n_B + \hat{d} + 1$. For an ARIMAX model ($n_B = n_A - 1$) with $\hat{d} = 0$ this yields: $n_T \leq n_A$. Hence, increasing the robustness for high frequencies by using a T polynomial with a degree larger than n_A in (3.50) is not possible. In the case $T = \hat{A}$, (3.50) becomes:

$$|\Delta_a| < \hat{K}_{dc} \qquad \forall \omega \geq 0 \tag{3.51}$$

The transfer function of the nominal system (2.160) is in this case:

$$y(k) = \frac{q^{-\hat{d}-1}\hat{B}(q^{-1})\hat{A}(1)}{\hat{A}(q^{-1})\hat{B}(1)} Sp + \frac{\mathcal{R}}{\hat{A}}\xi(k) \tag{3.52}$$

Now, the robustness can obviously be increased simply by increasing the DC gain of the model. However, note that by changing the DC gain of the model $|\Delta_a|$ is changed too. $|\Delta_a|$ for frequency zero is given by:

$$|\Delta_a| = |K_{dc} - \hat{K}_{dc}| \quad \text{if } \omega = 0$$

If $\hat{K}_{dc} \geq K_{dc}$, then (3.51) is always satisfied for $\omega = 0$. However, if $\hat{K}_{dc} < K_{dc}$, then (3.51) can be satisfied only if $\hat{K}_{dc} > 0.5K_{dc}$.

Robustness of dead-beat controllers

In the case of dead-beat control $P = 1$ (see Table 3.2). Hence, the CLCE is given by:

$$\hat{A}\mathcal{R} + q^{-\hat{d}-1}\hat{B}S = T$$

Obviously, by choosing $T = \hat{A}T_1$ and $\hat{D} = \Delta$, the closed-loop characteristic equation of the dead-beat controller is identical to that of the mean-level controller with $T = T_1$ and $\hat{D} = \Delta$ (see (3.46)). Then, S is given by (3.49) with $T(1)$ replaced by $T_1(1)$:

$$S = \frac{\hat{A}(q^{-1})T_1(1)}{\hat{B}(1)} \tag{3.53}$$

and an upper bound on the additive modeling error is given by (3.50) with T replaced by T_1:

$$|\Delta_a| < \hat{K}_{dc}\left|\frac{\hat{A}(1)T_1(q^{-1})}{\hat{A}(q^{-1})T_1(1)}\right| \qquad \forall \omega \geq 0 \tag{3.54}$$

Hence, the robustness of the dead-beat controller with $T = \hat{A}$ and $\hat{D} = \Delta$ is identical to that of the mean-level controller with $T = 1$. Moreover, by choosing $T = \hat{A}^2$, the robustness properties of the dead-beat controller coincide with those of the mean-level controller with $T = \hat{A}$. The transfer function of the nominal system (2.160) is in this case:

$$y(k) = \frac{q^{-d-1}\hat{B}(q^{-1})}{\hat{B}(1)} Sp + \frac{\mathcal{R}}{\hat{A}}\xi(k)$$

and disturbances are rejected in the same way as they are rejected in mean-level control with $T = \hat{A}$ (see (3.52)). However, the servo behavior of the closed-loop system is still that of dead-beat control because T and $\hat{D}$ do not influence the nominal closed-loop system (see Theorem 2.11).

Remark: the same results have been obtained by Clarke [43] using other methods.

3.5.2 The Noise Model and its Influence on the Robustness and Regulator Behavior

In the previous section, stability robustness was considered for some particular settings of the UPC design parameters. However, the polynomials T and $\hat{D}$ also have an influence on the regulator behavior of the closed-loop system. If knowledge about the noise spectrum is available, the noise model $T/\hat{D}\hat{A}$ can be chosen according to this spectrum. For example, if the noise measured on the process output (= $\xi(k)$) is white, then T can be chosen equal to $\hat{D}\hat{A}$. When the C and D polynomials of the process are available (they may be estimated by a suitable identification method), then T and $\hat{D}$ can be chosen equal to C and D respectively. An example of incorporating knowledge of the noise spectrum in the UPC design is discussed in Chapter 5.

In order to avoid steady-state errors at the sampling instants, $\hat{D}$ should at least contain $\max(p, r) - n$ factors Δ (p and r are the type of the disturbance and the reference trajectory and n is the number of integrators in the process).

In this section, some examples are shown which illustrate the influence of T on the robustness and regulator behavior of the closed-loop system.

Example 3.20 The influence of T on the robustness of the closed-loop system.

Settings:	$H_p = 15,\ H_s = 1,\ H_c = 3,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = 1,$ $\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)} \quad T_s = 0.5s \Longrightarrow$ $H(z^{-1}) = \dfrac{0.029z^{-1}(1+0.928z^{-1})}{(1-0.882z^{-1})(1-0.905z^{-1})}$
Model:	identical to process
Parameters:	$T = (1-0.9q^{-1})(1-\mu q^{-1})$ where $0 \leq \mu \leq 1$

In the example, T is structured as $T = (1-0.9q^{-1})(1-\mu q^{-1})$. The variable μ is varied between 0 and 1. This choice for T seems rather peculiar, but will be explained later on. The upper gain and delay margin as a function of μ are shown in Figure 3.53 (the lower gain margin is equal to zero and the lower delay margin is not of interest because the process and model do not contain time delay). The figure shows that both the upper gain and delay margin increase as μ increases. For $\mu = 0.88$, they are both infinite. Increasing μ beyond $\mu = 0.88$ makes the gain margin decrease. The robustness criteria being infinite for $\mu = 0.88$ can be explained by applying the following theorem:

Theorem 3.5 *If* $PT = \hat{A}\hat{D}$, *then* $S = 0$.

Proof. If $PT = \hat{A}\hat{D}$, then solving $\overline{E}_i$ and $\overline{F}_i$ from

$$\frac{PT}{\hat{A}\hat{D}} = \overline{E}_i + q^{-i}\frac{\overline{F}_i}{\hat{A}\hat{D}}$$

yields: $\overline{E}_i = 1$ and $\overline{F}_i = 0\ \forall i \geq 1$. The MV i-step-ahead predictor is now:

$$P\hat{y}(k+i) = \frac{\hat{B}\hat{D}}{T}u(k+i-1) \qquad i \geq 1$$

By using the assumption $PT = \hat{A}\hat{D}$, this predictor becomes:

$$\hat{y}(k+i) = \frac{\hat{B}}{\hat{A}}u(k+i-1) \qquad i \geq 1$$

The predictions of the process output do not depend on $y(k), y(k-1), \cdots$. Hence the criterion function (2.158) does not depend on $y(k), y(k-1), \cdots$. Therefore, the optimization problem and hence its solution ($= u(k)$) do not depend on $y(k)$, $y(k-1)$, $\cdots$. Finally, if $u(k)$ does not depend on $y(k), y(k-1), \cdots$, then $S = 0$.

□

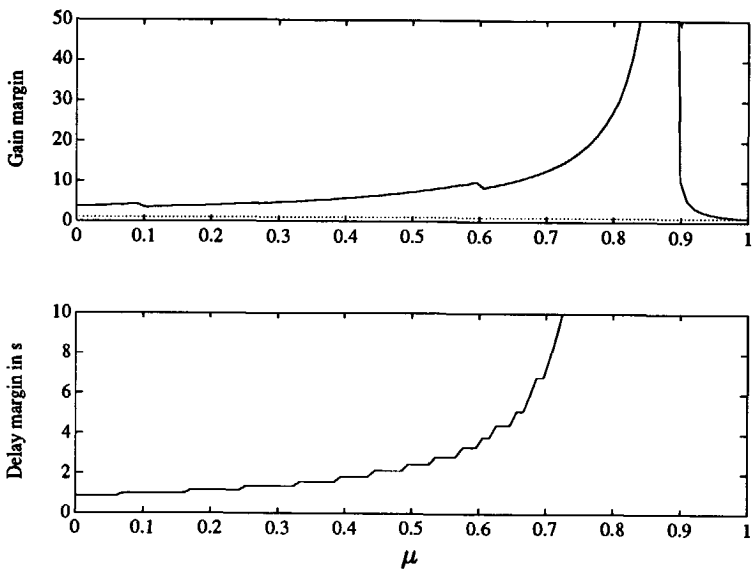

Figure 3.53: Gain and delay margins as a function of μ where $T = (1-0.9q^{-1})(1-\mu q^{-1})$ and $\hat{D} = 1$.

If $\mu = 0.88$, then PT is equal to $\hat{A}\hat{D}$ and by applying Theorem 3.5, $S = 0$. Then there is no longer feedback and the closed-loop system is stable if the process is stable. The gain and delay margin are in this case infinitely large.

As has already been discussed in section 2.3.3, the noise model does not affect the servo behavior of the closed-loop system if the process is correctly estimated. Thus, also when $T = \hat{A}\hat{D}$ the servo behavior is not affected. Because in this case feedback is no longer present, the desired servo behavior is achieved by cancellation of the poles of the process. This can easily be seen from the transfer function of the closed-loop system (2.192):

$$y(k) = \frac{q^{-d-1}BT}{AR + q^{-d-1}BS} w(k+H_p) + \frac{AR}{AR + q^{-d-1}BS} \xi(k) \tag{3.55}$$

In the case $P = 1$ and $T = \hat{D}\hat{A}$ then $S = 0$. Now, (3.55) becomes:

$$y(k) = \frac{q^{-d-1}B\mathcal{T}}{A\mathcal{R}}w(k + H_p) + \xi(k) \tag{3.56}$$

Because T is a factor of $\mathcal{T}$ (see (2.191)) and $T = \hat{A}\hat{D}$, $\hat{A}$ is a factor of $\mathcal{T}$ and hence the poles of the process are canceled if the process is correctly estimated.

The transfer function (3.56) shows that disturbances on the output of the process are neither suppressed nor amplified.

In practice the choice $PT = \hat{D}\hat{A}$ is not very realistic. Feedback is necessary in almost all applications. For example, integral action may be required in order to prevent steady-state errors. As shown before, integral action can be introduced by selecting $\hat{D} = \Delta$ and by selecting Q_n and β such that the conditions (2.204) and (2.205) are satisfied. Because this choice for $\hat{D}$ is the most commonly used (for example in GPC and DMC), subsequently the influence of the T polynomial is examined using $\hat{D} = \Delta$ only.

Now, a good choice for T (if $P = 1$) might be: $T = \hat{A}(1 - \mu q^{-1})$. If $\mu = 1$, then $T = \hat{A}\Delta$. As a result, $S = 0$ and the stability margins are infinite. Decreasing μ makes the robustness smaller. Moreover, if $\hat{D} = \Delta^{\delta}$ where $\delta > 1$ (this may be required in order to avoid steady-state errors for non-constant reference trajectories and/or disturbances, see section 2.3.3), then a logical choice for T is: $T = \hat{A}(1 - \mu q^{-1})^{\delta}$.

In the next example, the influence of μ on the robustness of the closed-loop system when $T = \hat{A}(1 - \mu q^{-1})$ and $\hat{D} = \Delta$, is shown.

Example 3.21 The influence of T on the robustness of the closed-loop system when $\hat{D} = \Delta$.

Settings:	same as in Example 3.20 but now $\hat{D} = \Delta$.
Process:	same as in Example 3.20
Model:	identical to process
Parameters:	$T = \hat{A}(1 - \mu q^{-1})$ with $0 \leq \mu \leq 1$

Figure 3.54 shows the upper gain and delay margin as a function of μ. The figure shows that both the gain and delay margin increase monotonously as μ increases. For $\mu = 0$, the robustness is already quite large: $\overline{gm} = 3.0$ and $\overline{dm} = 0.7$s.

In this example H_p, H_s, H_c and ρ are chosen such that dead-beat control is approximated. It was shown in section 3.5.1, that dead-beat control with $T = \hat{A}$

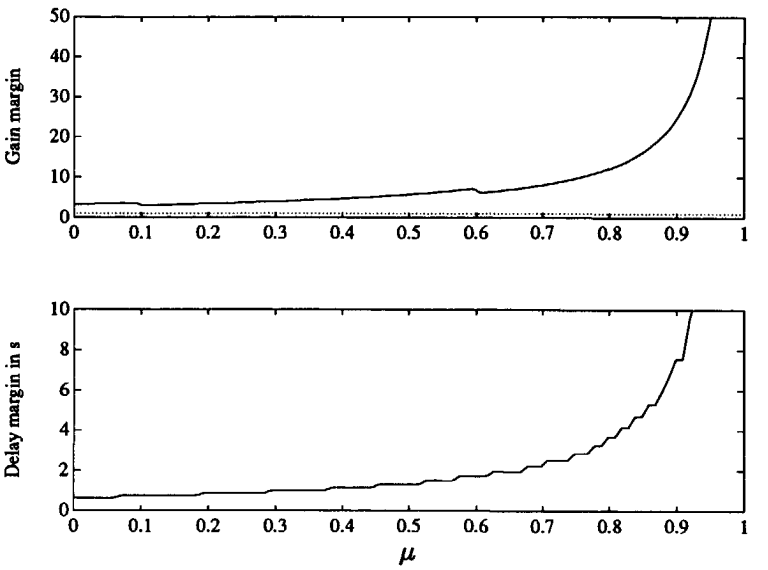

Figure 3.54: Gain and delay margins as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ and $\hat{D} = \Delta$.

yields the same robustness as mean-level control with $T = 1$. Because mean-level control yields a quite robust closed-loop system, this explains why for $\mu = 0$ (hence $T = \hat{A}$) the robustness is already quite large.

Thus far the influence of T on the robustness of the closed-loop system. In the next section, the influence of T on the regulator behavior of the closed-loop system is discussed.

The influence of T on the regulator behavior

The regulator performance is judged by calculating the variances of $y(k)$ and $u(k)$ as a function of μ. The disadvantage of of analyzing the regulator performance in this way is that it can be used to judge only the performance of the controller for a particular choice of the C and D polynomials of the process. In order to overcome the problem of the analysis being useful only for a particular choice for C and D, three combinations for C and D are examined:

1. $C = 1$, $D = 1$: The noise that appears on the output of the process is white noise filtered by $1/A$.

2. $C = A$, $D = 1$: Now, the output of the process is disturbed by white noise. This type of disturbance can be considered as measurement noise.

3. $C = A$, $D = \Delta$: For this choice of the noise model, the output of the process is corrupted by integrated white noise. This type of noise is often called Brownian motion or random walk.

Example 3.22 The influence of T on the regulator behavior of the closed-loop system.	
Settings:	same as in Example 3.21
Process:	same as in Example 3.21
Model:	identical to process
Parameters:	$T = \hat{A}(1 - \mu q^{-1})$ with $0 \leq \mu \leq 1$

The Figures 3.55a, 3.55b and 3.56 show the influence of μ on the variances of $y(k)$ and $u(k)$ for the noise models mentioned above. The variance of $y(k)$ is normalized with σ_e^2 (the variance of $e(k)$). Hence, a value of 1 corresponds to minimum variance. Figure 3.55a shows that for noise filtered by $1/A$, increasing μ makes the process output variance increase. Table 3.4 shows σ_y^2 and σ_u^2 for $\mu = 0$, $\mu = 1$ and for minimum-variance control. For $\mu = 1$ the output variance is equal

	σ_y^2	σ_u^2
$\mu = 0$	5.5	0.6
$\mu = 1$	218	0
MV	1	21

Table 3.4: Regulator performance criteria for different values of μ and for minimum-variance (MV) control.

to the uncontrolled output variance. For $\mu = 0$, the process output variance is 5.5 times worse than that obtained when using minimum-variance control. However, still a rather large noise reduction ($218/5.5 = 40$) is obtained. The controller output variance is quite small compared to the controller output variance required to obtain minimum-variance control.

Figure 3.55b shows the effect of μ on σ_y^2 and σ_u^2 when the process output is disturbed by measurement noise. For $\mu = 1$, measurement noise is exactly modeled and hence minimum variance is achieved by doing nothing at all ($\sigma_u^2 = 0$). Making μ

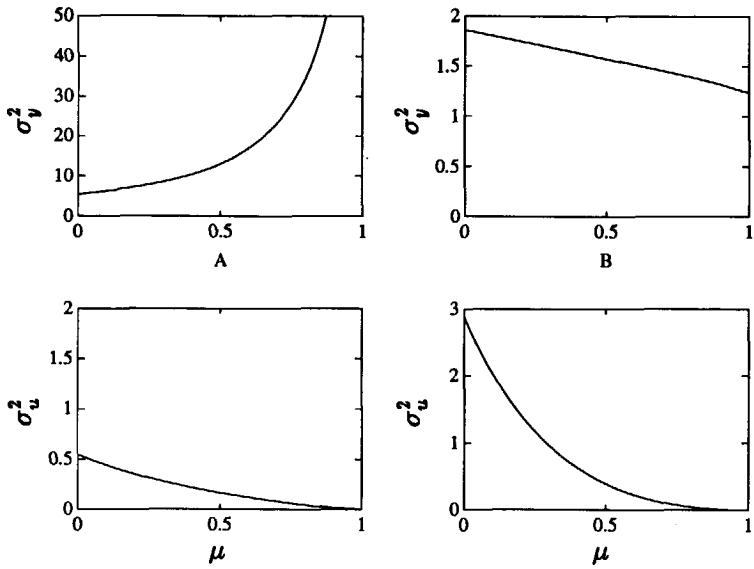

Figure 3.55: Regulator performance criteria as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$, $C = 1$ and $D = 1$ (Figure A) and where $T = \hat{A}(1 - \mu q^{-1})$, $C = A$ and $D = 1$ (Figure B).

smaller than 1, causes the controller output variance and the process output variance to increase. Even when $\mu = 0$ both variances may still be acceptable; the process output variance is only a factor 1.8 worse than in the minimum-variance case.

Finally, Figure 3.56 shows that when $C = A$ and $D = \Delta$, increasing μ makes σ_y^2 increase and σ_u^2 decrease. For $\mu = 0$, the noise model is equal to that of the process. Hence, for $H_p = H_c$ and $\rho = 0$, minimum-variance control is obtained. In this case, $H_p = 15$ and $H_c = 3$ and hence dead-beat control is approximated. Despite this, the process output variance is still quite small: $\sigma_y^2 = 1.5$.

The Figures 3.57a and 3.57b illustrate the use of T in reducing the controller output variance when the process is disturbed by measurement noise (hence, $C = A$ and $D = 1$). In Figure 3.57a, $T = 1$ was used while in Figure 3.57b, $T = \hat{A}$ was used. All other settings were equal to those given in Example 3.22. Obviously, by choosing $T = \hat{A}$ instead of $T = 1$, the controller output variance is significantly decreased while the process output variance is hardly affected. The response to set point changes is not affected at all because the process is correctly estimated (see Theorem 2.11).

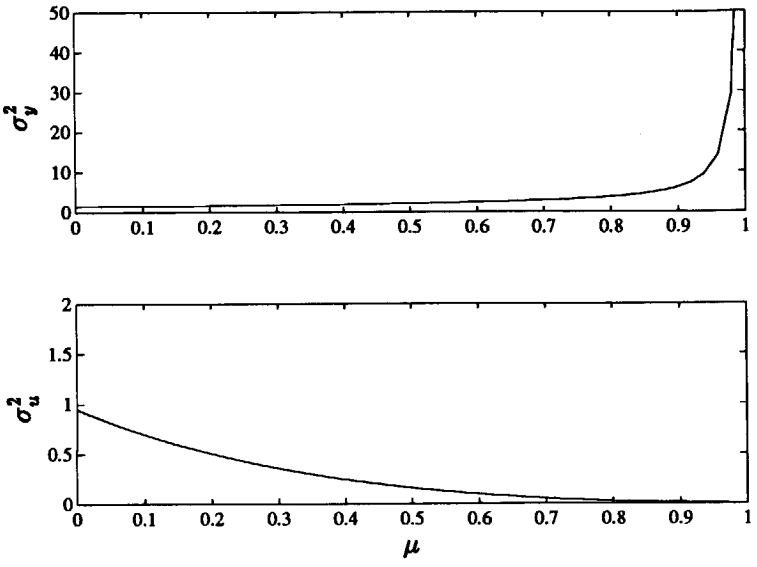

Figure 3.56: Regulator performance criteria as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$, $C = A$ and $D = \Delta$.

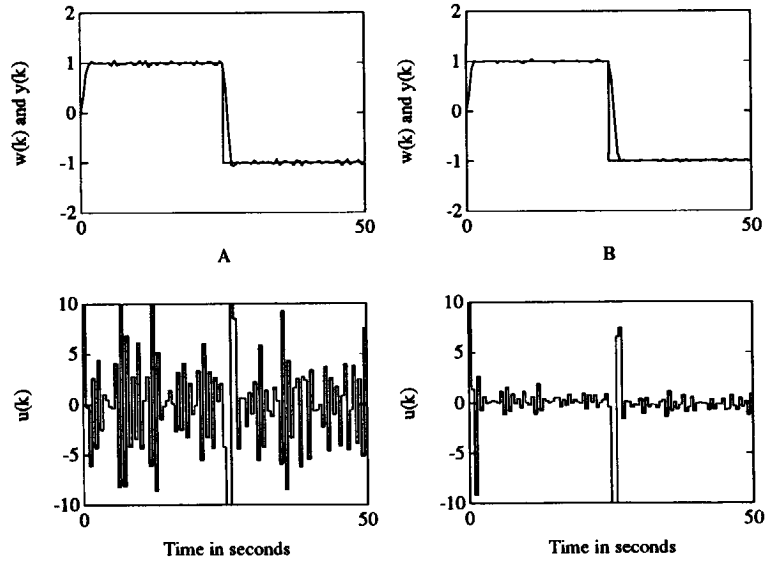

Figure 3.57: Responses where $T = 1$ (Figure A) and where $T = \hat{A}$ (Figure B) when the process output is disturbed by measurement noise.

Robustness and regulator behavior if $T = (1 - \mu q^{-1})^{n_A}$

Thus far, T has been chosen equal to $T = \hat{A}(1 - \mu q^{-1})$. However, Clarke [6] suggested the following choice for T: $T = (1 - \mu q^{-1})^{n_A}$ with as default $\mu = 0.8$. In order to examine the effect of this choice for T, the simulations of Example 3.22 were repeated where $T = (1 - \mu q^{-1})^2$ and $\mu = 0, \ldots, 1$.

Example 3.23 The influence of T on the regulator behavior of the closed-loop system using $T = (1 - \mu q^{-1})^{n_A}$ and $\hat{D} = \Delta$.

Settings:	same as in Example 3.22
Process:	same as in Example 3.22
Model:	identical to process
Parameters:	$T = (1 - \mu q^{-1})^2$ with $0 \leq \mu \leq 1$

Figure 3.58 shows the upper and lower gain margin and the time delay margin as a function of μ. Obviously, also choosing $T = (1 - \mu q^{-1})^2$ makes the robustness

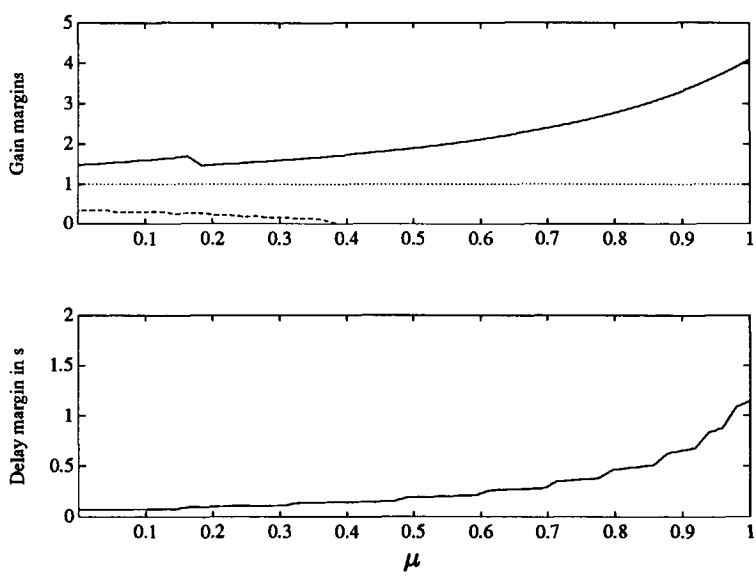

Figure 3.58: Gain and delay margins as a function of μ for $T = (1 - \mu q^{-1})^2$ and $\hat{D} = \Delta$.

increase as μ increases. However, both the gain and time delay margins remain at a lower level than those obtained when using $T = \hat{A}(1 - \mu q^{-1})$ (see Figure 3.54).

Now, the upper gain margin remains between 1.4 and 4.1 and the delay margin remains between 0.07s and 1.1s (when choosing $T = \hat{A}(1 - \mu q^{-1})$ these values are respectively: 3.0 and infinity for the gain margin and 0.7s and infinity for the delay margin). Hence, choosing $T = \hat{A}(1 - \mu q^{-1})$ results in a much wider robustness

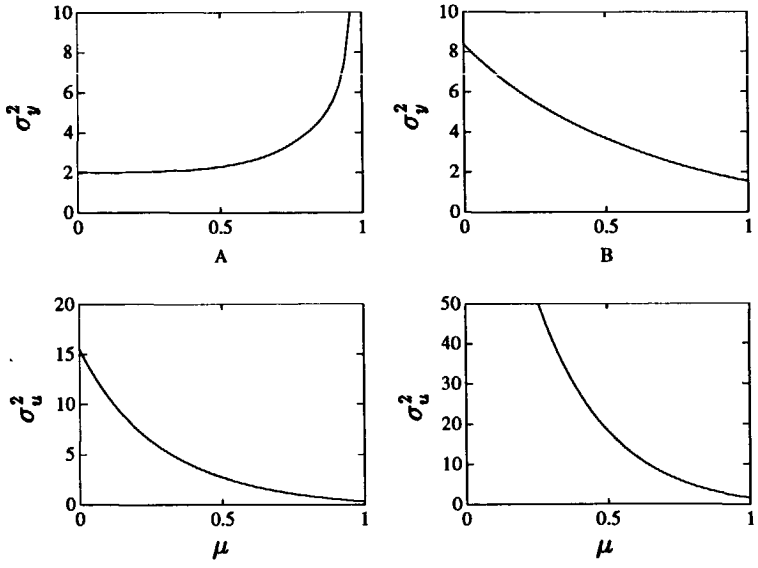

Figure 3.59: Regulator performance criteria as a function of μ where $T = (1 - \mu q^{-1})^2$, $C = 1$ and $D = 1$ (Figure A) and where $T = (1 - \mu q^{-1})^2$, $C = A$ and $D = 1$ (Figure B).

range.

Figure 3.59a shows the regulator performance criteria for $C = 1$ and $D = 1$. For $\mu = 0$ good noise rejection is obtained. However, the controller output variance is rather large. Compared to the results obtained with the previous choice of T, the noise rejection is 2.8 times better. The controller output variance is approximately 25 times larger (see Figure 3.55a). Increasing μ makes the controller output decrease while the process output variance increases slightly for $\mu < 0.8$. For the chosen process noise model ($C = 1$ and $D = 1$) using $\mu = 0.9$ yields variances which are comparable to those obtained when $T = \hat{A}$ is chosen. Not a surprising result, because the roots of A are given by 0.882 and 0.905. Hence, for $\mu = 0.9$, T is approximately equal to $\hat{A}$. Using the default choice suggested by Clarke ($\mu = 0.8$), also yield comparable results. The default value for μ in the case $T = (1 - \mu q^{-1})^{n_A}$ is obviously related to the sampling period T_s. When the process has n_A time

constants of τ seconds, the discrete time process has n_A poles at $e^{\frac{-T_s}{\tau}}$. If, as a rule of thumb, the sampling period is chosen ten times smaller than the major time constant of the process, the discrete time process has n_A poles at 0.9. By choosing, $T = (1-0.9q^{-1})^{n_A}$, the T polynomial is equal to the $\hat{A}$ polynomial which results in good robustness and good regulator behavior as is shown above. In this example, the process is sampled ten times faster than the major time constant making $T = (1-0.9q^{-1})^2$ a good choice.

Figure 3.59b shows the influence of μ on the regulator behavior when the process is disturbed by measurement noise. It shows clearly that a small value for μ

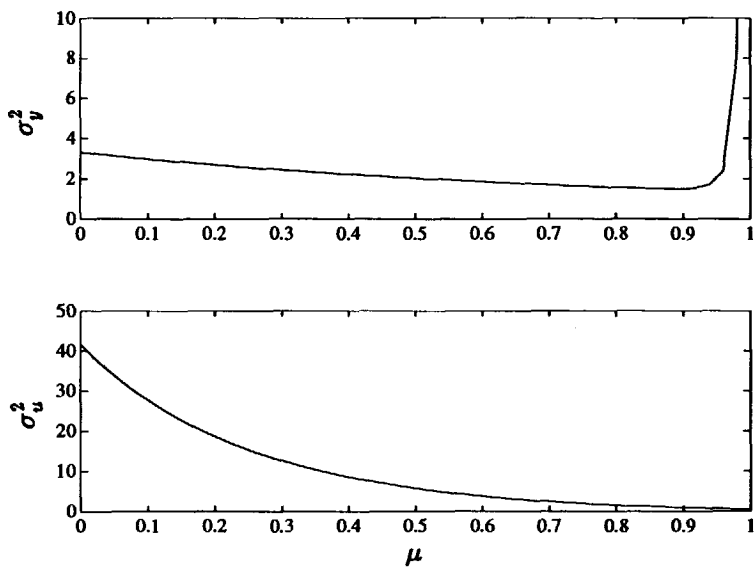

Figure 3.60: Regulator performance criteria as a function of μ where $T = (1-\mu q^{-1})^2$, $C = A$ and $D = \Delta$.

yields unacceptably large variances for both $y(k)$ and $u(k)$. Again choosing $\mu = 0.9$ results in a much better performance.

Finally, Figure 3.60 shows the regulator performance criteria when $C = A$ and $D = \Delta$. For a small μ the controller output variance is again extremely large. Increasing μ makes the controller output variance decrease and again for $\mu = 0.9$ an acceptable value is obtained. Note that increasing μ beyond $\mu = 0.9$ makes σ_y^2 increase. This is caused by the fact that for $\mu = 0.9$, $T \approx C$ and $\hat{D} = D$. Hence, the noise acting on the output of the process is well modeled which results in good

noise rejection. Obviously, other values for μ yield larger values for σ_y^2.

Conclusion: For this simple process, the choice $T = \hat{A}$ is a good one. It results in good robustness and good regulator behavior without a large controller output variance. The robustness can be further increased by choosing $T = \hat{A}(1 - \mu q^{-1})$ with $\mu > 0$ as shown in Figure 3.54. As a consequence, the regulator behavior becomes worse except when the process output is disturbed by measurement noise.

Using $T = (1 - \mu q^{-1})^{n_A}$ yields small robustness margins for small values of μ (especially the delay margin) and a large controller output variance. Using $\mu = 0.9$ yields, at least for this process, a good compromise between controller output variance, noise rejection and robustness. However, as mentioned above, for this choice $T \approx \hat{A}$ and hence both methods yield approximately the same results.

Thus far the influence of T on the robustness and regulator behavior has been examined for a simple second-order process. But what about other processes with complex poles, time delay and so on? The effect of the noise model on the robustness and regulator behavior of the closed-loop system for a more complex process, a non-minimum phase process, is discussed in the following example.

Example 3.24 The influence of T on the robustness of the closed-loop system with a non-minimum phase process.

Settings:	$H_p = 25,\ H_s = 1,\ H_c = 2,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = \Delta,$ $\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$
Process:	$H(s) = \dfrac{-4(s-1)}{s^2 + 1.6s + 4} \quad T_s = 0.2s \Longrightarrow$ $H(z^{-1}) = \dfrac{-0.5954z^{-1}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}}$
Model:	identical to process
Parameters:	$T = \hat{A}(1 - \mu q^{-1})$ or $T = (1 - \mu q^{-1})^2$ where $0 \leq \mu \leq 1$

The Figures 3.61a and 3.61b show the gain and time delay margins as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ and $T = (1 - \mu q^{-1})^2$, respectively. Comparing both figures yields:

- Using $T = \hat{A}(1 - \mu q^{-1})$ results in monotonously increasing robustness criteria as μ increases. Even when $\mu = 0$, the robustness margins are quite good: $\overline{gm} = 2.5$, $\underline{gm} = 0$ and $\overline{dm} = 0.34$s.
- If $T = (1 - \mu q^{-1})^2$, the robustness criteria have a maximum value for $\mu \approx 0.7$. Other values for μ yield criteria which are significantly worse. When $\mu = 0$, the

robustness margins are quite small: $\overline{gm} = 1.07$, $\underline{gm} = 0.97$ and $\overline{dm} = 0.09$s.

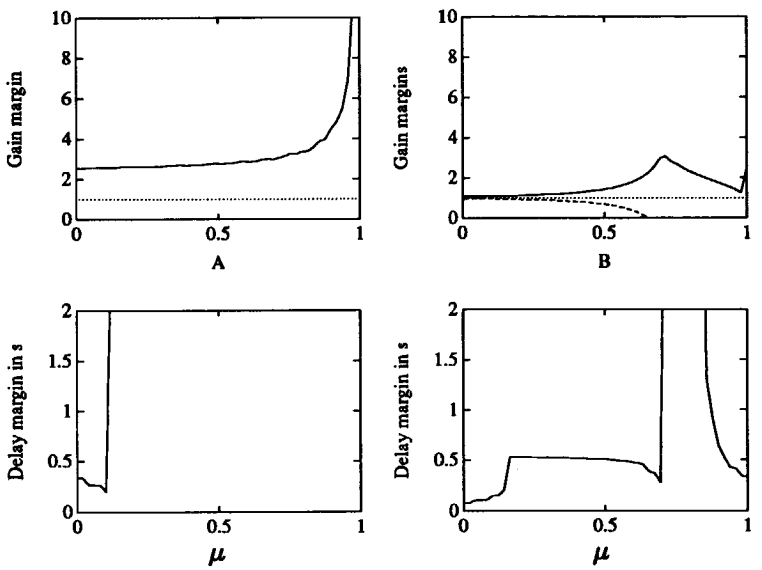

Figure 3.61: Robustness criteria as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ (Figure A) and with $T = (1 - \mu q^{-1})^2$ (Figure B).

A comparison of the regulator behavior for both choices for T yielded:

- $C = 1$, $D = 1$: Here comparable results were obtained if $\mu \geq 0.7$. If $\mu < 0.7$, using $T = (1 - \mu q^{-1})^2$ has shown to result in much worse (= larger) process and controller output variance.

- $C = A$, $D = 1$: Where $T = \hat{A}(1 - \mu q^{-1})$ the regulator criteria were hardly sensitive to changes in μ (for example, $1 \leq \sigma_y^2/\sigma_e^2 < 1.5$ for $0 \leq \mu \leq 1$). When $T = (1 - \mu q^{-1})^2$, the regulator performance criteria were extremely large for $\mu < 0.4$: $\sigma_y^2/\sigma_e^2 > 50$ for $\mu < 0.4$. Further, in this case the controller output variance was more than 1000 times larger. Using $\mu > 0.7$ has shown to yield comparable results.

- $C = A$, $D = \Delta$: Comparable results were obtained if $0.5 \leq \mu \leq 0.7$. For all other values of μ, using $T = (1 - \mu q^{-1})^2$ has shown to result in worse values for the process and controller output variance.

Conclusion: The choice $T = \hat{A}$ is again a good choice with respect to the robustness and regulator behavior. The robustness can be increased by using $T = \hat{A}(1 - \mu q^{-1})$ where $\mu > 0$. The choice $T = (1 - \mu q^{-1})^2$ with $\mu = 0.7$ is a good alternative. However, the robustness cannot be increased when using a different value for μ.

In the next example, the non-minimum phase process is extended by a time delay of 3 samples (= 0.6s)

Example 3.25 The influence of T on the robustness of the closed-loop system with a non-minimum phase process including time delay.

Settings:	Same as in Example 3.24 but now $\underline{d} = 3$ and $\overline{d} = 3$
Process:	$H(s) = \dfrac{-4(s-1)e^{-0.6s}}{s^2 + 1.6s + 4} \overset{T_s = 0.2s}{\Longrightarrow}$
	$H(z^{-1}) = \dfrac{-0.5954z^{-4}(1 - 1.2269z^{-1})}{1 - 1.5910z^{-1} + 0.7261z^{-2}}$
Model:	identical to process
Parameters:	$T = \hat{A}(1 - \mu q^{-1})$ or $T = (1 - \mu q^{-1})^2$ with $0 \leq \mu \leq 1$

Figure 3.62a shows the upper gain and the time delay margins as a function of μ when $T = \hat{A}(1 - \mu q^{-1})$ for $\mu = 0, \ldots, 1$. It shows again that increasing μ increases the stability robustness margins monotonously. For $\mu = 0.5$ the delay margins are quite good ($\overline{dm} = 3.8$s, $\underline{dm} = -17.6$s): hence, the closed-loop system remains stable if the true delay of the process is within 0 and 22 samples. For $\mu = 0$ (hence, $T = \hat{A}$) the delay margin is less than one sample.

The regulator criteria, shown in Table 3.5, show that for both $\mu = 0$ and $\mu = 0.5$, noise reduction is not achieved when $C = 1$ and $D = 1$ (the uncontrolled variance is equal to 15.3). Further, for $\mu = 0.5$ the robustness is significantly increased compared to when $\mu = 0$. The regulator performance criteria are hardly changed when μ is increased.

Figure 3.62b shows the gain and delay margins as a function of μ when $T = (1 - \mu q^{-1})^2$. This figure shows that for small μ ($\mu < 0.5$) the robustness is quite poor. A comparable robustness as when $T = \hat{A}(1 - \mu q^{-1})$ is obtained for $\mu > 0.65$. When $\mu = 0.8$ the robustness criteria are quite good: $\overline{gm} = 2.6$, $\underline{gm} = 0$, $\overline{dm} = 5.2$s and $\underline{dm} = -24.9$s. The regulator performance criteria for this choice of μ are shown in Table 3.6. When $C = 1$ and $D = 1$, the regulator performance criteria when $T = (1 - 0.8q^{-1})^2$ are better, compared to those obtained when $T = \hat{A}(1 - 0.5q^{-1})$ (compare Table 3.5). However, noise reduction is not achieved either. The regulator performance criteria when $C = A$ and $D = \Delta$ are worse compared to those obtained when using $T = \hat{A}(1 - 0.5q^{-1})$.

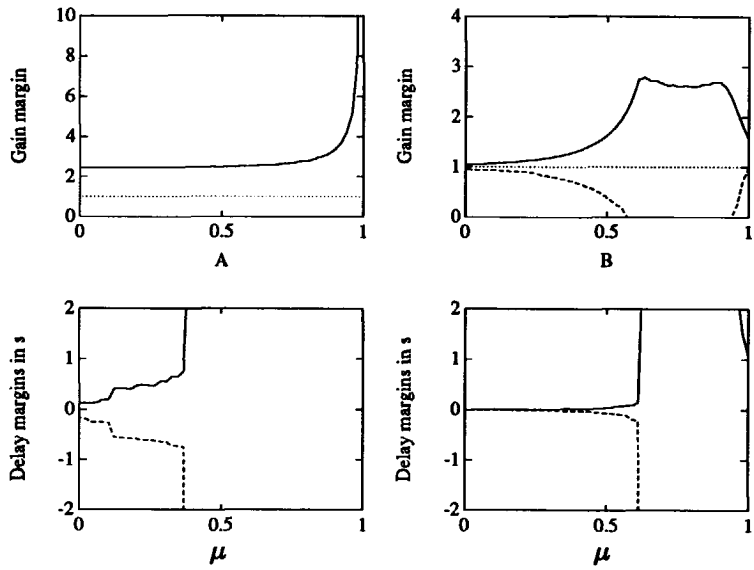

Figure 3.62: Robustness criteria as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ (Figure A) and where $T = (1 - \mu q^{-1})^2$ (Figure B).

		$C = 1$ $D = 1$	$C = A$ $D = 1$	$C = A$ $D = \Delta$
$\mu = 0$	σ_y^2	30.9	1.38	18
	σ_u^2	0.004	0.0011	0.42
$\mu = 0.5$	σ_y^2	27.5	1.16	19.1
	σ_u^2	0.004	0.0002	0.42

Table 3.5: Regulator performance criteria as a function of μ when $T = \hat{A}(1 - \mu q^{-1})$.

	$C = 1$ $D = 1$	$C = A$ $D = 1$	$C = A$ $D = \Delta$
σ_y^2	16.4	1.1	29.1
σ_u^2	0.0027	0.0001	0.42

Table 3.6: Regulator performance criteria if $T = (1 - 0.8q^{-1})^2$.

In the previous examples, stable processes only were investigated. The following example shows the effect of the noise model on the robustness of the closed-loop system for an unstable process.

Example 3.26 The influence of T on the robustness of the closed-loop system for an unstable process.

Settings: $H_p = 17,\ H_c = 3,\ H_s = 1,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = 1,$
$\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$

Process: $$H(s) = \frac{4.04}{(s+1)(s^2 - 0.4s + 4.04)} \quad T_s = 0.3s \Longrightarrow$$
$$H(z^{-1}) = \frac{1.710 \cdot 10^{-3} z^{-1}(1 + 3.505z^{-1})(1 + 0.2611z^{-1})}{1 - 2.4936z^{-1} + 2.4260z^{-2} - 0.8353z^{-3}}$$

Model: identical to process

Parameters: $T = (1 - \mu q^{-1})^3$ with $0 \leq \mu \leq 1$

Figure 3.63 shows the influence of μ on the stability margins of the closed-loop system. Note that the choice $T = \hat{A}$ is not possible because the roots of T appear

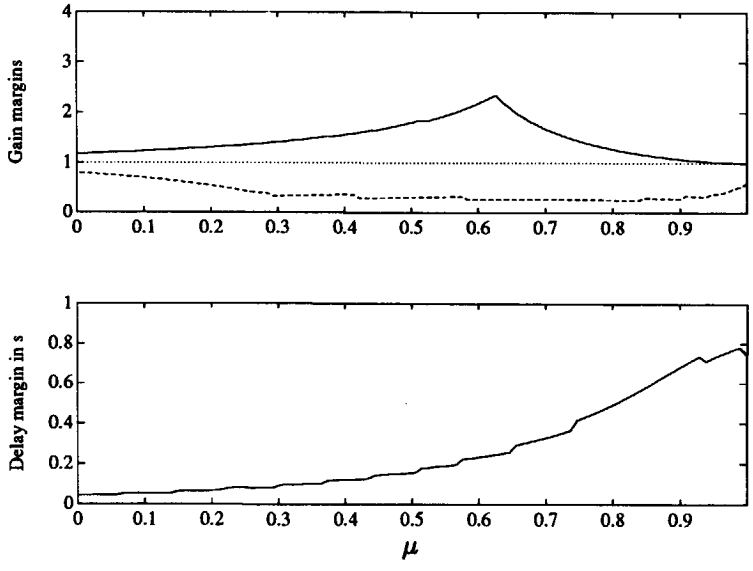

Figure 3.63: Robustness criteria as a function of μ where $T = (1 - \mu q^{-1})^3$.

as closed-loop poles (see Theorem 2.10). Hence, if $T = \hat{A}$, the closed-loop system would be unstable.

Figure 3.63 shows that increasing μ increases the robustness criteria. However, increasing μ beyond 0.62 makes the gain margin drop.

3.5.3 The Noise Model and Load Changes

Another way of judging the regulator behavior of the closed-loop system is how well disturbances due to load changes are rejected. Load changes can be modeled as:

$$\xi(k) = \frac{A(1)}{A} l(k) \tag{3.57}$$

in which $l(k)$ is a unit step: $L(z^{-1}) = \dfrac{1}{1 - z^{-1}}$ (the numerator of (3.57) is chosen such that the steady-state value of $\xi(k)$ is equal to 1). The criteria used to judge the regulator behavior with respect to load changes are discussed in section 3.1. The steady-state error is not taken into account because the controller parameters are selected such that steady-state errors do not occur. Further, it is assumed that $w(k) = 0$ so that the only disturbance in the system is caused by the load change.

It was shown in section 2.2.1 that the closed-loop transfer function in the case of minimum-variance control, and if the model is identical to the process, is given by (2.86):

$$y(k) = q^{-1} Sp + \frac{\hat{A}\hat{D}}{T} \xi(k) \tag{3.58}$$

Hence, if $\xi(k)$ represents a load change given by (3.57) and the set point is equal to zero, then (3.58) becomes:

$$y(k) = \frac{A(1)\hat{D}}{T} l(k)$$

where $l(k)$ is a unit step. By choosing $T = 1$ and $\hat{D} = \Delta$, the load change is rejected in one sample. In the case of pole-placement control, the CLTF is given by (2.111):

$$y(k) = \frac{q^{-1}RB}{B(1)P} Sp + \frac{A\mathcal{R}}{PT} \xi(k) \tag{3.59}$$

When $\xi(k)$ is given by (3.57) and the set point is equal to zero, (3.59) becomes:

$$y(k) = \frac{A(1)\mathcal{R}}{PT} l(k) \tag{3.60}$$

By choosing $P = T = 1$ and $\hat{D} = \Delta$, load changes are rejected in n_R samples. Note that if $\hat{D} = \Delta$ and a pole-placement controller is selected (hence, $\beta = 1$ and $\rho = 0$), Δ is a factor of $\mathcal{R}$ (see section 2.3.3).

As shown in section 3.5.2, T must usually chosen different from 1 in order to obtain good robustness of the closed-loop system. The effect of choosing T different from 1 on the rejection of load changes is easy to analyze. For example, for a pole-placement controller, (3.60) describes the behavior of the closed-loop system for a load change. Obviously, load changes are filtered by P and T making the rejection of load changes slower if P and/or T are chosen different from 1. In the previous sections, it was shown that in order to gain good robustness, T can be chosen according to $T = \hat{A}(1 - \mu q^{-1})$ with $\mu = 0, \ldots, 1$. However, if the model has badly damped poles, the response to a load change will be oscillatory. When the other choice for T: $T = (1 - \mu q^{-1})^{n_A}$ is used, the response to a load change will be smooth. The following example illustrates both choices for T in controlling a process that has badly damped poles.

Example 3.27 The influence of T on the rejection of load changes in controlling a process with badly damped poles.

Settings:	$H_p = 6$, $H_s = 3$, $H_c = 4$, $\beta = 1$, $P = N = 1$, $R = 1$, $\hat{D} = \Delta$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	$H(s) = \dfrac{101}{(s+4)(s^2+s+25.25)}$ $\overset{T_s = 0.1s}{\Longrightarrow}$ $H(z^{-1}) = \dfrac{0.0147z^{-1}(1+3.26z^{-1})(1+0.239z^{-1})}{1-2.34z^{-1}+2.024z^{-2}-0.607z^{-3}}$
Model:	identical to process
Parameters:	$T = \hat{A}$ or $T = (1 - 0.8q^{-1})^3$

Note that the parameter settings are chosen such that dead-beat control is obtained. The complex poles of the process have a relative damping ratio of 0.1. The step response of the process used in this example is shown in Figure 3.64. Figure 3.65a shows the response to a load change where $T = \hat{A}$. In this case the load change occurred at $t = 1$s. Because the complex poles of the process are badly damped, this response is oscillatory. Note that the controller output is constant and equal to

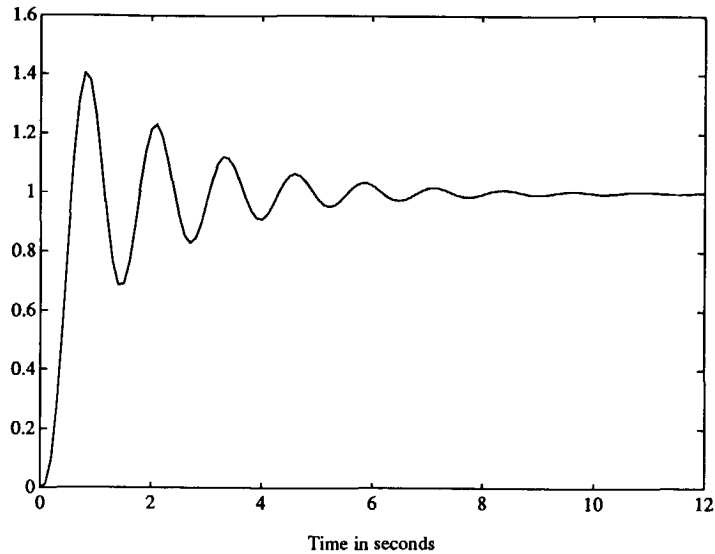

Figure 3.64: Step response of a process with badly damped poles.

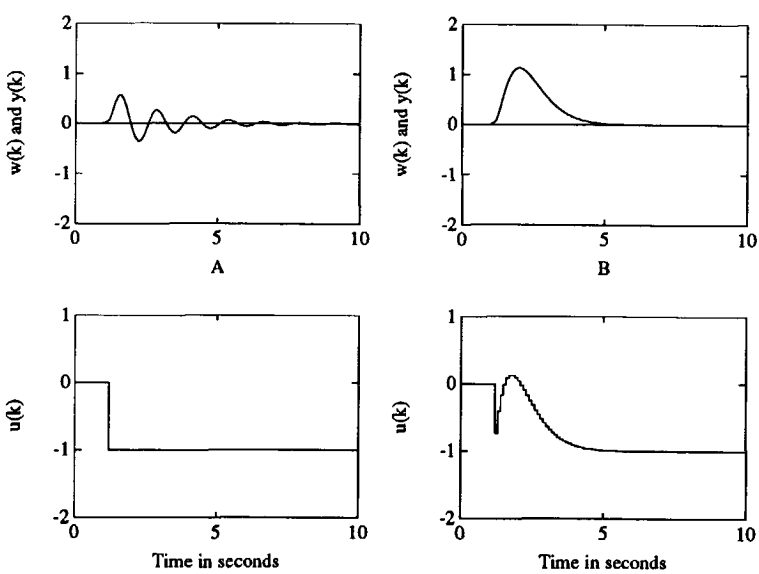

Figure 3.65: Response to a load change at $t = 1s$ where $T = \hat{A}$ (Figure A) and where $T = (1 - 0.8q^{-1})^3$ (Figure B).

-1. This is caused by the fact that a dead-beat controller is selected where $\hat{D} = \Delta$: The transfer function from $\xi(k)$ to $u(k)$ is given by:

$$u(k) = -\frac{AS}{A\mathcal{R} + q^{-d-1}BS}\xi(k) \tag{3.61}$$

It was shown in section 3.5.1 that for dead-beat controllers in which $\hat{A}$ is a factor of T and $\hat{D} = \Delta$, S is given by (3.53):

$$S = \frac{\hat{A}(q^{-1})T_1(1)}{\hat{B}(1)} \tag{3.62}$$

where $T = \hat{A}T_1$. In the case of dead-beat control $A\mathcal{R} + q^{-d-1}BS = T$ and hence (3.61) becomes:

$$u(k) = -\frac{AS}{T}\xi(k) \tag{3.63}$$

Using (3.57) and (3.62) taking into account $T_1 = 1$ (remember that $T = \hat{A}$), (3.63) becomes:

$$u(k) = -\frac{1}{K_{dc}}l(k)$$

The process used in the example has a DC gain of 1 and hence a load change results in a constant controller output: $u(k) = -1$.

When using $T = (1 - 0.8q^{-1})^3$, the response to a load change at $t = 1$s is slow but nicely damped as illustrated in Figure 3.65b.

3.5.4 The Noise Model and Model Mismatch

In this section, the influence of the noise model is shown when the process is different from the model. Two cases are discussed: unmodeled dynamics and parameter mismatch.

Unmodeled dynamics

In order to demonstrate the use of the T polynomial in the case of unmodeled dynamics, a third-order process with high frequency dynamics is controlled by using a first-order model in which the high frequency dynamics are neglected. The process and model are proposed by Rohrs [44].

Example 3.28 The use of T in the case of unmodeled dynamics.

Settings: $H_p = 20,\ H_s = 1,\ H_c = 2,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = \Delta,$
$\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$

Process:
$$H(s) = \frac{458}{(s+1)(s^2+30s+229)} \quad T_s = 0.04s \Longrightarrow$$
$$H(z^{-1}) = \frac{3.612 \cdot 10^{-3} z^{-1}(1+2.7629z^{-1})(1+0.1947z^{-1})}{1-2.0549z^{-1}+1.3524z^{-2}-0.2894z^{-3}} \quad (3.64)$$

Model:
$$\hat{H}(s) = \frac{2}{s+1} \quad T_s = 0.04s \Longrightarrow$$
$$\hat{H}(z^{-1}) = \frac{0.07842z^{-1}}{1-0.9608z^{-1}}$$

Parameters: $T = \hat{A}(1-\mu q^{-1})$ or $T = (1-\mu q^{-1})$ with $\mu = 0, \ldots, 1$

Note that the model used for calculating the controller parameters is given by the first-order part of (3.64). In the model the two complex poles at $-15 \pm 2j$ are neglected. Figure 3.66 shows a root-locus plot as a function of μ with $T = (1-\mu q^{-1})$. Obviously, the closed-loop system is unstable for all μ. Hence, $T = (1-\mu q^{-1})$ cannot be used.

Remark: Remember that the symbols 'x' denote the closed-loop poles for $\mu = 0$ while the symbols 'o' denote the closed-loop poles for $\mu = 1$. The closed-loop poles for $\mu = 0$ and $\mu = 1$ are, in general, not related to the poles and zeros of the process.

As discussed earlier, choosing $PT = \hat{A}\hat{D}$ makes the gain and time delay margins of the closed-loop system infinite if the process is stable. Therefore, let us examine the choice $T = \hat{A}(1-\mu q^{-1})$ for $\mu = 0, \ldots, 1$. For $\mu = 1$, the above-mentioned condition is satisfied and infinite stability margins are obtained. Figure 3.67 shows a root-locus plot as a function of μ with $T = \hat{A}(1-\mu q^{-1})$. For $\mu > 0.25$ the closed-loop system is stable and for $\mu > 0.9$ the system is well damped. Figure 3.68a shows a time response for $T = \hat{A}(1-0.9q^{-1})$. Although, the poles of the closed-loop system are well damped, a large overshoot occurs. This is caused by the roots of the T polynomial which appear as zeros of the closed-loop system (see (2.191) and (2.192)). If the process had been correctly estimated these zeros would

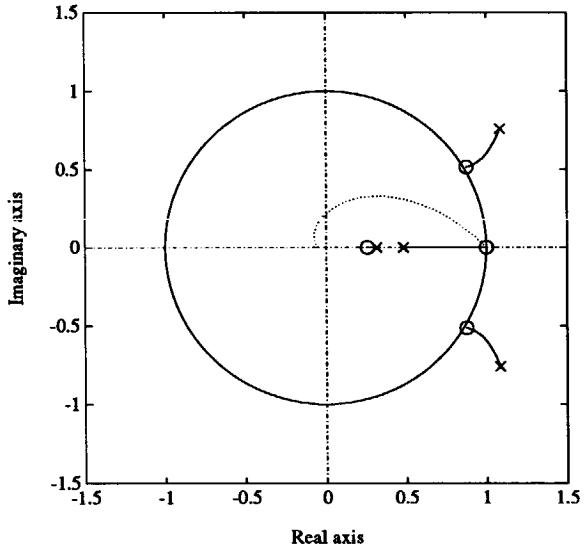

Figure 3.66: Root-locus plot as a function of μ with $T = (1 - \mu q^{-1})$ and $H_c = 2$.

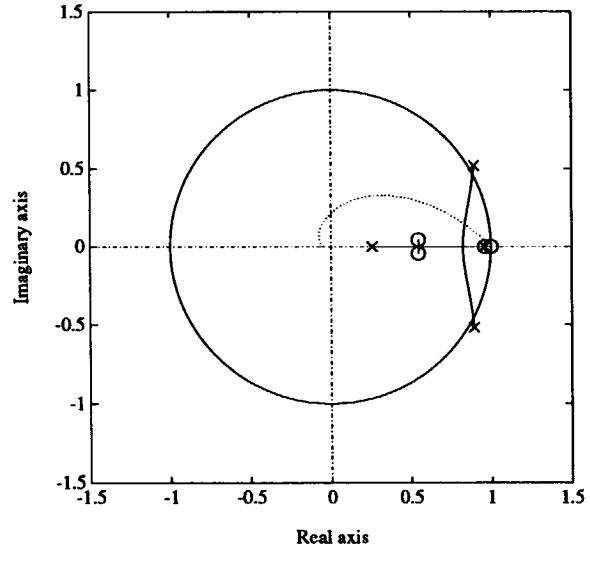

Figure 3.67: Root-locus plot as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ and $H_c = 2$.

have been canceled by the closed-loop poles and hence they would not affect the servo behavior of the system.

Another way of increasing the robustness of the closed-loop system is by making H_c smaller (see section 3.4.1). In Figure 3.69 the root-locus plot is shown as

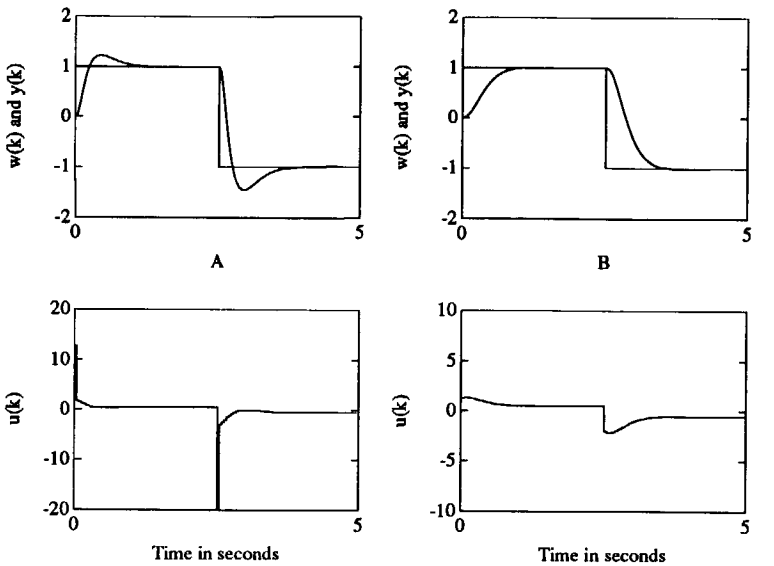

Figure 3.68: Time responses where $T = \hat{A}(1 - 0.9q^{-1})$ and $H_c = 2$ (Figure A) and where $T = \hat{A}$ and $H_c = 1$ (Figure B).

a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ and $H_c = 1$. The figure shows that now the closed-loop system is stable for all μ. For $\mu = 0$ the closed-loop poles are already well damped. Figure 3.68b shows the resulting time response. For this choice of H_c, $T = (1 - \mu q^{-1})$ can also be used to stabilize the closed-loop system. Then, the closed-loop system is stable for $\mu > 0.4$.

Remark: for both $H_c = 2$ and $H_c = 1$ the T polynomial must be chosen different from 1 in order to stabilize the closed-loop system.

A stable closed-loop system can also be obtained by using the results of section 3.5.1. For the process and model used in Example 3.28, the frequency response of the difference between process and model (= $|\Delta_a|$) is shown in Figure 3.70. In this figure, the DC gain of the model (= $\hat{K}_{dc}$) is shown too. Figure 3.70 shows clearly that $|\Delta_a| < \hat{K}_{dc} \quad \forall\omega \geq 0$. Now the closed-loop system is guaranteed to be stable if mean-level control is selected where $T = \hat{A}$ and $\hat{D} = \Delta$ or if dead-beat control is

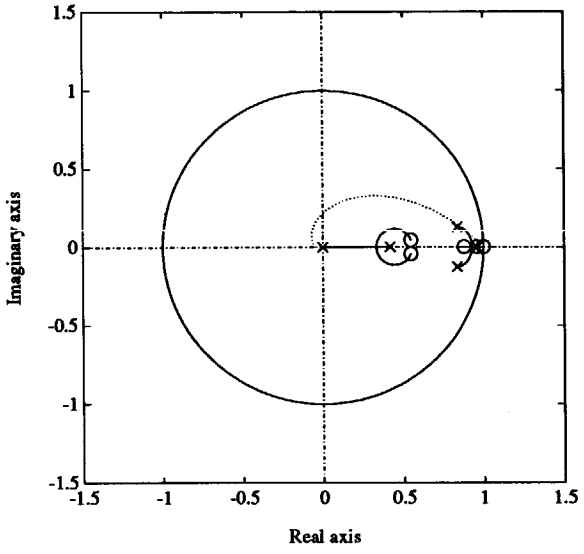

Figure 3.69: Root-locus plot as a function of μ where $T = \hat{A}(1 - \mu q^{-1})$ and $H_c = 1$.

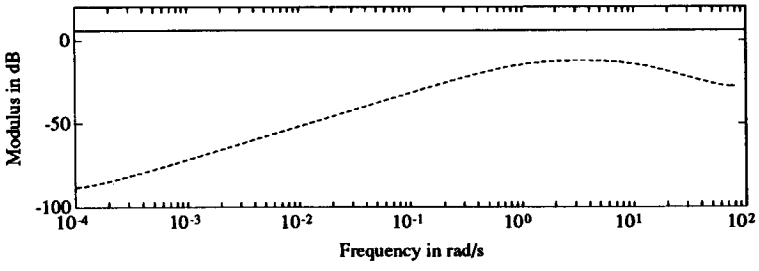

Figure 3.70: Frequency responses of Δ_a (dashed line) and $\hat{K}_{dc}$ (solid line).

selected where $T = \hat{A}^2$ and $\hat{D} = \Delta$ (see section 3.5.1, page 163 and page 165). The corresponding time responses are shown in Figure 3.71a and 3.71b for mean-level and dead-beat control, respectively.

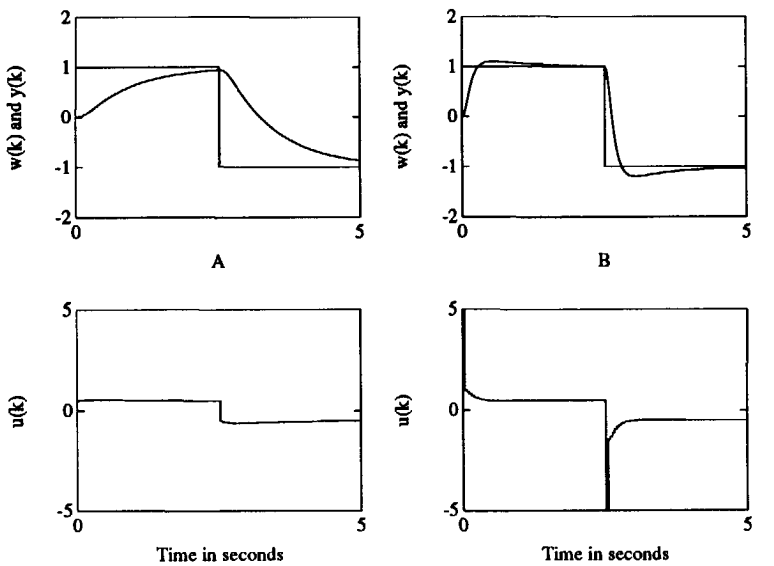

Figure 3.71: Time responses using mean-level control where $T = \hat{A}$ *and* $\hat{D} = \Delta$ *(Figure A) and using dead-beat control where* $T = \hat{A}^2$ *and* $\hat{D} = \Delta$ *(Figure B).*

Model Mismatch and the Effect on the Servo Behavior

It was mentioned in section 3.5.3 that if the model has badly damped poles, using $T = \hat{A}(1 - \mu q^{-1})$ results in an oscillatory response to load changes. If the process is correctly estimated, the response to set point changes does not depend on T and hence T does not result in an oscillatory response even if $T = \hat{A}$ and the model has badly damped poles. It was shown in the proof of Theorem 2.11 that this is due to the fact that the roots of T appear as zeros in the transfer function from set point to output: the badly damped poles are canceled. Therefore, it can be expected that the slightest mismatch between process and model results in an oscillatory response to set point changes too. The following example illustrates this effect.

Example 3.29 The effect of T on the servo behavior of the closed-loop system in the presence of model mismatch.

Settings: $H_p = 25,\ H_s = 1,\ H_c = 3,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = \Delta,$
$\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$

Process: $$H(s) = \frac{68}{(s+4)(s^2+2s+17)} \quad T_s = 0.1s \Longrightarrow$$
$$H(z^{-1}) = \frac{9.697 \cdot 10^{-3} z^{-1}(1+3.191z^{-1})(1+0.232z^{-1})}{1-2.337z^{-1}+1.936z^{-2}-0.549z^{-3}}$$

Model: $$\hat{H}(s) = \frac{101}{(s+4)(s^2+s+25.25)} \quad T_s = 0.1s \Longrightarrow$$
$$\hat{H}(z^{-1}) = \frac{0.0147z^{-1}(1+3.26z^{-1})(1+0.239z^{-1})}{1-2.340z^{-1}+2.024z^{-1}-0.607z^{-3}}$$

Parameters: $T = \hat{A}$ or $T = (1-0.8q^{-1})^3$

The complex poles of the process are situated at $-1 \pm 4j$ (relative damping ratio 0.24) while the complex poles of the model are situated at $-0.5 \pm 5j$ (relative damping ratio 0.1).

Figure 3.72a shows a step response using $T = \hat{A}$. The figure shows that due to the fact that the process is different from the model the choice $T = \hat{A}$ yields an oscillatory step response. In Figure 3.72b, T was taken equal to $(1-0.8q^{-1})^3$. The response is in this case not oscillatory due to the fact that the roots of T are well damped. Hence, it can be concluded that if the model has badly damped poles, $\hat{A}$ must not be a factor of T and hence the choice $T = \hat{A}(1-\mu q^{-1})$ is not applicable. As an alternative $T = (1-\mu q^{-1})^{n_{\hat{A}}}$ can be used.

Remark: although the closed-loop system is badly damped, the stability robustness of the system when $T = \hat{A}$ is still larger than when $T = (1-0.8q^{-1})^3$ (see Table 3.7).

T	$\overline{gm}$	$\overline{dm}$ in s
$\hat{A}$	4.0	0.35
$(1-0.8q^{-1})^3$	1.7	0.27

Table 3.7: Robustness criteria for different T.

The next example illustrates the use of the T polynomial in the case of an unstable process and an unstable model different from the process.

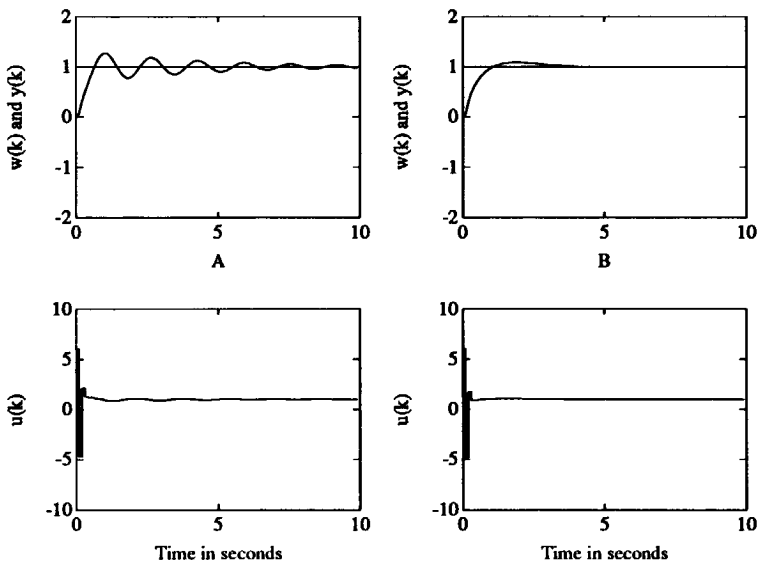

Figure 3.72: Step response in the presence of model mismatch and a process with badly damped poles where $T = \hat{A}$ (Figure A) and where $T = (1 - 0.8q^{-1})^3$ (Figure B).

Example 3.30 The effect of T on the stability and servo behavior of the closed-loop system in the presence of model mismatch and an unstable process.

Settings: $H_p = 20,\ H_s = 1,\ H_c = 2,\ \beta = 1,\ P = N = 1,\ R = 1,\ \hat{D} = \Delta,\ \underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$

Process:
$$H(s) = \frac{6.34}{(s+1)(s^2 - 0.6s + 6.34)} \quad T_s = 0.3s \Longrightarrow$$
$$H(z^{-1}) = \frac{2.696 \cdot 10^{-2} z^{-1}(1 + 3.519z^{-1})(1 + 0.268z^{-1})}{1 - 2.342z^{-1} + 2.383z^{-2} - 0.887z^{-3}}$$

Model:
$$\hat{H}(s) = \frac{4.04}{(s+1)(s^2 - 0.4s + 4.04)} \quad T_s = 0.3s \Longrightarrow$$
$$\hat{H}(z^{-1}) = \frac{1.710 \cdot 10^{-3} z^{-1}(1 + 3.505z^{-1})(1 + 0.2611z^{-1})}{1 - 2.4936z^{-1} + 2.4260z^{-2} - 0.8353z^{-3}}$$

Parameters: $T = (1 - \mu q^{-1})^3$ with $\mu = 0, \ldots, 1$.

The process has two complex poles at $0.3 \pm 2.5j$ while the model has two complex poles at $0.2 \pm 2j$. Figure 3.73 shows the step responses of both $H(s)$ and $\hat{H}(s)$. In order to show the influence of T on the stability of the closed-loop system a

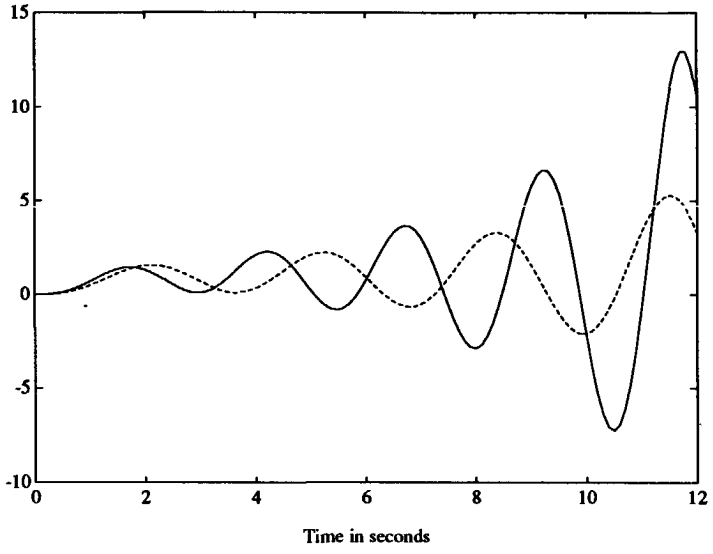

Figure 3.73: Step responses of $H(s)$ (dashed line) and $\hat{H}(s)$ (solid line).

root-locus plot is shown in Figure 3.74 with $T = (1 - \mu q^{-1})^3$ and $\mu = 0, \ldots, 1$. Remember that the symbol 'x' denotes the closed-loop poles if $\mu = 0$ while the symbol 'o' denotes the closed-loop poles if $\mu = 1$ (note that for $\mu = 1$, the closed-loop system has three poles at $z = 1$). The closed-loop poles for $\mu = 0$ and $\mu = 1$ are, in general, not related to the poles and zeros of the process.

The closed-loop system is stable for $\mu > 0.2$. Hence, T must be chosen different from 1 in order to stabilize the closed-loop system. Further increasing μ makes the closed-loop poles better damped but, because the model is different from the process, the closed-loop system becomes slower. This effect is illustrated by Figure 3.75 in which the rise and settling time are shown as a function of μ. A good choice for μ in this case might be $\mu = 0.7$. The servo behavior is hardly affected but the closed-loop poles are well damped. The robustness margins for this choice of μ are: $\underline{gm} = 0.39$, $\overline{gm} = 1.24$ and $\overline{dm} = 0.39$s. Thus, although the robustness margins are not extremely large, the unstable process can be controlled quite well even if the model is different from the process as is shown by Figure 3.76. This figure shows a time response using $T = (1 - 0.7q^{-1})^3$ without noise (Figure A) and with measurement noise (hence, $C = A$) with variance $\sigma_e^2 = 5 \cdot 10^{-3}$ added to the output of the process (Figure B).

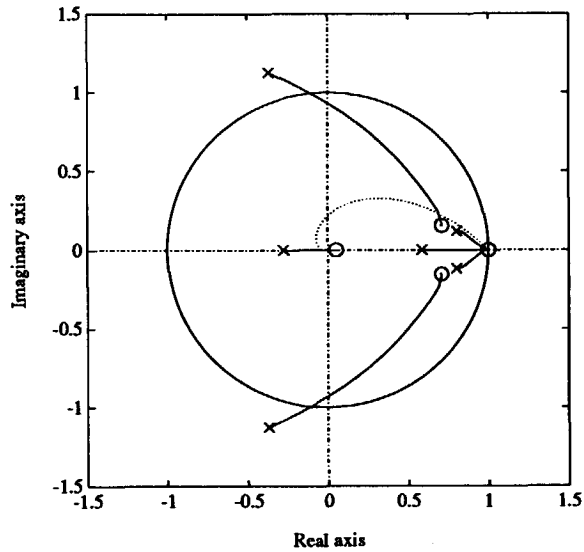

Figure 3.74: Root-locus plot as a function of T where $T = (1 - \mu q^{-1})^3$.

3.5.5 Conclusions

It was shown, by using a wide variety of processes, that the noise model can be used to improve the robustness of the closed-loop system. When the model is different from the process the noise model sometimes must be used in order to stabilize the closed-loop system. Two choices for T were discussed:

1. $T = \hat{A}(1 - \mu q^{-1})$ where $\hat{D} = \Delta$ and $0 \leq \mu \leq 1$ yields, for all processes that have been considered, monotonously increasing robustness criteria as μ increases. When $\mu = 0$ (hence, $T = \hat{A}$) a compromise between robustness and regulator performance is obtained. However, this choice for T cannot be used if the model has badly damped poles.

2. Using $T = (1 - \mu q^{-1})^{n_A}$ where $\hat{D} = \Delta$ and $0 \leq \mu \leq 1$ mostly yields a maximum robustness for a particular value for μ (usually for $0.6 \leq \mu \leq 0.9$). This maximum robustness has shown to be approximately equal to that obtained when using $T = \hat{A}$ and is hence equal to the minimum robustness that is obtained when using $T = \hat{A}(1 - \mu q^{-1})$. Further, using small values for μ ($\mu < 0.6$) usually result in an unacceptable regulator performance (because the

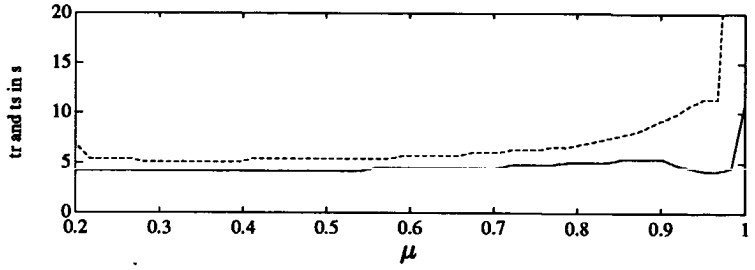

Figure 3.75: Rise (solid line) and settling time (dashed line) as a function of μ with $T = (1 - \mu q^{-1})^3$.

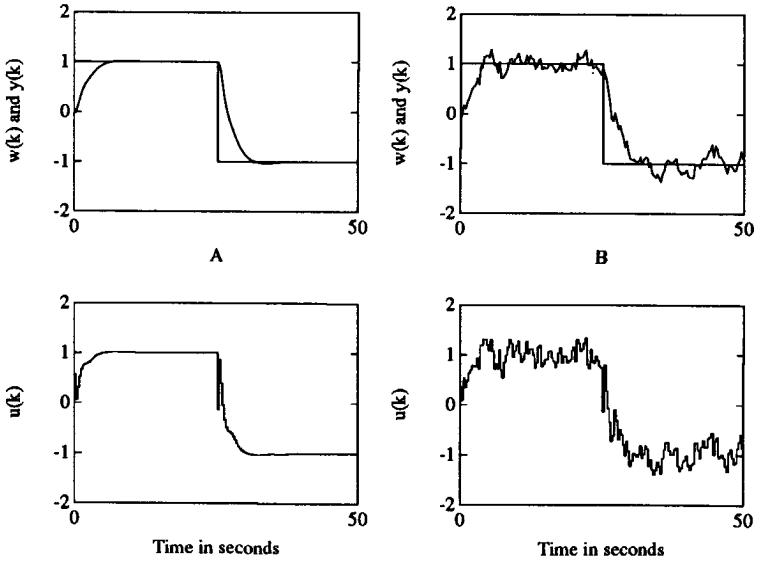

Figure 3.76: Predictive control of an unstable process in the presence of model mismatch where $T = (1 - 0.7q^{-1})^3$ without noise (Figure A) and with measurement noise (Figure B).

controller output variance is very large). However, $T = (1-\mu q^{-1})^{n_A}$ can also be used with unstable processes. It was shown that the choice for μ depends on the sampling period T_s. If a process with real poles is sampled ten times faster than its major time constant, this results in $\mu = 0.9$. The default value suggested by Clarke: $T = (1-0.8q^{-1})^{n_A}$ has shown to yield acceptable results in most cases.

In the case of load changes, the optimal choices for T and $\hat{D}$ are: $T = 1$ and $\hat{D} = \Delta$. Choosing T different from 1 increases the robustness of the closed-loop system but makes load changes be rejected more slowly.

It was shown that increasing μ where $T = (1-\mu q^{-1})^{n_A}$ makes the closed-loop system respond more slowly to set point changes if the model is different from the process. This is not the case if the process is correctly modeled.

3.6 Reference Trajectory and the use of P and R

In this section the reference trajectory and the design parameters P and R are discussed. As discussed in section 2.2, the P polynomial can be used to define the desired servo behavior of the closed-loop system. However, usually P has an effect on the regulator behavior and the robustness of the closed-loop system too.

3.6.1 Regulator Behavior and Robustness versus the use of P

In this section the influence of P on the regulator behavior and the robustness of the closed-loop system is discussed. When a pole-placement controller is selected (see Table 2.4), this influence is easy to analyze. Now the closed-loop transfer function, if the process is correctly modeled, is given by (2.160):

$$y(k) = \frac{q^{-d-1}B(q^{-1})P(1)}{P(q^{-1})B(1)}Sp + \frac{A(q^{-1})\mathcal{R}(q^{-1})}{P(q^{-1})T(q^{-1})}\xi(k) \tag{3.65}$$

Now, P can be used to define the transfer function from the set point (Sp) to the output ($y(k)$). However, (3.65) also shows that P affects the transfer function from the disturbance $\xi(k)$ to the output of the system and hence the regulator behavior of the system. Moreover, because the controller polynomials $\mathcal{R}$ and $\mathcal{S}$ are given by solving:

$$A\mathcal{R} + q^{-d-1}B\mathcal{S} = PT$$

P has the same effect on the regulator behavior and on the robustness of the closed-loop system as T has. T, however, does not influence the servo behavior, if the process is correctly estimated (see Theorem 2.11).

The discussion above is valid only when a pole-placement controller is selected. However, Corollary 3.4 shows that also for other settings of the UPC design parameters, P and T have exactly the same effect on the regulator behavior and robustness of the system.

From the design point of view, it may be desirable to make the regulator behavior and the robustness independent of P. Then P can be used to define the servo behavior of the closed-loop system while T can be used to define its regulator behavior and its robustness. Two ways to realize this are:

1. Use $P = 1$ in the criterion function (2.158) and generate the reference trajectory by filtering the set point by $1/P$. Now P is present in the feed-forward path of Figure 2.12 only and hence has no effect on the regulator behavior and the robustness of the system.

2. Use P as a factor of $\hat{D}$. Corollary 3.5 shows that in this case P has no effect on the regulator behavior and the robustness of the closed-loop system. Corollary 3.5 is based on the following theorem.

Theorem 3.6 *If $A = \hat{A}$, $B = \hat{B}$, $d = \hat{d}$, $N = P$ and P is a factor of Q_n if $\rho > 0$, then P is a factor of the closed-loop characteristic equation.*

Proof. Consider the criterion function that is minimized in UPC when using the assumptions mentioned in the theorem:

$$J = \sum_{i=H_e}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p-\underline{d}} \left[\frac{Q_n^*}{Q_d} Pu(k+i-1)\right]^2 \tag{3.66}$$

where $Q_n^* P = Q_n$ and $R = P(1)$. This criterion function is minimized taking into account the constraint:

$$\Delta^\beta Pu(k+i-1) = 0 \qquad 1 \le H_c < i \le H_p - \underline{d} \tag{3.67}$$

Further, define:

$$\begin{aligned} y'(k+i) &= Py(k+i) \quad \forall i \\ u'(k+i) &= Pu(k+i) \quad \forall i \end{aligned}$$

Now the optimal controller output sequence over the control horizon is obtained by minimizing:

$$J = \sum_{i=H_e}^{H_p} [\hat{y}'(k+i) - P(1)w(k+i)]^2 + \rho \sum_{i=1}^{H_p-\underline{d}} \left[\frac{Q_n^*}{Q_d} u'(k+i-1)\right]^2$$

taking into account:

$$\Delta^{\beta} u'(k+i-1) = 0 \qquad 1 \leq H_c < i \leq H_p - \underline{d} \tag{3.68}$$

The process model that is used for predicting the process output is given by:

$$Ay(k) = q^{-d} Bu(k-1) + \frac{C}{D} e(k) \tag{3.69}$$

Multiplication by P yields:

$$Ay'(k) = q^{-d} Bu'(k-1) + \frac{PC}{D} e(k)$$

Replacing A, B, d, C and D by their estimates yields:

$$\hat{A} y'(k) = q^{-\hat{d}} \hat{B} u'(k-1) + \frac{PT}{\hat{D}} e(k) \tag{3.70}$$

This model is used for predicting $y'(k+i)$ over the prediction horizon. For this purpose the following Diophantine equation is used:

$$\frac{PT}{\hat{A}\hat{D}} = E_i + q^{-i} \frac{F_i}{\hat{A}\hat{D}}$$

The i-step-ahead predictor is now derived in the same way as described in section 2.1.2. Minimization of the criterion function taking into account (3.68) with respect to $u'(k), \ldots, u'(k+H_c-1)$ yields the optimal controller output sequence over the control horizon for the augmented system (3.70). The closed-loop transfer function for this system obviously becomes :

$$y'(k) = \frac{q^{-d-1} B\mathcal{T}}{A\mathcal{R} + q^{-d-1} B\mathcal{S}} Sp + \frac{\frac{PC\mathcal{R}}{D}}{A\mathcal{R} + q^{-d-1} B\mathcal{S}} e(k)$$

where the controller polynomials $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ can be calculated by minimization of (3.66) in which $P = 1$ with respect to $u(k), \ldots, u(k+H_c-1)$ taking into account (3.67) where $P = 1$ and by using the process (3.69) with T substituted by PT for predicting the process output over the prediction horizon.

Using $y'(k) = Py(k)$ and the assumption that the input/output behavior of the process is correctly estimated, the closed-loop transfer function can be written as:

$$y(k) = \frac{q^{-d-1}BT}{P(A\mathcal{R} + q^{-d-1}BS)}Sp + \frac{\frac{PC\mathcal{R}}{D}}{P(A\mathcal{R} + q^{-d-1}BS)}e(k)$$

This function clearly shows that P is a factor of the closed-loop characteristic equation.

□

Based on this theorem, the following corollaries can be stated.

Corollary 3.3 *Under the conditions mentioned in Theorem 3.6 the closed-loop poles are given by P and the poles that would have been obtained when $P = N = 1$ and $Q_n = Q_n^*$ had been used.*

Proof. The proof follows directly from the proof of Theorem 3.6.

□

Corollary 3.4 *The effect of P on the regulator behavior of the closed-loop system is, under the assumptions mentioned in Theorem 3.6, identical to the influence of T.*

Proof. The regulator behavior of the closed-loop system is, under the assumptions mentioned in Theorem 3.6, given by:

$$\begin{aligned} y(k) &= \frac{\frac{PC\mathcal{R}}{D}}{P(A\mathcal{R} + q^{-d-1}BS)}e(k) \\ &= \frac{\frac{C\mathcal{R}}{D}}{A\mathcal{R} + q^{-d-1}BS}e(k) \end{aligned} \tag{3.71}$$

Further, because the same transfer function is obtained when minimizing (3.66) where $P = 1$ taking into account (3.67) where $P = 1$ and by using PT instead of T in (3.69), P has exactly the same influence on the regulator behavior as T.

□

Corollary 3.5 *If P is a factor of $\hat{D}$, then P does not have influence on the robustness and the regulator behavior of the closed-loop system.*

Proof. The regulator behavior of the closed-loop system where $P \neq 1$ is given by (3.71). The only difference between this function and the one that is obtained when using $P = 1$ is that PT instead of T is used to calculate $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$. By making P a factor of D, the i-step-ahead predictors no longer depend on P. As a result, (3.71) is independent of P and consequently the regulator behavior no longer depends on P.

□

Remarks:

- Theorem 3.6 shows that, as far the servo behavior of the closed-loop system is concerned, the roots of P are added to the set of closed-loop poles obtained when using $P = 1$. If, for example, a dead-beat controller is selected, the closed-loop poles are all in the origin. By choosing P different from 1, the roots of P are added to the set of closed-loop poles which in this case results in the closed-loop poles being equal to the roots of P. Then, by tuning P, the desired response time of the process to set point changes can be obtained (see section 3.6.2). However, if, for example, a mean-level controller is selected, the closed-loop poles are given by the roots of A. Then, adding extra poles to the closed-loop system by selecting P different from 1, can only yield a closed-loop system which responds more slowly to set point changes than the open-loop system. In general, selecting a desired response time by means of P is possible only if the closed-loop poles determined by P dominate the closed-loop poles that are obtained when $P = 1$.
- If $P \neq 1$ and $\hat{D} = \Delta$, then the choice $T = \hat{A}(1 - \mu q^{-1})$ where $\mu = 0, \ldots, 1$ is no longer a good one because for $\mu = 1$ Theorem 3.5 can no longer be applied. By making P a factor of $\hat{D}$ this problem is solved and hence by using Theorem 3.5 the stability margins are guaranteed to be infinite if $\mu = 1$ and if the process is stable.
- In section 3.4 it was argued that the closed-loop system does not depend on H_p if H_p is chosen larger than the settling time of the (discrete) closed-loop system. However, this was based on the situation in which $P = 1$. It has been shown above that P can be used to select the desired closed-loop servo behavior. Hence, with P the settling time of the closed-loop

system can be made arbitrarily large. This gives rise to the question: is the value of H_p for which the closed-loop system becomes independent of H_p affected by P? The answer is simply: if P is a factor of Q_n, then P does not affect the value of H_p for which the system becomes independent of H_p. This can be explained by inspecting the proof of Theorem 3.6. The minimization problem with $P \neq 1$ can be solved by solving a related minimization problem in which $P = 1$. Hence, the prediction horizon for which the system becomes independent of H_p is determined by the settling time of $y'(k)$ and $u'(k)$. This settling time is determined by the settling time of $y(k)$ and $u(k)$ if $P = 1$. Hence, P does not affect the value of H_p for which the closed-loop system becomes independent of H_p.

The following example illustrates the effect of P on the servo and regulator behavior of the closed-loop system.

Example 3.31 Influence of P on the servo and regulator behavior of the closed-loop system.

Settings:	$H_p = 4,\ H_s = 2,\ H_c = 3,\ \beta = 1,\ R = P(1),\ T = 1,\ \hat{D} = 1,$ $\underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)} \quad T_s = 0.5s \Longrightarrow$ $H(z^{-1}) = \dfrac{0.029z^{-1}(1+0.928z^{-1})}{(1-0.882z^{-1})(1-0.905z^{-1})}$
Model:	identical to process
Parameters:	$P = N = 1$ and $P = N = (1 - 0.85q^{-1})$.

Note that H_p, H_s, H_c and ρ are selected such that a pole-placement controller is obtained. Further, white noise filtered by $1/A$ was added to the output of the process (hence, $C = D = 1$). Figure 3.77a shows the response when $P = N = 1$. Now the controller is a dead-beat controller and consequently the response to set point changes is quite fast. Figure 3.77b shows the response when $P = N = (1 - 0.85q^{-1})$. Now, the servo behavior of the closed-loop system is that of a first-order system with a time constant of 3s. However, the figure also shows that the regulator behavior is drastically changed.

Remark: because the DC gain of the process is equal to 5 and the set point is either 1 or -1, changes in the DC level of the controller output are hardly visible in the figure.

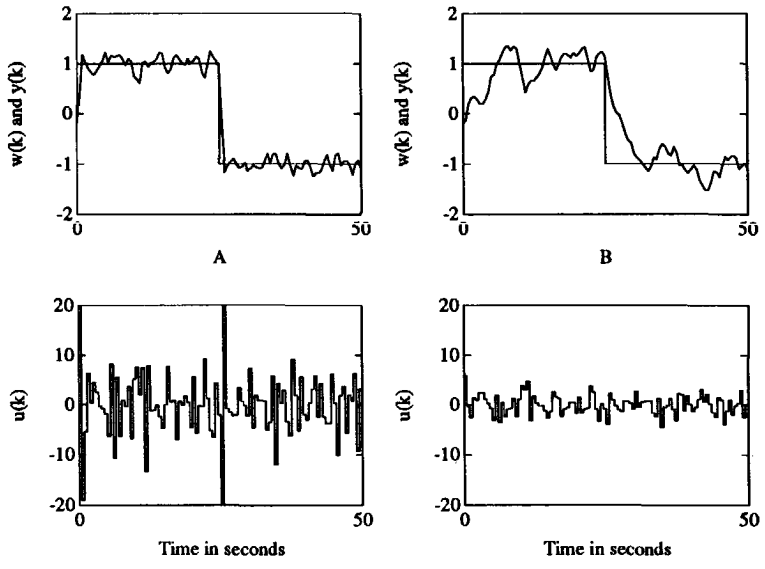

Figure 3.77: Effect of P on the servo and regulator behavior of the closed-loop system. Figure A: $P = N = 1$; Figure B: $P = N = (1 - 0.85q^{-1})$.

In the next example, the effect of P on the regulator behavior is eliminated by making P a factor of $\hat{D}$.

Example 3.32 Influence of P on the servo and regulator behavior of the closed-loop system.

Settings:	Same as in Example 3.31 except that now $\hat{D} = P$
Process:	Same as in Example 3.31
Model:	identical to process
Parameters:	$P = N = (1 - 0.85q^{-1})$.

Figure 3.78 shows the resulting time response. Obviously, by making P a factor of $\hat{D}$, the effect of P on the regulator behavior is canceled (compare the Figures 3.77a and 3.78). The servo behavior of the closed-loop system is still that of a first-order system with a time constant of 3s.

Note: in GPC and in some other predictive controllers, the reference trajectory is generated by using a first-order filter initiated at $y(k)$. It was shown in section 2.4 that this way of generating the reference trajectory is similar to using $P = 1 - \alpha q^{-1}$

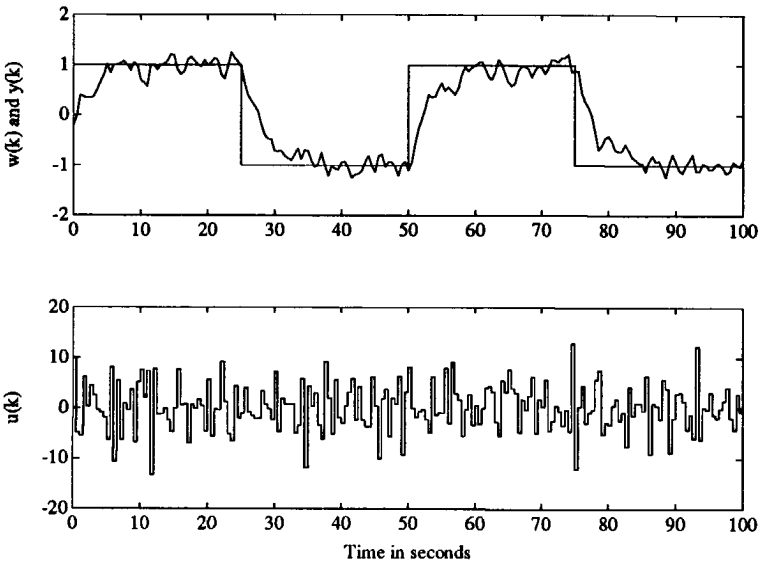

Figure 3.78: Time response where $P = N = (1 - 0.85q^{-1})$ *and* $\hat{D} = P$.

in the criterion function (2.158). Therefore, it can be expected that the influence of the reference trajectory on the regulator behavior and the robustness of the system is similar to using $P = 1 - \alpha q^{-1}$ or $T = (1 - \alpha q^{-1})$.

3.6.2 Design of P and R using a Single Parameter

When the desired closed-loop system is known in terms of poles and zeros the choices for R and P are straightforward. However, it may also be desirable to define the closed-loop servo behavior using a single parameter. For example, the rise time of the response to a step change in the set point.

If a first-order trajectory is desired, R and P can easily be calculated: $R = 1 - \alpha$ and $P = 1 - \alpha q^{-1}$. The parameter α can be related to the desired rise time of the trajectory by using:

$$\alpha = e^{-\frac{2.3T_s}{t_r}}$$

The disadvantage of using a first-order trajectory is that it is quite fast in the beginning and reaches its steady-state value slowly. This typically results in a large controller output after a set point change and a large settling time.

In order to overcome this problem, a second-order trajectory can be used. Now two parameters must be specified: the damping ratio ζ and the natural frequency ω_n. The damping ratio can be fixed at $\zeta = 0.7$. In this case the overshoot is approximately 4.6% and the natural frequency coincides with the desired bandwidth of the closed-loop system. The natural frequency ω_n can be related to the peak time of the step response:

$$t_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} \Longrightarrow \omega_n = \frac{\pi}{t_p\sqrt{1-\zeta^2}}$$

Hence, given the desired peak time of the closed-loop system and assuming $\zeta = 0.7$, ω_n can easily be calculated. Now, the reference model is known in the s-domain. Transforming this continuous model to the discrete domain using a zero-order hold circuit yields for P:

$$P = 1 + p_1 q^{-1} + p_2 q^{-2}$$

where:

$$\begin{aligned} p_1 &= -2e^{-\zeta\omega_n T_s}\cos(\omega_n T_s\sqrt{1-\zeta^2}) \\ p_2 &= e^{-2\zeta\omega_n T_s} \end{aligned}$$

It is common practice not to include the zero and the unit delay of the discrete reference model in the model that is used to define the characteristics of the closed-loop system (see, for example, [45]). Hence, R is given by:

$$R = 1 + p_1 + p_2$$

By using one of the methods mentioned above, the desired speed of the closed-loop system can be adjusted by using a single parameter: the rise or peak time of the closed-loop step response. The following example illustrates the use of a first- and second-order reference trajectory.

Example 3.33 The use of a first and second-order reference trajectory to define the servo behavior of the closed-loop system.

Settings:	Same as in Example 3.31
Process:	Same as in Example 3.31
Model:	identical to process
Parameters:	$P = N = 1 - 0.72q^{-1}$ or $P = N = 1 - 1.398q^{-1} + 0.540q^{-2}$.

The P (and N) polynomials were selected such that both reference trajectories have a rise time of 3.5s. Figure 3.79a shows the response when a first-order reference trajectory is selected ($P = 1 - 0.72q^{-1}$). Figure 3.79b shows the use of a second-order trajectory ($P = 1 - 1.398q^{-1} + 0.540q^{-2}$). Both responses have the same rise time. However, the second-order trajectory makes the system settle faster to the set point. Further, the controller output is smoother than when using a first-order trajectory.

Remark: although the rise times of both responses are equal, the bandwidth of the closed-loop system in the case of a second-order reference trajectory is greater than that when using a first-order trajectory: 0.89 rad/s and 0.66 rad/s, respectively.

3.6.3 Alternative Way of generating the Reference Trajectory

In some predictive controllers (e.g. [26, 5, 1]), the reference trajectory is a first-order trajectory initiated at $y(k)$ rather than at $w(k)$. It was shown in section 2.4 that this way of generating the reference trajectory is closely related to using R and P. Moreover, if $H_p = H_c = 1$, both ways of generating the reference trajectory are identical. If $H_p > 1$, then both ways of generating the reference trajectory are identical only if $\hat{y}(k+i-1) = w(k+i-1)$. However, initiating the reference signal at $y(k)$ introduces an obscure feed-back loop. It is hard to predict what the effect of this feedback will be on the closed-loop system, because it is not obtained as a result of minimization of the criterion function, but added in a heuristic way. That this additional feedback can even make the closed-loop system unstable is shown by the following example.

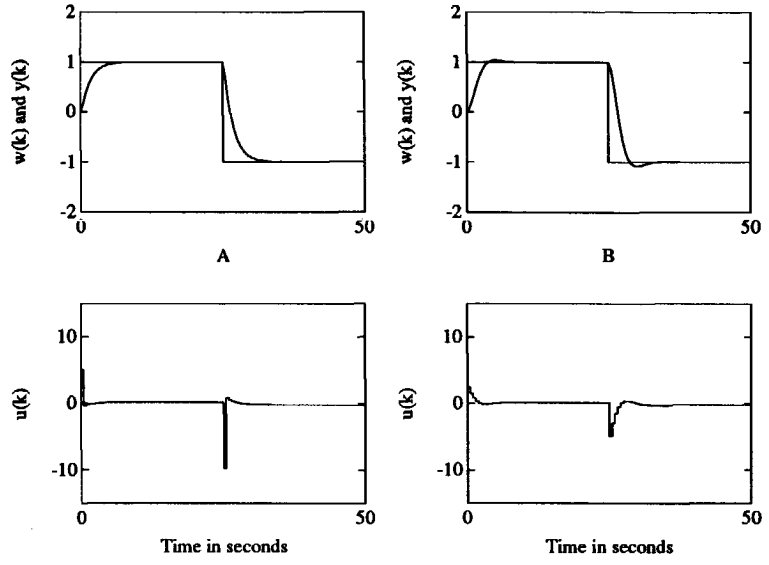

Figure 3.79: The use of a first-order (Figure A) and a second-order reference trajectory (Figure B).

Example 3.34 Effect of different ways of generating the reference trajectory on the servo behavior of the closed-loop system.	
Settings:	$H_p = 25$, $H_s = 1$, $H_c = 3$, $\beta = 1$, $R = P = N = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	Same as in example 3.31
Model:	identical to process
Parameters:	none

In this example, the reference trajectory is generated by using:

$$w(k+i) = (1 + p_1 + p_2)Sp - p_1 w(k+i-1) - p_2 w(k+i-2) \qquad (3.72)$$

where $i = 1, \ldots, H_p$. The parameters p_1 and p_2 are calculated as proposed in the previous section. The desired peak time was selected to be 10s. This results in $p_1 = -1.693$ and $p_2 = 0.735$. Figure 3.80a shows the response when the reference trajectory is initiated at $w(k)$ and $w(k-1)$. The response is as expected. Figure 3.80b shows the response when the reference trajectory is initiated at $y(k)$ and $y(k-1)$.

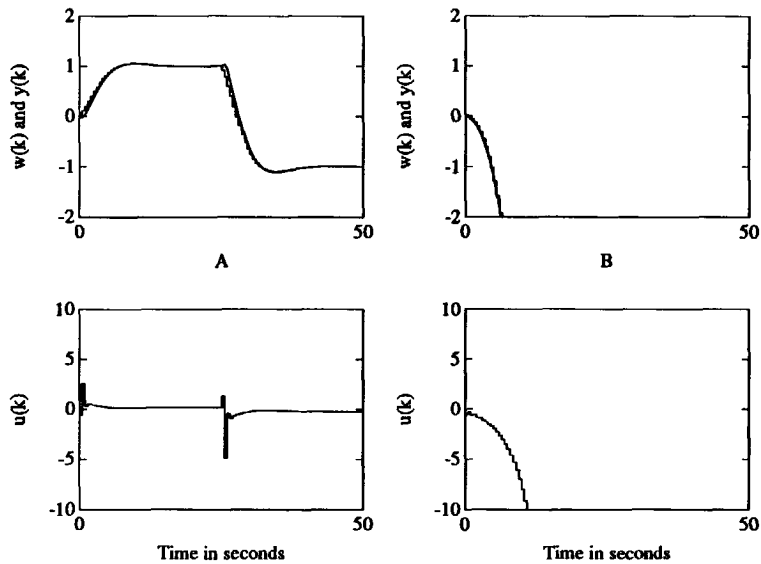

Figure 3.80: Initiating the reference trajectory at $w(k)$ and $w(k-1)$ (Figure A) and at $y(k)$ and $y(k-1)$ (Figure B).

Now, the closed-loop system is unstable due to the additional feed-back loop.

Other simulations showed that in the latter case the behavior of the closed-loop system strongly depends on the prediction horizon. For example, choosing $H_p = 20$ makes the closed-loop system stable, but extremely slow. Decreasing H_p makes the system faster. When the reference trajectory is initiated using $w(k)$ and $w(k-1)$, this effect did not occur.

The following example shows that the effects mentioned above do not appear when using P in the criterion function. Note that now the reference trajectory is generated by: $w(k+i) = Sp$ with $i = 1, \ldots, H_p$.

Example 3.35 Using P in the criterion function in order to define the closed-loop servo behavior.	
Settings:	$H_p = 25$, $H_s = 1$, $H_c = 3$, $\beta = 1$, $R = P(1)$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $P = N = 1 - 1.693q^{-1} + 0.735q^{-2}$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	Same as in Example 3.31
Model:	identical to process
Parameters:	none

The resulting response is shown in Figure 3.81. The figure clearly shows that the

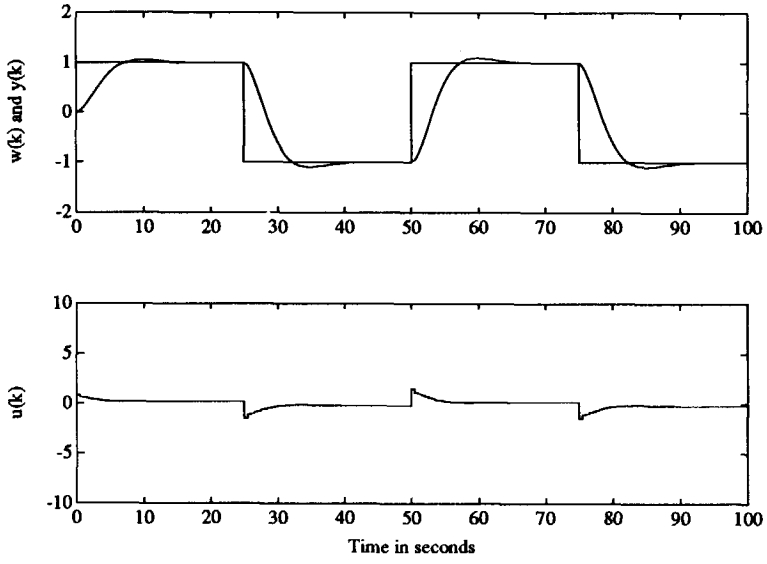

Figure 3.81: Time response in which the desired servo behavior is defined by using R and P in the criterion function.

system behaves as defined by P. Undesirable effects do not occur.

3.6.4 Set Point changes known a priori

In this section the influence of prior knowledge about set point changes on the closed-loop servo behavior is discussed. In section 2.4 it was shown that if set point

changes are known a priori, then the $\tilde{V}$ polynomial is present in the feed-forward path. That this polynomial may result in rather peculiar servo behavior is shown by the following example.

Example 3.36 Influence of using prior knowledge about set point changes in generating the reference trajectory.

Settings:	$H_p = 10$, $H_s = 1$, $H_c = 2$, $\beta = 1$, $R = P = N = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$
Process:	Same as in example 3.31
Model:	identical to process
Parameters:	none

Figure 3.82 shows the response when set point changes are known in advance. Although the response is still optimal with respect to the criterion function that is

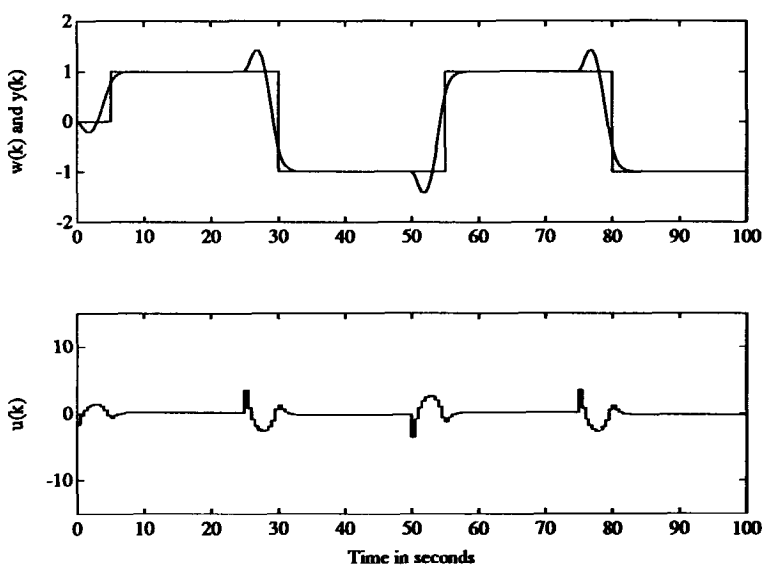

Figure 3.82: Time response when set point changes are known in advance.

minimized, the closed-loop system becomes non-minimum phase. Increasing the prediction horizon does not make this effect disappear. In order to explain this phenomenon, consider the criterion (2.158) with $P = R = 1$, $H_s = 1$, $\rho = 0$ and $\beta = 1$:

$$J = \sum_{i=1}^{H_p} [\hat{y}(k+i) - w(k+i)]^2$$

This criterion is minimized taking into account:

$$\Delta u(k+i) = 0 \qquad H_c < i \leq H_p - 1$$

In the example $H_c = 2$. Hence, $u(k)$ and $u(k+1)$ can freely be chosen. All other controller outputs over the prediction horizon are equal to $u(k+1)$. There are four possibilities for the controller to choose $u(k)$ and $u(k+1)$:

1. $u(k)$ is negative and $u(k+1)$ is positive. All controller outputs over the prediction horizon are positive except $u(k)$. Figure 3.83a shows a typical response.
2. $u(k)$ and $u(k+1)$ are both positive. Now, all controller outputs over the prediction horizon are positive. A typical response is shown in Figure 3.83b.
3. $u(k)$ and $u(k+1)$ are both negative. As a result, all controller outputs are negative making the predicted process outputs over the prediction horizon also negative which results in a bad criterion value. Therefore, this possibility is not considered.
4. $u(k)$ is positive and $u(k+1)$ is negative. Now, all controller outputs except $u(k)$ are negative resulting in a negative going process output. This possibility, having the same disadvantage as the previous one, is not considered either.

Comparing the responses depicted in the Figures 3.83a and 3.83b shows that when $u(k)$ is negative, a smaller criterion value is obtained. Because the controller outputs are calculated such that the criterion function is minimal, the first possibility is used to control the process. This obviously results in the response shown in the Figures 3.83a and 3.82. Hence, the criterion function in combination with the control horizon causes the typical response shown in Figure 3.82. Knowing this, two ways to avoid the non-minimum phase effect are:

- Use $H_c = 1$. Now, only $u(k)$ can freely be chosen the controller. All other controller outputs are equal to this value. Obviously, $u(k)$ is positive which results in the response shown in Figure 3.83b. Figure 3.84a shows the response obtained in a simulation where $H_c = 1$. All other settings were equal to those given in Example 3.36.

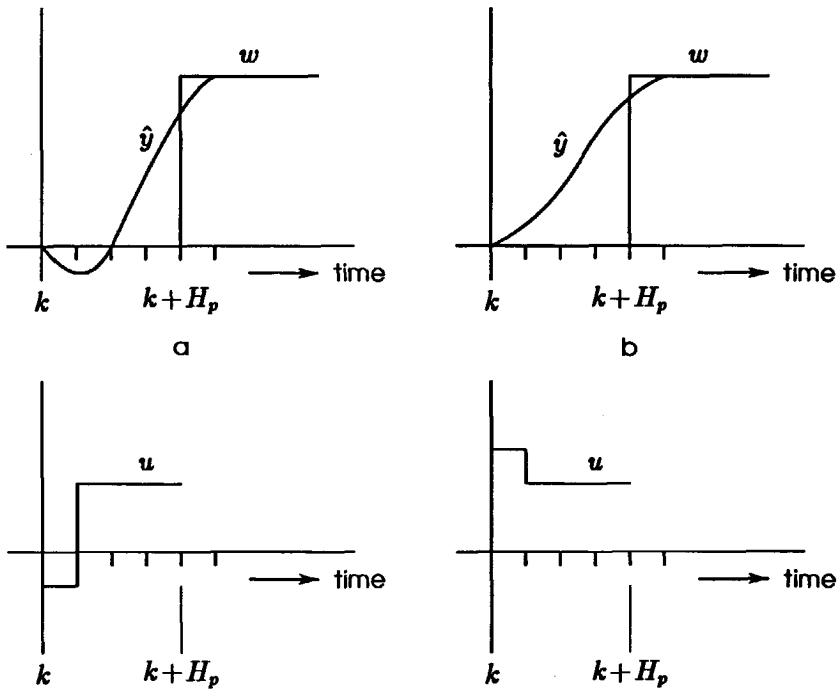

Figure 3.83: Two possibilities to generate the controller outputs when the set point change is known in advance and $H_c = 2$.

- Use $H_s = H_p$. Now, a large tracking error at $t = k+1, \ldots, k+H_p-1$, as in Figure 3.83b, is not included in the criterion function and hence will not affect the controller outputs. The resulting response is shown in Figure 3.84b. All settings were equal to those in Example 3.36 except that $H_s = 10$. Further, a small value for ρ ($\rho = 10^{-6}$) has been used in order to obtain a unique solution to the optimization problem.

3.6.5 Conclusions

It was shown that the design parameters R and P can be used to define the servo behavior of the closed-loop system. However, the P polynomial has the same influ-

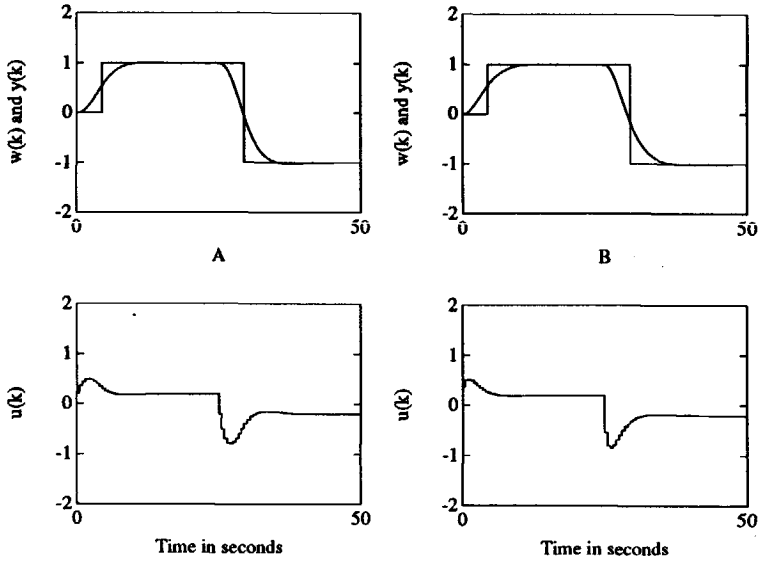

Figure 3.84: Time responses when set point changes are known in advance where $H_c = 1$ *(Figure A) and* $H_s = 10$ *(Figure B).*

ence as T on the regulator behavior and the robustness of the system. This influence can be eliminated by making P a factor of $\hat{D}$ or by using $P = 1$ in the criterion together with generating the reference trajectory by using (3.72) initiated at $w(k)$ and $w(k-1)$. Then, P can be used to define the servo behavior and T can be used to define the regulator behavior and the robustness of the closed-loop system.

Generating the reference trajectory by using (3.72) initiated at $y(k)$ and $y(k-1)$, as an alternative to using P in the criterion function, is not advised because then again the regulator behavior and the robustness are influenced by the trajectory. Moreover, an obscure feed-back loop is introduced which in some situations can make the closed-loop system unstable.

Finally, it was shown that using prior knowledge about set point changes can yield a non-minimum phase response although the process is minimum phase. It has been explained that this effect is caused by the criterion function in combination with the control horizon.

3.7 The Sampling Period

As in any discrete controller, the sampling period plays an important role. In [19] a detailed discussion can be found on how the sampling period can be chosen. There, as a rule of thumb, it is suggested to choose T_s 10−20 times smaller than the settling time of the closed-loop system. In this section, only the influence of the sampling period on the UPC design is discussed. It has already been shown that the prediction horizon can be chosen equal to $H_p = \text{int}(t_s(5\%)/T_s)$. Therefore choosing a small sampling time yields a large prediction horizon. Depending on the computer UPC is implemented, this can cause problems because of the large amount of memory that is involved in using large prediction horizons. However, if the rule of thumb mentioned above for selecting the sampling period is used, the value for H_p is not extremely large: $H_p = 10 - 20$.

In fact, all UPC design parameters that are polynomials are affected by the sampling period. For most polynomials the relation is quite clear. The polynomials of the model are directly estimated or obtained by applying the z-transform to the Laplace representation of the model. The P, Q_n and Q_d polynomials are derived in a similar way. The influence of T_s on the T polynomial was discussed in section 3.5.2. If, for example, $T = (1 - \mu q^{-1})^2$ with $\mu = 0.8$ is selected, making the sampling period smaller yields a larger value for μ. When $T = \hat{A}$ is used, the influence of T_s on T is automatically taken into account.

Another problem that occurs if the sampling period is small is that if the process contains time delay, this results in a large value for $\hat{d}$ which in turn results in large matrices and a large controller order.

To conclude: as in any discrete controller, the sampling time is an important parameter. In UPC (or in fact in any predictive controller) some additional problems can be expected if the sampling period is small in relation to the settling time of the process. This results in a large prediction horizon and a large controller order (if the process contains time delay) requiring a large amount of memory and possibly causing numerical problems. However, the smaller T_s the better a reference trajectory can be tracked or a disturbance can be rejected.

Up to now, numerical and memory problems have not occurred for $H_p \leq 25$ and $\hat{d} \leq 10$ (on a PC with 640 kB of memory).

3.8 Review of some well-known Predictive Controllers

In this section a review is given of the predictive controllers: DMC [2], PCA [26], MAC [1], GPC [5], EPSAC [7, 11] and EHAC [10].

Because the UPC controller is a unification of the controllers mentioned above, the results presented in Chapter 2 and this chapter can be used to make some remarks concerning these controllers. The following table recalls how the UPC design parameters must be selected in order to obtain the criterion function that is minimized in the above-listed controllers (see also Table 2.3). In Table 3.8 the

Controller	H_p	H_s	H_c	N	β	P	R	ρ	Q_n	Q_d
DMC[a]	$\surd$	$d+1$	$\surd$	1	1	1	1	$\surd$	Δ	1
PCA	$\surd$	$d+1$	H_p-d	-	-	1	1	$\surd$	Δ	1
MAC	$\surd$	$d+1$	H_p-d	-	-	1	1	$\surd$	Δ	1
GPC	$\surd$	$\surd$	$\surd$	1	1	$\surd$	$\surd$	$\surd$	Δ	1
EPSAC[b]	$\surd$	$d+1$	1	1	1	$\surd$	$\surd$	0	-	-
EHAC[c]	$\surd$	H_p	1	1	1	1	1	0	-	-

Table 3.8: Criterion parameters for several well-known controllers.

[a] In the original paper on DMC [2], controller output weighting was not considered. However, more recently ([11, 46]) controller output weighting is considered by means of ρ in DMC.

[b] EPSAC is considered in the case $\lambda = 1$. Only in this case, can the criterion that is minimized in EPSAC be obtained by selecting particular parameter settings in the UPC criterion function. See also section 2.2.2 and 3.8.5.

[c] Within EHAC different strategies can be used to obtain the controller. Here only one of the possibilities is considered. See also section 3.8.6.

symbol '$\surd$' denotes that the parameter is a design parameter of the UPC controller and the controller that is listed in the same row of the table. Further, the symbol '-' denotes that the corresponding UPC design parameter may have any value.

As discussed in Chapter 2, not only the criterion function but also the model that is used plays an important role in predictive controller design. Table 3.9 shows how the parameters of the unified process model (2.1) must be selected to obtain the model that is used in the predictive controllers mentioned above. Because in DMC, PCA and MAC, the predictions based on the FSR and FIR models are corrected by adding the term $y(k) - \hat{y}(k)$ to the predictions, the noise model is given by $C = 1$ and $D = \Delta$ (see section 2.1.3, page 25 for a more detailed discussion).

Because all predictive controllers are based on the same concept (see Chapter

Controller	Model	A	B	C	D
DMC	FSR	1	$\surd$	1	Δ
PCA	FIR	1	$\surd$	1	Δ
MAC	FIR	1	$\surd$	1	Δ
GPC	ARIMAX	$\surd$	$\surd$	$\surd$	Δ
EPSAC	ARIMAX	$\surd$	$\surd$	$\surd$	Δ
EHAC[a]	ARIX	$\surd$	$\surd$	1	Δ

Table 3.9: Process models used by predictive controllers.

[a]Originally, the EHAC controller uses $D = 1$. However, in [11] an incremental version of EHAC is proposed yielding $D = \Delta$. This version is considered here.

1), they all can handle processes with time delay. In fact, this is probably the reason why predictive controllers have been used successfully in real-life applications.

In the following sections each controller mentioned in Table 3.8 is discussed separately.

3.8.1 The DMC Controller

A detailed discussion with respect to DMC can be found in [46]. Based on Table 3.8 and Table 3.9 the following conclusions can be drawn with respect to the DMC controller:

- The use of a finite step response model has as a consequence that only stable processes without integrators can be modeled and controlled.
- Because $\hat{D} = \Delta$, $\beta = 1$ and $Q_n = \Delta$, steady-state errors do not occur for constant set points and/or disturbances. However, non-constant reference trajectories and/or disturbances can yield steady-state errors.
- The noise model $T/\hat{A}\hat{D}$ is fixed to $1/\Delta$. Therefore, the controller cannot be optimal, in the sense of minimum variance, to other kinds of noise.
- The polynomial Q_n is fixed to Δ and Q_d is fixed to 1. As a result, increments are weighted. It was shown in section 3.3, that a badly damped or even an unstable closed-loop system can be the result. Further, because Q_n and Q_d are fixed, eliminating certain frequencies in $u(k)$ is not possible.

- The design parameters of the DMC controller make it possible to realize mean-level and minimum-settling time controllers only (see Table 2.4).
- The robustness of the DMC controller can be influenced by H_p, H_c and ρ. If $\rho = 0$, $H_c = 1$ and H_p large, mean-level control is obtained. The upper bound for additive modeling errors as given by (3.50) now becomes:

$$|\Delta_a| < \hat{K}_{dc} \qquad \forall \omega \geq 0$$

  Obviously, the robustness of the DMC controller with $H_c = 1$ and H_p large, is equal to that of mean-level control based on an ARIMAX model with $T = \hat{A}$ and $\hat{D} = \Delta$. Moreover, also for other settings of H_c and H_p, the robustness of DMC is equal to that of the UPC controller with $T = \hat{A}$ and $\hat{D} = \Delta$. It was shown in section 3.5 that for this choice for T and $\hat{D}$, a compromise is obtained between robustness and regulator behavior. This is probably the reason why many successful applications have been reported using DMC.
- Since, $R = P = 1$, the servo behavior must be defined by generating an appropriate reference trajectory.

3.8.2 The PCA Controller

The PCA controller [26] is based on a finite impulse response model of the process. In section 2.1.3 it was shown that the noise model implicitly used in PCA is given by $T = 1$ and $\hat{D} = \Delta$. The reference trajectory is generated in the same way as in GPC. The major difference between the PCA controller and, for example, the DMC controller is that the PCA controller does not consider a control horizon (i.e. $H_c = H_p - d$). Now the following remarks can be made with respect to the PCA controller:

- One consequence of the use of a finite impulse response model is that only stable processes without integrators can be modeled and controlled.
- Because $\hat{D} = \Delta$, $\beta = 1$ and $Q_n = \Delta$, steady-state errors do not occur for constant set points and/or disturbances. However, non-constant reference trajectories and/or disturbances can yield steady-state errors.
- The noise model $T/\hat{A}\hat{D}$ is fixed to $1/\Delta$. Therefore, the controller cannot be optimal, in the sense of minimum variance, to other kinds of noise.

- Owing to the fact that the control horizon is fixed to $H_p - d$, it is not possible to select mean-level, dead-beat or pole-placement controllers. When $\rho = 0$, minimum-variance control is obtained independently of H_p. As a result, ρ must be taken different from zero in almost all applications. However, it is quite difficult to select a value for ρ a priori.

- The polynomial Q_n is fixed to Δ and Q_d is fixed to 1. As a result, increments are weighted. Consequently, the closed-loop system can be badly damped or even unstable. Further, because Q_n and Q_d are fixed, eliminating certain frequencies in $u(k)$ is not possible.

- As in MAC, the reference trajectory is generated by using a first-order filter initiated at $y(k)$. This way of generating the reference trajectory has been shown to be similar to using $P = 1 - \alpha q^{-1}$ in the criterion function (see section 2.4). However, in section 2.2.1, it was shown that by doing so problems with $\rho = 0$ and $H_c = H_p - d$ are not solved. Moreover, it was shown that now an obscure feed-back loop is activated which in some situations can make the closed-loop system unstable (see Example 3.34 at page 206).

3.8.3 The MAC Controller

The MAC controller, introduced by Richalet in 1977 [1], is also based on a finite impulse response model of the process. Originally, there was no correction of the predictions based on the FIR model. Hence, the noise model implicitly used was given by: $T = 1$ and $\hat{D} = 1$. As a result, a constant reference trajectory and/or constant disturbances can yield steady-state errors. In more recent papers (for example, [11]), the predictions based on the FIR model are also corrected by adding $y(k) - \hat{y}(k)$ to the predictions. Now, this version of MAC implicitly uses $T = 1$ and $\hat{D} = \Delta$ as noise model. However, the major difference between the MAC controller and the PCA controller is that in PCA the criterion is minimized analytically while in MAC an iterative optimization method is used for this purpose.

3.8.4 The GPC Controller

The GPC controller proposed by Clarke *et al.* [5, 6] uses an ARIMAX model. Hence, stable as well as unstable processes can be modeled and the $\hat{D}$ polynomial is equal to

Δ. The reference trajectory is often generated by a first-order filter initiated at $y(k)$ as described in section 2.4. However, also the use of P is considered for defining the servo behavior of the closed-loop system. Relations between GPC, pole-placement design and LQ control were reported in [31] and [18]. There, it is shown that pole-placement design can be considered as a subset of GPC.
The major differences with UPC are:

- The $\hat{D}$ polynomial is fixed to Δ. Therefore, not all noise spectra can be dealt with. However, the most important ones: measurement noise, Brownian motion, noise filtered by the system dynamics and load changes can be taken into account.
- Because $\hat{D} = \Delta$, $Q_n = \Delta$ and $\beta = 1$, non-constant reference trajectories and/or disturbances can yield steady-state errors.
- The polynomial Q_n is fixed to Δ and Q_d is fixed to 1. As a result, increments are weighted. It was shown in section 3.3, that a badly damped or even an unstable closed-loop system can be the result. Further, because Q_n and Q_d are fixed, eliminating certain frequencies in $u(k)$ is not possible.
- In GPC, the polynomials P and N are not explicitly present in the criterion function and control horizon definition. They can, however, be taken into account in a different way [31].

Similar to UPC, GPC considers the T polynomial as a tool to influence the robustness and regulator behavior of the closed-loop system. The choice $T = (1 - \mu q^{-1})^{n_A}$ as used in GPC yields, in most cases, satisfactory results. The default value for μ ($\mu = 0.8$) suggested by Clarke has been shown to yield good results in almost all simulations.

3.8.5 The EPSAC Controller

The EPSAC controller is described in detail in [7] and [11]. As discussed in section 2.2.2, the EPSAC controller has an additional parameter λ in the criterion function that realizes weighting of the future tracking error. Such a parameter is not present in the UPC criterion function. As a result, EPSAC cannot be considered as a full subset of UPC. However, it has been shown that the effect of this parameter is similar to that of the minimum cost horizon H_s (in fact, this was the reason not to include λ

in the UPC design).
The major differences between EPSAC and UPC are:

- The $\hat{D}$ polynomial is fixed to Δ. Therefore, not all noise spectra can be dealt with. However, the most important ones: measurement noise, Brownian motion, noise filtered by the system dynamics and load changes can be taken into account.

- Because $\hat{D} = \Delta$ and $\beta = 1$, non-constant reference trajectories and/or disturbances can yield steady-state errors.

- Controller output weighting by means of ρ is not considered. As a result, eliminating certain frequencies in the controller output is not possible.

- Because $H_c = 1$, only mean-level control and minimum-variance control can be obtained for particular settings of H_p. It has been shown that with $H_c = 1$ the robustness, the servo and the regulator behavior of the closed-loop system heavily depend on H_p. As a result, an a priori choice for H_p can be quite difficult to select. Therefore, in contrast with UPC, one must be careful when choosing H_p. The rule of thumb derived in section 3.4 cannot always be used because it yields a large prediction horizon which, together with $H_c = 1$, results in mean-level control.

  In [11], the prediction horizon is chosen equal to $H_p = d+5$. This value for H_p results in an acceptable behavior for most processes. However, using this value in controlling the non-minimum phase process, used in Example 3.12 on page 141, yields an unstable closed-loop system. Then, a larger value for H_p should have been chosen. In this case, choosing $H_p \geq 9$ results in a stable closed-loop system. However, now the unstable process used in Example 3.15 (page 147) can no longer be controlled.

  To conclude: Compared with UPC, EPSAC is less well suited for controlling unstable processes. For well-damped and stable processes, the choice $H_c = 1$ together with a prediction horizon which is not extremely large (for example, $H_p = d + 5$), mostly yields acceptable results.

  An advantage of choosing $H_c = 1$ is that the controller output can easily be calculated (no matrix inverse is required).

- The minimum cost horizon H_s is not included in the EPSAC design. As has already been mentioned, the parameter λ that is present in the EPSAC criterion function has a similar effect. However, in contrast to GPC and UPC, pole-placement control cannot be considered as a subset of EPSAC.

- As in UPC, in EPSAC the P polynomial is used to define the desired servo behavior of the closed-loop system. However, because the P polynomial is not a factor of $\hat{D}$, P not only affects the servo behavior, but also the regulator behavior and the robustness in a way similar to T.

3.8.6 The EHAC Controller

The EHAC controller was proposed by Ydstie in 1984 [10]. Originally, a noise model was not considered. However, in [11] an incremental version of EHAC has been proposed that guarantees the absence of steady-state errors in the presence of constant disturbances. This results in $T = 1$ and $\hat{D} = \Delta$. This version of EHAC is discussed here.

The main characteristic of EHAC is that the minimum cost horizon is equal to the prediction horizon and that controller output weighting by means of ρ is absent. In combination with $R = P = 1$, the criterion function that is minimized with respect to the controller output sequence over the prediction horizon, is:

$$J = [\hat{y}(k + H_p) - w(k + H_p)]^2$$

Obviously, there is no unique solution to this minimization problem. There are an infinite number of controller output sequences that yield $J = 0$ and hence minimize the criterion. There are many strategies that lead to unique controls. One of them is to use $H_c = 1$. Now, all controller outputs over the prediction horizon are equal to $u(k)$. The minimization problem can now be formulated as: find such a $u(k)$ that J is minimal, taking into account that $\Delta u(k + i - 1) = 0$ for $1 < i \leq H_p - d$. It can easily be shown that also in this case the minimal value of J is equal to zero. Now, the prediction horizon obviously influences the response time of the closed-loop system (see also section 3.4.2, page 156).
The following remarks can be made with respect to EHAC:

- In the case of a well-damped minimum phase process, H_p can be chosen equal to the desired response time of the closed-loop system. However, if the process is badly damped, unstable or non-minimum phase, finding a good value for H_p is more difficult. This is, among other reasons, due to the fact that H_c is equal to one, which results in mean-level control if H_p is large. Mean-level control is unadvisable for controlling badly damped or unstable processes because it results in a badly damped or an unstable closed-loop system.

- Controller output weighting is not considered. Hence, eliminating certain frequencies in the controller output is not possible.
- A T polynomial different from 1 is not considered. Therefore, the only parameter that can be used to change the robustness or regulator behavior is H_p. However, by changing H_p not only the robustness and regulator behavior of the system is changed, but also the system's servo behavior.
- The tracking of a preprogrammed reference trajectory is not possible because the tracking error at $t = k + H_p$ is included in the criterion function only.
- Pole-placement design is not a subset of EHAC. Only mean-level and minimum-variance control can be obtained by selecting the EHAC design parameters.

3.8.7 Conclusions

In this section a number of well-known predictive controllers have been reviewed. It has been shown that GPC especially is closely related to UPC. However, UPC has some additional features:

- Steady-state errors can be avoided also for non-constant disturbances and reference trajectories.
- In UPC, Q_n and Q_d are not fixed. Therefore, certain frequencies in the controller output can be attenuated by selecting Q_n and Q_d in combination with $\rho > 0$.
- The polynomials P and N can be directly used to obtain a pole-placement controller. In combination with making P a factor of $\hat{D}$, P can be used to specify the desired servo behavior and T can be used to specify the desired robustness and regulator behavior of the closed-loop system.

Both GPC and UPC are capable of controlling badly damped or unstable processes. Further, they both allow pole-placement design. All other controllers discussed in this section are less, or not at all, suited for controlling such processes. Further, for all controllers discussed in this section, the following holds:

- Processes with time delay can be controlled.

- Constant reference trajectories and constant disturbances do not result in steady-state errors.
- Attenuation of certain frequencies in the controller output is not considered.

In the next section it is shown that although UPC has more parameters than any other predictive controller, it can be tuned more easily.

3.9 Tuning of the UPC design Parameters

In this section the tuning of the UPC design parameters is discussed (see also [47]). The tuning procedure of the UPC design parameters can be divided into two parts:

- Find some initial settings for the parameters. Rule of thumb methods are used for this purpose.
- If the characteristics of the closed-loop system tuned by means of the rule of thumb methods are not satisfactory, use fine tuning procedures to achieve better results.

In the next section the rule of thumb methods are discussed.

3.9.1 Rule of Thumb Methods

Based on the results shown in the various sections, rule of thumb methods can be formulated that provide (initial) settings for the UPC design parameters. Because there is a slight difference between the rule of thumb methods used for well-damped and badly-damped processes (among which unstable processes), they are discussed separately.

Rule of thumb methods for well-damped processes

In this section the rule of thumb methods are formulated for processes with well-damped poles. Processes containing integrators are also considered as such.

$H_s = \underline{d} + 1$. This value for H_s is based on the simulation results that have been presented in this chapter, as well as on the fact that, now, the tracking error over the complete prediction horizon is included in the criterion. It was shown that other values for H_s sometimes cause unpredictable results. Only in the case where UPC is operated as a pole-placement controller, can other values for H_s be considered (see Table 3.2).

$H_c = n_A$. This value for H_c has been shown to yield a compromise between robustness and performance. It is also based on the fact that for $H_c = n_A + 1$, pole-placement control is obtained (if H_s and H_p are chosen accordingly). A (slightly) smaller value for H_c then yields a more robust control system which still has an acceptable performance.

$H_p = \text{int}(t_s(5\%)/T_s)$ where $t_s(5\%)$ is the 5% settling time of the step response of the (continuous) process. Simulations have shown that this value for H_p yields low sensitivity of the closed-loop system to changes in H_p. Note that H_p is directly related to the sampling time of the process. Using a small sampling time in relation to the settling time of the step response of the process yields a large prediction horizon.

$\beta = \max(\max(r,p) - n, 1)$ where r and p denote the type of the reference trajectory and disturbance and n denotes the number of integrators in the process. Now steady-state errors do not occur and β is as small as possible in order to avoid stability problems.

$\rho = 0$. For most applications, the weighting factor ρ can be taken equal to zero. Then, Q_n and Q_d can be taken equal to 1.

$\hat{D} = P\Delta^\beta$. Together, with the choice of β (Q_n does not play a role because $\rho = 0$), steady-state errors do not occur. Further, P does not affect the servo-behavior and the robustness of the closed-loop system.

$T = \hat{A}$. This value for T can be used only if the poles of the model are well damped. Further, simulations in combination with theoretical results have shown that for this choice of T a nice compromise between good robustness and good regulator behavior is obtained. Ideally, T and $\hat{D}$ should be taken equal to C and D, respectively, in order to get good noise rejection. However, since C and D are often not known, T can be chosen as described above.

$P = 1 + p_1 q^{-1} + p_2 q^{-2}$ where p_1 and p_2 are calculated by specifying the desired peak time of the closed-loop system (see section 3.6.2). The reference trajectory is in this case equal to the set point (hence, $\boldsymbol{w} = [Sp, \cdots, Sp]^T$).

Further, by assumption $R = P(1)$ and $N = P$.

Rule of thumb methods for badly-damped processes

In this section, the rule of thumb methods are formulated for processes with badly-damped poles. Most of the methods mentioned above for well-damped processes can also be used for badly-damped processes. Only those that are different are discussed here.

$H_p = \text{int}(3t_r/T_s)$ where t_r is the rise time of the process.

$T = (1 - \mu q^{-1})^{n_A}$ with $\mu = 0.8$. Because the roots of T appear as poles in the closed-loop system using $T = \hat{A}$ is not possible. The choice $T = (1-\mu q^{-1})^{n_A}$, however, can also be used for well-damped processes.

Thus far, it has not been made clear when a process is called badly damped. Whether or not a process is badly damped is for the designer to decide. If the controller is tuned by means of the rule of thumb methods for well-damped processes, then a load change results in an oscillating process output if the process is underdamped. This is caused by the fact that the roots of T, now being equal to $\hat{A}$, appear as closed-loop poles. If such an oscillation is not acceptable, then the rule of thumb methods for badly-damped processes must be used.

By using the rule of thumb methods mentioned above, initial settings for the UPC design parameters are obtained. Usually, the resulting response is quite acceptable. However, in some situations fine tuning may be desired. In the next section, some guidelines are given.

3.9.2 Fine Tuning of the UPC design Parameters

In this section some fine tuning procedures are discussed that can be used if the closed-loop system tuned by means of the rule of thumb methods does not behave satisfactorily. Based on theoretical results and the simulations presented in the Chapters 2 and 3, the following parameters are <u>not</u> considered as tuning parameters: Q_n, Q_d, β, H_s, $\hat{D}$ and H_p. These parameters have either less influence on the closed-loop system (e.g. H_p) or their influence is hard to predict (e.g. H_s). Only in the case where $H_c = 1$, can H_p be considered as a tuning parameter because then its influence

on the closed-loop system is rather great. The parameters Q_n, Q_d, β and $\hat{D}$ are obtained by applying theorems concerning the steady-state behavior and the regulator behavior. This leaves T, P, ρ and H_c as possible tuning parameters (remember that, by definition, $R = P(1)$ and $N = P$).

Fine tuning may be required in many situations. Here, only a few will be discussed, which are, in my opinion, the most important ones. Fine tuning for stable processes is considered in the following sections.

Controller output variance too large

If the controller output variance is too large (the controller is too active) basically two approaches can be used to decrease the controller output variance.

1. If the too-large controller output variance is caused by noise acting on the system, then the T polynomial can best be used to decrease the variance of the controller output. When $T = \hat{A}$, T can be extended to $T = \hat{A}(1-\mu q^{-1})^{\delta}$. Now increasing μ makes the controller output less active. Consequently, the noise rejection is worse and load changes are rejected more slowly. However, the robustness of the system is improved. Further, if the process is different from the model, the response of the system to set point changes becomes slower. When $T = (1 - \mu q^{-1})^{n_A}$, μ can be increased in the way described above to reduce the controller output variance. Then, however, the robustness of the closed-loop system generally does not improve continuously as μ increases.

2. If set point changes cause large controller outputs, then you have probably selected too small a response time for the system. Increasing the desired peak time of the system and therefore retuning P makes the controller output less active. As a result, the closed-loop system responds more slowly to set point changes. The robustness and the regulator behavior are not influenced by P and they, therefore, do not change.

Usually, the desired result can be obtained by using the methods described above. However, if this is not the case, the following methods can be utilized:

1. Decrease H_c. Because, as a rule of thumb, H_c is chosen equal to n_A, there are only a few possibilities for choosing H_c (in the case of a second-order process $H_c = 1$ is the only possible choice). Decreasing H_c makes the closed-loop system slower and more robust. Further, the rejection of noise and load changes

is worse. Only in the case where decreasing H_c results in $H_c = 1$, can the prediction horizon be used as a (fine) tuning parameter. Now, the closed-loop system is quite sensitive to H_p. Increasing H_p results in mean-level control which makes the system more robust but slower.

2. Increase the weighting factor ρ. To avoid steady-state errors and possible stability problems, the choice for Q_n and Q_d becomes important. Choosing $Q_n = P\Delta^\beta$ and $Q_d = P(1 - 0.95q^{-1})^\beta$ guarantees that there are no steady-state errors and simultaneously ensures that instability due to ρ is avoided. Further, P was made a factor of Q_n in order to make sure that the roots of P remain poles of the closed-loop system when $\rho > 0$ (see Theorem 3.6).

As long as the poles specified by P dominate over the closed-loop poles that would have been obtained by choosing $P = 1$, the response of the system to set point changes is determined by P and hence by the specified peak time.

Load changes rejected too slowly

By choosing β, $\hat{D}$ and Q_n as specified by the rule of thumb methods, load changes are always rejected. However, they can be rejected too slowly. In this case, the tuning parameters ρ, T and H_c (in this order) can be used to make the load changes be rejected faster (remember that P does not influence the regulator behavior of the system).

If, for some reason, ρ was taken different from zero, decreasing ρ makes load changes be rejected faster. However, ρ is usually taken equal to zero. Then, T and H_c must be used.

- If $T = \hat{A}$, T can be chosen equal to to $T = (1 - \mu q^{-1})^{n_A}$ with $\mu = 0.8$. Now decreasing μ ensures that load changes are rejected faster. Consequently, the controller output becomes more active. In general, the noise rejection improves in contrast to the robustness of the system which becomes worse. Further, if the process is different from the model, the response of the closed-loop to set point changes becomes faster.

- If $H_c = 1$, H_c can be increased in order to make the rejection of load changes faster. However, H_c must not be taken greater than n_A. Further, it might be necessary to use T in combination with H_c to obtain the desired result.

The set point tracked too slowly

If the response to set point changes is slower than specified by P (and hence by the specified peak time), the roots of P obviously do not dominate over the poles of the closed-loop system that would have been obtained where $P = 1$. Hence, the only way to improve the tracking of the set point, is to increase the speed of the system when $P = 1$. This can be done as follows:

1. If ρ is different from zero, decrease ρ.
2. Increase H_c if $H_c < n_A$.
3. Decrease μ if $T = \hat{A}(1 - \mu q^{-1})^{\delta}$ or $T = (1 - \mu q^{-1})^{n_A}$. Only when the slow tracking is caused by model mismatch, does this result in a faster closed-loop system.

All of the above-mentioned methods yield a less robust control system. Therefore, their use may be restricted.

Insufficient robustness

If the system has insufficient robustness, the closed-loop system can become unstable in the presence of unmodeled dynamics. In practice, unmodeled dynamics are always present because the model that is used is, in general, of a lower order than the real process. Therefore, some robustness is always required. Three ways to improve the robustness of the closed-loop system are:

1. Increase μ in $T = \hat{A}(1-\mu q^{-1})^{\delta}$. Now the robustness of the closed-loop system increases monotonously with μ. For $\mu = 1$, infinite robustness margins are obtained because then there is no longer feedback. Consequently, the rejection of load changes and noise becomes worse as μ increases. When $\mu = 1$, load changes and noise are no longer rejected. The servo behavior of the system is influenced only if the process is different from the model. The greater this difference, the greater the influence of T on the system's servo behavior.
2. Decrease H_c. Because, as a rule of thumb, H_c is chosen equal to n_A, there are only a few possibilities for choosing H_c (in the case of a second-order process, $H_c = 1$ is the only possible choice). Decreasing H_c also makes the closed-loop

system slower if the poles of the system that would have been obtained when $P = 1$, dominate over the poles defined by P. Further, the rejection of noise and load changes is worse. Only if decreasing H_c results in $H_c = 1$, can the prediction horizon be used as a (fine) tuning parameter. Now, the closed-loop system is quite sensitive to H_p. Increasing H_p results in mean-level control which makes the system more robust but slower.

3. Increase ρ in the way described in the section on decreasing the controller output variance.

Undesired frequencies in the controller output

In order to attenuate undesired frequencies in the controller output, ρ can be used together with Q_n and Q_d. Then the value for ρ must be determined by using Bode plots and simulations as shown in section 3.3. The polynomials Q_n and Q_d must be chosen such that the frequencies that must be attenuated are passed by the filter Q_n/Q_d. All other frequencies must not be passed by this filter. However, one must take care that $P\Delta^\gamma$ is a factor of Q_n and P is a factor of Q_d in order to prevent steady-state errors and to ensure that the roots of P appear as closed-loop poles.

3.9.3 Fine tuning for unstable processes

When the process to be controlled is unstable or badly damped, the above-mentioned guidelines can still be used. However, the following remarks must be made:

- T can be used in the way described above. However, T must always be chosen equal to $T = (1 - \mu q^{-1})^{n_A}$. Further, choosing a large value for μ ($\mu \rightarrow 1$) can make feedback disappear which results in an unstable closed-loop system.
- Further, ρ can be used in the way described above. However, a large ρ results in an unstable closed-loop system because then there is no longer feedback.
- H_c may not be chosen equal to 1 because then, mean-level control would be obtained which, for an unstable process, would result in an unstable closed-loop system.

3.9.4 Conclusions

The above-mentioned rule of thumb methods and the fine tuning rules make tuning of the UPC controller relatively simple. Moreover, in my opinion, the tuning of UPC is easier than the tuning of the other predictive controllers considered in the thesis. This is because the others lack one or more of the UPC design parameters which have been shown to be useful for simple tuning. In other predictive controllers, tuning parameters are used that can have an unpredictable effect on the system or that are hard to determine a priori.

For example, the possibility of making P a factor of $\hat{D}$ and Q_n in order to obtain separate tuning of the servo and regulator behavior is not considered in the predictive controllers discussed in this chapter. Further, for example, in PCA and MAC, tuning of the closed-loop system is done by means of the prediction horizon and the weighting factor ρ. It has been argued that this way of tuning the system is rather difficult. For example, a rule of thumb on how to choose ρ is not available.

The above-mentioned rule of thumb and fine tuning methods are well-suited for implementation in an expert system [48].

3.10 Conclusions

In this chapter, the design parameters of the UPC controller have been analyzed with respect to the servo and regulator performance and the robustness of the closed-loop system. This analysis was based on simulations in combination with theoretical results. It was shown that a wide variety of processes can be controlled by using UPC, even if the model is different from the process. Further, it is believed that the major goal of this chapter, to provide insight into the influence of the controller parameters on the closed-loop system and coming up with rule of thumb methods on how to select them, has been achieved. The rule of thumb methods generally yield good initial values for the UPC controller parameters. However, for some processes fine tuning may be useful. Some guidelines on how this can be done have also been presented.

Based on the results obtained with respect to the unified predictive controller, some remarks have been made regarding several predictive controllers well known in the literature. It has been shown that these controllers can be regarded as a subset of UPC. The GPC controller has been shown to be the closest to UPC. However, UPC has some additional features. The tuning of the well-known predictive controllers is,

in my opinion, more difficult than the tuning of UPC, despite the fact that UPC has more design parameters than any other predictive controller considered in this thesis.

Chapter 4

Predictive Control with Controller Output Constraints

In the previous two chapters, constraints on the controller output were not considered. However, constraints on the controller output are common practice. The most frequently appearing constraints are level constraints: The controller output is limited between two values: the upper limit ($\overline{u}$) and the lower limit ($\underline{u}$):

$$\underline{u}(k) \leq u(k) \leq \overline{u}(k) \qquad \underline{u}(k) < \overline{u}(k) \tag{4.1}$$

Often, the upper and lower limit are constant. However, for generality, they are assumed to be time varying.

Another type of constraint, appearing less frequently in practice, is the rate constraint. Now, the change of the controller output per sample is limited between two values:

$$\underline{\Delta u}(k) \leq \Delta u(k) \leq \overline{\Delta u}(k) \qquad \underline{\Delta u}(k) < 0 \text{ and } \overline{\Delta u}(k) > 0 \tag{4.2}$$

where $\overline{\Delta u}$ and $\underline{\Delta u}$ are the upper and lower limit of the rate constraint, respectively. Also, the rate constraints may be time varying. Level and rate constraints are, for example, present in the steering machine of a ship: The rudder angle is limited to between, for example, plus or minus 30° while the speed of the rudder is limited to between, for example, plus or minus 10°/s.

One way to ensure that the controller output satisfies the constraints (4.1) and (4.2) is by clipping the calculated output of the controller by using:

$$\begin{aligned} u^*(k) \;=\; & \max\left(\min\left(u(k), \overline{u}(k), \overline{\Delta u}(k) + u(k-1)\right), \underline{u}(k),\right. \\ & \left.\underline{\Delta u}(k) + u(k-1)\right) \end{aligned} \tag{4.3}$$

where $u^*(k)$ is the constrained controller output that can be used to control the process. However, by using (4.3), a nonlinear element is introduced in the closed-loop system. The influence of such an element on the stability and the performance of the closed-loop system can be rather large. This effect is illustrated by the following example.

Example 4.1 Influence of controller output constraints on the performance and stability of the closed-loop system.

Settings: $H_p = 4,\ H_s = 2,\ H_c = 3,\ \beta = 1,\ R = P = N = 1,\ T = 1,$ $\hat{D} = 1,\ \underline{d} = 0,\ \overline{d} = 0,\ \rho = 0,\ Q_n = 1,\ Q_d = 1$

Process: $H(s) = \dfrac{5}{(4s+1)(5s+1)} \quad T_s = 0.5s \Longrightarrow$

$$H(z^{-1}) = \frac{0.029z^{-1}(1 + 0.928z^{-1})}{(1 - 0.882z^{-1})(1 - 0.905z^{-1})}$$

Model: identical to process

Parameters: $\underline{u}$, $\overline{u}$, $\underline{\Delta u}$ and $\overline{\Delta u}$

The UPC design parameters are selected such that dead-beat control is obtained. Figure 4.1a shows the response when constraints are absent. Figure 4.1b shows what happens if level constraints with $\underline{u} = -3$ and $\overline{u} = 15$ are imposed on the controller output. The response becomes slower and overshoot occurs for positive set point changes. Figure 4.2a shows what happens if rate constraints are present with $\overline{\Delta u} = 15$ and $\underline{\Delta u} = -15$. Obviously, the closed-loop system is close to instability due to the nonlinear element (4.3) that is present in the loop. If the rate constraints

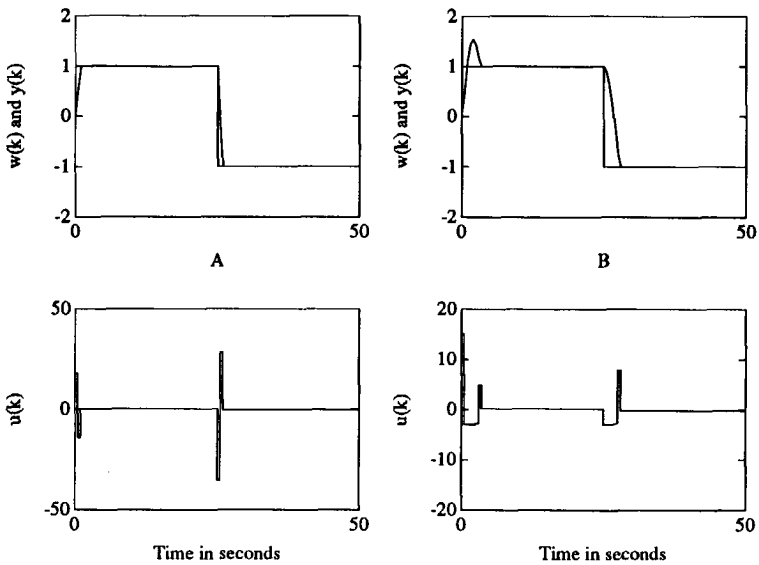

Figure 4.1: Response without constraints (Figure A) and with level constraints: $\underline{u} = -3$ *and* $\overline{u} = 15$ *(Figure B).*

are $\overline{\Delta u} = 10$ and $\underline{\Delta u} = -10$, the closed-loop system becomes unstable as is shown in Figure 4.2b.

In predictive controllers a multi-step criterion function is minimized, usually without taking the constraints on the controller output into account. For example, in UPC, criterion function (4.4) is minimized taking into account (4.5) only:

$$J = \sum_{i=H_e}^{H_p} [P\hat{y}(k+i) - Rw(k+i)]^2 + \rho \sum_{i=1}^{H_p - \underline{d}} \left[\frac{Q_n}{Q_d} u(k+i-1)\right]^2 \qquad (4.4)$$

where

$$N\Delta^{\beta} u(k+i-1) = 0 \qquad 1 \leq H_c < i \leq H_p - \underline{d} \qquad (4.5)$$

Equation (4.5) can be considered as an equality constraint and the optimization problem, therefore, as a constrained optimization problem. However, it was shown in section 2.3.1 that (4.5) can easily be incorporated into the unconstrained optimization problem which can be solved analytically. For this reason, and for the sake

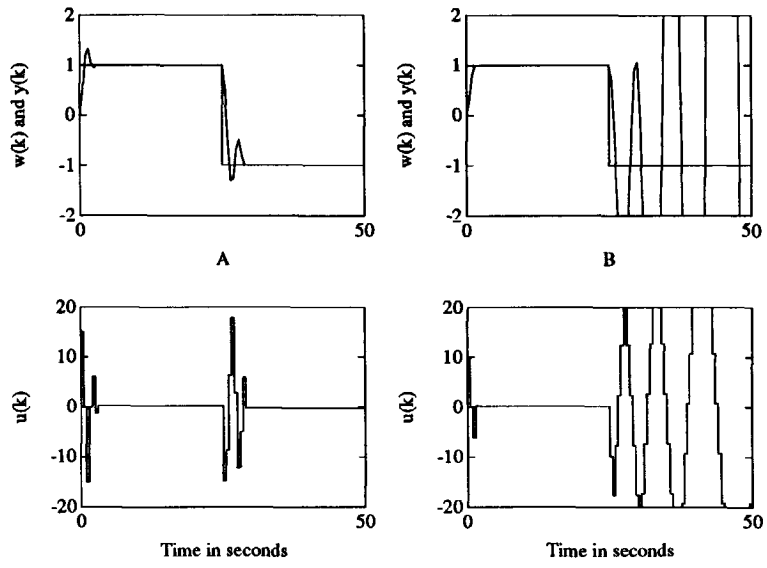

Figure 4.2: Response in the case of rate constraints on the controller output. $\overline{\Delta u} = 15$ *and* $\underline{\Delta u} = -15$ *in Figure A and* $\overline{\Delta u} = 10$ *and* $\underline{\Delta u} = -10$ *in Figure B.*

of convenience, finding the controller output sequence $u(k), \ldots, u(k + H_c - 1)$ that minimizes (4.4) subject to (4.5) is called the *unconstrained optimization problem* although, from a theoretical point of view, this is not correct.

When the inequality constraints (4.1) and/or (4.2) are present on the controller output, the optimization problem is called the *constrained optimization problem.* The optimal solution to the unconstrained problem may violate the constraints. Then, this solution is not a feasible solution to the constrained optimization problem. In order to find the optimal solution to the constrained optimization problem, the constraints must be explicitly taken into account. Further, because the criterion function (4.4) is minimized with respect to $u(k), \cdots, u(k + H_c - 1)$, also constraints on the future controller outputs must be taken into account. Now the optimization problem can be formulated as: Minimize criterion function (4.4) with respect to $u(k), \cdots, u(k + H_c - 1)$ subject to the following constraints:

$$\underline{u}(k+i-1) \leq u(k+i-1) \leq \overline{u}(k+i-1) \quad i = 1, \ldots, H_c \tag{4.6}$$

$$\underline{\Delta u}(k+i-1) \leq \Delta u(k+i-1) \leq \overline{\Delta u}(k+i-1) \quad i = 1, \ldots, H_c \tag{4.7}$$

$$N\Delta^{\beta} u(k+i-1) = 0 \quad 1 \leq H_c < i \leq H_p - \underline{d} \tag{4.8}$$

In general, optimization can denote maximization as well as minimization. In this thesis, optimization always denotes minimization.

The above-mentioned optimization problem is known as a quadratic programming (QP) problem [49], pp.314. A number of techniques have been presented to solve such a problem (see e.g. [49], pp.315). In predictive controllers, QP techniques have been used, for example, by García [50, 51] in the QDMC controller. Because QP techniques minimize a quadratic criterion function with general linear inequality constraints, not only constraints on the controller output, but also constraints on the process output and the states of the process, can easily be taken into account. A disadvantage is, however, that QP algorithms are rather complex and extremely (computation) time consuming [52]. This issue may not be important in the process industry where sampling times may be in the order of seconds or even minutes but in controlling, for example, a motor of a robot arm, sampling times may be in the order of ms and the issue of computation time is more important.

In order to limit the complexity and the computation time of the control algorithm, various other related methods have been presented in the literature. For example, in Tsang [52] a method is presented to be used with GPC. However, due to the fact that in GPC the criterion function is minimized with respect to $\Delta u(k), \cdots, \Delta u(k + H_c - 1)$, rather than $u(k), \cdots, u(k + H_c - 1)$ as in UPC, only rate constraints could be taken into account. Level constraints were considered only if $H_c = 2$. Further, in this approach, assumptions must be made with respect to the optimal solution of the constrained optimization problem: It must be known which of the optimal controller outputs $\Delta u(k), \cdots, \Delta u(k + H_c - 1)$ are on the feasibility boundary. In [52] it is assumed that they are given by the controller output increments optimal for the unconstrained criterion that violate the constraints (4.7).

In this chapter, Rosen's gradient projection method [53, 54] is used to solve the constrained optimization problem. This method is a general optimization method that can be used to minimize a criterion function (not necessarily quadratic) in the presence of linear inequality constraints. The reason to select this method is that it boils down to a simple and fast method in the case of level constraints. In the case of rate constraints, the complexity of the algorithm is greater. A summary of the results presented in this chapter can be found in [27] and [28].

4.1 Rosen's Gradient Projection Method

In searching for an optimum, most iterative optimization methods make use of the following algorithm.

Algorithm 4.1 Basic algorithm of an iterative optimization method.

Step 1: Initialize $\boldsymbol{x} \Longrightarrow \boldsymbol{x}_0$ and $n = 0$.

Step 2: `If` stop criterion satisfied `Then`
Optimal solution found.
`End If`

Step 3: $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \lambda_n \boldsymbol{d}_n$ (4.9)
$n = n + 1$
`Goto` Step 2

where $\boldsymbol{x}_n$ is a vector of dimension $n_X \times 1$ containing the variables for which the criterion is optimized. Further, n denotes the iteration number ($n \geq 0$), $\boldsymbol{d}_n$ denotes the search direction at iteration n and λ_n is a scalar denoting the step size in the search direction. Because the search direction is determined by $\boldsymbol{d}_n$, the step size λ_n is positive. The way $\boldsymbol{d}_n$ and λ_n are calculated distinguishes optimization algorithms.

Also, Rosen's gradient projection method makes use of Algorithm 4.1. How the search direction, and the step size in the search direction are calculated, is discussed in the following sections.

4.1.1 The Search Direction

As has already been mentioned, Rosen's method can be used to minimize a criterion function subject to linear inequality (and equality) constraints. In order to take the constraints into account, they must satisfy:

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \tag{4.10}$$

where $\boldsymbol{A}$ is a matrix of dimension $a \times n_X$ and $\boldsymbol{b}$ is a vector of dimension $a \times 1$. The variable a denotes the number of (in)equality constraints. Now define the terms *active* constraints and a feasible point.

Definition 4.1 *A constraint j is defined as active if $\boldsymbol{a}_j^T \boldsymbol{x} = b_j$ where $\boldsymbol{a}_j^T$ and b_j denote the jth row of $\boldsymbol{A}$ and the jth element of $\boldsymbol{b}$, respectively.*

Remark: equality constraints are always active.

Definition 4.2 *A point $\boldsymbol{x}$ is defined to be feasible if $\boldsymbol{x}$ satisfies (4.10).*

Obviously, $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$ in (4.9) must be feasible points.

Further, in Rosen's gradient projection method [53], the search direction $\boldsymbol{d}_n$ is determined by the negative gradient, projected on the *active* constraints:

$$\boldsymbol{d}_n = -\boldsymbol{P}_n \boldsymbol{g}_n \tag{4.11}$$

where $\boldsymbol{P}_n$ is the projection matrix and $\boldsymbol{g}_n$ is the gradient of the criterion function J with respect to $\boldsymbol{x}_n$:

$$\boldsymbol{g}_n = \frac{\partial J}{\partial \boldsymbol{x}_n}$$

The active constraints at iteration n are assumed to be given by:

$$\boldsymbol{K}_n \boldsymbol{x}_n = \boldsymbol{b}_n^* \qquad [\boldsymbol{K}_n] = r \times n_X \tag{4.12}$$

where $\boldsymbol{K}_n$ is called the constraint matrix, r denotes the number of active constraints and the vector $\boldsymbol{b}_n^*$ represents the values of the bounds. Now the projection matrix $\boldsymbol{P}_n$ is given by [53], pp.390:

$$\boldsymbol{P}_n = \boldsymbol{I} - \boldsymbol{K}_n^T (\boldsymbol{K}_n \boldsymbol{K}_n^T)^{-1} \boldsymbol{K}_n \qquad [\boldsymbol{P}_n] = n_X \times n_X \tag{4.13}$$

where $\boldsymbol{I}$ is the identity matrix of dimension $n_X \times n_X$.

4.1.2 The Optimal Step in the Search Direction

In Rosen's gradient projection method, λ_n is the optimal step size in the search direction. Finding the optimal step $\lambda_{opt,n}$ in the search direction is a one-dimensional minimization problem. Moreover, it is also a constrained optimization problem. Because $\boldsymbol{x}_{n+1}$ must be a feasible point, the step size is limited by λ_{max}: the shortest distance to the inactive constraints j:

$$\lambda_{max} = \min_j \left\{ \frac{b_j - \boldsymbol{a}_j^T \boldsymbol{x}_n}{\boldsymbol{a}_j^T \boldsymbol{d}_n} \middle| \boldsymbol{a}_j^T \boldsymbol{d}_n > 0 \right\} \tag{4.14}$$

Because finding $\lambda_{opt,n}$ is a one-dimensional optimization problem, the optimal step in the search direction is given by:

$$\lambda^*_{opt,n} = \min(\lambda_{opt,n}, \lambda_{max}) \tag{4.15}$$

where $\lambda^*_{opt,n}$ is the optimal step size in the search direction $\boldsymbol{d}_n$ for the constrained optimization problem. If $\lambda^*_{opt,n} = \lambda_{max}$, then one or more inactive constraints become active. Consequently, the constraint matrix $\boldsymbol{K}_n$ and the projection matrix $\boldsymbol{P}_n$ change.

In order to find $\lambda_{opt,n}$ an iterative line search method can be used. Many methods are available. For example, fixed and variable step methods and parabolic interpolation methods [53], pp.252. However, it will be shown that due to the special form of the constraints (4.6) and (4.7), $\lambda_{opt,n}$ and hence $\lambda^*_{opt,n}$ can be calculated analytically.

4.1.3 Stop Criteria

In any optimization problem, conditions are required that determine whether or not an optimum has been found and the iterative optimization algorithm can be stopped. In Rosen's method, two conditions must be satisfied for an optimal solution:

1. If the search direction $\boldsymbol{d}_n$ is equal to zero an optimum has been found. In practice, the following condition must be satisfied for a solution to the optimization problem:

$$\max_j |d_j| < \epsilon \quad \text{for } j = 1, \ldots, n_X \tag{4.16}$$

where d_j is the jth element of $\boldsymbol{d}_n$ and ϵ is a small positive value determining the accuracy of the solution.

2. The Kuhn-Tucker condition must be satisfied (see [53], pp.391):

$$n_j \geq 0 \quad \text{for } j = 1, \ldots, n_X \tag{4.17}$$

where n_j is the jth element of the Kuhn-Tucker vector $\boldsymbol{n}_n$ given by:

$$\boldsymbol{n}_n = -(\boldsymbol{K}_n \boldsymbol{K}_n^T)^{-1} \boldsymbol{K}_n \boldsymbol{g}_n \tag{4.18}$$

If (4.17) is not satisfied for all j, then an active constraint corresponding to a negative value of n_j must be removed from the constraint matrix $\boldsymbol{K}_n$. Usually, the constraint corresponding to the most negative value of n_j is removed. Hence, if n_j is the most negative element of $\boldsymbol{n}_n$, then delete row j of $\boldsymbol{K}_n$. The iterative search continues until both (4.16) and (4.18) are satisfied. Then, an optimal solution to the constrained optimization algorithm is found. This optimal solution is also the global solution (thus the unique solution) if the Hessian matrix (the matrix of second-order derivatives) is positive definite.

4.1.4 Visualization of Rosen's Gradient Projection Method

Rosen's gradient projection method is depicted in Figure 4.3. In the initialization step, $\boldsymbol{K}_n$ and $\boldsymbol{x}_n$ must be initialized yielding $\boldsymbol{K}_0$ and $\boldsymbol{x}_0$. Of course, $\boldsymbol{x}_0$ must be a feasible point. If the initial point $\boldsymbol{x}_0$ has no active constraints (hence, $r = 0$), then $\boldsymbol{P}_0 = \boldsymbol{I}$ and $\boldsymbol{n}_0 = 0$. In general, choosing $\boldsymbol{x}_0$ close to the optimal solution of the optimization problem yields a much faster convergence. In section 4.2.1 on page 246, various ways of initializing $\boldsymbol{x}_n$ are discussed.

Note that Rosen's gradient projection method can also be used to find a solution to an unconstrained optimization problem. Then, the algorithm boils down to the method of steepest descent [53], pp.323.

4.2 UPC in the Presence of Constraints

In UPC, the optimization problem in the presence of constraints can be formulated as:

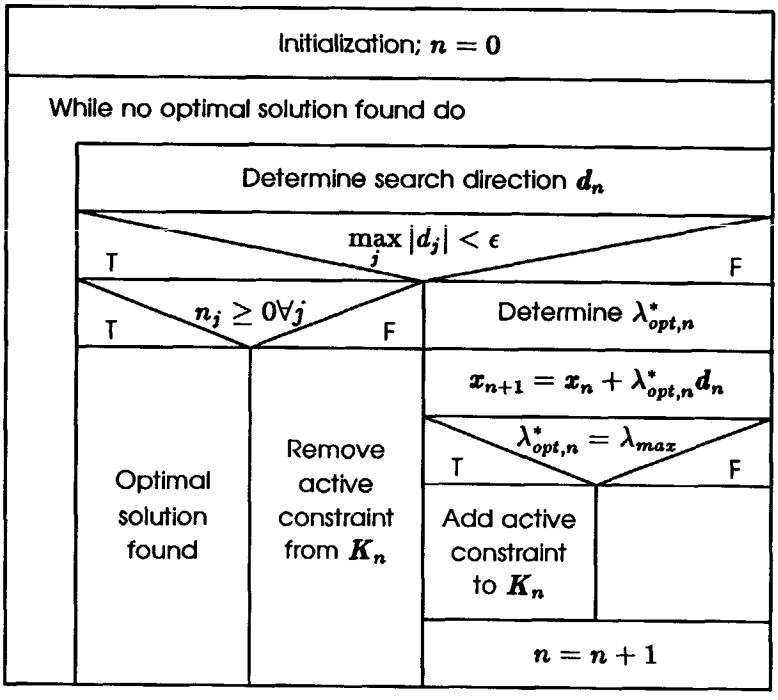

Figure 4.3: Structure diagram of Rosen's gradient projection method.

Minimize criterion function (4.4) with respect to $u(k), \cdots, u(k + H_c - 1)$ subject to the constraints (4.6), (4.7) and (4.8). As shown in section 2.3.1, the equality constraint (4.8) can be simply incorporated. For finding the optimal solution subject to the constraints (4.6) and (4.7), the gradient projection method of Rosen, as discussed in the previous section, is used. Because the criterion function must be minimized with respect to the controller output sequence over the control horizon, the vector x_n, as used in the previous section, becomes:

$$x_n = [u(k), \cdots, u(k + H_c - 1)]^T = \overline{u}_n$$

where $\overline{u}_n$ denotes the vector $\overline{u}$ at iteration n.

Further, in order to be able to incorporate the constraints (4.6) and (4.7) in Rosen's method, they must be written as in (4.10):

$$u(k + j - 1) \leq \overline{u}(k + j - 1) \tag{4.19}$$

$$-u(k+j-1) \leq -\underline{u}(k+j-1) \tag{4.20}$$
$$\Delta u(k+j-1) \leq \overline{\Delta u}(k+j-1) \tag{4.21}$$
$$-\Delta u(k+j-1) \leq -\underline{\Delta u}(k+j-1) \tag{4.22}$$

with $j = 1, \ldots, H_c$. Because there are significant differences between the minimization of (4.4) in the presence of level and of rate constraints, both types of constraints are discussed separately.

Remark: the constraints can be time variant. However, for convenience, $\overline{u}(k+j-1)$, $\underline{u}(k+j-1)$, $\overline{\Delta u}(k+j-1)$ and $\underline{\Delta u}(k+j-1)$ are denoted as $\overline{u}$, $\underline{u}$, $\overline{\Delta u}$ and $\underline{\Delta u}$, respectively.

4.2.1 Level Constraints

When there are level constraints, the constraints (4.19) and (4.20) are present only and (4.10) becomes:

$$A\overline{u} \leq b: \left[\begin{array}{cccc} 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \\ \hline -1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -1 \end{array}\right] \overline{u} \leq \left[\begin{array}{c} \overline{u} \\ \vdots \\ \overline{u} \\ \hline -\underline{u} \\ \vdots \\ -\underline{u} \end{array}\right] \qquad [A] = 2H_c \times H_c \tag{4.23}$$

Hence, $2H_c$ inequality constraints are present numbered from $1, \ldots, 2H_c$. Further, because $u(k+j-1)$ is limited by either $\overline{u}$ or $\underline{u}$, the constraints j and $j+H_c$ cannot be active simultaneously. Hence, the maximum number of active constraints is equal to H_c. Now, element ij of the constraint matrix K_n becomes:

- If constraint j is the ith active constraint, then:

$$k_{ij} = 1 \quad k_{il} = 0 \ \forall l \neq j \text{ and } k_{lj} = 0 \ \forall l \neq i \tag{4.24}$$

- If constraint $j + H_c$ is the ith active constraint, then:

$$k_{ij} = -1 \quad k_{il} = 0 \ \forall l \neq j \text{ and } k_{lj} = 0 \ \forall l \neq i \tag{4.25}$$

Example 4.2 Assume $H_c = 3$ and two active constraints:

$$-u(k) = -\underline{u} \tag{4.26}$$

$$u(k+2) = \overline{u} \tag{4.27}$$

Then, (4.12) becomes:

$$\boldsymbol{K}_n \overline{\boldsymbol{u}}_n = \boldsymbol{b}_n^* : \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \end{bmatrix} = \begin{bmatrix} -\underline{u} \\ \overline{u} \end{bmatrix}$$

and hence the constraint matrix $\boldsymbol{K}_n$ is of dimension 2×3.

In the following sections, each part of Rosen's method is discussed separately.

The search direction

The search direction is given by (4.11):

$$\boldsymbol{d}_n = -\boldsymbol{P}_n \boldsymbol{g}_n \qquad [\boldsymbol{d}_n] = H_c \times 1 \tag{4.28}$$

The gradient of the criterion (4.4) with respect to $\overline{\boldsymbol{u}}_n$ is given by (2.167) with $\overline{\boldsymbol{u}}$ replaced by $\overline{\boldsymbol{u}}_n$. The projection matrix is given by (4.13):

$$\boldsymbol{P}_n = \boldsymbol{I} - \boldsymbol{K}_n^T (\boldsymbol{K}_n \boldsymbol{K}_n^T)^{-1} \boldsymbol{K}_n \qquad [\boldsymbol{P}_n] = H_c \times H_c \tag{4.29}$$

Due to the simplicity of the constraints (4.19) and (4.20), the projection matrix can be simply calculated. Using (4.24) and (4.25), element ij of the matrix product $\boldsymbol{K}_n \boldsymbol{K}_n^T$ becomes:

$$(\boldsymbol{K}_n \boldsymbol{K}_n^T)_{ij} = \sum_{l=1}^{H_c} k_{il} k_{jl} = 1 \text{ if } i = j$$

$$= 0 \text{ if } i \neq j$$

Hence, $\boldsymbol{K}_n \boldsymbol{K}_n^T$ is the identity matrix of dimension $r \times r$. As a result, the projection matrix $\boldsymbol{P}_n$ given by (4.29), becomes:

$$P_n = I - K_n^T K_n$$

The ijth element of $K_n^T K_n$ is given by:

$$\begin{aligned}(K_n^T K_n)_{ij} = \sum_{l=1}^{r} k_{li} k_{lj} &= 0 \quad \text{if } i \neq j \\ &= 1 \quad \text{if } i = j \\ &\qquad \text{and constraint } j \text{ or } j + H_c \text{ is active} \\ &= 0 \quad \text{if } i = j \\ &\qquad \text{and constraint } j \text{ or } j + H_c \text{ is not active}\end{aligned}$$

Clearly, the projection matrix is a diagonal matrix. Element jj of P_n is given by:

$$\begin{aligned}p_{jj} &= 0 \text{ if constraint } j \text{ or } j + H_c \text{ is active} \\ p_{jj} &= 1 \text{ if constraint } j \text{ or } j + H_c \text{ is not active}\end{aligned}$$

The search direction d_n given by (4.28) now becomes:

$$d_j = 0 \text{ if constraint } j \text{ or } j + H_c \text{ is active} \tag{4.30}$$

$$d_j = -g_j \text{ if constraint } j \text{ or } j + H_c \text{ is not active} \tag{4.31}$$

where g_j is the jth element of g_n.

Note that constraint j is active if $u(k + j - 1) = \overline{u}$ and constraint $j + H_c$ is active if $u(k + j - 1) = \underline{u}$.

Conclusion: in the case of level constraints, the search direction is easy to determine.

The Optimal Step in the Search Direction

In order to calculate the optimal step in the search direction, recall the criterion function (2.165):

$$J = (\hat{y}^* - w^*)^T (\hat{y}^* - w^*) + \rho u^{*T} u^* \tag{4.32}$$

where $\boldsymbol{u}^*$ and $\boldsymbol{y}^*$ are given by (2.179) and (2.181), respectively:

$$\boldsymbol{u}^* = \boldsymbol{\Phi}\boldsymbol{M}\overline{\boldsymbol{u}} + \boldsymbol{\Omega}\tilde{\boldsymbol{u}} + \boldsymbol{\Phi}\boldsymbol{N}\check{\boldsymbol{u}} \tag{4.33}$$

$$\hat{\boldsymbol{y}}^* = \boldsymbol{\Gamma}\boldsymbol{M}\overline{\boldsymbol{u}} + \boldsymbol{\Psi}\boldsymbol{s} + \boldsymbol{k} + \boldsymbol{\Gamma}\boldsymbol{N}\check{\boldsymbol{u}} \tag{4.34}$$

In order to determine the optimal step size in the search direction $\boldsymbol{d}_n$, criterion function (4.32) can be rewritten as:

$$J_{n+1} = (\hat{\boldsymbol{y}}^*_{n+1} - \boldsymbol{w}^*)^T(\hat{\boldsymbol{y}}^*_{n+1} - \boldsymbol{w}^*) + \rho \boldsymbol{u}^{*T}_{n+1}\boldsymbol{u}^*_{n+1} \tag{4.35}$$

where

$$\hat{\boldsymbol{y}}^*_{n+1} = \boldsymbol{\Gamma}\boldsymbol{M}\overline{\boldsymbol{u}}_{n+1} + \boldsymbol{\Psi}\boldsymbol{s} + \boldsymbol{k} + \boldsymbol{\Gamma}\boldsymbol{N}\check{\boldsymbol{u}} \tag{4.36}$$

$$\boldsymbol{u}^*_{n+1} = \boldsymbol{\Phi}\boldsymbol{M}\overline{\boldsymbol{u}}_{n+1} + \boldsymbol{\Omega}\tilde{\boldsymbol{u}} + \boldsymbol{\Phi}\boldsymbol{N}\check{\boldsymbol{u}} \tag{4.37}$$

$$\overline{\boldsymbol{u}}_{n+1} = \overline{\boldsymbol{u}}_n + \lambda_n \boldsymbol{d}_n \tag{4.38}$$

Substitution of (4.36), (4.37) and (4.38) in (4.35) yields:

$$\begin{aligned} J_{n+1} = {} & \lambda_n^2 \boldsymbol{d}_n^T \boldsymbol{M}^T(\boldsymbol{\Gamma}^T\boldsymbol{\Gamma} + \rho\boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{M}\boldsymbol{d}_n + \\ & 2\lambda_n\left(\boldsymbol{d}_n^T\boldsymbol{M}^T(\boldsymbol{\Gamma}^T\boldsymbol{\Gamma} + \rho\boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{M}\overline{\boldsymbol{u}}_n + \boldsymbol{d}_n^T\boldsymbol{M}^T\boldsymbol{\Gamma}^T\boldsymbol{f} + \rho\boldsymbol{d}_n^T\boldsymbol{M}^T\boldsymbol{\Phi}^T\boldsymbol{f}_2\right) + \\ & \text{terms not depending on } \lambda_n \end{aligned}$$

where:

$$\begin{aligned} \boldsymbol{f} &= \boldsymbol{\Psi}\boldsymbol{s} + \boldsymbol{k} + \boldsymbol{\Gamma}\boldsymbol{N}\check{\boldsymbol{u}} - \boldsymbol{w}^* \\ \boldsymbol{f}_2 &= \boldsymbol{\Omega}\tilde{\boldsymbol{u}} + \boldsymbol{\Phi}\boldsymbol{N}\check{\boldsymbol{u}} \end{aligned}$$

Now, the derivative of J_{n+1} with respect to λ_n is:

$$\begin{aligned} \frac{\mathrm{d}J_{n+1}}{\mathrm{d}\lambda_n} = {} & 2\lambda_n \boldsymbol{d}_n^T\boldsymbol{M}^T(\boldsymbol{\Gamma}^T\boldsymbol{\Gamma} + \rho\boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{M}\boldsymbol{d}_n + \\ & 2\boldsymbol{d}_n^T\boldsymbol{M}^T(\boldsymbol{\Gamma}^T\boldsymbol{\Gamma} + \rho\boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{M}\overline{\boldsymbol{u}}_n + \\ & 2\boldsymbol{d}_n^T\boldsymbol{M}^T\boldsymbol{\Gamma}^T\boldsymbol{f} + 2\rho\boldsymbol{d}_n^T\boldsymbol{M}^T\boldsymbol{\Phi}^T\boldsymbol{f}_2 \end{aligned} \tag{4.39}$$

The second-order derivative is given by:

$$\frac{\mathrm{d}^2 J_{n+1}}{\mathrm{d}\lambda_n^2} = 2\boldsymbol{d}_n^T \boldsymbol{M}^T (\boldsymbol{\Gamma}^T \boldsymbol{\Gamma} + \rho \boldsymbol{\Phi}^T \boldsymbol{\Phi}) \boldsymbol{M} \boldsymbol{d}_n$$

Clearly, the second-order derivative of J_{n+1} with respect to λ_n is positive. Therefore, the minimal value of J_{n+1} with respect to λ_n can be found by setting (4.39) equal to zero. By using the fact that $\boldsymbol{g}_n$ is given by (2.182) with $\overline{\boldsymbol{u}}$ replaced by $\overline{\boldsymbol{u}}_n$, the optimal value for λ_n becomes:

$$\lambda_{opt,n} = -\frac{\boldsymbol{d}_n^T \boldsymbol{g}_n}{2\boldsymbol{d}_n^T \boldsymbol{M}^T (\boldsymbol{\Gamma}^T \boldsymbol{\Gamma} + \rho \boldsymbol{\Phi}^T \boldsymbol{\Phi}) \boldsymbol{M} \boldsymbol{d}_n} = -\frac{\boldsymbol{d}_n^T \boldsymbol{g}_n}{\boldsymbol{d}_n^T \boldsymbol{H} \boldsymbol{d}_n} \tag{4.40}$$

where $\boldsymbol{H}$ is the Hessian given by (2.183). Equation (4.40) shows that the optimal step size in the search direction can be calculated analytically. An iterative line search method is not required.
The maximum step size in the search direction is given by:

$$\lambda_{max} = \min \left\{ \min_j \left(\frac{\overline{u} - u(k+j-1)}{d_j} \middle| d_j > 0 \right), \right.$$

$$\left. \min_j \left(\frac{\underline{u} - u(k+j-1)}{d_j} \middle| d_j < 0 \right) \right\} \tag{4.41}$$

Further, (4.15) can be used to calculate $\lambda^*_{opt,n}$:

$$\lambda^*_{opt,n} = \min(\lambda_{opt,n}, \lambda_{max}) \tag{4.42}$$

Stop Criteria

The two stop criteria that must be satisfied for an optimal solution are (4.16) and (4.17). In the case of level constraints, the Kuhn-Tucker vector is easy to calculate. Because the matrix product $\boldsymbol{K}_n \boldsymbol{K}_n^T$ equals the identity matrix, the vector $\boldsymbol{n}_n$ in (4.18) is given by:

$$\boldsymbol{n}_n = -\boldsymbol{K}_n \boldsymbol{g}_n$$

Using (4.24) and (4.25) yields for the ith element of $\boldsymbol{n}_n$:

$$n_i = -g_j \text{ if the } i\text{th active constraint is constraint } j \tag{4.43}$$

$$n_i = g_j \text{ if the } i\text{th active constraint is constraint } j + H_c \tag{4.44}$$

If (4.16) and (4.17) are satisfied, then an optimal solution is found. Moreover, because the Hessian (2.183) is positive definite, a global minimum is obtained.

Initialization of $\overline{u}$

As is generally known, a well-chosen starting point yields a much faster convergence of an optimization algorithm. Therefore, the choice of $\overline{u}_0$ is an important one. $\overline{u}_0$ can be chosen in many ways. The only restriction is that $\overline{u}_0$ must be a feasible point. Three ways of choosing $\overline{u}_0$ are:

1. Use a fixed initialization of $\overline{u}_0$. For example, $\overline{u}_0 = 0$. This choice can be used if $\underline{u} \leq 0$ and $\overline{u} \geq 0$ (then $\overline{u}_0 = 0$ will always be a feasible point).

2. Given the optimal controller output sequence $\overline{u}$ for the unconstrained optimization problem, a feasible starting point $\overline{u}_0$ can be obtained (if the constraints are time invariant) by clipping the vector $\overline{u}$ by the constraints:

$$u^*(k+j-1) = \min\left(\max\left(u(k+j-1), \underline{u}\right), \overline{u}\right) \tag{4.45}$$

   where $j = 1, \ldots, H_c$. This results in the vector $\boldsymbol{u}^*$:

$$\boldsymbol{u}^* = [u^*(k), \cdots, u^*(k+H_c-1)]^T$$

   This vector can be used as a starting point of the optimization. Hence, $\overline{u}_0 = \boldsymbol{u}^*$.

3. Utilize the optimal controller output sequence calculated in the previous sample. Let $\overline{u}_{k-1}$ be the vector $\overline{u}$ at $t = k-1$ and let $u_{k-1,j}$ be the jth element of $\overline{u}_{k-1}$. Then, the following algorithm can be used to initialize $\overline{u}$ at $t = k$.

$$u_{0,j} = u_{k-1,j+1} \qquad j = 1, \ldots, H_c - 1 \tag{4.46}$$

$$u_{0,j} = u_{k-1,j} \qquad j = H_c$$

   where $u_{0,j}$ denotes the jth element of $\overline{u}_0$ at $t = k$.

   If $k = 0$, the solution to the optimization problem in the previous sample is not available. Then, for example, a fixed initialization can be used instead.

When using one of the methods suggested above for finding a starting point of the optimization, this point can still be far from the optimal solution resulting in a slow convergence. In order to overcome this problem, a special algorithm has been developed that calculates an approximate solution of the optimization problem. This algorithm is called ACH (Active Constraint Handling) and is discussed in the following section.

4.2.2 The ACH algorithm

The ACH algorithm is based on the relation between controller output weighting by means of weighting factors and the optimal solution of the unconstrained optimization problem as discussed in section 2.3.1. For this purpose, each controller output $u(k+j-1)$ is equipped with its own weighting factor ρ_j. Consequently, criterion function (2.165) becomes:

$$J = (\hat{\boldsymbol{y}}^* - \boldsymbol{w}^*)^T(\hat{\boldsymbol{y}}^* - \boldsymbol{w}^*) + \boldsymbol{u}^{*T}\boldsymbol{\Theta}^*\boldsymbol{u}^* \tag{4.47}$$

where $\boldsymbol{\Theta}^*$ is a diagonal matrix given by:

$$\boldsymbol{\Theta}^* = diag(\underbrace{\rho_1, \cdots, \rho_{H_c}}_{H_c}, \underbrace{0, \cdots, 0}_{H_p - H_c - \underline{d}}) \qquad [\boldsymbol{\Theta}^*] = (H_p - \underline{d}) \times (H_p - \underline{d}) \tag{4.48}$$

The ACH algorithm is an iterative algorithm that increases the weighting factors ρ_j such that all controller outputs over the control horizon satisfy the constraints (4.19) and (4.20). The way these weighting factors are calculated is such that the corresponding solution is close to the optimal solution of the constrained optimization problem.
The ACH algorithm can be divided into three steps:

Algorithm 4.2 The ACH algorithm.

Step 1: Initialization step: Calculate the optimal controller output sequence of the unconstrained optimization problem where $Q_n = Q_d = P = N = \beta = 1$ by using (2.184). Further, $m = 0$.

Step 2: **If** $|u(k+j-1)| \leq \eta |u_b| \ \forall j$ **Then**
Solution found
End If

Step 3: **For** $j = 1, \ldots, H_c$ **Do**
If $|u(k+j-1)| > \eta |u_b|$ **Then**
Calculate ρ_j in (4.48) such that $u(k+j-1) = u_b$
$m = m + 1$
End If
End For
Goto Step 2.

where m is the iteration number. Here it is assumed that $\underline{u} < 0$ and $\overline{u} > 0$. Further, u_b is either $\underline{u}$ or $\overline{u}$, depending on which of the constraints (4.19) or (4.20) is violated. The variable η is a variable > 1 determining the accuracy of the solution. For example, if $\eta = 1.1$, then all controller outputs will not exceed the constraints by more than 10 %.

Increasing the weighting factor ρ_j on $u(k+j-1)^2$, obviously decreases $u(k+j-1)^2$ and $|u(k+j-1)|$. Because the ACH algorithm makes $|u(k+j-1)|$ smaller if $|u(k+j-1)| > \eta |u_b|$, ρ_j is increased in each iteration of the ACH algorithm. Further, because infinite weights on $u(k+j-1)^2$ yield $u(k+j-1) = 0 \ \forall j$ (see (4.50) with (4.51)), and because in this case the stop criterion of the ACH algorithm is satisfied, convergence is always achieved. Moreover, because the weighting factors are not made larger than necessary, a solution close to the optimal solution of the constrained optimization problem can be expected. The crucial part of the ACH algorithm, how to calculate ρ_j such that $u(k+j-1) = u_b$, is discussed in the following section.

Calculation of ρ_j such that $u(k+j-1) = u_b$

In order to show how ρ_j can be calculated such that $u(k+j-1) = u_b$, recall (2.184) with $\Phi = I$ and $\Omega = 0$ (because $Q_n = Q_d = 1$) and $N = 0$ (because $N = \beta = 1$):

$$\overline{u} = \left[M^T \left(\Gamma^T \Gamma + \rho I\right) M\right]^{-1} M^T \Gamma^T (w^* - \Psi s - k)$$

When each $u(k+j-1)$ for $j = 1, \ldots, H_c$ has its own weighting factor ρ_j, it is easy to verify that minimization of (4.47) yields:

$$\overline{u} = \left[M^T \left(\Gamma^T \Gamma + \Theta^*\right) M\right]^{-1} M^T \Gamma^T (w^* - \Psi s - k) \tag{4.49}$$

where Θ^* is given by (4.48). Now, (4.49) can be rewritten into:

$$\overline{u} = R^{-1} A^T z \tag{4.50}$$

where:

$$\begin{aligned} R &= A^T A + \Theta & [R] &= H_c \times H_c \\ A &= \Gamma M & [A] &= H_p - \underline{d} \times H_c \\ z &= w^* - \Psi s - k & [z] &= H_p - \underline{d} \times 1 \\ \Theta &= diag(\rho_1, \cdots, \rho_{H_c}) & [\Theta] &= H_c \times H_c \end{aligned} \tag{4.51}$$

In order to be able to calculate ρ_j such that $u(k+j-1) = u_b$, (4.50) is rewritten into:

$$\Theta \overline{u} = -A^T A \overline{u} + A^T z \tag{4.52}$$

Now, set the jth element of $\overline{u}$ equal to u_b, yielding:

$$\overline{u} = [u(k), \cdots, \underbrace{u_b}_{j}, \cdots, u(k + H_c - 1)]^T$$

Consequently we have to solve H_c unknowns (ρ_j and $u(k), \cdots, u(k+H_c-1)$ with $u(k+j-1)$ excluded) from H_c equations (the dimension of the system given by (4.52)). This can be accomplished by writing (4.52) into a system of H_c equations:

$$\begin{aligned} \rho_1 u(k) &= -{}_r(A^T A)_1 \overline{u} + {}_r(A^T)_1 z \\ &\vdots \\ \rho_j u_b &= -{}_r(A^T A)_j \overline{u} + {}_r(A^T)_j z \\ &\vdots \\ \rho_{H_c} u(k + H_c - 1) &= -{}_r(A^T A)_{H_c} \overline{u} + {}_r(A^T)_{H_c} z \end{aligned} \tag{4.53}$$

where ${}_r(\boldsymbol{A}^T\boldsymbol{A})_j$ denotes the jth row of $\boldsymbol{A}^T\boldsymbol{A}$ and ${}_r(\boldsymbol{A}^T)_j$ denotes the jth row of $\boldsymbol{A}^T$. Now introduce:

$$\begin{aligned}
\boldsymbol{\lambda}_j &= \text{the } j\text{th column of } \boldsymbol{A} \\
\tilde{\boldsymbol{A}}_j &= \boldsymbol{A} \text{ with column } j \text{ removed} \\
\tilde{\boldsymbol{u}}_j &= [u(k), \cdots, u(k+j-2), u(k+j), \cdots, u(k+H_c-1)]^T \\
\tilde{\boldsymbol{\Theta}}_j &= diag(\rho_1, \cdots, \rho_{j-1}, \rho_{j+1}, \cdots, \rho_{H_c})
\end{aligned}$$

Then (4.53) can be rewritten into:

$$\rho_j u_b = -\boldsymbol{\lambda}_j^T \boldsymbol{A}\bar{\boldsymbol{u}} + \boldsymbol{\lambda}_j^T \boldsymbol{z} \tag{4.54}$$

$$\tilde{\boldsymbol{\Theta}}_j \tilde{\boldsymbol{u}}_j = -u_b \tilde{\boldsymbol{A}}_j^T \boldsymbol{\lambda}_j - \tilde{\boldsymbol{A}}_j^T \tilde{\boldsymbol{A}}_j \tilde{\boldsymbol{u}}_j + \tilde{\boldsymbol{A}}_j^T \boldsymbol{z} \tag{4.55}$$

Note that if $j = 1$, $\tilde{\boldsymbol{u}}_j$ and $\tilde{\boldsymbol{\Theta}}_j$ are given by:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_1 &= [u(k+1), \cdots, u(k+H_c-1)]^T \\
\tilde{\boldsymbol{\Theta}}_1 &= diag(\rho_2, \cdots, \rho_{H_c})
\end{aligned}$$

Rewriting (4.55) results in:

$$\tilde{\boldsymbol{u}}_j = \boldsymbol{R}_j^{-1} \left(\tilde{\boldsymbol{A}}_j^T \boldsymbol{z} - u_b \tilde{\boldsymbol{A}}_j^T \boldsymbol{\lambda}_j \right) \tag{4.56}$$

in which:

$$\boldsymbol{R}_j = \left(\tilde{\boldsymbol{A}}_j^T \tilde{\boldsymbol{A}}_j + \tilde{\boldsymbol{\Theta}}_j \right) \tag{4.57}$$

The subscript j in (4.57) denotes that ρ_j is calculated such that $u(k+j-1) = u_b$. Now $\tilde{\boldsymbol{u}}_j$ can be solved from (4.56). Subsequently, ρ_j can be solved from (4.54). A disadvantage of solving $\tilde{\boldsymbol{u}}_j$ from (4.56) is that a matrix inverse is required. However, the following theorems can be used to compute $\boldsymbol{R}_j^{-1}$ with a small computational effort.

Theorem 4.1 *If $\boldsymbol{R}^{-1}$ is the inverse of a symmetric matrix $\boldsymbol{R}$, then the inverse of $\boldsymbol{R}$ in which the jth row and column are removed is given by:*

$$\boldsymbol{R}_j^{-1} = \tilde{\boldsymbol{R}}_j^{-1} - \frac{\tilde{\boldsymbol{r}}_j \tilde{\boldsymbol{r}}_j^T}{(\boldsymbol{R}^{-1})_{jj}} \tag{4.58}$$

where:

- $\tilde{\boldsymbol{R}}_j^{-1}$ *is matrix* $\boldsymbol{R}^{-1}$ *in which the* j*th row and column are removed.*
- $\tilde{\boldsymbol{r}}_j$ *is the* j*th column of* $\boldsymbol{R}^{-1}$ *in which the* j*th element is removed.*
- $(\boldsymbol{R}^{-1})_{jj}$ *is element* jj *of* $\boldsymbol{R}^{-1}$.

Proof. The proof follows directly by applying the method of partitioning.

□

From (4.57) follows that the matrix $\boldsymbol{R}_j$ equals the matrix $\boldsymbol{R}$ in which the jth row and column are removed. Hence, Theorem 4.1 can be used to calculate $\boldsymbol{R}_j^{-1}$ once $\boldsymbol{R}^{-1}$ is known. Because, only ρ_j is changed in each iteration of the ACH algorithm, the matrix $\boldsymbol{R}^{-1}$ can be updated quite easily.

Suppose that the matrices $\boldsymbol{R}_m$ and $\boldsymbol{\Theta}_m$ are the matrices $\boldsymbol{R}$ and $\boldsymbol{\Theta}$ at iteration m. Further, $\rho_{j,m}$ denotes the weighting factor ρ_j at iteration m. Now, the following theorem can be used to calculate $\boldsymbol{R}_{m+1}^{-1}$ if weighting factor $\rho_{j,m}$ is changed.

Theorem 4.2 *If weighting factor* $\rho_{j,m}$ *is increased by* δ*, then* $\boldsymbol{R}_{m+1}^{-1}$ *is given by:*

$$\boldsymbol{R}_{m+1}^{-1} = \boldsymbol{R}_m^{-1}(\boldsymbol{I} + \boldsymbol{\Delta}_m \boldsymbol{R}_m{}^{-1})^{-1} \tag{4.59}$$

where:

$$\boldsymbol{\Delta}_m = diag(0, \cdots, 0, \underbrace{\delta}_{j}, 0, \cdots, 0)$$

Proof. At iteration m weighting factor $\rho_{j,m}$ is increased by δ:

$$\rho_{j,m+1} = \rho_{j,m} + \delta$$

Matrix Θ_{m+1} then becomes:

$$\Theta_{m+1} = \Theta_m + \Delta_m \tag{4.60}$$

The inverse of R at iteration m is:

$$R_m^{-1} = (A^T A + \Theta_m)^{-1} \tag{4.61}$$

Substitution of (4.60) in (4.61) yields:

$$R_m^{-1} = (A^T A + \Theta_{m+1} - \Delta_m)^{-1} \tag{4.62}$$

Rewriting (4.62) using $R_{m+1} = (A^T A + \Theta_{m+1})$ yields:

$$R_{m+1}^{-1} = R_m^{-1}(I + \Delta_m R_m{}^{-1})^{-1}$$

where I is the identity matrix of dimension $H_c \times H_c$.

□

Because Δ_m has only one element different from zero, the inverse in (4.59) is easy to calculate.

Theorem 4.3 *If* $\Delta_m = diag(0, \cdots, 0, \underbrace{\delta}_{j}, 0, \cdots, 0)$, *then* $(I + \Delta_m R_m{}^{-1})^{-1}$ *is given by:*

$$(I + \Delta_m R_m{}^{-1})^{-1} =$$

$$\begin{bmatrix}
1 & 0 & & & \cdots & & & 0 \\
0 & & & & & & & \\
\vdots & \ddots & \ddots & \ddots & & & & \vdots \\
0 & \cdots & 0 & 1 & 0 & & \cdots & 0 \\
z_{j1} & & \cdots & z_{jj-1} & z_{jj} & z_{jj+1} & \cdots & z_{jH_c} \\
0 & & \cdots & & 0 & 1 & 0 \cdots & 0 \\
\vdots & & & & & \ddots & \ddots \ddots & \vdots \\
 & & & & & & & 0 \\
0 & & & & \cdots & & 0 & 1
\end{bmatrix} \tag{4.63}$$

where:

$$z_{ji} = -\frac{\delta r_{ji}}{\delta r_{jj}+1} \quad \text{if } i \neq j$$

$$z_{ji} = \frac{1}{\delta r_{jj}+1} \quad \text{if } i = j$$

and r_{ji} is element ji of matrix $\boldsymbol{R}_m^{-1}$.

Proof. If r_{ji} is element ji of matrix $\boldsymbol{R}_m^{-1}$, $\boldsymbol{I} + \boldsymbol{\Delta}_m \boldsymbol{R}_m{}^{-1}$ is:

$$\boldsymbol{I} + \boldsymbol{\Delta}_m \boldsymbol{R}_m{}^{-1} = \begin{bmatrix} 1 & 0 & & & \cdots & & & & 0 \\ 0 & & & & & & & & \\ \vdots & \ddots & \ddots & \ddots & & & & & \vdots \\ 0 & \cdots & 0 & 1 & 0 & & \cdots & & 0 \\ \delta r_{j1} & & \cdots & & 1+\delta r_{jj} & & \cdots & & \delta r_{jH_c} \\ 0 & & \cdots & & 0 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & \ddots & \ddots & \ddots & \vdots \\ & & & & & & & & 0 \\ 0 & & & & \cdots & & & 0 & 1 \end{bmatrix}$$

Due to the simple structure of $\boldsymbol{I} + \boldsymbol{\Delta}_m \boldsymbol{R}_m{}^{-1}$, its inverse is straightforward and given by (4.63).

□

Calculation of $\boldsymbol{R}_j^{-1}$ in (4.56) by using (4.58), (4.59) and (4.63) instead of using a standard matrix inversion algorithm yields a considerable decrease in the number of floating point operations (FLOPs) required to do so. Table 4.1 shows the number of additions ($\sum$), multiplications ($\prod$) and divisions (/) that are required to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$ if ρ_j is changed for a standard algorithm (see section A.4) and a dedicated algorithm based on the Theorems 4.1, 4.2 and 4.3. For both algorithms the symmetry of matrix $\boldsymbol{R}_j$ has been exploited. Further, Table 4.2 shows the number of FLOPs that are required to calculate $\boldsymbol{R}_j^{-1}$ for both algorithms. The Tables 4.1 and 4.2 show that for small values for H_c the number of FLOPs required to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$ is approximately equal for both algorithms. However, for larger values of H_c, the number of FLOPs required to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$ is considerably smaller when using the dedicated algorithm.

Algorithm	Σ	Π	/
Standard	$\frac{3}{2}H_c^3 - 3H_c^2 + \frac{5}{2}H_c - 1$	$\frac{3}{2}H_c^3 - 2H_c^2 + \frac{1}{2}H_c$	$\frac{1}{2}H_c^3 + \frac{1}{2}H_c - 1$
$H_c = 2$	4	5	4
$H_c = 5$	124	140	64
Dedicated	$H_c^2 + 1$	$H_c^2 + H_c$	$\frac{1}{2}H_c^2 + \frac{3}{2}H_c - 1$
$H_c = 2$	5	6	4
$H_c = 5$	26	30	19

Table 4.1: Numerical complexity for a standard and a dedicated algorithm to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$.

Algorithm	FLOPs
Standard	$\frac{7}{2}H_c^3 - 5H_c^2 + \frac{7}{2}H_c - 2$
$H_c = 2$	13
$H_c = 5$	328
Dedicated	$\frac{5}{2}H_c^2 + \frac{5}{2}H_c$
$H_c = 2$	15
$H_c = 5$	75

Table 4.2: Number of FLOPs required to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$.

Utilizing the dedicated algorithm for calculating $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$, Algorithm 4.2 becomes:

Algorithm 4.3 The ACH algorithm using a dedicated algorithm to calculate $\boldsymbol{R}_j^{-1}$ and $\boldsymbol{R}_{m+1}^{-1}$.

Step 1: Initialization step: Calculate the optimal controller output sequence of the unconstrained optimization problem with $Q_n = Q_d = P = N = \beta = 1$ by using (2.184). This yields, among others: $\boldsymbol{R}_0^{-1}$, $\boldsymbol{\Theta}_0$ and $m = 0$.

Step 2: **If** $|u(k+j-1)| \leq \eta |u_b| \;\; \forall j$ **Then**
Solution found.
End if

Step 3: For $j = 1, \ldots, H_c$ Do
If $|u(k+j-1)| > \eta |u_b|$ Then
Calculate R_j^{-1} using (4.58)
Calculate $\tilde{u}_j$ using (4.56) and form with u_b the vector $\bar{u}$
Calculate ρ_j using (4.54) and form with $\tilde{\Theta}_j$
the matrix Θ_{m+1}
Calculate R_{m+1}^{-1} using (4.59) and (4.63)
$m = m + 1$
End If
End For
Goto Step 2.

In the initialization step Θ_0 can be taken equal to:

$$\Theta_0 = diag(\rho, \cdots, \rho)$$

where ρ is the weighting factor originally set by the designer. Note that the ACH algorithm must be calculated in every sample. However, because R_0^{-1} is time invariant, if the model and the UPC design parameters are constant, R_0^{-1} need not be calculated in every sample.

Numerical complexity of the ACH algorithm

In order to examine the numerical complexity of the ACH algorithm , the number of additions, multiplications and divisions necessary in order to evaluate (4.56) and (4.54) (thus including the calculation of R_j^{-1} and R_{m+1}^{-1}) are shown in Table 4.3. Moreover, the total number of FLOPs necessary for each iteration of the ACH algorithm is shown in Table 4.4. The numbers in the Tables 4.3 and 4.4 are calculated

	Σ	Π	/
$\tilde{u}_j$	$2H_c^2 - H_c + 1$	$2H_c^2 - H_c + 2$	$\frac{1}{2}H_c^2 + \frac{3}{2}H_c - 1$
ρ_j	$H_c + 1$	H_c	1

Table 4.3: Number of FLOPs required to calculate $\tilde{u}_j$ and ρ_j.

based on the fact that if $A^T z$ and $A^T A$ are available, $\tilde{A}_j^T z$ can be obtained by eliminating element j of $A^T z$. Further, $\tilde{A}_j^T \lambda_j$ is equal to the jth column of $A^T A$ with the

jth element eliminated. Because, $\boldsymbol{A}^T\boldsymbol{z}$ and $\boldsymbol{A}^T\boldsymbol{A}$ do not change during the period the ACH algorithm seeks for a solution, no floating point computations are required to find $\tilde{\boldsymbol{A}}_j^T\boldsymbol{z}$ and $\tilde{\boldsymbol{A}}_j^T\boldsymbol{\lambda}_j$ in (4.56).

In evaluating (4.54), $\boldsymbol{\lambda}_j^T\boldsymbol{A}$ and $\boldsymbol{\lambda}_j^T\boldsymbol{z}$ are required. However, $\boldsymbol{\lambda}_j^T\boldsymbol{A}$ is given by the jth row of $\boldsymbol{A}^T\boldsymbol{A}$ while $\boldsymbol{\lambda}_j^T\boldsymbol{z}$ is given by the jth element of $\boldsymbol{A}^T\boldsymbol{z}$. Hence, also in this case no extra computations are required. Table 4.4 also gives an impression

H_c	Σ	Π	/	FLOPs	T_c in ms
H_c	$2H_c^2+2$	$2H_c^2+2$	$\frac{1}{2}H_c^2+\frac{3}{2}H_c$	$\frac{9}{2}H_c^2+\frac{3}{2}H_c+4$	
2	10	10	5	25	0.7
5	52	52	20	124	3.0

Table 4.4: Total number of FLOPs and calculation time required to perform one iteration of the ACH algorithm.

of the calculation times (T_c) required for each iteration of the ACH algorithm on a Tulip SX compact 2 (a PC with 386 SX processor) with coprocessor. The ACH algorithm was programmed in Turbo C V2.0.

4.2.3 Implementation and Numerical Complexity

Based on the previous discussion on how to calculate the search direction and the optimal step in the search direction, Rosen's gradient projection method can be dedicated to minimize (4.4) subject to level constraints. The most complicated calculation that is required is that of calculating the gradient $\boldsymbol{g}_n$. This gradient must be calculated every iteration using (2.182). However, in (2.182) only $\bar{\boldsymbol{u}}_n$ depends on the iteration number. Therefore, many computations can be done before starting the iterative search. Let $\boldsymbol{h}$ be given by:

$$\boldsymbol{h} = 2\boldsymbol{M}^T\left[\boldsymbol{\Gamma}^T\left(\boldsymbol{\Psi}\boldsymbol{s}+\boldsymbol{k}+\boldsymbol{\Gamma}\boldsymbol{N}\check{\boldsymbol{u}}-\boldsymbol{w}^*\right)+\rho\boldsymbol{\Phi}^T\left(\boldsymbol{\Omega}\tilde{\boldsymbol{u}}+\boldsymbol{\Phi}\boldsymbol{N}\check{\boldsymbol{u}}\right)\right] \tag{4.64}$$

Then $\boldsymbol{g}_n$ is given by:

$$\boldsymbol{g}_n = \boldsymbol{H}\bar{\boldsymbol{u}}_n + \boldsymbol{h} \tag{4.65}$$

where $\boldsymbol{H}$ is the Hessian given by (2.183). The vector $\boldsymbol{h}$ does not depend on the iteration number n and hence need not be calculated each iteration. Moreover, the solution to the unconstrained optimization problem is now given by:

$$\overline{\boldsymbol{u}} = -\boldsymbol{H}^{-1}\boldsymbol{h} \tag{4.66}$$

The resulting algorithm is called DGP (Dedicated Gradient Projection) and is described below.

Algorithm 4.4 The DGP algorithm.

Step 1: Initialization step. Calculate $\overline{\boldsymbol{u}}_0$, $\boldsymbol{K}_0$, $\boldsymbol{h}$ and set $n = 0$.

Step 2: **While** no optimal solution found **Do**
Calculate $\boldsymbol{g}_n$ using (4.65)
Determine $\boldsymbol{d}_n$ using (4.30) and (4.31)
If $\max_j |d_j| < \epsilon$ **Then**
Determine $\boldsymbol{n}_n$ using (4.43) and (4.44)
If $n_j > 0 \; \forall j$ **Then**
Optimal solution found.
Else
Remove active constraint from $\boldsymbol{K}_n$
End If
Else
Calculate $\lambda_{opt,n}$ using (4.40)
Calculate λ_{max} using (4.41)
Calculate $\lambda^*_{opt,n}$ using (4.42)
$\overline{\boldsymbol{u}}_{n+1} = \overline{\boldsymbol{u}}_n + \lambda^*_{opt,n}\boldsymbol{d}_n$
If $\lambda^*_{opt,n} = \lambda_{max}$ **Then**
Add inactive constraints to $\boldsymbol{K}_n$
End If
$n = n + 1$
End If
End While
Goto Step 2

Now, the numerical complexity of the DGP algorithm can be calculated. Table 4.5 shows the number of floating point operations and the resulting calculation time that are required to perform one iteration of the DGP algorithm.

H_c	Σ	Π	/	FLOPs	T_c in ms
H_c	$2H_c^2+3H_c$	$2H_c^2+2H_c$	H_c+1	$4H_c^2+6H_c+1$	
2	14	12	3	29	0.8
5	65	60	6	131	3.2

Table 4.5: Number of FLOPs and calculation times required to perform one iteration of the DGP algorithm.

Remark: the number of FLOPs to perform one iteration of the DGP algorithm depends on the number of active constraints. The numbers shown in Table 4.5 are calculated for the worst case i.e. $r = H_c$.

Embedding the DGP algorithm in a control scheme that calculates the optimal controller output sequence over the control horizon in every sample, yields:

Algorithm 4.5 Using the DGP algorithm to calculate $\overline{u}$ every sample.

Step 1: Initialization step.

Step 2: Measure $y(k)$.

Step 3: Calculate $\overline{u}$ optimal to the unconstrained optimization problem by using (4.66).

Step 4: **If** $u(k+j-1) < \underline{u} \vee u(k+j-1) > \overline{u}$ $\forall j$ **Then**
Use the DGP algorithm 4.4 to find $\overline{u}$.
End If
Use $u(k)$ to control the process.
Wait until next sample
Goto Step 2.

In the initialization step, all time invariant matrices, vectors and other variables can be calculated. If the model is not estimated on line, the matrices of the prediction model Γ, Ψ and K_c can be calculated in this step. Moreover, if the UPC design parameters do not change in time, the matrices M, N, Φ and Ω in (2.179) are time invariant. As a result, the Hessian H (2.183) and its inverse can be calculated in the initialization step and hence need not be calculated every sample or iteration step of the DGP algorithm. Now, the solution to the unconstrained optimization problem is given by (4.66). The vector h must be calculated every sample using (4.64). This requires xH_c additions and multiplications in which $x = \max(n_P - H_c, n_A -$

$1) + n_B + \bar{d} + n_K + 1 + \beta + n_N - H_c + H_p - \underline{d} + \max(Q_n, Q_d)$ (of course, all matrix products in (4.64) should be calculated in the initialization step). Hence, in order to calculate $\bar{u}$ optimal to the unconstrained optimization problem, $H_c^2 + xH_c$ additions and multiplications are required. Note that if the constraints are not taken into account as in Chapters 2 and 3, only x additions and multiplications are required in every sample (see (2.185)).

The total number of FLOPs that are required to calculate $u(k)$ depends on the constraints and the number of iterations required by the DGP algorithm to solve the constrained optimization problem. Assume that the number of iterations required by the DGP algorithm is given by I_{DGP}, then the total number of FLOPs that are required every sample is given by: $2H_c^2 + 2xH_c + I_{DGP}(4H_c^2 + 6H_c + 1)$. Further, if the ACH algorithm is used to calculate the starting point of the DGP algorithm, then the total number of FLOPs is given by: $2H_c^2 + 2xH_c + I_{DGP}(4H_c^2 + 6H_c + 1) + I_{ACH}(\frac{9}{2}H_c^2 + \frac{3}{2}H_c + 4)$ where I_{ACH} is the number of iterations of the ACH algorithm.

Note: the number of iterations required by the DGP and ACH algorithms are time variant. Subsequently, $I_{DGP}(k)$ and $I_{ACH}(k)$ denote the number of iterations required by DGP and ACH at sample k, respectively.

4.2.4 Simulation Results

In this section, the DGP and ACH algorithms are illustrated by some examples. First it is shown that the performance of the closed-loop system in the presence of level constraints increases significantly when the constraints are explicitly taken into account in the minimization of the criterion function. The performance is judged by using the performance criteria O and t_s which, in my opinion, are the most important. If both O and t_s decrease or increase, the performance of the closed-loop system is increased or decreased, respectively. If, on the other hand, one of the criteria increases and the other decreases, it is up to the designer to decide which of the criteria is the most important.

Further, the influence of the starting point of DGP on the number of iterations required to solve the constrained optimization problem is shown. The DGP algorithm is implemented using the Algorithms 4.4 and 4.5.

Remark: In Chapter 3 all simulations were performed by using the discrete time equivalent of the continuous process. Consequently, the accuracy of the performance criteria t_r, t_p, t_s and O depend on the sampling period. For the simulations discussed

in Chapter 3 this accuracy was sufficient. However, for some simulations that are presented in this chapter, this is no longer the case. Therefore, all simulations results shown in this chapter were obtained by simulating the continuous process by using a numerical integration method. Here, the second-order Runge-Kutta method was used with a fixed integration step I_s. The integration step is selected such that the accuracy of the performance criteria is sufficient.

Using the DGP algorithm to control the process in the presence of level constraints

The following example illustrates the use of the DGP algorithm in the case of level constraints on the controller output.

Example 4.3 The use of the DGP algorithm in controlling a process in the presence of level constraints.

Settings:	$H_p = 4$, $H_s = 2$, $H_c = 3$, $\beta = 1$, $R = P = N = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$, $\underline{u} = -3$, $\overline{u} = 15$, $\underline{\Delta u} = -20$, $\overline{\Delta u} = 20$, $I_s = 0.05$s
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)}$
Model:	identical to process
Parameters:	None

Note that the rate constraints cannot become active. Figure 4.4a shows the response when the level constraints are not taken into account. Hence, the solution to the unconstrained optimization problem is used to control the process. Due to the asymmetrical constraints, a large overshoot occurs ($\approx$ 53 %). Activating the DGP algorithm where $\epsilon = 10^{-4}$ initialized at $\overline{u}_0 = 0$ results in the response shown in Figure 4.4b. Definitely, the performance of the closed-loop system is significantly increased by taking the constraints into account. Now, the overshoot is $\approx$ 7.2 %. In order to show the influence of the parameter ϵ on the performance of the closed-loop system, a number of simulations have been performed using different ϵ. The performance criteria O and t_s as a function of ϵ are shown in Figure 4.5. The Figure shows that choosing $\epsilon < 8 \cdot 10^{-3}$ yields results close to the optimal one (obtained for $\epsilon \to 0$). The steady-state error is when $\epsilon \leq 0.6$ equal to zero. If $\epsilon > 0.6$, the steady-state error is 100 % because in this case the starting point (= 0) already satisfies the stop criterion of the DGP algorithm.

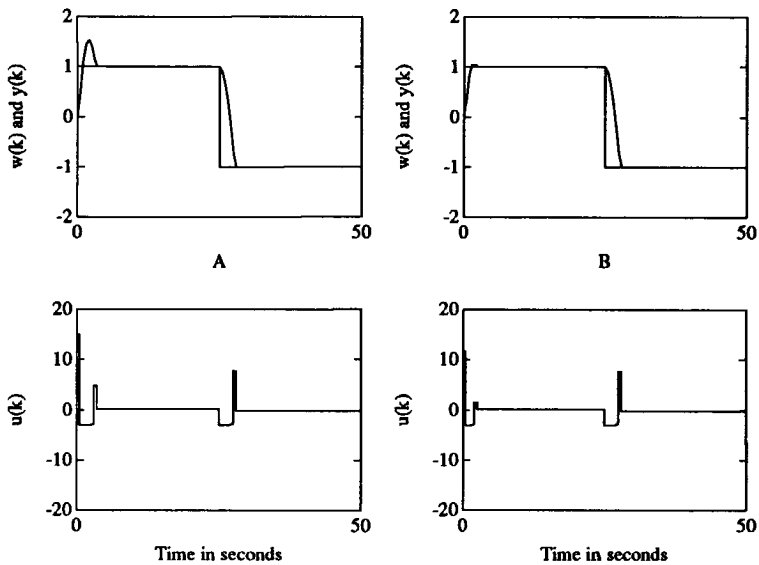

Figure 4.4: Response without DGP (Figure A) and with DGP (Figure B) in the case of level constraints: $\underline{u} = -3$ *and* $\overline{u} = 15$.

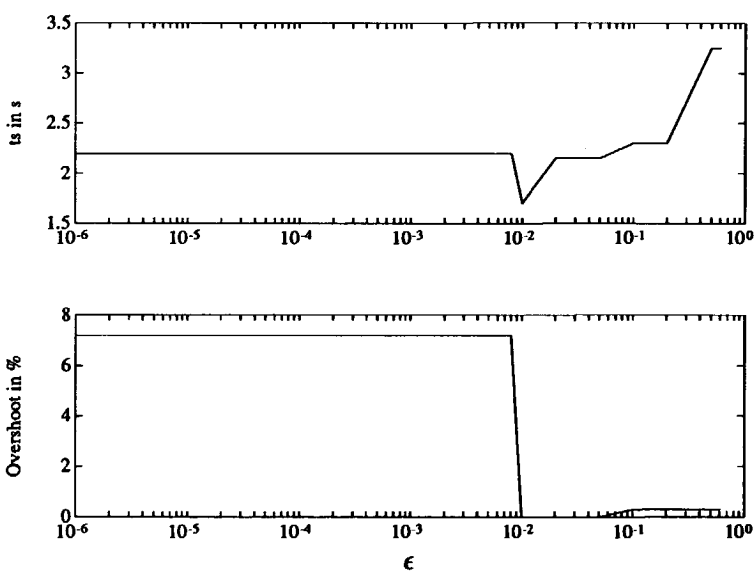

Figure 4.5: Settling time and overshoot as a function of ϵ.

The effect of the prediction horizon in the case of level constraints

When constraints are active, the closed-loop system and the UPC controller are nonlinear. Therefore it is hard to predict what the effect of the UPC design parameters will be on the response of the closed-loop system. However, the effect of the prediction and control horizon can easily be predicted. Using a control horizon equal to one results in a one-dimensional optimization problem: only $u(k)$ has to be determined. Now, the optimal solution to the constrained optimization problem can be simply obtained by clipping $u(k)$ by using (4.45). Hence, in this case, using the DGP algorithm will not yield a different performance of the closed-loop system. When the prediction and control horizon are different from 1, the DGP algorithm is expected to give a better performance (i.e. a smaller overshoot and settling time) of the system when the constraints become active. Because by minimization of (4.4), taking into account the constraints, the effect of these constraints on the predicted process output is taken into account. Now, a large overshoot caused by the constraints will result in a large value of the criterion function. Minimization of this criterion function taking into account the constraints will obviously make the overshoot smaller. Therefore, it can be expected that using a larger prediction horizon makes the performance of the closed-loop system increase. The following example illustrates this.

Example 4.4 Influence of H_p on the performance and stability of the closed-loop system in the presence of level constraints.

Settings:	$H_s = 2$, $H_c = 3$, $\beta = 1$, $R = P = N = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$, $\underline{u} = -1$, $\overline{u} = 10$, $\underline{\Delta u} = -20$, $\overline{\Delta u} = 20$, $\epsilon = 10^{-4}$, $I_s = 0.05$s
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)}$
Model:	identical to process
Parameters:	H_p with $H_p \geq 4$

Note that when the constraints are not taken into account, the closed-loop system does not depend on H_p. For all $H_p \geq 4$ a dead-beat controller is obtained according to Theorem 2.4. Consequently, if this (fixed) controller is used to control the process in the presence of constraints, the performance of the system is independent of H_p. When the constraints are taken into account in the optimization problem, the prediction horizon does affect the system's performance. Table 4.6 shows the performance indices as a function of the prediction horizon when the constraints are taken into account and when they are not. The steady-state error is equal to zero for all $H_p \geq 4$.

When $H_p = 4$, the performance is already significantly better when the constraints are taken into account. The prediction horizon can be further increased to improve the overshoot and settling time. Note, however, that increasing H_p from 6 to 8 makes the performance of the system slightly worse compared to when $H_p = 5$. Simulations using $\epsilon = 10^{-6}$ showed that this effect is not caused by the finite accuracy of the solution. Figure 4.6a shows the response when the constraints are not

H_p	O in %	t_s in s
4	21.3	4.1
5	12.5	3.6
6	9.7	3.5
7	10.4	3.5
8	12.2	3.6
9	5.3	3.1
≥ 10	0	2.2
Without DGP	54.1	5.2

Table 4.6: Performance criteria as a function of H_p in the case of level constraints.

taken into account. Figure 4.6b shows the response when DGP is used with $H_p = 10$.

Initialization of $\overline{u}$ and the effect on the calculation time

So far the only concern has been the performance of the closed-loop system in the presence of constraints. It was shown that this performance is dependent on ϵ and, if ϵ is chosen small enough, it is independent of the starting point of the optimization algorithm.

In this section, the effect of the initialization of $\overline{u}$ on the speed of convergence of the DGP algorithm is examined. As a measure for the speed of convergence of the DGP algorithm, the number of iterations required to find the solution to the constrained optimization problem with accuracy ϵ, is used. Because the number of iterations required to find an optimal solution with accuracy ϵ is time variant, the number of iterations is considered for every sample.

Figure 4.7b shows how many iterations are required by DGP in every sample to find the optimal solution in the case $\overline{u}_0 = 0$ and $\epsilon = 10^{-4}$. Note that most of the

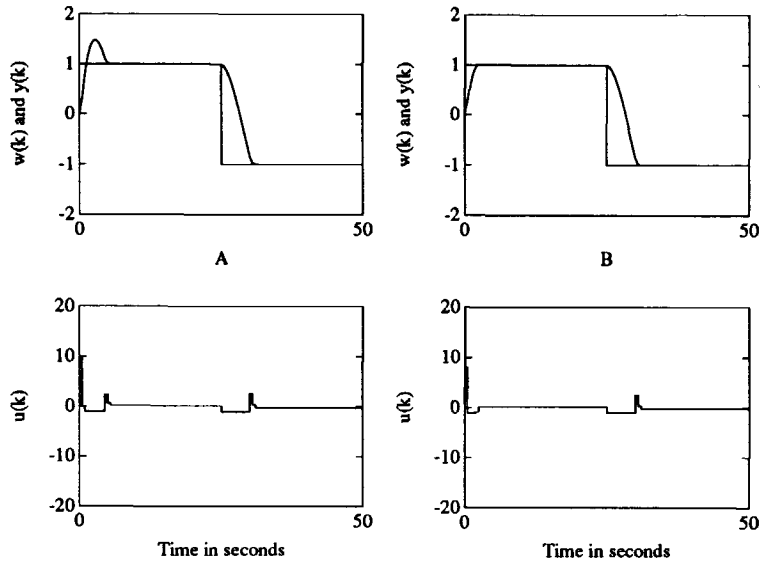

Figure 4.6: Response without DGP (Figure A) and with DGP and $H_p = 10$ (Figure B) in the case of level constraints: $\underline{u} = -1$ and $\overline{u} = 10$.

time $I_{DGP}(k)$ is equal to zero. This is caused by the fact that if the solution to the unconstrained problem does not violate the constraints, this solution is directly used to control the process (see also Algorithm 4.5). All other settings were equal to those in Example 4.4 with $H_p = 10$. Figure 4.7a shows the results if the starting point is generated using the solution calculated in the previous sample as discussed in section 4.2.1 on page 246. In this case the number of iterations that are required by DGP can be even worse. The figures show that usually the number of iterations required by DGP in each sample is rather small. However, sometimes a large peak in the number of iterations occurs which causes a long calculation time for that particular sample.
Four ways to ensure that the number of iterations remains small are:

- Use a larger value for ϵ. By doing so, the accuracy of the solution becomes smaller. When ϵ is made too large, the solution can be far from the optimal solution. That ϵ has a strong effect on the number of iterations required by the DGP algorithm is shown by Table 4.7. In this table, $\max_k I_{DGP}(k)$ is shown as a function of ϵ. Clearly, increasing ϵ results in a considerable decrease of iterations. The effect of ϵ on the performance of the system is also shown in

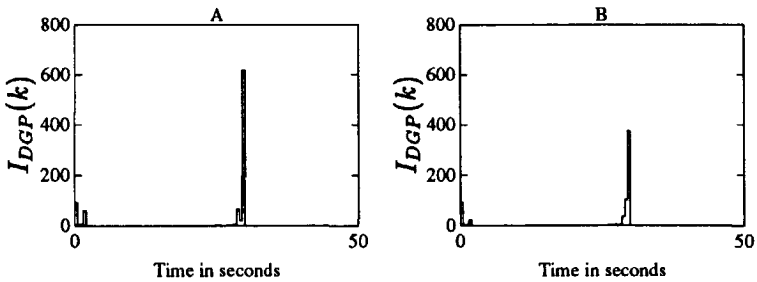

Figure 4.7: Number of iterations required by DGP ($I_{DGP}(k)$) if the solution found at $t = k - 1$ is used (Figure A) and if $\overline{u}_0 = 0$ (Figure B).

ϵ	$\max_k I_{DGP}(k)$	$\mathcal{O}$ in %	t_s in s
10^{-5}	484	0	2.2
10^{-4}	376	0	2.2
10^{-3}	270	0	2.2
10^{-2}	69	0	2.2
10^{-1}	7	0	3.1

Table 4.7: $\max_k I_{DGP}(k)$ and performance as a function of ϵ.

Table 4.7. Obviously, for this system using $\epsilon \leq 10^{-2}$ yields solutions close to the optimal one. Hence, using $\epsilon = 10^{-2}$ results in a good performance with a relatively small number of iterations. Using a smaller value for ϵ does not result in a better performance but makes the number of iterations increase. Other simulations have shown that using a larger value for ϵ results in solutions far from the optimal one.

- Impose a maximum on the number of iterations that is required by the DGP algorithm in every sample. By doing so, the calculation time of the DGP algorithm is bounded. However, the solution can be far from the optimal solution if the maximum allowed number of iterations is much smaller than required to find the solution with accuracy ϵ.

- Calculate a better starting point for the optimization. In this case, the accuracy of the solution is not affected. However, if the starting point is chosen closer to the optimal solution, the number of iterations required will be much smaller. An algorithm that can be used for this purpose is the ACH algorithm

described in section 4.2.2. As shown by the Tables 4.4 and 4.5, the number of FLOPs required for one iteration of the DGP and ACH algorithms are of the same order. Hence, in order to get an improvement in the calculation time, $\max_k I_{ACH}(k) + I_{DGP}(k)$ must be smaller than $\max_k I_{DGP}(k)$ when not using the ACH algorithm. Table 4.8 shows the number of iterations required by ACH and DGP for different values of η and ϵ. Note that, independent of η, the accuracy of the solution that is found is always less than ϵ because the ACH algorithm is used only to calculate the starting point for the DGP algorithm. The symbol '-' in Table 4.8 denotes the case in which the ACH

η	ϵ	$\max_k I_{ACH}$	$\max_k I_{DGP}$	$\max_k I_{ACH}+I_{DGP}$	$\max_k T_c$ in ms
-	10^{-4}	0	376	376	870
-	10^{-2}	0	69	69	160
2	10^{-4}	3	11	11	30
1.5	10^{-4}	7	42	44	104
1.1	10^{-4}	10	42	44	105
1.01	10^{-4}	15	42	44	105
2	10^{-2}	3	5	5	16
1.5	10^{-2}	8	7	10	26
1.1	10^{-2}	10	8	10	25
1.01	10^{-2}	16	8	17	28

Table 4.8: Number of iterations and total calculation times of ACH and DGP for different values for η and ϵ.

algorithm was not used. The table shows that when using the ACH algorithm to calculate $\overline{u}_0$, a considerable decrease of computation time occurs compared to the case where $\overline{u}_0 = 0$ (which is the starting point in the case the ACH algorithm is not used).

In order to get an impression of the total calculation time required to calculate the optimal controller output in the presence of constraints in each sample, some measurements were performed on a Tulip SX compact 2 (a PC with 386 SX processor) with numeric coprocessor. All algorithms were programmed in Turbo C V2.0. Because the calculation time can be different each sample, $\max_k T_c(k)$ is used as a measure. The time required to calculate the optimal solution to the unconstrained optimization problem is $\approx$ 5.6 ms. Although the total calculation time when using the DGP algorithm in combination

with the ACH algorithm is about 5 times greater if $\epsilon = 10^{-2}$, the calculation times are still relatively small.

- If the set of active constraints remains constant while searching for an optimal solution in the $H_c - r$ dimensional subspace, the DGP algorithm is a steepest-descent algorithm. It is well known that methods using conjugate search directions usually converge much faster. For example, in [55] a modified version of the conjugate-gradient method of Fletcher and Reeves [56] is proposed which can be used to minimize a criterion function subject to linear (in)equality constraints. This method proceeds as follows. The first search direction is the projected gradient. Then, as long as the set of active constraints remains constant, successive search directions are generated that are conjugated to the previously calculated ones. Thus, this method uses conjugate search directions to find an optimum in the $H_c - r$ dimensional subspace. However, when the set of active constraints changes (because either $\max_j |d_j| < \epsilon$ or $\lambda^*_{opt,n} = \lambda_{max}$) the procedure must be restarted using the projected gradient for the first search direction. Hence, if the set of active constraints changes many times during the search for an optimal solution of the constrained optimization problem, generating conjugate search directions is not expected to decrease the number of iterations significantly [55]. However, if the set of active constraints does not change many times during the iterative search, the method described in [55] is expected to give a faster convergence. Further research in this respect would, therefore, be useful.

4.2.5 Rate Constraints

In this section only rate constraints are considered. The reason why rate constraints and level constraints are not considered here is that, in this case, some extra problems arise. Section 4.2.6 discusses the case in which level and rate constraints are present simultaneously.

Also, in the case of rate constraints Rosen's gradient projection method, as depicted in Figure 4.3, can be used. However, due to the fact that active rate constraints yield a different constraint matrix, the calculation of the search direction and the Kuhn-Tucker vector can no longer be done by using (4.30), (4.31), (4.43) and (4.44). The optimal step in the search direction can still be calculated using (4.40). However, the maximum step in the search direction can no longer be calculated using (4.41). Also, the initialization of $\overline{u}$ needs some attention. In the following sections,

each part of Rosen's gradient projection method that is different from that in the case of level constraints is discussed separately.

The search direction in the case of rate constraints

When the rate constraints (4.21) and (4.22) are present on the controller output, (4.10) becomes:

$$\boldsymbol{A}\overline{\boldsymbol{u}} \leq \boldsymbol{b}: \left[\begin{array}{ccccc} 1 & 0 & \cdots & & 0 \\ -1 & 1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & 1 & 0 \\ 0 & \cdots & 0 & -1 & 1 \\ \hline -1 & 0 & \cdots & & 0 \\ 1 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & -1 & 0 \\ 0 & \cdots & 0 & 1 & -1 \end{array}\right] \overline{\boldsymbol{u}} \leq \left[\begin{array}{c} \overline{\Delta u} + u(k-1) \\ \overline{\Delta u} \\ \vdots \\ \overline{\Delta u} \\ \hline -\underline{\Delta u} - u(k-1) \\ -\underline{\Delta u} \\ \vdots \\ -\underline{\Delta u} \end{array}\right]$$

The search direction is still given by (4.28). Further, the projection matrix $\boldsymbol{P}_n$ is given by (4.29). Because $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is no longer equal to the identity matrix, the calculation of $\boldsymbol{P}_n$ is more difficult than in the case of level constraints. However, also when there are rate constraints, the matrix product $\boldsymbol{K}_n\boldsymbol{K}_n^T$ has a special structure.

Theorem 4.4 *In the case of active rate constraints, the matrix product* $\boldsymbol{K}_n\boldsymbol{K}_n^T$ *is a tridiagonal matrix.*

Proof. Element ij of $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is calculated using:

$$(\boldsymbol{K}_n\boldsymbol{K}_n^T)_{ij} = {}_r(\boldsymbol{K}_n)_i \; {}_r(\boldsymbol{K}_n)_j^T$$

Hence, element ij of $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is given by calculating the inner product of row i and row j of $\boldsymbol{K}_n$. Suppose row i (denotes the ith active constraint) of matrix $\boldsymbol{K}_n$ corresponds to a rate constraint on $u(k+l-1)$ and row j corresponds to a rate

constraint on $u(k+m-1)$ where $l \neq m$ if $i \neq j$ and $l = m$ if $i = j$. Then, row i and row j of matrix K_n are given by:

$$
\begin{aligned}
{}_r(K_n)_i &= [0, \cdots, 0, k_{il-1}, k_{il}, 0, \cdots, 0] && l > 1 \\
{}_r(K_n)_i &= [k_{il}, 0, \cdots, 0] && l = 1 \\
{}_r(K_n)_j &= [0, \cdots, 0, k_{jm-1}, k_{jm}, 0, \cdots, 0] && m > 1 \\
{}_r(K_n)_j &= [k_{jm}, 0, \cdots, 0] && m = 1
\end{aligned}
$$

where if $l > 1$ and $m > 1$:

$$
\begin{aligned}
k_{il-1} = -1 \quad \wedge \quad k_{il} = 1 \quad & \text{if } \Delta u(k+l-1) = \overline{\Delta u} \\
k_{il-1} = 1 \quad \wedge \quad k_{il} = -1 \quad & \text{if } \Delta u(k+l-1) = \underline{\Delta u} \\
k_{jm-1} = -1 \quad \wedge \quad k_{jm} = 1 \quad & \text{if } \Delta u(k+m-1) = \overline{\Delta u} \\
k_{jm-1} = 1 \quad \wedge \quad k_{jm} = -1 \quad & \text{if } \Delta u(k+m-1) = \underline{\Delta u}
\end{aligned}
$$

and if $l = 1$

$$
\begin{aligned}
k_{il} &= 1 \quad \text{if } \Delta u(k+l-1) = \overline{\Delta u} \\
k_{il} &= -1 \quad \text{if } \Delta u(k+l-1) = \underline{\Delta u}
\end{aligned}
$$

and if $m = 1$

$$
\begin{aligned}
k_{jm} &= 1 \quad \text{if } \Delta u(k+m-1) = \overline{\Delta u} \\
k_{jm} &= -1 \quad \text{if } \Delta u(k+m-1) = \underline{\Delta u}
\end{aligned}
$$

Because $K_n K_n^T$ is symmetric, the case $i \leq j$ only is considered. Taking into account the fact that there cannot be more than one active rate constraint on $\Delta u(k+l-1)$ and $\Delta u(k+m-1)$, the ijth element of $K_n K_n^T$ becomes:

$$
(KK^T)_{ij} = {}_r(K_n)_i \; {}_r(K_n)_j^T =
$$

$$
\begin{aligned}
&= 1 \quad \text{if } i = j \wedge l = 1 \\
&= 2 \quad \text{if } i = j \wedge l > 1 \\
&= -1 \quad \text{if } i = j - 1,\ l = m - 1 \text{ and} \\
&\qquad \Delta u(k+l-1) = \Delta u(k+m-1) = \overline{\Delta u} \text{ or} \\
&\qquad \Delta u(k+l-1) = \Delta u(k+m-1) = \underline{\Delta u}
\end{aligned}
$$

$$
\begin{aligned}
&= 1 \quad \text{if } i = j-1,\ l = m-1,\ \Delta u(k+l-1) = \overline{\Delta u} \text{ and} \\
&\qquad \Delta u(k+m-1) = \underline{\Delta u} \text{ or } \Delta u(k+l-1) = \underline{\Delta u} \text{ and} \\
&\qquad \Delta u(k+m-1) = \overline{\Delta u} \\
&= 0 \quad \text{for all other values for } i,\ j,\ l \text{ and } m
\end{aligned}
$$

Hence in the case of rate constraints, $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is a tridiagonal matrix (= a matrix with nonzero elements only on the diagonal plus and minus one column).

□

Because $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is a tridiagonal matrix, the search direction can be calculated quite easily. Rewriting (4.28) and (4.29) yields:

$$\boldsymbol{d}_n = -\boldsymbol{g}_n - \boldsymbol{K}_n^T\boldsymbol{n}_n \tag{4.67}$$

where $\boldsymbol{n}_n$ is the Kuhn-Tucker vector given by (4.18):

$$
\begin{aligned}
\boldsymbol{n}_n &= (\boldsymbol{K}_n\boldsymbol{K}_n^T)^{-1}\boldsymbol{q} \quad [\boldsymbol{n}_n] = r \times 1 \\
\boldsymbol{q} &= -\boldsymbol{K}_n\boldsymbol{g}_n \qquad\quad [\boldsymbol{q}] = r \times 1
\end{aligned} \tag{4.68}
$$

The matrix inverse in (4.68) need not be calculated explicitly. Only the solution to the linear system $(\boldsymbol{K}_n\boldsymbol{K}_n^T)\boldsymbol{n}_n = \boldsymbol{q}$ is required. It is shown in [57], pp. 47 that if $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is tridiagonal, $\boldsymbol{n}_n$ can be calculated from (4.68) in $8r - 7$ FLOPs where r is the number of active constraints. If a standard algorithm had been used to calculate the inverse of $\boldsymbol{K}_n\boldsymbol{K}_n^T$, $O(r^3)$ FLOPs would have been necessary.

Because the elements of the constraint matrix $\boldsymbol{K}_n$ are equal to 0, 1 or -1 and because each row and column of $\boldsymbol{K}_n$ have at most two elements different from zero, the calculation of $\boldsymbol{K}_n^T\boldsymbol{n}_n$ and $\boldsymbol{q}$ is simple and requires at most H_c additions or subtractions. In comparison with the number of FLOPs required when there are level constraints, $11H_c - 7$ additional FLOPs are necessary in the case of rate constraints to calculate $\boldsymbol{d}_n$ and $\boldsymbol{n}_n$ (this number is based on the worst case i.e. $r = H_c$).

The maximal step in the search direction

The maximal step in the search direction λ_{max} can be calculated using (4.14). In the case of rate constraints (4.14) becomes:

$$\lambda_{max} = \min\left\{\min_j\left(\frac{\overline{\Delta u} - \Delta u(k+j-1)}{\Delta d_j}\middle|\, \Delta d_j > 0\right),\right.$$

$$\left.\min_j\left(\frac{\underline{\Delta u} - \Delta u(k+j-1)}{\Delta d_j}\middle|\, \Delta d_j < 0\right)\right\} \quad (4.69)$$

where $\Delta d_j = d_j - d_{j-1}$ if $j > 1$ and $\Delta d_1 = d_1$.

Initialization of $\overline{u}$

Because $\overline{u}_0$ must be a feasible point, a fixed value for $\overline{u}_0$ cannot be used when there are rate constraints. However, the second and third method suggested in section 4.2.1 on page 246 to initialize $\overline{u}$ when there are level constraints can also be used in the case of rate constraints.

The third method can be used unmodified. The second method, however, requires some modifications. Given $\overline{u}$, the optimal solution to the unconstrained optimization problem, then a feasible starting point of the optimization algorithm can be found by clipping $\overline{u}$ by the constraints (4.21) and (4.22):

$$u^*(k+j-1) = \min\left(\max\left(u(k+j-1), u(k+j-2) + \underline{\Delta u}\right),\right.$$

$$\left. u(k+j-2) + \overline{\Delta u}\right)$$

This yields the vector u^*:

$$u^* = [u^*(k), \cdots, u^*(k+H_c-1)]^T$$

Now u^* can be used as a starting point.

Simulation have shown that using the optimal solution calculated in the previous sample results in a faster convergence than when the starting point is calculated by clipping $\overline{u}$. Subsequently, the starting point is calculated based on the optimal solution found in the previous sample.

Implementation and Numerical Complexity

In the case of rate constraints, the DGP algorithm 4.4 can still be used. However, the calculation of $\boldsymbol{d}_n$, $\boldsymbol{n}_n$ and λ_{max} must be carried out by using (4.67), (4.68) and (4.69), respectively.

The number of FLOPs that are required to perform one iteration of the DGP algorithm for rate constraints only is shown in Table 4.9. Obviously, in the case of rate constraints some extra calculations are necessary. This is mainly because of the fact that the search direction and the Kuhn-Tucker vector are harder to compute.

H_c	Σ	Π	/	FLOPs
H_c	$2H_c^2+9H_c-3$	$2H_c^2+5H_c-3$	$3H_c$	$4H_c^2+17H_c-6$
2	23	15	6	44
5	92	72	15	179

Table 4.9: Number of FLOPs required to perform one iteration of the DGP algorithm in the case of rate constraints.

4.2.6 Level and Rate Constraints

When level and rate constraints are present on the controller output, some problems arise in calculating the search direction $\boldsymbol{d}_n$ and the Kuhn-Tucker vector $\boldsymbol{n}_n$. When both types of constraints are present, (4.10) becomes:

$$A\overline{u} \le b: \left[\begin{array}{ccccc} 1 & 0 & \cdots & & 0 \\ 0 & 1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \\ & & \ddots & 1 & 0 \\ 0 & & \cdots & 0 & 1 \\ \hline -1 & 0 & \cdots & & 0 \\ 0 & -1 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \\ & & \ddots & -1 & 0 \\ 0 & & \cdots & 0 & -1 \\ \hline 1 & 0 & \cdots & & 0 \\ -1 & 1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & 1 & 0 \\ 0 & \cdots & 0 & -1 & 1 \\ \hline -1 & 0 & \cdots & & 0 \\ 1 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & -1 & 0 \\ 0 & \cdots & 0 & 1 & -1 \end{array}\right] \overline{u} \le \left[\begin{array}{c} \overline{u} \\ \vdots \\ \overline{u} \\ \hline -\underline{u} \\ \vdots \\ -\underline{u} \\ \hline \Delta u + u(k-1) \\ \overline{\Delta u} \\ \vdots \\ \overline{\Delta u} \\ \hline -\underline{\Delta u} - u(k-1) \\ -\underline{\Delta u} \\ \vdots \\ -\underline{\Delta u} \end{array}\right]$$

Now we have to optimize (4.4) taking into account $4H_c$ constraints. However, since $u(k+j-1)$ is limited by either $\overline{u}$ or $\underline{u}$, the constraints j and $j+H_c$ cannot become active at the same time. Further, because $\Delta u(k+j-1)$ is limited by either $\overline{\Delta u}$ or $\underline{\Delta u}$, the constraints $j+2H_c$ and $j+3H_c$ cannot become active at the same time. As a result, $2H_c$ constraints can become active simultaneously. For example, the constraints j and $j+2H_c$ can become active simultaneously. However, in an H_c dimensional optimization problem at most H_c independent constraints can be active. Moreover, in this case the search direction is equal to zero and hence need not be calculated explicitly. Obviously, in the case of level and rate constraints, constraints can become active that are a linear combination of other, already active, constraints. Therefore, problems arise in determining K_n. The following example illustrates this effect.

Example 4.5 Suppose that $H_c = 3$ and the following constraints are active:

$$u(k) = \underline{u} \tag{4.70}$$

$$u(k+1) = \overline{u} \tag{4.71}$$

$$\Delta u(k+1) = \overline{\Delta u} \tag{4.72}$$

Then the number of active constraints is equal to 3 and the constraint matrix $\boldsymbol{K}_n$ becomes:

$$\boldsymbol{K}_n = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 0 \end{bmatrix} \qquad [\boldsymbol{K}_n] = 3 \times 4$$

The matrix product $\boldsymbol{K}_n\boldsymbol{K}_n^T$, the inverse of which is required to calculate the projection matrix, is given by:

$$\boldsymbol{K}_n\boldsymbol{K}_n^T = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix} \qquad [\boldsymbol{K}_n\boldsymbol{K}_n^T] = 3 \times 3$$

The matrix $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is singular and the projection matrix cannot be calculated. Obviously, one of the active constraints is redundant. When one of the active constraints (4.70), (4.71) or (4.72) is redundant. Leaving out, for example, constraint (4.71) yields:

$$\boldsymbol{K}_n\boldsymbol{K}_n^T = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \qquad [\boldsymbol{K}_n\boldsymbol{K}_n^T] = 2 \times 2$$

This matrix can be inverted and can therefore be used to calculate the projection matrix $\boldsymbol{P}_n$ utilizing (4.13). Note that leaving out constraint (4.70) or (4.72) yields the same result for $\boldsymbol{K}_n\boldsymbol{K}_n^T$.

This example shows that when level and rate constraints are present, one must be careful when including active constraints in $\boldsymbol{K}_n$. Only the nonredundant active constraints must be included. The following section discusses an algorithm that can be used to determine which of the active constraints must be included in $\boldsymbol{K}_n$.

Calculation of the constraint matrix K_n

In this section, an algorithm that determines which of the active constraints can be included in the constraint matrix K_n is proposed. It is based on the fact that $u(k+j-1)$ must not be determined by more than one active constraint. All other active constraints that determine $u(k+j-1)$ are redundant. In Example 4.5, $u(k)$ is determined by active constraint (4.70) and $u(k+1)$ is determined by (4.71). However, if $u(k)$ and $u(k+1)$ are determined, active constraint (4.72) is redundant because $\Delta u(k+1)$ $(= u(k+1) - u(k))$ is already determined by (4.70) and (4.71).

The following algorithm can be used to determine whether or not an active constraint can be added to the current set of active constraints.

Algorithm 4.6 An algorithm that determines whether or not an active constraint is redundant. Here u_b is either $\overline{u}$ or $\underline{u}$ depending on which of the level constraints is active and Δu_b is either $\overline{\Delta u}$ or $\underline{\Delta u}$ depending on which of the rate constraints is active.

Step 1: Calculate those $u(k+j-1)$ that are determined by the current set of active constraints which is assumed to consist of nonredundant constraints only. The following algorithm can be used for this purpose.

```
For j = 1, ..., H_c Do
    If u(k+j-1) = u_b Then
        u(k+j-1) is determined
    End If
End For

For j = H_c, ..., 2 Do
    If u(k+j-1) is determined and Δu(k+j-1) = Δu_b Then
        u(k+j-2) is determined.
    End If
End For

For j = 1, ..., H_c - 1 Do
    If u(k+j-1) is determined and Δu(k+j) = Δu_b Then
        u(k+j) is determined.
    End If
End For
```

Step 2: Suppose a level constraint on $u(k+j-1)$ becomes active. Then this level constraint is redundant if $u(k+j-1)$ is determined by the current set of active constraints. If a rate constraint on $u(k+j-1)$ becomes active, this constraint is redundant if $u(k+j-1)$ and $u(k+j-2)$ are determined by the current set of active constraints.

Calculation of the search direction

The search direction and the Kuhn-Tucker vector can be calculated by using (4.67) and (4.68). In general the matrix $\boldsymbol{K}_n\boldsymbol{K}_n^T$ is a pentdiagonal matrix (= a matrix with nonzero elements only on the diagonal plus or minus two columns). As a result, calculating $\boldsymbol{n}_n$ using (4.68) requires more floating point operations than when rate constraints only are present.

Calculation of the maximum step size in the search direction

The maximal step in the search direction λ_{max} can be calculated using (4.14). When there are level and rate constraints this yields for λ_{max}:

$$\lambda_{max} = \min(\lambda_{max,level}, \lambda_{max,rate})$$

where $\lambda_{max,level}$ is the maximum step size for level constraints given by (4.41) and $\lambda_{max,rate}$ is the maximum step size for rate constraints given by (4.69).

Initialization of $\overline{u}$

Two methods can be used to initialize $\overline{u}$:

1. Use the optimal solution in the previous sample by evaluating (4.46). This way of initializing $\overline{u}$ always results in a feasible starting point if the constraints are time invariant. It is used in the simulations presented in the next section.
2. Clip $\overline{u}$ by the constraints (4.19), (4.20), (4.21) and (4.22) yielding $\boldsymbol{u}^*$:

$$\begin{aligned} u^*(k+j-1) \;=\; \max\Big(\min\Big(u(k+j-1), \overline{u}, \overline{\Delta u} + u(k+j-2)\Big), \\ \underline{u}, \underline{\Delta u} + u(k+j-2)\Big) \end{aligned}$$

with $j = 1, \ldots, H_c$. Now $\boldsymbol{u}^*$ can be used as the starting point of the optimization algorithm.

4.2.7 Simulation results

In section 4 it was shown that rate constraints especially have a great influence on the stability of the closed-loop system. The closed-loop system can even become unstable as is shown by Figure 4.2. In order to show what happens if the rate constraints are taken into account when minimizing the criterion function, consider the following example.

Example 4.6	The use of the DGP algorithm when there are rate constraints.
Settings:	$H_p = 4$, $H_s = 2$, $H_c = 3$, $\beta = 1$, $R = P = N = 1$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$, $\overline{u} = 100$, $\underline{u} = -100$, $\overline{\Delta u} = 10$, $\underline{\Delta u} = -10$, $\epsilon = 10^{-3}$, $I_s = 0.05$s
Process:	$H(s) = \dfrac{5}{(4s+1)(5s+1)}$
Model:	identical to process
Parameters:	None

Figure 4.8a shows the response when the rate constraints are not taken into account. The closed-loop system becomes unstable. When the DGP algorithm is used to calculate the optimal controller output sequence over the control horizon, the closed-loop system is stable as is shown by Figure 4.8b. Moreover, its performance is quite good.

The effect of the prediction horizon when there are rate constraints

As shown in section 4.2.4, the prediction horizon can be used to improve the performance of the closed-loop system when there are level constraints. Obviously, by predicting the process output over a larger horizon, the effect of the constraints can be taken into account. The following example shows that also in the case of rate constraints increasing the prediction horizon makes the performance of the system improve.

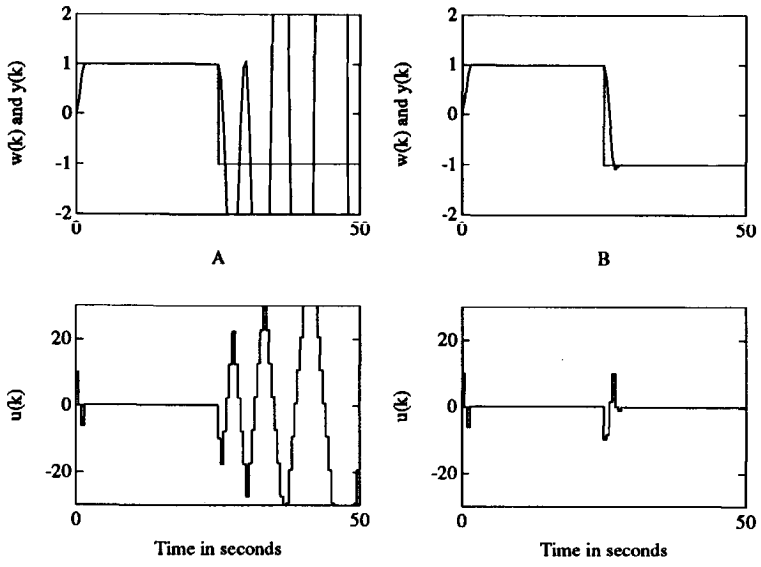

Figure 4.8: Responses when there are rate constraints without DGP (Figure A) and with DGP (Figure B).

Example 4.7 The influence of the prediction horizon on the performance of the closed-loop system in the case of rate constraints.

Settings: $H_s = 2$, $H_c = 3$, $\beta = 1$, $R = 0.25$, $P = N = 1 - q^{-1} + 0.25q^{-2}$, $T = 1$, $\hat{D} = 1$, $\underline{d} = 0$, $\overline{d} = 0$, $\rho = 0$, $Q_n = 1$, $Q_d = 1$, $\overline{u} = 100$, $\underline{u} = -100$, $\overline{\Delta u} = 0.1$, $\underline{\Delta u} = -0.1$, $\epsilon = 10^{-3}$, $I_s = 0.05$s

Process: $H(s) = \dfrac{5}{(4s+1)(5s+1)}$

Model: identical to process

Parameters: $H_p \geq 4$.

Note that when there are no constraints, choosing $H_p \geq 4$ yields a pole-placement controller. The closed-loop poles are determined by P. Here, the closed-loop system behaves as a second-order system with two time constants both equal to 0.72s. The response using UPC without the constraints being present is shown in Figure 4.9a. If rate constraints are imposed on the controller output, the response deteriorates. The resulting response is shown in Figure 4.9b. Clearly, the performance of the system is now unacceptable. When the DGP algorithm with $H_p = 4$ is used to control the process, the system's performance is much better as is shown by Figure

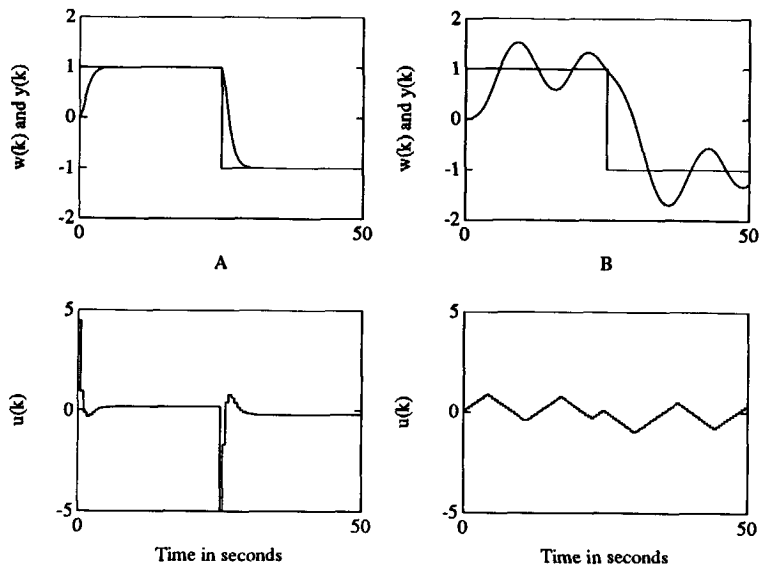

Figure 4.9: Responses using UPC without rate constraints (Figure A) and with rate constraints (Figure B).

4.10a. However, still the overshoot is rather large ($\approx$ 22 %). Table 4.10 shows the performance of the closed-loop system as a function of H_p. Clearly, increasing the prediction horizon makes the overshoot and settling time of the system decrease and therefore makes the performance of the system increase. However, for $H_p \geq 8$ the system gets slightly slower compared to when $H_p = 7$. Other simulations have shown that this effect is not caused by the finite accuracy of the solution.

Remark: if the DGP algorithm is not used, the prediction horizon does not affect the closed-loop system: For all $H_p \geq 4$ the same pole-placement controller yielding the same performance is obtained. Figure 4.10b shows the response when the DGP algorithm is used with $H_p = 8$.

Numerical complexity of the DGP algorithm when there are rate constraints

As has already been shown in Tables 4.5 and 4.9, the number of FLOPs required in each iteration of the DGP algorithm in the case of rate constraints is larger than if level constraints are present. However, what really matters is the number of FLOPs required to find an optimal solution of accuracy ϵ. This number is given by the

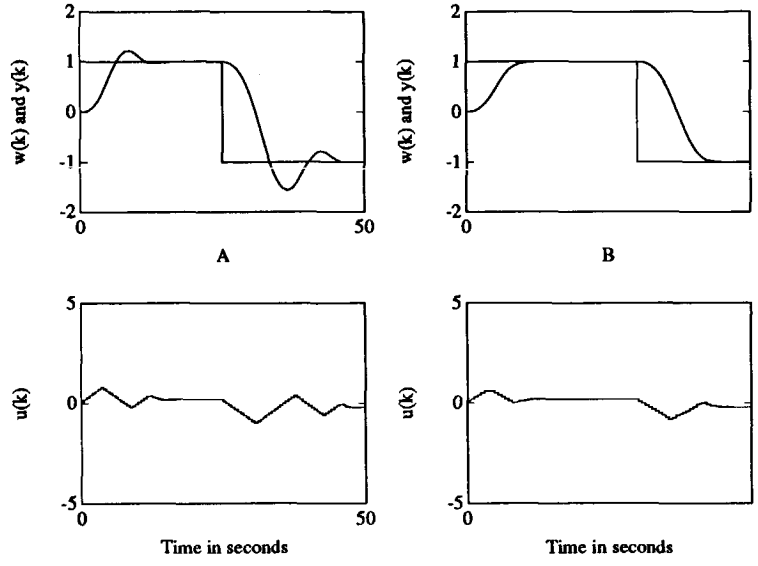

Figure 4.10: Responses using DGP where $H_p = 4$ (Figure A) and where $H_p = 8$ (Figure B) when there are rate constraints.

H_p	O in %	t_s in s
4	22.3	11.9
5	15.4	11.7
6	5.4	10.5
7	0.7	7.5
8	0	8.3
9	0	8.6
≥ 10	0	8.6
Without DGP	52.7	32.0

Table 4.10: Performance criteria as a function of H_p in the case of rate constraints and $\epsilon = 10^{-3}$.

number of iterations times the number of FLOPs required to perform one iteration.

In order to give an illustration of the number of iterations required by DGP in every sample, Example 4.7 was repeated with $H_p = 8$. The Figures 4.11a and 4.11b show the number of iterations that are required by DGP in every sample for accuracy $\epsilon = 10^{-3}$ and $\epsilon = 10^{-2}$, respectively. Obviously, the number of iterations

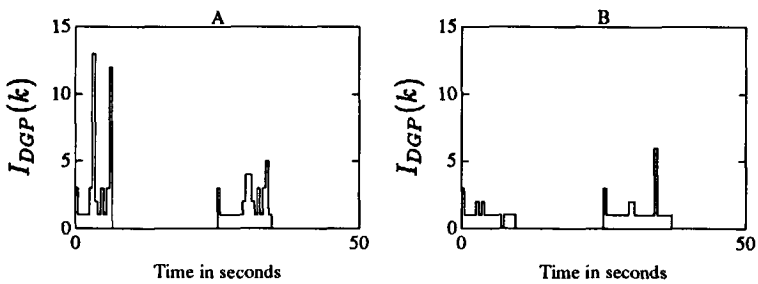

Figure 4.11: Number of iterations required by DGP ($I_{DGP}(k)$) to find the optimal solution with accuracy $\epsilon = 10^{-3}$ (Figure A) and $\epsilon = 10^{-2}$ (Figure B) when there are rate constraints.

required when $\epsilon = 10^{-2}$ is significantly smaller. Simulations have shown that the resulting response is hardly affected.

In order to get an impression of the calculation time and the iterations required in every sample, some measurements were performed for different values for ϵ and H_p. The results are shown in Table 4.11. Table 4.11 shows that the number of

H_p	ϵ	$\max_k I_{DGP}(k)$	$\max_k T_c(k)$ in ms
4	10^{-2}	3	13
4	10^{-3}	3	13
8	10^{-2}	6	19
8	10^{-3}	13	40

Table 4.11: Calculation times and iterations as function of ϵ and H_p in the case of rate constraints and $H_p = 8$.

iterations and the calculation time required are quite small. However, in comparison with the calculation time of the UPC controller that does not take into account the constraints, they are about 3 times larger when $\epsilon = 10^{-2}$. Note that although H_p does not influence the number of calculations required in each iteration of DGP, the

number of iterations (and hence the total calculation time) required to find an optimal solution with accuracy ϵ does depend on H_p.

4.3 Conclusions

In this chapter, predictive control when there are level and rate constraints on the controller output has been discussed. If the constraints are not taken into account, the performance of the system can be rather bad. Moreover, when there are rate constraints, the system can easily become unstable. Minimization of the criterion function over the prediction horizon, taking into account the constraints, has been shown to result in much better system performance, owing to the fact that the effect of the constraints on the predicted process output is taken into account. Therefore, increasing the prediction horizon increases the system's performance also in the case the prediction horizon does not affect the system's performance when there are no constraints.

For minimization of the criterion function, Rosen's gradient projection method has been used. It has been shown that due to the simplicity of the constraints and the criterion function, this method is simple and fast. The resulting algorithm has been called DGP. Further, it was shown that the starting point of DGP has a major effect on the number of iterations required to find the optimal solution with a particular accuracy. If the desired accuracy is sufficiently small, the starting point hardly influences the solution that is found. Calculating the starting point of DGP by using the ACH algorithm has been shown to result in a considerable decrease in the number of FLOPs required in every sample. For a typical application, calculation times smaller than 20 ms can be obtained on a Tulip SX compact 2 with numeric coprocessor.

Chapter 5

Applications

Many applications of predictive controllers have been presented over the last few years. For example, in [58] the application of GPC to a spray drying tower and to an industrial soap dryer has been described. Applications of (Q)DMC to chemical processes have been reported in, for example, [50] and [59]. Further, applications of predictive controllers to robots have been considered in, for example, [14]. These references represent only a small number of the many papers that have been written about the applications of predictive controllers. In the proceedings of the CIM Europe Workshop on Computer Integrated Design of Controlled Industrial Systems, which was held in Paris, April 26-27, 1990, many other papers (and references) to this subject can be found.

In this chapter, two applications of the UPC controller are discussed. The first ap-

plication concerns the control of the Mach number in a transonic wind tunnel. This application was carried out in cooperation with the National Aerospace Laboratory in Amsterdam, the Netherlands. The purpose of this application was to show that if accurate models of the process and disturbances are available, a high-performance control system can be obtained by using UPC. In this application, a great deal of effort has been put into obtaining accurate models of the process and the disturbances.

The second application shows UPC as a temperature controller for a distillation column. The purpose of this application was to show that even if a simple model of the process is used, approximately the same performance is obtained as is obtained when using the controller normally used to control the temperature of the column. This application was carried out in cooperation with the Royal Gist-brocades N.V. in Delft, the Netherlands.

5.1 Mach Control of a Transonic Wind Tunnel

The high-speed wind tunnel of the National Aerospace Laboratory in Amsterdam, the Netherlands, is one of the major transonic facilities in Europe and is used for aerodynamic research on scale models. Aerodynamic data of scale models are measured at a given Mach number, stagnation pressure, temperature and angle of attack. The Mach number, stagnation pressure and temperature specify the experimental conditions and must remain constant throughout a series of measurements, while the angle of attack is varied. However, changing the angle of attack influences the Mach number. Without some kind of control, the variation in the Mach number can be as high as 8 percent. However, it is required to keep the variations of the Mach number within one-tenth percent over the range of the angle of attack. Recently, a well-tuned proportional-integral-derivative (PID) controller based on extensive experiments was implemented [60]. The PID controller performs well for relatively slow changes in the angle of attack [60]. In order to improve the efficiency of the wind tunnel, the rate of change of the angle of attack must be increased which, however, results in larger deviations of the Mach number.

5.1.1 Description of the Wind Tunnel

The high-speed wind tunnel of the National Aerospace Laboratory is a closed-circuit continuously driven transonic tunnel. The wind tunnel is used for testing scale models, mostly of airplanes, in the speed region of zero to Mach 1.3. The main body of

the wind tunnel is a pressure shell (see Figure 5.1) which allows stagnation pressures between 12 kPa and 400 kPa. Evacuation is normally carried out to realize high

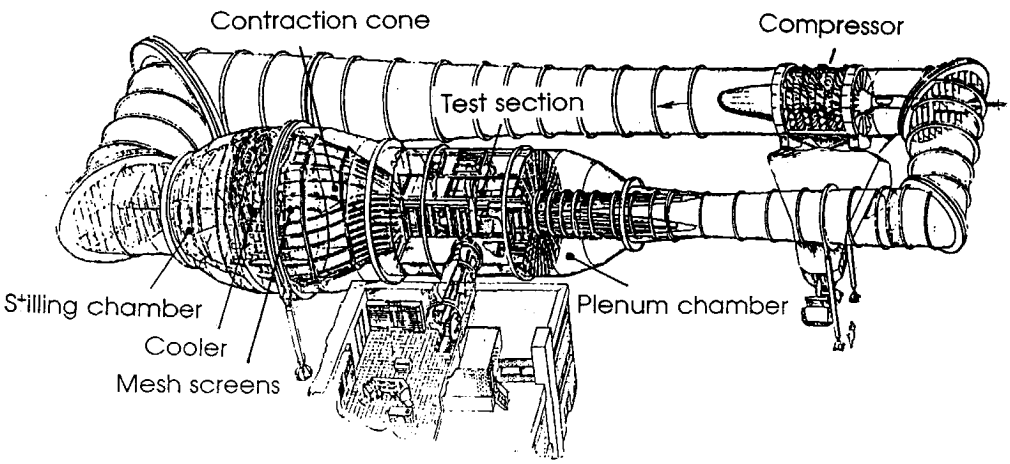

Figure 5.1: The high-speed wind tunnel of the National Aerospace Laboratory.

Mach numbers with the limited amount of power available while, by pressurizing, the Reynolds number can be increased in order to decrease the influence of model factors on the measurements. The wind tunnel is driven by four electric motors on one axis with a maximum power of 15 MW at 620 rpm. The power is conveyed to the air by an axial compressor which consists of three stages. The amount of power, which is related to the Mach number, can be controlled hydraulically by changing the pitch of the rotor blades of the compressor. A more detailed description of the wind tunnel can be found in [60] and [61].

5.1.2 The Control Problem

It is important to keep the Mach number constant at a predefined set point because most measured variables are a function of the Mach number, for given test conditions of stagnation pressure and temperature. The wind tunnel is driven by hand to the desired Mach number, stagnation pressure and temperature. During measurements the Mach number only is controlled. The Mach number is influenced by many factors but the pressure drop in the test section caused by varying the angle of attack is the most

important. Without control, the variation in Mach number can be as high as 8 percent. However, it is required to keep the variations of the Mach number within one-tenth percent in the transonic region ($0.5 \leq$ Mach ≤ 1.2) during a continuous variation of the angle of attack. Formerly, the Mach number was controlled manually by adjusting the pitch of the compressor blades when the actual Mach number deviated from the set point. This resulted in poor Mach number accuracy, bad reproduction and low efficiency because of the limited rate of change of the angle of attack. In the worst case, the angle of attack had to be adjusted discontinuously to allow retuning of the Mach number. In order to overcome these problems, a PID controller was designed [60] which has been in use now for over three years. Compared to manual control, a considerable increase in accuracy and higher efficiency was obtained. The major disadvantage of the PID controller is that only small rates of change of the angle of attack can be used without violating the accuracy requirement on the Mach number. For reasons of efficiency, it is desirable to increase the speed of the angle of attack. The speed can be adjusted to between 0.075°/s and 0.51°/s.

5.1.3 The Process Model

The static and dynamic behavior of the process to be controlled depends on the hydraulic blade angle system, the wind-tunnel process and the Mach measuring system. The blade angle system primarily consists of a cylinder, a 4/3 valve and two flow valves, for each direction one. The system can be modeled by an integrator, a time constant and a time delay. The parameters of this system were obtained by supplying a number of pulses to the input. Analysis of the resulting response has yielded the following relation between the input U and the pitch Φ (which can be measured):

$$\frac{\Phi(s)}{U(s)} = \frac{e^{-0.1s}}{s(0.3s+1)}$$

Further, the control signal U is limited to between plus or minus 0.5 Volts.

The wind-tunnel process can be divided into the closed-tunnel circuit and the compressor which causes a pressure difference so that air will flow around in the tunnel. The physical model is very complicated and of high order. For developing a controller, a simple low-order model is to be preferred. In order to obtain such a model, pulses were supplied to the blade angle system. Curve fitting methods were used to obtain the transfer function from the pitch of the rotor blades to the Mach number. It was shown in [60] that this complicated process can be represented reasonably well by a simple first-order model with time delay.

$$\frac{\text{Mach}(s)}{\Phi(s)} = \frac{K_{dc}e^{-0.3s}}{s\tau + 1} \tag{5.1}$$

The parameters K_{dc} and τ depend on the Mach number as is shown in Table 5.1. Finally, the dynamic behavior of the Mach number measuring system was found to be negligible in comparison with that of the blade angle system and the wind-tunnel process.

For more details on the modeling of the wind-tunnel process see [60].

Mach	K_{dc}	τ in s
0.5	0.035	0.35
0.6	0.0365	0.55
0.7	0.0375	0.70
0.8	0.0395	1.0

Table 5.1: Parameters of the wind-tunnel process as a function of the Mach number.

Disturbances

The Mach number in the test section is influenced by several disturbances. The main disturbance is caused by the pressure drop in the test section due to varying the angle of attack. The value and the sign of this disturbance are not known beforehand and depend on the scale model. For an "average model", the maximum sensitivity of the Mach number to the angle of attack is about 0.0045 Mach/°. Here, the nonlinear effect of the angle of attack on the Mach number is approximated by adding a triangular signal $\xi(t)$ to the Mach number. Figure 5.2 shows the signal added to the Mach number in the case where the angle of attack α is changed from -7.5° to 7.5° at a rate of change of 0.5°/s. At time $t = 50$s, the polar is activated, at $t = 65$s, the point with the smallest pressure drop is reached, and at $t = 80$s the model has reached its final position. Note that in this case the smallest pressure drop is obtained for $\alpha = 0°$.

Further, a considerable amount of noise is present on the Mach number. The spectrum and level of this disturbance are also dependent on the Mach number. Spectral analysis of the noise present on the output of the process with the input of the blade angle system equal to zero has shown that the noise can be considered as white noise filtered by the dynamics of the wind-tunnel process described by (5.1).

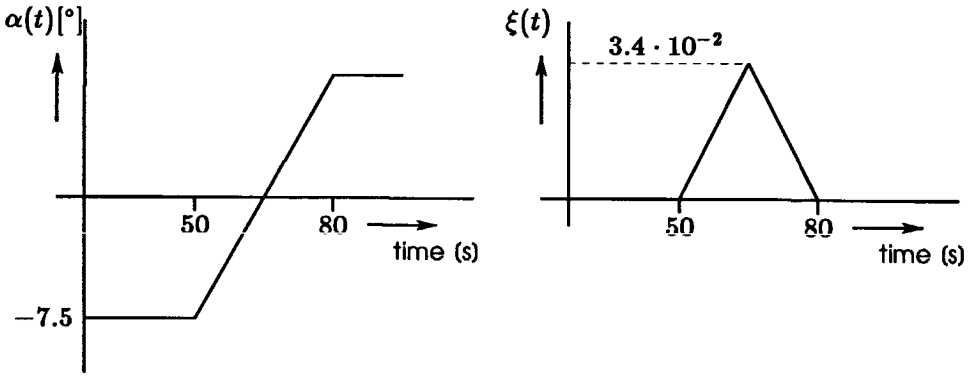

Figure 5.2: Triangular disturbance on the process output.

The complete process is visualized in Figure 5.3 where $\xi(t)$ is the triangular signal shown in Figure 5.2 and $e(t)$ is white noise.

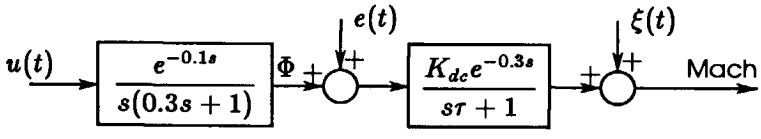

Figure 5.3: Process model of the high-speed wind tunnel.

Because predictive controllers are especially suited for controlling processes with time delay, and because models of the process and the disturbances are available, it can be expected that a predictive controller yields good performance. Next, the tuning of the UPC controller and some simulation results are shown.

5.1.4 Controller Design and Simulation Results

To apply the Unified Predictive Controller to the wind tunnel, the model described earlier is used. In this section the model for Mach 0.8 only is considered. The same design procedure can, however, be used for the models for other Mach numbers.

Applying the z-transform to the continuous model using a zero-order hold circuit and a sampling period of 0.1s yields:

$$\text{Mach} = \frac{1.973 \cdot 10^{-5} q^{-4}(1 + 3.356q^{-1})(1 + 0.240q^{-1})}{(1 - q^{-1})(1 - 0.717q^{-1})(1 - 0.905q^{-1})} u(k-1) + \frac{1}{1 - 0.905q^{-1}} e(k) + \xi(k)$$

This transfer function shows that the process has an unstable inverse. Therefore, the minimum-variance control strategy cannot be applied.

Because the model has a time delay of 4 samples: $\underline{d} = \overline{d} = \hat{d} = 4$. Further, the reference trajectory is constant, the disturbances on the output of the process are triangular and the process has an integrator. This yields for r, p and n: $r = 1$, $p = 2$ and $n = 1$.

According to the rule of thumb methods presented in section 3.9, the following settings are found for the parameters H_p, H_c, H_s and β:

- Choosing H_p according to the 5% settling time of the process' step response yields $H_p = 25$.
- $H_c = n_A = 3$. However, simulations have shown that choosing this value for H_c results in an unacceptably large controller output variance. Fine tuning of H_c has yielded $H_c = 2$.
- $H_s = \underline{d} + 1 = 5$
- $\beta = \max(\max(p, r) - n, 1) = 1$

Because ρ will be used as a tool to decrease the controller output variance, applying the rule of thumb methods with respect to ρ, Q_n and Q_d yield $Q_n = \Delta$ and $Q_d = 1 - 0.95q^{-1}$. For the time being, the weighting factor ρ is taken equal to zero. Further, because the controller will only be used in a regulator system, the design parameters P and R can both be taken equal to one: $P = R = 1$. Using the rule of thumb to select the $\hat{D}$ polynomial yields: $\hat{D} = \Delta$. Now all UPC design parameters except T have been selected. The T polynomial can be selected in different ways as is shown later in this section.

The controller obtained by using the above-mentioned settings is close to a dead-beat controller. Consequently, the controller output variance can be expected to be quite large and hence cannot be applied to the blade angle system of the

wind tunnel. However, in simulation, this set of parameters can be used to get an impression of the disturbance rejection capabilities of the controller. The disturbance rejection is judged by using several criteria. When the triangular disturbance $\xi(k)$ is absent and only $e(k)$ is present, approximations of the variances of the error signal and the controller output are used as criteria. These criteria are defined as:

$$\sigma_y^2 = \frac{1}{1000}\sum_{k=1}^{1000}(y(k)-w)^2 \qquad \sigma_u^2 = \frac{1}{1000}\sum_{k=1}^{1000}u(k)^2$$

in which w is the Mach set point and $y(k)$ is the Mach number.
Note: subsequently, the controller output variance is normalized to the controller output variance of the PID controller normally used to control the Mach number.

Further, the noise reduction ratio N is used as a measure of the regulator performance. This ratio is defined as $N = \overline{\sigma}_y^2/\sigma_y^2$, in which $\overline{\sigma}_y^2$ is the variance of the error signal in open loop. Hence, a value greater than 1 implies a noise reduction while a value smaller than 1 implies that the controller makes the noise appear amplified on the process output.

In order to judge the regulator behavior with respect to the triangular disturbance, two other criteria are used: the maximum deviation from the set point (|Max|) and the settling time of the response (t_s) caused by the disturbance $\xi(k)$ (= the time it takes the controller to reject the disturbance $\xi(k)$).

Thus far, T and hence the noise model which is given by $T/\hat{D}\hat{A}$ (see (2.1) in which A, C and D and are replaced by their estimates) is not yet fully determined. As mentioned above, there are different ways of selecting T:

1. Select T 'optimal' for disturbances caused by $e(k)$. Then T must be chosen according to the estimated noise model.
2. Select T 'optimal' for disturbances caused by the triangular disturbance $\xi(k)$.

Ad.1 Selecting T according to the estimated noise model yields:

$$\frac{T}{\Delta\hat{A}} = \frac{1}{1-0.905q^{-1}} \Longrightarrow T = \Delta^2(1-0.717q^{-1})$$

However, by doing so, the term Δ^2 in the numerator of the noise model cancels the term Δ^2 in its denominator. As a result, the absence of steady-state errors is no longer guaranteed. In order to prevent cancellation of the term Δ^2, T can be chosen as:

$$T = (1 - 0.9q^{-1})^2(1 - 0.717q^{-1})$$

Simulations using this choice for T showed that, because of the limitation of the controller output and the time delay present in the process, noise reduction cannot be obtained: the noise reduction ratio was equal to 0.99. Further, the controller output variance was very large. This being a fact, it is better to model the noise as measurement noise, and then the objective is to obtain a noise reduction ratio of 1 in combination with a small controller output variance.

Tuning for measurement noise

Measurement noise can be considered as white noise acting on the output of the process. Therefore, measurement noise can be modeled by choosing T such that $T/\hat{D}\hat{A} = 1$. In this case, however, the double integrator in the denominator of the noise model is again canceled and the triangular disturbance $\xi(k)$ is not rejected. Therefore, T is taken equal to $(1 - 0.9q^{-1})^2(1 - 0.717q^{-1})(1 - 0.905q^{-1})$. The controller with the above-mentioned parameters is called UPC_{meas}. Simulation results obtained when using this set of controller parameters are shown in Table 5.2. In the simulations the disturbance $\xi(k)$ as shown in Figure 5.2 is used (hence, the rate of change of the angle of attack is 0.5°/s). The table shows that the controller

	t_s in s	|Max|	N	σ_u^2	gm	dm in s
UPC_{meas}	5.6	$3.8 \cdot 10^{-3}$	0.89	1.1	2.3	0.71
$\text{UPC}_{\text{trian}}$	2.2	$2.4 \cdot 10^{-3}$	0.45	219.0	1.5	0.10
UPC_{comp}	4.3	$3.2 \cdot 10^{-3}$	0.83	9.0	2.2	0.51
UPC_{exp}	10	$3.8 \cdot 10^{-3}$	0.70	1.2	2.3	0.45
PID	18	$4.2 \cdot 10^{-3}$	0.83	1.0	2.4	0.74

Table 5.2: Performance and robustness of the PID and the Unified Predictive Controller for different settings of its design parameters.

output variance is quite small. The noise reduction is smaller than 1 and hence the noise appears amplified (by a factor 1.1) on the Mach number. Further, the rejection of the triangular disturbance is quite slow. This is not very surprising because the controller was tuned to handle the disturbances caused by $e(k)$. Further, in Table 5.2 the following (stability) robustness criteria are shown: The gain margin gm and the delay margin dm. Let us now concentrate on the rejection of the triangular disturbance.

Ad.2 In order to select T as 'optimal' for rejecting the triangular disturbance, the noise model should be taken equal to $1/\Delta^2$ resulting in:

$$T = (1 - 0.905q^{-1})(1 - 0.717q^{-1})$$

Simulation results obtained when using this noise model are shown also in Table 5.2 (= $\text{UPC}_{\text{trian}}$). Obviously, the triangular disturbance is now rejected much better. The values for |Max| and t_s are the best we can get taking the limiter and the time delay into account. However, a major disadvantage of this choice for T is the poor noise reduction and the large controller output variance caused by the dead-beat character of the controller output. This large controller output variance is not desired because it will cause extensive wear in the (mechanical) blade angle system. Overhaul of this system is very expensive. Further, the robustness of the closed-loop system is much worse than that of the predictive controller tuned for measurement noise (= UPC_{meas}).

Thus far, two controllers have been designed: one optimal with respect to the noise caused by $e(k)$ and the other optimal with respect to the triangular disturbance. Obviously, a compromise must be made.

The most important objective is to reject the triangular disturbance as fast as possible. However, the noise reduction must not be much smaller than 1 and the controller output variance must not be too large. Three parameters can be used to decrease the controller output variance: ρ, H_c and T. It is quite difficult to use the weighting factor ρ because its value cannot be calculated a priori. The control horizon is much easier to select. It is an integer variable with, in this case, only one other possibility: $H_c = 1$ (making H_c larger will increase the controller output variance). Figure 5.4 shows a time response where $H_c = 1$ and T is still selected as optimal for the triangular disturbance. Note that at $t = 50$s, the triangular disturbance $\xi(k)$ is activated (see Figure 5.2). Table 5.2 also shows the criteria for this set of parameters (= UPC_{comp}). The noise reduction ratio is almost as good as that obtained when using the predictive controller tuned for measurement noise (= UPC_{meas}). The rejection of the triangular disturbance is better, although not as good as when $H_c = 2$ (= $\text{UPC}_{\text{trian}}$) is used. Hence, this controller shows a nice compromise between the two controllers derived earlier in this section.

Further, Table 5.2 shows that the closed-loop system remains stable if the true time delay of the process is between 0s and 0.91s. Hence, the closed-loop system is quite robust with respect to variations in the process time delay. Also, the gain margin has an acceptable value.

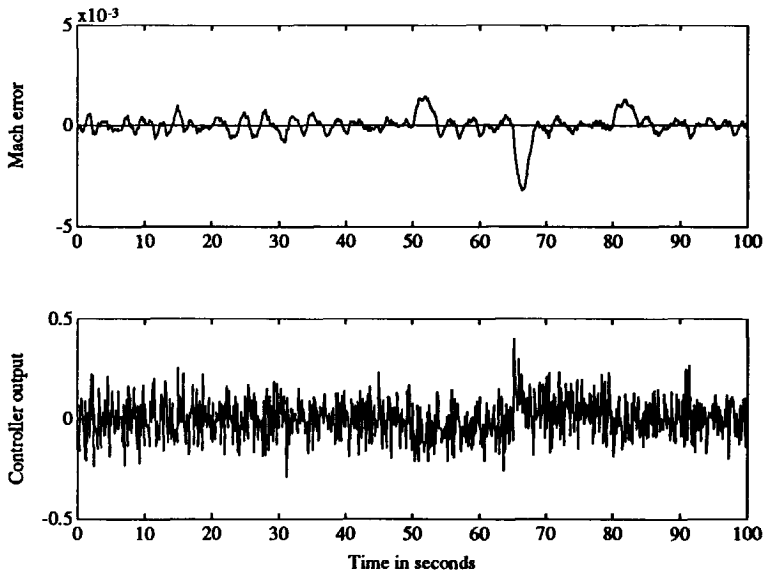

Figure 5.4: Simulated time response using the predictive controller UPC_{comp}.

Before applying the Unified Predictive Controller, discussed above, to the wind tunnel, a comparison is made with the PID controller currently used to control the Mach number. Figure 5.5 shows the time response using the PID controller. The figure clearly shows that the PID controller rejects disturbances caused by changing the angle of attack worse than the proposed predictive controller. Table 5.2 shows that the settling time t_s especially is quite long. This is caused by an oscillation in the Mach number during the triangular disturbance. The noise reduction does not differ from that obtained when using the predictive controller. However, the controller output variance is significantly smaller (compare the Figures 5.4 and 5.5). Further, the robustness criteria when using the PID controller are slightly better than those obtained when using the predictive controller. Obviously, the PID controller is much worse in respect to the rejection of the triangular disturbances. The performance of the PID controller in respect to the triangular disturbance can be improved by adjusting its parameters. However, in [60] it was shown that it is not possible to increase this performance without making the noise reduction ratio N smaller.

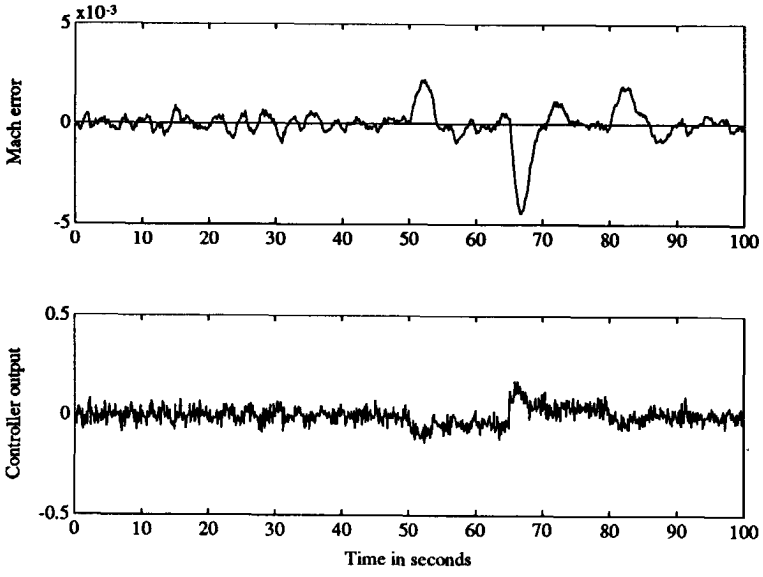

Figure 5.5: Simulated time response using the PID controller.

5.1.5 Experiments

Because of the limited memory capacity of the control computer of the National Aerospace Laboratory, the Unified Predictive Controller designed above could not directly be implemented. The major problem was the large prediction horizon which caused large matrices. Therefore, different parameter settings were used during the experiments. These settings were: $H_p = 14$, $H_c = 2$, $\rho = 3 \cdot 10^{-4}$ and $T = (1-0.7q^{-1})^3$. All other settings were not changed. Subsequently, the UPC controller used in the experiments will be called UPC_{exp}. In simulation, these controller settings result in a worse performance than that of the predictive controller derived above. The performance criteria are shown in Table 5.2. The robustness criteria, however, do not differ much from those of the predictive controller derived in the simulations ($= \text{UPC}_{\text{comp}}$). During the experiments, the model, on which the Unified Predictive Controller is based, is adapted as a function of the Mach number according to Table 5.1. By doing this, the controller is 'optimal' for different Mach numbers. The PID controller, designed for Mach 0.8 only, is not adapted. Hence, for Mach numbers other than 0.8, better results with this controller can be expected if it is redesigned for these Mach numbers too. However, it is more time consuming to redesign the PID controller for other process models than it is to redesign the Unified Predictive

Controller. In the latter case, only the T polynomial needs to be adjusted.

During two days of measurements at the wind tunnel of the National Aerospace Laboratory in Amsterdam, the PID and the predictive controller were examined at different Mach numbers. On the first day, a number of experiments were carried out at Mach=0.7, 0.8 and 0.9. On the second day, all experiments were repeated in order to examine the reproducibility of the results. Also, throughout the second day, there was another scale model present in the wind tunnel. This model caused much more disturbances on the Mach number. In all experiments, the rate of change of the angle of attack was 0.51°/s. To compare the PID controller with the predictive controller, the following four criteria were used:

1) **Noise reduction.** This criterion measures the noise reduction of the predictive controller in comparison with that of the PID controller. It is defined as the ratio between the variance of the Mach number obtained when using the PID controller and that obtained when using UPC.

2) **Low frequency noise reduction.** This criterion measures the noise reduction for low frequencies only. It is defined in a way similar to that of criterion 1.

3) **High frequency noise reduction.** This criterion measures the noise reduction for the higher frequencies. Also this criterion is defined in a way similar to that of criterion 1.

4) **Output variance.** The controller output variance of UPC normalized to that of the PID controller.

The objective of using different criteria is that the second criterion is a measure of how well the triangular disturbances are rejected, and the third criterion is a measure of how well the predictive controller rejects the noise in comparison with the PID controller. The second and the third criterion are obtained by filtering the data by a moving-average filter taking 100 samples in the past and 100 samples in the future into account. By using such a filter, phase shifts between the filtered and the original signal do not occur. The filtered signal is used to calculate the second criterion while the differences between the measured and the filtered data are used to calculate the third criterion. The original signal is used to calculate the first criterion. Note that values greater than one for the first three criteria imply that the predictive controller performs better than the PID controller. In Table 5.3, the above-mentioned criteria are shown for the measurements obtained on the first and on the second day. Although there was another model present in the wind tunnel throughout the second

day of measurements, the criteria are approximately the same as those obtained the first day. The table shows clearly what was expected from the simulations: The

Day	Mach	1) Noise	2) Low Freq.	3) High Freq.	4) Output variance
#1	0.7	1.31	4.86	1.07	0.59
	0.8	1.56	4.34	1.18	0.74
	0.9	1.39	4.56	1.13	0.75
#2	0.7[a]	-	-	-	-
	0.8	1.50	4.20	1.13	0.77
	0.9	1.36	4.11	1.23	0.67

Table 5.3: Performance of the PID and the predictive controller throughout two measurement days.

[a]Unfortunately, measurements taken at Mach=0.7 during the second day of measurements are not available.

predictive controller rejects the triangular disturbance more than 4 times better than the PID controller. Further, Table 5.3 shows that when using the predictive controller, no significant improvement in reducing the higher frequencies is obtained. However, the overall performance (measured by criterion 1) is 30-60% better when using the predictive controller. Further, it can be expected that by increasing the rate of change of the angle of attack, the overall performance when using the predictive controller compared to that when using the PID controller increases too, because then the lower frequencies will dominate over the higher frequencies. To achieve these results, the predictive controller required a larger control output variance. Because the results obtained during the experiments confirm those obtained in the simulations, the model of the process and disturbances are believed to be quite accurate.

Figure 5.6 shows the response of the PID controller at Mach = 0.8 while Figure 5.7 shows the response of the Unified Predictive Controller at the same Mach number. The dashed line in the top windows of both figures shows the error between set point and Mach number filtered by the moving-average filter. In both experiments the angle of attack was changed over 15° at a rate of change of 0.51°/s. The polar was activated at $t = 5$s and stopped at $t = 35$s. Then, the rate of change was reversed and the angle of attack was changed until it returned at its original value (at $t = 65$s).

Figures 5.6 and 5.7 clearly show the better behavior of the predictive controller in rejecting the disturbance caused by changing the angle of attack. Also, for other Mach numbers the rejection of the disturbance caused by changing the angle

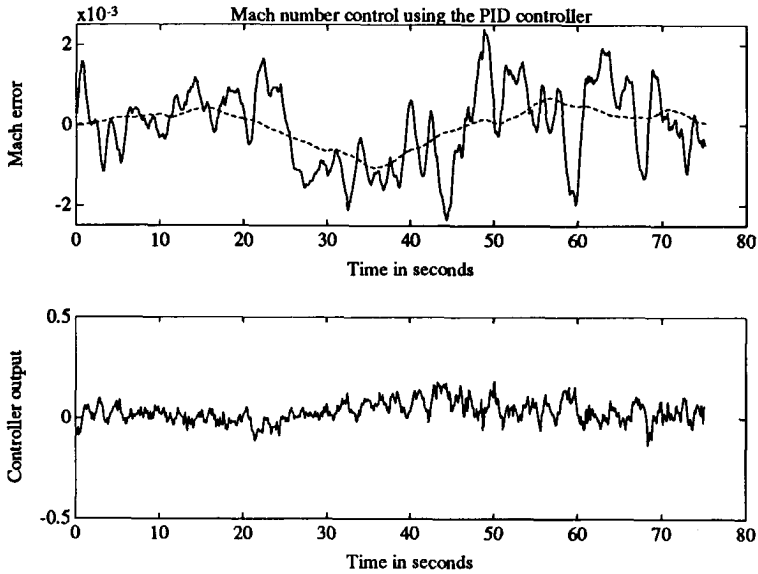

Figure 5.6: Measured response of the PID controller at Mach = 0.8.

of attack is much better when the predictive controller is used, as is shown in Table 5.3. Note that the deviations of the Mach number exceed the required ± 0.001. This is caused by the high rate of change of the angle of attack. The predictive controller derived by means of the simulations ($= \text{UPC}_{\text{comp}}$) showed (in simulation) even better results than the predictive controller used in the experiments. Therefore, we can expect that implementation of this controller will yield better results too.

5.1.6 Conclusions

The use of the Unified Predictive Controller in controlling the Mach number of a transonic wind tunnel has been examined. It has been shown, by means of simulations and experiments, that the Unified Predictive Controller yields an overall performance improvement of 30-60% in comparison with the PID controller that is normally used to control the Mach number. The rejection by the Unified Predictive Controller of disturbances caused by changing the angle of attack is a factor 4 better than that of the PID controller. This makes it possible to change the angle of attack faster so that the efficiency of the wind tunnel operation can be improved. It is our belief that this is mainly because knowledge about the disturbances can be incorporated into

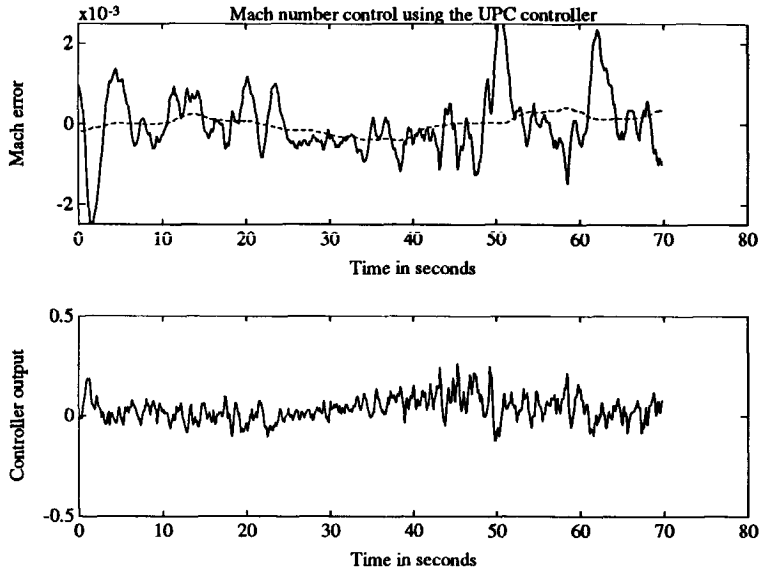

Figure 5.7: Measured response of the Unified Predictive Controller at Mach = 0.8.

the predictive controller design.

Simulations have shown that even better results can be expected. These results have not yet been verified by experiments. However, we hope to provide verification in the near future.

5.2 Temperature Control of a Distillation Column

In this section, the application of the UPC controller as a temperature controller of a distillation column is discussed. It is shown that the controller can easily be tuned by using the rule of thumb methods described in section 3.9. The model on which the predictions are based is a first-order model with time delay. Although this model cannot correctly describe the complex nonlinear process, experiments have shown that it can successfully be used in combination with the UPC controller. The design parameters of the UPC controller are selected such that the resulting closed-loop system is robust with respect to model mismatch.

5.2.1 Process Description

The purpose of the distillation column under investigation is to separate water containing organic components (butyl-acetate (BA) and butanol (BuOH)) into an overhead product containing water-BA-BuOH and a bottom product containing water with traces of the organic components. Figure 5.8 shows a schematic diagram of the distillation column and Figure 5.9 shows a photo of the column. The column has 20 sieve trays and is heated by direct steam (4 bar). The vapour is condensed against the feed (C1) and cooled by chilled water (H2; T $\approx$ 30°C). In the separator S1, the product is separated into water and BA. Together with an almost equal amount of waste BA at a temperature of 150°C fed in from another process unit, the water is returned to the column as reflux. The total reflux is approximately 1.6 m^3/h. The feed to the column is heated by the vapour and by two heat exchangers in the bottom stream (H1A/B). Both the feed and the reflux enter the column at tray 5. The temperature of the column is controlled by the steam input. During a run, the dynamics

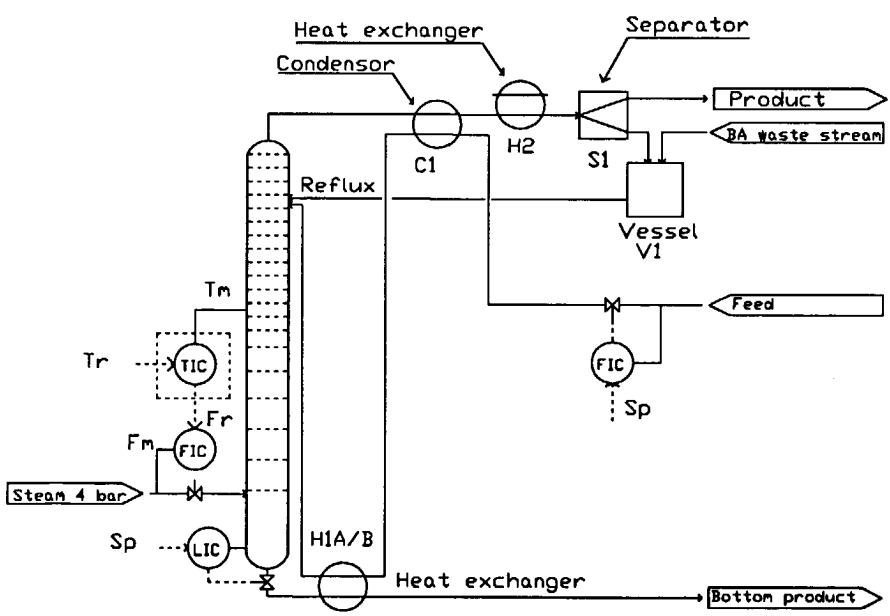

Figure 5.8: Schematic diagram of the distillation column.

of the column change, mainly as a result of fouling. This makes it necessary to clean

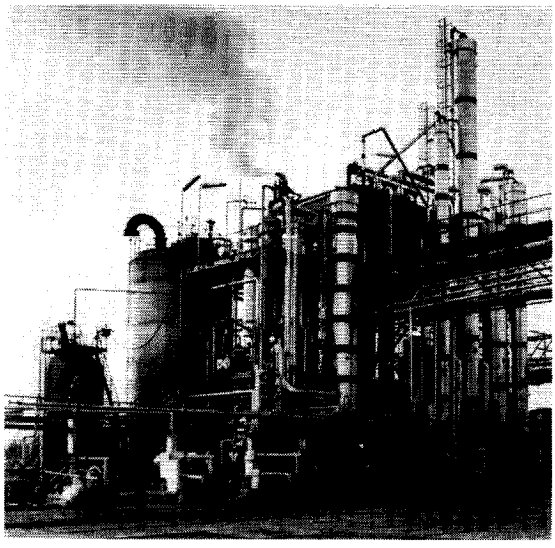

Figure 5.9: The distillation column at the Royal Gist-brocades N.V., Delft, the Netherlands (Photo: Royal Gist-brocades N.V.).

the column every two or three days.

Traditional control scheme

The feed to the column is adjusted by the operator and the adjustment depends on the level in the storage tank. The bottom level in the column is controlled by a P controller (LIC) and the temperature at tray 13 is controlled by a PI controller TIC which generates the set point for the steam flow controller FIC. All controllers are pneumatic.

Process disturbances

In order to test the UPC controller, the following disturbances were applied to the process:

1. step-wise changes in the feed flow. These kinds of disturbance often occur during the normal operation of the column because the operator changes the

feed of the column manually, depending on the level of the storage tank. During the experiments the feed flow was varied between 18 and 22 m^3/h.

2. pulse disturbances via the (hot) BA-waste stream. The BA-waste stream consists of about 99 % BA at a temperature of about 150°C. For 30 seconds, the flow was increased from approximately 0.8 m^3/h to 2.1 m^3/h which resulted in an addition of 22l BA to the column. Because the vessel V1 has a volume of about 0.1 m^3, the reflux temperature and the BA concentration in the reflux quickly rise (see Figures 5.12 and 5.13). Despite the higher reflux temperature, the overall result was a drop in the temperature T_m caused by a higher concentration of BA in the reflux.

Process dynamics

The dynamics of the distillation column and the adjoining equipment is rather complex. By using the traditional mass and heat balances, a high-order dynamic model can be derived. Complications arise as a result of the fact that on the (top)trays there is already a separation into water and BA which is difficult to describe. The resulting model would be highly nonlinear.

Another approach is to use a simple model of the process which describes its most important properties in combination with a robust controller. A first-order model with time delay is used to describe the most important dynamics of the transfer function from the steam flow set point F_r to the temperature T_m.

$$T_m = \frac{K_{dc}e^{-T_d s}}{s\tau + 1}F_r + \delta \tag{5.2}$$

in which δ is an offset term, K_{dc} is the DC gain, T_d is the time delay and τ is the time constant of the process. Note that transfer function (5.2) also includes the dynamics of the flow control loop. However, experiments showed that this loop is quite fast and can be described by a unit gain. Of course, model (5.2) cannot correctly describe the dynamics of the distillation column. For instance, second-order effects introduced by the heat exchangers and nonlinearities are neglected. Therefore, it is required that the UPC controller, which is based on this model, be robust with respect to unmodeled dynamics.

The parameters of (5.2) were determined by making several step responses of the open-loop process. Steam flow changes ≤ 0.2 ton/h were used for this purpose. The flow changes were taken this small in order to keep the process in one operating

point. Analyzing the resulting responses yielded the following values for K_{dc}, T_d, τ and δ:

$$\begin{aligned} K_{dc} &= 5 \\ T_d &= 30s \\ \tau &= 40s \\ \delta &= 85°C \end{aligned}$$

When a sampling period of 5s is used, the discrete model becomes:

$$H(z^{-1}) = \frac{0.588z^{-7}}{1 - 0.883z^{-1}}$$

The disturbances mentioned in the previous section can be simulated by adding a disturbance model to the output of the process:

$$T_m(k) = \frac{0.588q^{-6}}{1 - 0.883q^{-1}} F_r(k-1) + \delta + \frac{1}{1 - 0.883q^{-1}} l(k)$$

Simulations and experiments have shown that step-wise changes in the feed-flow between 18 and 22 m^3/h can be simulated by using a step of magnitude 0.2 for $l(k)$. In the case of pulse disturbances via the BA-waste stream, simulations have shown that $l(k)$ is a pulse of magnitude -1.

5.2.2 Controller Design

As discussed in the previous section, the design parameters of the UPC controller must be selected such that the resulting control system is robust. Applying the rule of thumb methods presented in section 3.9 yields:

H_p: Choosing H_p according to the 5% settling time of the step response of the process yields $H_p = 25$.

H_e: Because the model has a time delay of 6 samples, $\underline{d} = \overline{d} = \hat{d} = 6$ and hence $H_e = \underline{d} + 1 = 7$.

H_c: To obtain a robust controller, a mean-level controller is selected: $H_c = 1$.

β: The process does not have an integrator: $n = 0$. Further, the reference trajectory and disturbances are constant: $r = p = 1$. This yields for β: $\beta = \max(\max(p,r) - n, 1) = 1$.

During the simulations and experiments, controller output weighting by means of ρ was not used. Hence, $\rho = 0$ and $Q_n = Q_d = 1$. Further, because the control system is a regulator system, P and R can be taken equal to 1: $P = R = 1$. A rule of thumb on how to choose $\hat{D}$ is to take $\hat{D}$ equal to $P\Delta^\beta$. With the above-mentioned parameters this yields for $\hat{D}$: $\hat{D} = \Delta$.

The T polynomial must be selected such that the robustness and regulator behavior is acceptable. Because the control system must be very robust, T has been chosen equal to $T = (1-0.9^{-1})^2$ (which is close to using $T = \hat{A}(1-0.9q^{-1})$ which guarantees a robust control system). Then the gain and delay margins are equal to 4.6 and 137s, respectively.

Figure 5.10 shows a time response using the model and the parameters of the controller selected as suggested above. At $t = 0.3$h, a pulse disturbance occurred. At $t = 0.7$h and $t = 1$h changes in the feed flow occurred. Clearly, the integral action makes sure the disturbance is rejected. Other simulations have shown that the pulse disturbance cannot be rejected better (faster and with a smaller maximum deviation from the set point) because of the time delay and the desire to have a robust control system.

5.2.3 Experiments

Set-up of the experiments

In this section, real-life experiments with the UPC controller applied to the distillation column as described in section 5.2.1 are discussed. The control scheme is as shown in Figure 5.11. The figure shows that the UPC controller is used in cascade with the (pneumatic) flow controller FIC. The pneumatic switch SW is used to select either the PI controller, normally used to control the temperature of the column, or the UPC controller. This switch has been installed for safety reasons and can be used to restore the original control scheme as shown in Figure 5.8. The UPC controller was implemented such that switching between the PI and the UPC controller does not result in changes in the flow set point F_r. The UPC controller was implemented

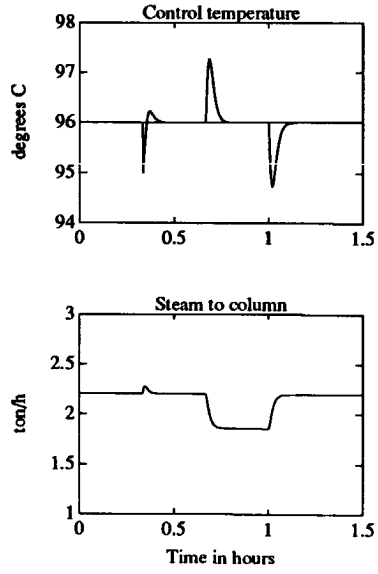

Figure 5.10: Simulated response using the UPC controller.

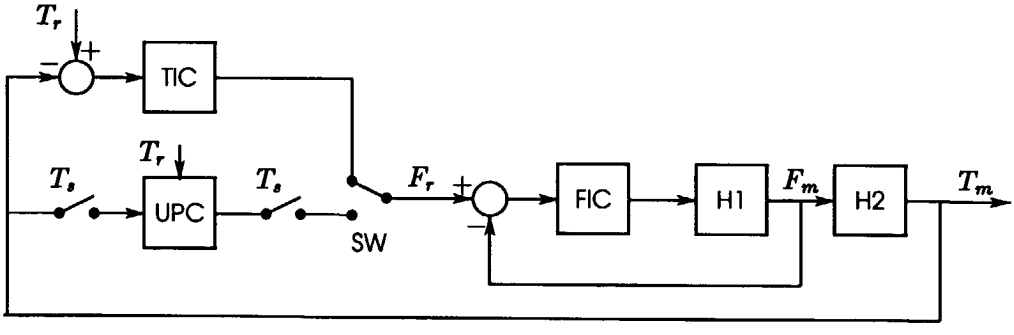

Figure 5.11: Control scheme using the PI and UPC controllers.

in Fortran on a μPDP-11/73. The MUSIC simulation and control package [62] was used to realize the sampling, i/o and data acquisition.

During the experiments the set point was kept at 96°C.

Experimental results

During the experiments, it was found that the variance of the disturbances on the process output changes rapidly with time. Because the process is a part of a continuous production process, changes in, for example, the BA concentration of the feed and reflux, caused by changes in another part of the production process, change the dynamics of the column and the variance of the disturbances. Taking the above into consideration, comparisons with the conventional PI controller are hard to make. Nevertheless, the response of the PI controller is shown in Figure 5.12 as a reference. Step changes in the feed flow occurred at $t = 0.85$h and $t = 1.2$h while an impulse disturbance in the BA concentration occurred at $t = 0.1$h. In Figure 5.13 the

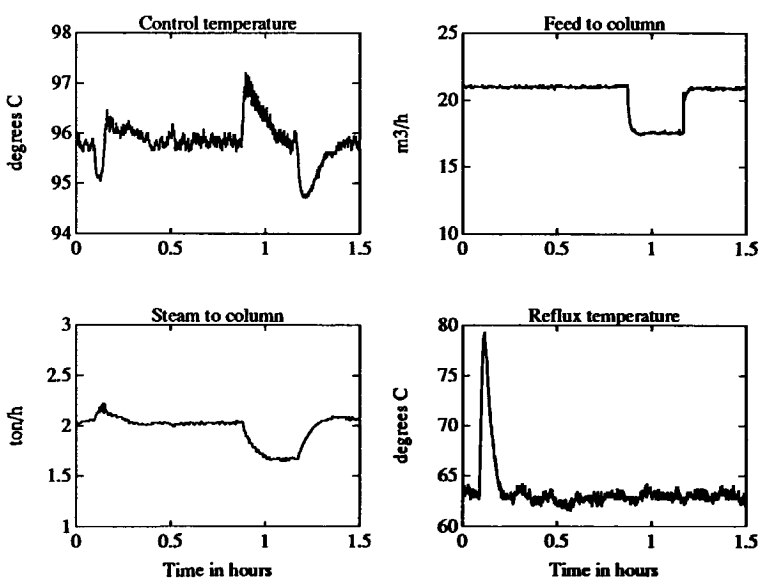

Figure 5.12: Responses using the conventional PI controller.

response of the UPC controller is shown. The figure clearly shows that temperature disturbances caused by changes in the feed flow (at $t = 0.5$h and $t = 1.1$h) are

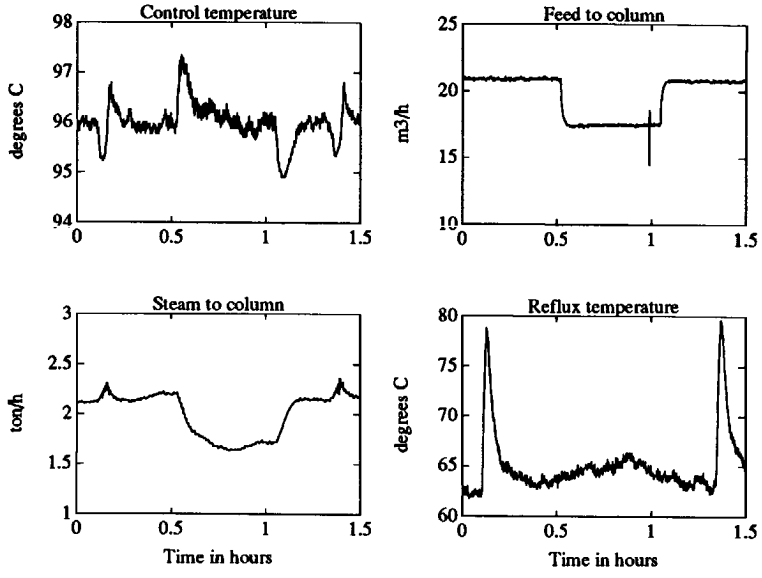

Figure 5.13: Responses using the UPC controller.

rejected because of the integral action. Further, Figure 5.13 shows that impulse disturbances (occurred at $t = 0.1$h and $t = 1.3$h) in the BA concentration of the reflux are nicely rejected. Note that these impulse disturbances also yield large changes in the reflux temperature. Despite the fact that the UPC controller is based on a simple model of the plant, its performance is comparable to that of the PI controller.

5.2.4 Conclusions

An application of the UPC controller as a temperature controller of a distillation column has been discussed. It has been shown that good performance and robustness could be obtained in spite of the fact that the UPC controller was based on a simple first-order model of the plant. Further, it has been shown that if a model of the plant and some design requirements are known, the controller parameters can easily be selected by using the rule of thumb methods described in section 3.9.

5.3 Conclusions

In this chapter, two applications of UPC have been discussed. It has been shown that if a model of the process is available, the UPC design parameters (and hence the controller itself) can be made readily available by using the rule of thumb methods. In the case of the Mach controller, some fine tuning was necessary in order to meet the requirements. Owing to the fact that prior knowledge about the model and disturbances could be incorporated into the design of the UPC controller, better results could be obtained than those obtained when using the traditional controller. Especially the fact that the time delay, and the knowledge about the disturbances, can be incorporated into the controller design is believed to play an important role in the increase in the performance of the system.

The second application has shown that even if the controller is based on a simple model of the process, good results can be obtained. With little effort, an acceptable control system has been obtained.

Chapter 6

Conclusions and Suggestions

In this section, some conclusions are drawn and some suggestions for further research are given. Because conclusions have been given at the end of each chapter, here, only some general conclusions are drawn.

In this thesis a unified approach to predictive control design has been proposed. The approach has resulted in the Unified Predictive Controller (UPC). A profound study with respect to its properties and its design parameters has yielded many theoretical results. Based on these results and an extensive simulation study, the following conclusions can be drawn.

- The concept of predictive control, as discussed in Chapter 1, can be used to control many different kinds of 'difficult' processes including non-minimum phase and unstable processes and processes with time delay. However, it

has been shown that in order to control non-minimum phase processes and processes with time delay it is necessary to limit the number of degrees of freedom of the controller output sequence. The control horizon was introduced for this purpose.

- Because the process model that is used in UPC is quite a general input-output model, UPC can be used not only if an ARMAX, ARIX, ARIMAX or ARX model of the process is available but also if an impulse or step response model is available.

- It has been shown that, although predictive controllers use the receding horizon strategy, if the process is linear, the criterion function is quadratic and if there are no constraints, then the resulting controller is linear and the controller parameters are time invariant. In this case the control law can be written as:

$$u(k) = \boldsymbol{v}^T \boldsymbol{w} - \boldsymbol{l}^T \boldsymbol{s} \tag{6.1}$$

where $\boldsymbol{v}$ and $\boldsymbol{l}$ represent the controller parameters, $\boldsymbol{w}$ is the reference trajectory and $\boldsymbol{s}$ is a vector that contains past information of $y(k)$ and $u(k)$:

$$\boldsymbol{s} = [y(k), y(k-1), \cdots, u(k-1), u(k-2), \cdots]^T$$

The predictive control law can therefore easily be implemented in a computer. Moreover, the complexity of this control law is of the same order as that of a pole-placement controller or of a discrete PID controller. However, the calculation of the parameters of the predictive controller is rather complicated and can therefore be quite time consuming. In an adaptive context, it may be necessary to calculate the controller parameters at every sampling instant. The calculation time required to do so, however, may not be available. However, when the process does not change, or a fixed controller yields good results in spite of process changes, one can calculate the parameters of the predictive control law off line and calculate the controller output in every sampling instant by using (6.1).

- It has been shown that owing to the unified approach, many well-known predictive controllers can be regarded as a subset of the Unified Predictive Controller. Moreover, also pole-placement design methods, minimum-variance control and time-optimal control are a subset of UPC. The above-mentioned controllers can be obtained simply by selecting the UPC design parameters. Thus, some remarks concerning these controllers, can be made based on the results obtained with respect to UPC:

- All predictive controllers can handle non-minimum phase processes and processes with time delay. However, only GPC and UPC are suited to control unstable processes as well.
- The tuning of GPC and UPC in order to obtain a desired behavior of the closed-loop system is easier than that of the other predictive controllers discussed in this thesis.
- Except for UPC, none of the predictive controllers that are discussed in this thesis can deal with non-constant reference trajectories and/or disturbances. If there are such disturbances, steady-state errors occur.
- Most predictive controllers discussed in this thesis use weighting of the controller output increments. However, it has been shown that this can yield a badly damped or even an unstable closed-loop system for a certain range of the weighting factor. In this respect, it has been shown to be better to weight the controller outputs directly if the purpose of controller output weighting is to make the controller output less active.

  In UPC, filtered controller outputs are weighted. The filter that is used for this purpose is quite general. Because only the filtered controller outputs are weighted, certain frequencies in the controller output can be attenuated by choosing a filter that passes only those frequencies that must be attenuated. Further, it has been shown that this filter can be used successfully for obtaining a compromise between directly weighting the controller outputs and the conditions which must be satisfied for zero steady-state error.

- Because of the unified approach, UPC has more design parameters than any other predictive controller. This is mainly because of the fact that UPC has some additional features. For example, in contrast to most other predictive controllers, UPC allows the tracking of non-constant reference trajectories and the rejection of non-constant disturbances. At first glance, one could say that this makes the tuning of the design parameters more complicated. However, it has been shown that most design parameters can be tuned (and fixed) by rule of thumb methods which have been derived from theoretical arguments and experimental results. The only parameters for the designer to select are P and T. The response to set point changes can be tuned by P while the regulator behavior and robustness can be tuned by means of T. In the ideal situation where the input-output transfer function of the process is identical to that of the model (hence, $A = \hat{A}$, $B = \hat{B}$ and $d = \hat{d}$), T does not influence the servo behavior of the system. In general, this will hardly ever be the case. Then, the effect of T on the system's servo behavior depends on the amount of model

mismatch. In contrast, P never affects the regulator behavior and robustness of the system.

Usually, the UPC controller tuned by means of these rules of thumb yields good results for a wide variety of processes. However, in some circumstances, fine tuning may be required. Therefore, fine tuning methods have been discussed with respect to, among other things, the controller output variance, the robustness of the closed-loop system and the rejection of load changes. The rules obtained for initialization and fine tuning of the controller can be implemented in an expert system [48], facilitating a novice designer and operator to install and use the UPC controller quite easily.

- In practical applications, the robustness of the closed-loop system is of paramount importance. It has been shown that especially the noise model (which mainly consists of T and $\hat{D}$) has a major effect on the system's robustness. By a special way of writing the i-step-ahead predictors that are used to predict the process output over the prediction horizon, it has been shown that the filtered difference between process output and model output is added as a correction term to the predictions. The filter is determined by the noise model. By changing the noise model (thus changing the filter) the effect of model mismatch on the predictions can be influenced. This obviously affects the robustness of the control system.

  This special way of writing the i-step-ahead predictors has also been used to provide a theoretical background to the heuristically added correction term in the predictive controllers DMC, PCA and MAC. It has been shown that this correction term corresponds to a particular noise model which models Brownian motion and prevents steady-state errors caused by constant disturbances and/or reference trajectories. Based on the knowledge gained in this thesis, the heuristically added correction term can be regarded as being a good one in most practical situations.

- That heuristics do not always yield good results was shown in Chapter 3. In most predictive controllers, the reference trajectory is a first-order trajectory initialized at $y(k)$ which seems a good choice. By doing this, an obscure feed-back loop is introduced the effect of which on the behavior of the closed-loop system is hard to predict. Although for first-order trajectories this has not resulted in undesirable results, it has been shown that initializing a higher-order trajectory at the process output can easily result in an unstable closed-loop system because of the additional feed-back loop that is introduced by initializing the reference trajectory at the process output. For this reason, it is better to specify the desired servo behavior of the closed-loop system by using

the P polynomial in the criterion function, or by generating an appropriate reference trajectory initialized by using past values of the reference trajectory.

- Level and rate constraints on the controller output are two types of constraints that often appear in practice. It has been shown that if these constraints are not taken into account, the performance of the closed-loop system deteriorates significantly when they become active. Rate constraints especially can have a disastrous effect on the performance of the closed-loop system. It has been shown that taking the constraints into account when minimizing the criterion function yields much better results than when they are not. Then, the effect of the constraints on the predicted process output is taken into account. Together with the receding horizon policy (used only in predictive controllers) this has been shown to yield very good results compared to when the constraints are not incorporated in the optimization problem. In most other controller design methods, constraints on the controller outputs cannot be taken into account because they do not use the receding horizon strategy, and because they minimize a criterion with respect to the controller parameters instead of the controller outputs.

  However, in predictive controllers a constrained optimization problem must be solved at every sampling instant. Rosen's gradient projection method has been used for this purpose. It has been shown that because of the simplicity of the constraints, the criterion function and the model, Rosen's method can be reduced to a simple and fast optimization algorithm which has been called DGP. Further, as in all nonlinear optimization algorithms, the starting point has a major effect on the number of iterations required to find with a particular accuracy the optimal solution. In the case of level constraints, an algorithm called ACH has been proposed for calculating the starting point for DGP. This has resulted in a considerable decrease in the number of floating point operations that are required at every sampling instant. It has been shown that, for a typical example, calculation times smaller than 20 ms can be obtained on a Tulip SX Compact 2 with a numeric coprocessor. This makes it possible to use DGP for the in-line control of many processes.

- It has been shown that the rule of thumb methods derived in this thesis can also be applied successfully to tune the UPC controller for industrial processes. Further, it has been shown that if a good model of the process and disturbances is available, a high-performance control system can easily be obtained. However, in such a case, much effort is required in order to obtain the model.

When little effort is used to obtain a model of the process, the UPC controller has still been shown to yield acceptable results in controlling an industrial process (compared to the controller that is normally used to control the process).

This thesis has, in my opinion, significantly enhanced the insight into the influence of the design parameters of predictive controllers on the behavior of the closed-loop system. Further, although a number of theoretical results are already intuitively understood, it is my belief that this thesis has contributed to an improved understanding of how predictive controllers operate and why they can successfully be used in practical applications. Nevertheless, there is still a wide area open for further research.

6.1 Suggestions for further Research

In this thesis, linear single-input single-output processes have been considered only. Further research is required to extend the Unified Predictive Controller to linear MIMO processes. Some initial research has shown that the results obtained in this thesis for SISO processes can, to a certain extent, also be used to analyze predictive controllers for MIMO systems.

Further, in Chapter 1 it has been argued that the predictive control concept can be used if a model of the process is available that describes its input-output behavior. Hence, not only nonlinear models but also models based on neural networks can be used. However, little research with respect to this has been reported in literature. The major difficulty when developing a predictive controller using a nonlinear model is that the minimization of the criterion function can no longer be performed analytically. In [13] a steepest-descent method is used to minimize the criterion function because the gradient of the criterion function with respect to the controller outputs was readily available. However, for other applications this may not be the case and hence the minimization of the criterion function must be performed in a different way. Another difficulty is that the minimization of the criterion function must be performed on line: there is, in general, no analytical solution to the optimization problem. For this reason the calculation time of the controller can easily become unacceptably large making the application of the controller not possible.

In Chapter 4 controller output constraints have been considered. The resulting minimization problem has been solved by using Rosen's gradient projection method. Although a solution to the optimization problem with sufficient accuracy is found

relatively fast, other methods for finding the solution of the constrained optimization problem might be more efficient. The method proposed in [55] is, in my opinion, a good candidate because it generates conjugate search directions with only a modest increase of the numerical complexity of the optimization algorithm.

Further research is also required to incorporate constraints on the output of the process (or states). For example, a level constraint on the process output may be imposed thus allowing the designer to specify an upper bound on the overshoot of the control system.

Appendix A

A.1 General Solution of the Diophantine Equation

In this section, the solution of the following Diophantine equation is discussed.

$$\frac{X}{Y} = E_i + q^{-i}\frac{F_i}{Y} \tag{A.2}$$

where X and Y are polynomials of degree $n_X \geq 0$ and $n_Y \geq 0$ with no common factors and E_i and F_i are polynomials which form the solution of (A.2) for a certain integer variable $i \geq 1$. In general (A.2) has many solutions. However, the objective

in this thesis is to calculate the first i coefficients of the impulse response of $\frac{X}{Y}$ and hence $n_{E_i} \leq i-1$. Two cases must be considered separately: $n_Y = 0$ and $n_Y > 0$.

$n_Y = 0$ Now (A.2) becomes:

$$\frac{X}{y_0} = E_i + q^{-i}\frac{F_i}{y_0}$$

The degrees of the polynomials E_i and F_i are given by $\min(i-1, n_X)$ and $n_X - i$ respectively. If $i \geq n_X + 1$ then $n_{E_i} = n_X$ and $n_{F_i} < 0$. Hence, E_i is given by $E_i = X/y_0$ and $F_i = 0$. If, on the other hand, $i < n_X + 1$ then the degrees of E_i and F_i are given by $i-1$ and $n_X - i$ respectively. Hence, E_i is given by the first i elements of the polynomial X/y_0 and F_i is given by the last $n_X - i + 1$ elements of X.

$n_Y > 0$ In this case the degrees of E_i and F_i are given by $i-1$ and $\max(n_X - i, n_Y - 1)$ respectively. E_i and F_i can be derived by using the recursive relationship between two successive solutions of (A.2):

$$X = YE_i + q^{-i}F_i \tag{A.3}$$

$$X = YE_{i+1} + q^{-i-1}F_{i+1} \tag{A.4}$$

Subtracting (A.3) from (A.4) yields:

$$Y\left[E_{i+1} - E_i\right] + q^{-i}\left[q^{-1}F_{i+1} - F_i\right] = 0 \tag{A.5}$$

Because $n_{E_{i+1}} = i$ and $n_{E_i} = i-1$, the following holds:

$$E_{i+1} - E_i = \tilde{E} + q^{-i}e_{i+1,i} \tag{A.6}$$

in which $\tilde{E}$ is a polynomial of degree $i-1$ and $e_{i+1,i}$ is the ith coefficient of E_{i+1}. Substituting (A.6) in (A.5) yields:

$$Y\tilde{E} + q^{-i}\left[q^{-1}F_{i+1} - F_i + Ye_{i+1,i}\right] = 0 \tag{A.7}$$

From (A.7) follows:

$$\tilde{E} = 0 \tag{A.8}$$

$$q^{-1}F_{i+1} = F_i - Ye_{i+1,i} \tag{A.9}$$

Solving (A.9) yields:

$$e_{i+1,i} = f_{i,0}/y_0 \tag{A.10}$$

$$f_{i+1,j} = f_{i,j+1} - y_{j+1}e_{i+1,i} \quad j = 0,\ldots,n_{F_{i+1}} \tag{A.11}$$

where $n_{F_{i+1}} = \max(n_X - i - 1, n_Y - 1)$. Note that for a certain range of j parts of (A.11) are equal to zero:

$$f_{i+1,j} = f_{i,j+1} \quad \text{if } j+1 > n_Y \tag{A.12}$$

$$f_{i+1,j} = -y_{j+1}e_{i+1,i} \quad \text{if } j+1 > n_{F_i} \tag{A.13}$$

$$f_{i+1,j} = 0 \quad \text{if } j+1 > n_Y \wedge j+1 > n_{F_i} \tag{A.14}$$

The polynomial E_{i+1} is now given by (A.6), (A.8) and (A.10):

$$E_{i+1} = E_i + q^{-i}e_{i+1,i}$$

while F_{i+1} is given by (A.10), ..., (A.14).

In order to obtain the recursive solutions of (A.2), the solution for $i = 1$ is required. The polynomial E_1 is simply given by x_0/y_0. The polynomial F_1 can easily be derived from (A.3) for $i = 1$:

$$X = YE_1 + q^{-1}F_1 \Longrightarrow F_1 = q\left(X - Y\frac{x_0}{y_0}\right)$$

Starting from the solution for $i = 1$, the solutions for $i = 2,\ldots$ can be derived recursively by using (A.10), ..., (A.14).

A.2 Diophantine Equations and i-step-ahead Predictors

In this section, the general solution of the Diophantine as discussed in section A.1 is applied to the Diophantine equations (2.3) and (2.62) which have to be solved in order to derive the MV i-step-ahead predictor of the unified prediction model when there is an unknown time delay. Of course, the Diophantine equations mentioned above are also valid if the time delay is known or absent.

Recall the first Diophantine equation to be solved:

$$\frac{1}{A} = E_i + q^{-i}\frac{F_i}{A} \qquad i \geq \underline{d} + 1 \tag{A.15}$$

From section A.1 directly follows that if $n_A > 0$, the degrees of E_i and F_i are given

by $i-1$ and n_A-1. The solution of E_i and F_i are found by using the recursive relations derived in section A.1. When $n_A = 0$, the solution is straightforward: $E_i = 1$ and $F_i = 0$.

The second Diophantine equation to be solved is given by (2.62) and is repeated here:

$$E_i\tilde{B} = G_{i-\underline{d}} + q^{-i+\underline{d}}H_{i-\underline{d}} \qquad i \geq \underline{d}+1 \tag{A.16}$$

When $n_A > 0$ the general solution presented in section A.1 can be used. Now, the degrees of $G_{i-\underline{d}}$ and $H_{i-\underline{d}}$ are equal to $i-\underline{d}-1$ and $n_B+\overline{d}-1$ respectively. One way to calculate $G_{i-\underline{d}}$ and $H_{i-\underline{d}}$ is by using the general solution of the Diophantine equation where $n_Y = 0$. However, this requires the polynomial $E_i\tilde{B}$ to be calculated for $i = \underline{d}+1, \ldots, H_p$. Because, multiplying polynomials is a rather time consuming procedure, a different method is presented. This method is based on the recursive relationship between two successive solutions of (A.16). The derivation is roughly the same as the one used to derive the general solution of the Diophantine equation in section A.1.
Two successive solutions of (A.15) are given by:

$$\tilde{B} = A\tilde{B}E_i + q^{-i}\tilde{B}F_i \tag{A.17}$$

$$\tilde{B} = A\tilde{B}E_{i+1} + q^{-i-1}\tilde{B}F_{i+1} \tag{A.18}$$

Subtracting (A.17) from (A.18) and taking into account (A.16) yields:

$$A\left[G_{i+1-\underline{d}} - G_{i-\underline{d}}\right] + q^{-i+\underline{d}}\left[q^{-1-\underline{d}}\tilde{B}F_{i+1} - q^{-\underline{d}}\tilde{B}F_i + q^{-1}AH_{i+1-\underline{d}} - AH_{i-\underline{d}}\right] = 0 \tag{A.19}$$

Because $n_{G_{i+1-\underline{d}}} = i-\underline{d}$ and $n_{G_{i-\underline{d}}} = i-\underline{d}-1$, the following holds:

$$G_{i+1-\underline{d}} - G_{i-\underline{d}} = \tilde{G} + q^{-i+\underline{d}}g \tag{A.20}$$

in which $\tilde{G}$ is a polynomial of degree $i-\underline{d}-1$ and g is the last coefficient of $G_{i+1-\underline{d}}$. Substituting (A.20) in (A.19) yields:

$$\tilde{G} = 0 \tag{A.21}$$

$$q^{-1}AH_{i+1-\underline{d}} = AH_{i-\underline{d}} - Ag + q^{-\underline{d}}\tilde{B}F_i - q^{-1-\underline{d}}\tilde{B}F_{i+1} \tag{A.22}$$

Using the recursive relationship (A.9) with $Y = A$ and $e_{i+1,i} = f_{i,0}$ in (A.22) yields:

$$q^{-1}H_{i+1-\underline{d}} = H_{i-\underline{d}} - g + f_{i,0}q^{-\underline{d}}\tilde{B} \tag{A.23}$$

From (A.23) the recursive relations follow directly:

$$g = h_{i-\underline{d},0} \qquad \underline{d} > 0 \tag{A.24}$$

$$g = h_{i-\underline{d},0} + f_{i,0}\tilde{b}_0 \qquad \underline{d} = 0 \tag{A.25}$$

$$h_{i+1-\underline{d},j} = h_{i-\underline{d},j+1} + f_{i,0}\tilde{b}_{j+1-\underline{d}} \quad j = 0,\ldots,n_{H_{i+1-\underline{d}}} \tag{A.26}$$

where $n_{H_{i+1-\underline{d}}} = n_B + \overline{d} - 1$. Note that for a certain range of j parts of (A.26) are equal to zero:

$$h_{i+1-\underline{d},j} = f_{i,0}\tilde{b}_{j+1-\underline{d}} \quad \text{if } j+1 > n_{H_{i-\underline{d}}}$$

$$h_{i+1-\underline{d},j} = h_{i-\underline{d},j+1} \quad \text{if } j+1-\underline{d} < 0$$

$$h_{i+1-\underline{d},j} = 0 \quad \text{if } j+1 > n_{H_{i-\underline{d}}} \wedge j+1-\underline{d} < 0$$

In order to start the recursions, the solution for $i = \underline{d}+1$ is required. Setting $i = \underline{d}+1$ in (A.16) yields:

$$E_{\underline{d}+1}\tilde{B} = G_1 + q^{-1}H_1 \tag{A.27}$$

From (A.27) follows directly:

$$G_1 = \tilde{b}_0 e_{\underline{d}+1,0}$$

$$H_1 = q(E_{\underline{d}+1}\tilde{B} - G_1)$$

When $n_A = 0$ then $E_i = 1$ and $F_i = 0$. Hence, (A.16) becomes:

$$\tilde{B} = G_{i-\underline{d}} + q^{-i+\underline{d}}H_{i-\underline{d}} \qquad i \geq \underline{d}+1 \tag{A.28}$$

For solving (A.28), the general solution with $n_Y = 0$, presented in section A.1, can be used. Note that now $n_{G_{i-\underline{d}}} = \min(i-1, n_B + \overline{d}) - \underline{d}$ and $n_{H_{i-\underline{d}}} = n_B + \overline{d} - i$.

A.3 i-step-ahead Predictors predicting $Py(k+i)$

In this section, the MV i-step-ahead predictor predicting $Py(k+i)$ is derived. First the predictor for the unified process model where $T = \hat{D} = 1$ is derived. For this purpose the following Diophantine equation is used:

$$\frac{P}{A} = E_i + q^{-i}\frac{F_i}{A} \qquad i \geq \underline{d} + 1 \tag{A.29}$$

The polynomials E_i and F_i are of degree $i-1$ and $\max(n_P - i, n_A - 1)$ if $n_A > 0$. If $n_A = 0$, the degrees of E_i and F_i are $\min(i-1, n_P)$ and $n_P - i$ respectively. The recursive formulas derived in section A.1 can be used to obtain the solutions of (A.29). Using the same strategy as presented in section 2.1.1, the MV i-step-ahead predictor is given by:

$$P\hat{y}(k+i) = E_i\tilde{B}u(k+i-\underline{d}-1) + F_i y(k) \qquad i \geq \underline{d}+1$$

Separation of future and past terms is realized by solving (2.62):

$$E_i\tilde{B} = G_{i-\underline{d}} + q^{-i+\underline{d}}H_{i-\underline{d}} \qquad i \geq \underline{d}+1$$

Now the degrees of $G_{i-\underline{d}}$ and $H_{i-\underline{d}}$ are for $n_A > 0$ given by $i-\underline{d}-1$ and $n_B+\overline{d}-1$. If $n_A = 0$ they are equal to $\min(i-1, n_B+\overline{d}+n_P)-\underline{d}$ and $\min(i-1, n_P)+n_B+\overline{d}-i$ respectively.
The MV i-step-ahead predictor becomes:

$$P\hat{y}(k+i) = G_{i-\underline{d}}u(k+i-\underline{d}-1) + F_i y(k) + H_{i-\underline{d}}u(k-1) \tag{A.30}$$

where $i \geq \underline{d}+1$. Collecting all i-step-ahead predictors for $i = H_s, \ldots, H_p$ in a matrix notation yields if $n_A > 0$:

$$\hat{\boldsymbol{y}}^* = \boldsymbol{\Gamma}\boldsymbol{u} + \boldsymbol{\Psi}\boldsymbol{s} \tag{A.31}$$

in which:

$$\begin{aligned}
\hat{\boldsymbol{y}}^* &= [P\hat{y}(k+H_s), \cdots, P\hat{y}(k+H_p)]^T && [\hat{\boldsymbol{y}}] = H_p - H_s + 1 \times 1 \\
\boldsymbol{u} &= [u(k), \cdots, u(k+H_p-1-\underline{d})]^T && [\boldsymbol{u}] = H_p - \underline{d} \times 1 \\
\boldsymbol{s} &= [y(k), \cdots, y(k-n_F), \\
&\qquad u(k-1), \cdots, u(k-n_B-\overline{d})]^T && [\boldsymbol{s}] = (n_F + 1 + n_B + \overline{d}) \times 1
\end{aligned}$$

where $n_F = \max(n_P - H_s, n_A - 1)$ and the matrices $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$ are of dimension $(H_p - H_s + 1) \times (H_p - \underline{d})$ and $(H_p - H_s + 1) \times (\max(n_P - H_s + 1, n_A) + n_B + \overline{d})$, respectively.

When $n_A = 0$, (A.31) is still valid. However, the degrees of the matrices Γ and Ψ now become:

$$\begin{aligned}
[\Gamma] &= (H_p - H_s + 1) \times (\min(H_p, n_B + n_P + 1 + \bar{d}) - \underline{d}) \\
[\Psi] &= (H_p - H_s + 1) \times (2n_P + n_B + \bar{d} - 2H_s + 2) \quad \text{if } H_s \geq n_P + 1 \\
[\Psi] &= (H_p - H_s + 1) \times (n_P + n_B + \bar{d} - H_s + 1) \quad \text{if } H_s < n_P + 1
\end{aligned}$$

and $n_F = n_P - H_s$.

When $n_A > 0$ the recursive relations derived in section A.2 can be used directly. However, when $n_A = 0$ some modifications are necessary because the degrees of E_i and F_i strongly differ from the ones used in the original derivation. The basic recursive relationships (A.20) and (A.23) are still valid. However, (A.24), ..., (A.26) need some modifications. From the degree of $G_{i-\underline{d}}$ follows that g is different from zero only if $i \leq n_B + \bar{d} + n_P$. Further, $H_{i-\underline{d}}$ exists only if $i \leq n_P + n_B + \underline{d}$. Hence, when $n_A = 0$, g can be calculated recursively using the following equations:

$$\begin{aligned}
g &= h_{i-\underline{d},0} + f_{i,0}\tilde{b}_0 \quad i \leq n_P \wedge \underline{d} = 0 \\
g &= h_{i-\underline{d},0} \quad i \leq n_P \wedge \underline{d} > 0 \\
g &= h_{i-\underline{d},0} \quad n_P < i \leq n_B + \bar{d} + n_P
\end{aligned}$$

For the calculation of $H_{i-\underline{d}}$, (A.26) can be used with the exclusions previously mentioned and with the exclusion that $f_{i,0} = 0$ if $i \geq n_P + 1$.

Thus far the i-step-ahead predictor and prediction model when $T = \hat{D} = 1$. Because the derivation of the unified i-step-ahead predictor in section 2.1.2 is independent of P, the unified MV i-step-ahead predictor is again given by (2.56). However, when $n_A = 0$ the degrees of the polynomials are different:

$$\begin{aligned}
n_{\bar{Q}_i} &= n_d - 1 \\
n_{Q_i} &= \min(n_d - 1, n_p + n_c - i) \leq n_d - 1 \\
n_{N_i} &= n_c + \min(i-1, n_p) - n_d - i \leq n_c - n_d - 1 \\
n_{K_i} &= n_d - 1 && n_{N_i} = 0 \\
&= \max(n_c, n_d) - 1 && i - 1 \leq n_p \\
&= \max(n_d - 1, n_c + n_p - i) \leq \max(n_c, n_d) - 1 && i - 1 \geq n_p
\end{aligned}$$

The matrix notation of the unified prediction model now becomes:

$$\hat{\boldsymbol{y}}^* = \boldsymbol{\Gamma u} + \boldsymbol{\Psi s} + \boldsymbol{k} \tag{A.32}$$

where $\hat{\boldsymbol{y}}^*$, $\boldsymbol{\Gamma}$, $\boldsymbol{\Psi}$, $\boldsymbol{s}$ and $\boldsymbol{u}$ are as described above. The vector $\boldsymbol{k}$ is given by (2.58):

$$\boldsymbol{k} = \boldsymbol{K}_c \tilde{\boldsymbol{c}}$$

in which:

$$\begin{aligned} \tilde{\boldsymbol{c}} &= [\tilde{c}(k), \cdots, \tilde{c}(k - n_K)]^T \qquad [\tilde{\boldsymbol{c}}] = (n_K + 1) \times 1 \\ \tilde{c}(k) &= \frac{\hat{A}y(k) - \hat{\tilde{B}}u(k - 1 - \underline{d})}{T} \end{aligned}$$

The matrix $\boldsymbol{K}_c$ is again given by (2.46) but is now built up of the coefficients of the polynomials K_i for $i = H_s, \ldots, H_p$:

$$\boldsymbol{K}_c = \begin{bmatrix} k_{H_s,0} & \cdots & k_{H_s,n_K} \\ \vdots & & \vdots \\ k_{H_p,0} & \cdots & k_{H_p,n_K} \end{bmatrix} \qquad [\boldsymbol{K}_c] = (H_p - H_s + 1) \times (n_K + 1)$$

where $n_K = \max(n_T, n_D) - 1$.

A.4 Calculation of R_j^{-1}

In this section a standard way to calculate $\boldsymbol{R}_j^{-1}$ is discussed. As shown by (4.57), the calculation of $\boldsymbol{R}_j^{-1}$ is similar to calculating the inverse of a matrix $\boldsymbol{C}$ given by:

$$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B}^T \boldsymbol{B} \tag{A.33}$$

where $\boldsymbol{A}$ is a diagonal matrix of dimension $n \times n$ and $\boldsymbol{B}$ is a matrix of dimension $m \times n$. Matrix $\boldsymbol{C}$ in (A.33) can also be written as:

$$\boldsymbol{C} = \boldsymbol{A} + \sum_{i=1}^{m} {}_r(B)_i^T \cdot {}_r(B)_i \tag{A.34}$$

Using a recursive way of computing C, (A.34) becomes:

$$
\begin{aligned}
&C_0 = A \\
&\texttt{For } i = 1, \ldots, m \texttt{ Do} \\
&\quad C_i = C_{i-1} + {}_r(B)_i^T \cdot {}_r(B)_i \\
&\texttt{End For} \\
&C = C_m
\end{aligned}
\tag{A.35}
$$

Rewriting (A.35) yields:

$$
\begin{aligned}
I &= C_i^{-1} C_{i-1} + C_i^{-1} {}_r(B)_i^T {}_r(B)_i \Longrightarrow \\
C_{i-1}^{-1} &= C_i^{-1} + C_i^{-1} {}_r(B)_i^T {}_r(B)_i C_{i-1}^{-1} \Longrightarrow \qquad \text{(A.36)} \\
C_i^{-1} {}_r(B)_i^T &= \frac{C_{i-1}^{-1} {}_r(B)_i^T}{1 + {}_r(B)_i C_{i-1}^{-1} {}_r(B)_i^T} \qquad \text{(A.37)}
\end{aligned}
$$

Rewriting (A.36) yields:

$$
C_i^{-1} = C_{i-1}^{-1} - C_i^{-1} {}_r(B)_i^T {}_r(B)_i C_{i-1}^{-1} \tag{A.38}
$$

Substitution of (A.37) in (A.38) yields:

$$
C_i^{-1} = C_{i-1}^{-1} - \frac{C_{i-1}^{-1} {}_r(B)_i^T {}_r(B)_i C_{i-1}^{-1}}{1 + {}_r(B)_i C_{i-1}^{-1} {}_r(B)_i^T} \tag{A.39}
$$

Now the inverse of C can be calculated by using:

$C_0^{-1} = A^{-1}$
For $i = 1, \ldots, m$ Do
    Calculate C_i^{-1} by using (A.39)
End For
$C^{-1} = C_m^{-1}$

Note that the inverse of A can easily be calculated because A is a diagonal matrix.

Bibliography

[1] J. Richalet, A. Rault, J.L. Testud and J. Papon, "Model predictive heuristic control: applications to industrial processes", *Automatica*, Vol. 14, No. 5, pp. 413-428, 1978.

[2] C.R. Cutler and B.L. Ramaker, "Dynamic Matrix Control - A Computer Control Algorithm", *Proceedings JACC*, San Francisco, U.S.A., 1980.

[3] P.M. Bruijn and H.B. Verbruggen, "Model algorithmic control using impulse response models", *Journal A*, Vol. 25, No. 2, pp. 69-74, 1984.

[4] D.W. Clarke, P.S. Tuffs and C. Mohtadi, "Self-tuning Control of a Difficult Process", *Proceedings 7th IFAC Symposium on Identification and System Parameter Estimation*, York, U.K., pp. 1009-1014, 1985.

[5] D.W. Clarke, C. Mohtadi and P.S. Tuffs, "Generalized Predictive Control-Part I. The Basic Algorithm", *Automatica*, Vol. 23, No. 2, pp. 137-148, 1987.

[6] D.W. Clarke, C. Mohtadi and P.S. Tuffs, "Generalized Predictive Control-Part II. Extensions and Interpretations", *Automatica*, Vol. 23, No. 2, pp. 149-160, 1987.

[7] R.M.C. De Keyser and A.R. van Cauwenberghe, "Extended Prediction Self-Adaptive Control", *Proceedings 7th IFAC Symposium on Identification and System Parameter Estimation*, York, U.K. pp. 1255-1260, 1985.

[8] J. Richalet and J. Papon, "Industrial Applications of Internal Model Control", *Proceedings 7th IFAC-IMACS Conference on digital computer applications to process control*, Vienna, Austria, 1985.

[9] A.R.M. Soeterboek, H.B. Verbruggen and P.P.J. van den Bosch, "Unified Predictive Control - Analysis of Design Parameters", *Proceedings of the IASTED International Symposium on Modelling, Identification and Control*, Grindelwald, Switzerland, 1989.

[10] B.E. Ydstie, "Extended Horizon Adaptive Control", *Proceedings 9th IFAC World Congress*, Budapest, Hungary, 1984.

[11] R.M.C. De Keyser, Ph. G.A. van de Velde and F.A.G. Dumortier, "A Comparative Study of Self-adaptive Long-range Predictive Control Methods", *Automatica*, Vol. 24, No. 2, pp. 149-163, 1988.

[12] M. Kinnaert, "Adaptive generalized Predictive controller for MIMO systems", *Int. J. Control*, Vol. 50, No. 1, pp. 161-172, 1989.

[13] A.R.M. Soeterboek, H.B. Verbruggen and P.P.J. van den Bosch, "Predictive Control of Nonlinear Processes", *Proceedings of the IFAC Symposium on Intelligent Tuning and Adaptive Control ITAC 91*, Singapore, 1991.

[14] J. Richalet, S. Abu el Ata-Doss, Ch. Arber, H.B. Kuntze, A. Jacubasch and W. Schill, "Predictive Functional Control. Application to fast and accurate robots", *Proceedings 10th IFAC World Congress*, Munich, F.R.G., 1987.

[15] P. Djavdan, H.J.A.F. Tulleken, M.H. Voetter, H.B. Verbruggen and G.J. Olsder, "Probabilistic Robust Controller Design", *Proceedings 28th Conference on Decision and Control*, Tampa, Florida, U.S.A., pp. 2164-2172, 1989.

[16] U. Hoffmann, N. Wiesner, P. Schmitz and H. Rake, "A Predictive Adaptive on-off Controller for Temperature Control of Injection Moulding Machines", *Proceedings of the 12th IMACS World Congress on Scientific Computation*, Paris, France, 1988.

[17] H. Kwakernaak, "Robust Control", *Journal A*, Vol. 29, No. 4, pp. 17-27, 1988.

[18] C. Mohtadi and D.W. Clarke, "Generalised Predictive Control, LQ, or Pole-placement: A unified approach", *Proceedings of the 25th Conference on Decision and Control*, Athens, Greece, pp. 1536-1541, 1986.

[19] K.J. Åström and B. Wittenmark, *Computer Controlled Systems*, Prentice-Hall, Inc., 1984.

[20] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*, Wiley-Interscience, New York, 1972.

[21] K.J. Åström and B. Wittenmark, *Adaptive Control*, Addison-Wesley, 1989.

[22] H. Butler, *Model Reference Adaptive Control - Bridging the Gap between Theory and Practice*, Ph.D. thesis, Delft University of Technology, the Netherlands, 1990.

[23] C. Greco, G. Menga, E. Mosca and G. Zappa, "Performance Improvements of Self-tuning Controllers by Multi-step Horizons: The MUSMAR Approach", *Automatica*, Vol. 20, pp. 681-699, 1984.

[24] E. Mosca, G. Zappa and J.M. Lemos, "Robustness of Multipredictor Adaptive Regulators: MUSMAR", *Automatica*, Vol. 25, pp. 521-529, 1989.

[25] R.M.C. De Keyser, "Model Based Predictive Control Toolbox", *Proceedings CIM-Europe Workshop on Computer Integrated Design of Controlled Industrial Systems*, Paris, France, pp. 35-56, 1990.

[26] P.M. Bruijn, H.B. Verbruggen and O.V. Appeldoorn, "Predictive Control: A comparison and simple implementation", *Proceedings IFAC Conference on Low Cost Automation*, Valencia, Spain, 1986.

[27] A.R.M. Soeterboek, H.B. Verbruggen and P.P.J. van den Bosch, "Self-tuning Control in the Presence of Constraints Using Multi-step Criterion Functions", *Proceedings Fifth Yale Workshop on Applications of Adaptive Systems Theory*, New Haven. U.S.A., 1987.

[28] A.R.M. Soeterboek, H.B. Verbruggen and P.P.J. van den Bosch, "Self-tuning Control of Processes with Signal Level and Rate Constraints", *Proceedings 12th IMACS World Congress on Scientific Computation*, Paris, France, 1988.

[29] A.R.M. Soeterboek, H.B. Verbruggen, P.P.J. van den Bosch and H. Butler, "On the Unification of Predictive Control Algorithms", *Proceedings 29th IEEE Conference on Decision and Control*, Honolulu, U.S.A., 1990.

[30] V. Kučera, *Discrete Linear Control-The Polynomial Equation Approach*, John Wiley & Sons, 1979.

[31] D.W. Clarke and C. Mohtadi, "Properties of Generalized Predictive Control", *Automatica*, Vol. 25, No. 6, pp. 859-875, 1989.

[32] J.E. Marshall, "Extensions of O.J. Smith's method to digital and other systems", *Int. J. Control*, Vol. 19, No. 5, pp. 933-939, 1974.

[33] O.J.M. Smith, "A controller to overcome deadtime", *ISA Journal*, Vol. 6, No. 2, pp. 28-33, 1959.

[34] K.J. Åström, *Introduction to Stochastic Control Theory*, Academic Press New York and London, 1970.

[35] H.B. Verbruggen, *Digital Control Systems* (in Dutch), Delftse Universitaire Pers, 1975.

[36] D.W. Clarke and P.J. Gawthrop, "A Self-tuning Controller", *IEE Proc.*, No. 122, pp. 929-934, 1975.

[37] E. Mosca and G. Zappa, "ARX Modelling of Controlled ARMAX Plants and its Application to Robust Multipredictor Adaptive Control", *Proceedings 24th Conference on Decision and Control*, Ft. Lauderdale, U.S.A., pp. 653-660, 1985.

[38] A.R.M. Soeterboek, H.B. Verbruggen, P.P.J. van den Bosch and H. Butler, "Adaptive Predictive Control - A Unified Approach", *Proceedings Sixth Yale Workshop on Applications of Adaptive Systems Theory*, New Haven. U.S.A., 1990.

[39] E. Trulsson and L. Ljung, "Adaptive Control Based on Explicit Criterion Minimization", *Automatica*, Vol. 21, No. 4, pp. 385-399, 1985.

[40] R.C. Dorf, *Modern Control Systems* (third edition), Addison-Wesley, 1980.

[41] C.D. Doyle and G. Stein, "Multivariable Feedback Design: Concepts for a Classical/Modern Synthesis", *IEEE Trans. Autom. Control*, AC-26, No. 1, pp. 4-16, 1981.

[42] E.I. Jury and J. Blanchard, "A stability test for linear discrete systems in table form", *IRE Proc.* Vol. 49, No. 12, 1961.

[43] D.W. Clarke, "Generalized Predictive Control and its Application", *Proceedings CIM-Europe Workshop on Computer Integrated Design of Controlled Industrial Systems*, Paris, France, pp. 57-75, 1990.

[44] C.E. Rohrs, M. Athans, L. Valavani and G. Stein, "Some design quidelines for discrete-time adaptive controllers", *Automatica*, Vol. 20, No. 5, pp. 653-660, 1984.

[45] H.C. Lammers, *Towards the practical application of self-tuning adaptive controllers*, Ph.D. thesis, Delft University of Technology, the Netherlands, 1984.

[46] D.M. Prett and C.E. García, *Fundamental Process Control*, Butterworth Publishers, 1988.

[47] A.R.M. Soeterboek, H.B. Verbruggen and P.P.J. van den Bosch, "On the Design of the Unified Predictive Controller", *Proceedings of the IFAC Symposium on Intelligent Tuning and Adaptive Control ITAC 91*, Singapore, 1991.

[48] A.J. Krijgsman, H.B. Verbruggen and P.M. Bruijn, "Knowledge-Based Tuning and Control", *Proceedings of the IFAC Symposium on Intelligent Tuning and Adaptive Control ITAC 91*, Singapore, 1991.

[49] P.E. Gill and W. Murray, "Linearly-Constrained Problems including Linear and Quadratic Programming", *Proceedings of the Conference on The State of the Art in Numerical Analysis*, York, U.K., pp. 313-363, 1977.

[50] C.E. García, "Quadratic Dynamic Matric Control of nonlinear processes: An application to a batch reaction process", *AIChE Annual Meeting*, San Francisco, U.S.A., 1984.

[51] C.E. García and A.M. Morshedi, "Solution of the dynamic matrix control problem via quadratic programming", *Proceedings of the Conference of the Canadian Industrial Computing Society*, Ottawa, Canada, pp. 13.1-13.3, 1984.

[52] T.T.C. Tsang and D.W. Clarke, "Generalised Predictive Control with input constraints", *IEE Proceedings*, Vol. 135, No. 6, pp. 451-460, 1988.

[53] M.S. Bazaraa and C.M. Shetty, *Nonlinear Programming - Theory and Algorithms*, John Wiley & Sons, Inc., 1979

[54] J.B. Rosen, "The Gradient Projection Method for Nonlinear Programming, Part I, Linear Constraints", *SIAM J. Applied Mathematics*, No. 8, pp. 181-217, 1960.

[55] P.P.J. van den Bosch and F.A. Lootsma, "Scheduling of Power Generation via Large-Scale Nonlinear Optimization", *Journal of Optimization Theory and Applications*, Vol. 55, No. 2, pp. 313-326, 1987.

[56] R. Fletcher and C.M. Reeves, "Function minimization by conjugate gradients", *Computer Journal*, Vol. 7, pp. 149-154, 1964.

[57] W.H. Press, B.P. Flannery, S.A. Teukolsky and W.T. Vetterling, *Numerical Recipes in C - The Art of Scientific Computing*, Cambridge University Press, 1989.

[58] E.P. Lambert, *Process Control Applications of Long-Range Prediction*, D.Phil. thesis, Oxford University, U.K., 1987.

[59] C.R. Cutler and R.B. Hawkins, "Constrained multivariable control of a hydrocraker reactor", *Proceedings ACC*, Minneapolis, U.S.A., 1987.

[60] A.F. Pels, "Closed-loop Mach number control in a transonic wind tunnel", *Journal A*, Vol. 30, No. 4, pp. 25-32, 1989.

[61] A.R.M. Soeterboek, H.B. Verbruggen, A.F. Pels and G.C.A. van Langen, "A Predictive Controller for the Mach Number in a Transonic Wind Tunnel". To be published in the IEEE Control Systems Magazine, 1990.

[62] P.P.J. van den Bosch and A.R.M. Soeterboek, "CAD on Microcomputers for Control System Design", *Proceedings 12th IMACS World Congress*, Paris, France, 1988.

List of Symbols

General symbols

Italic printed upper case characters denote polynomials in q^{-1}. For example, A and B are polynomials in q^{-1}.

Italic printed lower case characters denote elements of polynomials, matrices, vectors or signals as a function of k (= discrete time).

Italic and bold printed upper case characters denote matrices. For example, $\boldsymbol{\Psi}$ and $\boldsymbol{M}$ are matrices.

Italic and bold printed lower case characters denote column vectors. For example, λ and $\boldsymbol{s}$ are vectors. A row vector is denoted by using the transpose operator. For example, λ^T and $\boldsymbol{s}^T$ are row vectors.

q^{-1}	denotes the backward shift operator: $q^{-1}x(k) = x(k-1)$
q	denotes the forward shift operator: $q\,x(k) = x(k+1)$
T_s	denotes the sampling period in seconds
z	denotes the complex variable used in the z-transform: $z = e^{T_s s}$ in which s is the complex variable used in the Laplace transform
${}_r(A)_j$	denotes the jth row of matrix A. If convenient, the symbol a_j^T is also used for this purpose. If matrix A is of dimension $n \times m$, then $[{}_r(A)_j] = 1 \times m$.
${}_c(A)_j$	denotes the jth column of matrix A. If matrix A is of dimension $n \times m$, then $[{}_c(A)_j] = n \times 1$.
$(A)_{ij}$	denotes element ij of matrix A. If convenient, the symbol a_{ij} is also used for this purpose.
x_i	denotes the ith element of a polynomial X.
n_X	denotes the degree of a polynomial X.
X_i	denotes a polynomial which is a function of i. Its degree is denoted by n_{X_i} and its jth element is given by $x_{i,j}$
$X(1)$	'gain' of a polynomial: $X(1) = \sum_{j=0}^{n_X} x_j$
$\hat{(\cdot)}$	denotes estimated variables
$\tilde{(\cdot)}$	signal filtered by T: $\tilde{x}(k) = \dfrac{x(k)}{T}$
x_{ss}	steady-state value of x(k): $$x_{ss} = \lim_{k\to\infty} x(k) = \lim_{z\to 1}(1 - z^{-1})X(z^{-1})$$
$(.)^T$	transpose operator

$[.]$	matrix dimension
Δ	differencing operator: $\Delta = 1 - q^{-1}$
J	criterion function
$E(.)$	expectation operator
$\partial(.)$	partial derivative operator
H_L	denotes the loop transfer function (= the loop transfer function based on the process)
$\hat{H}_L$	denotes the nominal loop transfer function (= the loop transfer function based on the model)
$P(.)$	transforms a row vector into a polynomial in q^{-1}: $P(x^T) = Y$ where: $x^T = [x_1, \ldots, x_n]$ $[x] = 1 \times n$; $Y = x_1 + x_2 q^{-1} + \cdots + x_{n-1} q^{-n+1}$ $n_Y = n - 1$
ω	denotes frequency in rad/s
$l(k)$	denotes a unit step: $L(z^{-1}) = \dfrac{1}{1 - z^{-1}}$

Model parameters

A, B, C, D	polynomials describing the process
$y(k)$	process output at $t = k$
$u(k)$	controller output and process input at $t = k$
$e(k)$	discrete white noise with zero mean
K_{dc}	DC-gain of the process: $K_{dc} = \dfrac{B(1)}{A(1)}$

$\hat{K}_{dc}$	DC-gain of the process model: $\hat{K}_{dc} = \dfrac{\hat{B}(1)}{\hat{A}(1)}$
d	time delay of the process in samples
$\hat{d}$	time delay of the model in samples
T_d	denotes the time delay of the process in seconds
T	estimate of C: $T = \hat{C}$
$\xi(k)$	disturbance acting on the output of the process
E_i, F_i, G_i, H_i	polynomials as a function of i which form the solution of a Diophantine equation with $T = \hat{D} = 1$
$\overline{E}_i, \overline{F}_i, \overline{G}_i, \overline{H}_i$	polynomials as a function of i which form the solution of a Diophantine equation with $T \neq 1$ and/or $\hat{D} \neq 1$
k	correction vector added to the prediction model based on a process model with $T = \hat{D} = 1$

Controller parameters

H_p	prediction horizon
H_s	minimum cost horizon
H_c	control horizon
β	denotes the type of the control horizon
$N(q^{-1})$	polynomial in definition of control horizon
ρ	weighting factor
P, Q_n, Q_d	polynomials in the unified criterion function
R	real variable equal to $P(1)$
$\underline{d}$	lower bound of the time delay of the process in samples
$\overline{d}$	upper bound of the time delay of the process in samples

$\underline{u}$	lower bound level constraint
$\overline{u}$	upper bound level constraint
$\underline{\Delta u}$	lower bound rate constraint
$\overline{\Delta u}$	upper bound rate constraint
$\mathcal{R}, S, \mathcal{T}$	controller polynomials
$u(k)$	controller output and process input at time $t = k$
$w(k)$	reference trajectory at time $t = k$
Sp	set point
ϵ	accuracy of solution found by DGP
η	accuracy of solution found by ACH

Servo performance criteria

t_r	denotes the rise time in seconds
t_p	denotes the peak time in seconds
t_s	denotes the settling time in seconds
O	denotes the overshoot in %
ϵ_{ss}	denotes the steady-state error in %

Robustness criteria

gm	denotes the gain margin
$\underline{gm}$	denotes the lower gain margin
$\overline{gm}$	denotes the upper gain margin
dm	denotes the delay margin in seconds
$\underline{dm}$	denotes the lower delay margin in seconds
$\overline{dm}$	denotes the upper delay margin in seconds

List of Abbreviations

ACH	Active Constraint Handling
ARIX	Auto-Regressive Integrated Exogenous
ARIMAX	Auto-Regressive Integrated Moving-Average eXogenous
ARMAX	Auto-Regressive Moving-Average eXogenous
ARX	Auto-Regressive eXogenous
CLCE	Closed-Loop Characteristic Equation
CLTF	Closed-Loop Transfer Function

DGP	Dedicated Gradient Projection method
DMC	Dynamic Matrix Control
EHAC	Extended Horizon Adaptive Control
EPSAC	Extended Prediction Self-Adaptive Control
ECM	Explicit Criterion Minimization
FIR	Finite Impulse Response
FLOP	Floating Point OPeration
FSR	Finite Step Response
GMV	Generalized Minimum Variance
GPC	Generalized Predictive Control
IMAC	Interlaced Multipredictor Adaptive Controller
LQ	Least Quadratic
MAC	Model Algorithmic Control
MIMO	Multi-Input Multi-Output
MSTC	Minimum-Settling Time Control
MV	Minimum Variance
PCA	Predictive Control Algorithm
PFC	Predictive Functional Control
RFRC	Ripple-Free Response Control
SISO	Single-Input Single-Output
UPC	Unified Predictive Control

List of Corollaries, Definitions and Theorems

Summary

All predictive controllers are based on the same concept: they calculate such a controller output sequence over the prediction horizon that the predicted process output is close to the reference trajectory. A model of the process is used to predict the process output while a criterion function is used to define how well the predicted process output tracks the reference trajectory. Minimization of the criterion function yields the optimal controller output sequence over the prediction horizon. Hence, for finding this optimal controller output sequence an optimization problem must be solved.

Despite the fact that predictive controllers are all based on the same principle, there are many approaches to predictive controller design. Using different models and criterion functions yield different predictive controllers each having its own properties.

In this thesis, a unified approach to predictive controller design for linear SISO processes is adopted by using a general process model and by using a general criterion function. The general process model unifies several, often used, models such as ARIMAX and impulse response models. Further, the process model includes a general noise or disturbance model. The resulting predictive controller is called the Unified Predictive Controller (UPC). In the thesis special attention is paid to the design of the UPC controller and to the influence of the noise model on the closed-loop system. It is shown that the optimization problem can be solved analytically resulting in a linear control law which can easily be implemented in a computer. Further, it is shown that the noise model plays an important role in determining the robustness and regulator behavior of the control system. A detailed analysis of the UPC controller shows that UPC not only unifies several well-known predictive controllers such as MAC, GPC and DMC, but also unifies minimum-variance and pole-placement controllers.

As a result of the unified approach, the UPC controller has many design parameters for the designer to select. In order to get an insight into the influence of these parameters on the closed-loop system, a simulation study is carried out. Based on theoretical results and on results obtained during the simulations, rule of thumb methods are derived that can be used to find initial settings of all design parameters. Usually, the control system behaves satisfactorily if the UPC design parameters are set by using the rule of thumb methods. However, in some situations, fine tuning may be required. Fine tuning methods are provided for this purpose.

In practice there are always constraints on the controller output. Not taking these constraints into account can deteriorate the performance of the system or can even make the control system unstable. In predictive control algorithms constraints on the controller output can be taken into account in a natural way. However, for finding the optimal controller output sequence when there are constraints, a nonlinear optimization problem must be solved. In the thesis only level and rate constraints, two often occurring constraints, are considered. For solving the optimization problem, Rosen's gradient projection method is used. Because of the simplicity of the constraints this method can much be simplified resulting in a dedicated gradient projection algorithm called DGP which is simple and fast. The DGP algorithm uses a heuristic algorithm called ACH (Active Constraint Handling) for finding a starting point of the optimization procedure. Because the starting point generated by ACH usually is close to the optimum, a fast convergence is obtained. It is shown that by taking the constraints on the controller output into account, a much better performance of the closed-loop system is obtained than when they are not.

Finally, two industrial applications show that the use of the Unified Predictive Controller is not restricted to simulations.

Samenvatting

Elke voorspellende regelaar is gebaseerd op hetzelfde concept: er wordt een zodanige stuurreeks berekend over de voorspellingshorizon dat de voorspelde procesuitgang het referentie traject zo goed mogelijk volgt. Een model van het te regelen proces wordt gebruikt om de uitgang van het proces te voorspellen. Een criteriumfunctie wordt gebruikt om te bepalen hoe goed het referentie traject wordt gevolgd. Minimalisatie van deze criteriumfunctie levert de optimale stuurreeks op die aan het proces moet worden toegevoerd. Er moet dus een optimalisatie probleem worden opgelost.

Ondanks het feit dat alle voorspellende regelaars zijn gebaseerd op hetzelfde concept zijn er vele mogelijkheden om zo'n regelaar te ontwerpen. Door gebruik te maken van verschillende modellen en criteriumfuncties ontstaan verschillende voorspellende regelaars met verschillende eigenschappen.

In dit proefschrift is een universele benadering gevolgd voor het ontwerp van voorspellende regelaars voor lineaire SISO processen. Dit is gedaan door gebruik te maken van een universeel procesmodel in combinatie met een universele criteriumfunctie. Het procesmodel kan worden gebruikt voor o.a. ARIMAX en impuls responsie modellen. Verder bevat het universele procesmodel een verstorings model. De resulterende voorspellende regelaar is Unified Predictive Controller (UPC) genoemd. In dit proefschrift is speciale aandacht besteed aan het ontwerp van de UPC regelaar en aan de invloed van het verstoringsmodel op het gedrag van het gesloten lus systeem. Het is aangetoond dat het optimalisatie probleem analytisch kan worden opgelost. Het resultaat is een lineaire regelaar die eenvoudig in een computer kan worden geïmplementeerd. Ook is aangetoond dat het verstoringsmodel een belangrijke invloed heeft op de robuustheid en het regelateurgedrag van het regelsysteem. Een gedetailleerde analyse van de UPC regelaar heeft laten zien dat deze regelaar niet alleen een aantal bekende voorspellende regelaars zoals de MAC, GPC en DMC regelaars generaliseert, maar ook een aantal andere regelaars zoals minimum variantie en poolplaatsende regelaars.

Een gevolg van de universele aanpak is dat de UPC regelaar een groot aantal parameters heeft die door de ontwerper moeten worden ingesteld. Een simulatiestudie is verricht om inzicht te krijgen in de invloed van deze parameters op het systeem gedrag. Gebaseerd op theoretische resultaten en op de resultaten verkregen uit de simulaties is een aantal vuistregels opgesteld die kunnen worden gebruikt om de regelaar (grof) in te stellen. Het instellen van de regelaar met behulp van de vuistregels leidt in de meeste gevallen tot acceptabele resultaten. Echter in sommige gevallen is een fijninstelling noodzakelijk. In het proefschrift zijn enige methoden voor fijninstelling van de regelaar beschreven.

In de praktijk is de regelaaruitgang altijd begrensd. Wanneer de begrenzingen niet worden meegenomen in het regelaarontwerp kan de performance van het regelsysteem zeer slecht worden; het systeem kan zelfs instabiel worden. In voorspellende regelaars kunnen begrenzingen op de uitgang van de regelaar op een natuurlijke manier worden geïntegreerd. Het resultaat is echter dat een niet-lineair optimalisatie probleem moet worden opgelost. In dit proefschrift zijn alleen amplitude- en snelheids begrenzingen beschouwd. Voor het oplossen van het optimalisatie probleem is de gradient projectie methode van Rosen gebruikt. Doordat de begrenzingen relatief simpel zijn kan deze methode sterk worden vereenvoudigd. Een en ander heeft geleid tot een optimalisatie methode die speciaal geschikt is gemaakt om het optimalisatie probleem met amplitude- en snelheids begrenzingen op te lossen. Deze methode is DGP (Dedicated Gradient Projection) genoemd. Een heuristisch algoritme genaamd ACH (Active Constraint Handling) is ontwikkeld om een startpunt te vinden voor de optimalisatie procedure. Omdat het door het ACH algoritme gevonden startpunt

in het algemeen dicht bij de optimale oplossing ligt, wordt een snelle convergentie van het DGP algoritme verkregen. Door de begrenzingen op de regelaar uitgang mee te nemen in het genereren van de optimale stuurreeks over de voorspellingshorizon wordt een veel beter gedrag van het regelsysteem verkregen vergeleken met het gedrag van het systeem wanneer de begrenzingen niet worden meegenomen.

Tenslotte laten twee industriële toepassingen zien dat de UPC regelaar niet alleen geschikt is om simulaties uit te voeren maar ook kan worden gebruikt om echte processen te regelen.

Curriculum Vitae

A. Ronald M. Soeterboek was born in Rotterdam, the Netherlands, on September 3, 1962. After graduating in 1980 from the pre-university education of the O.S.G. Albert Einstein in Hoogvliet, he studied Electrical Engineering at the Delft University of Technology, Delft, the Netherlands, from which he graduated in 1985 (cum laude) on a thesis on adaptive predictive control.

In October 1985, he became a research assistant at the Control Laboratory of the Electrical Engineering Department of the same university.

In January 1987, he became an assistant professor at the Control Laboratory, Department of Electrical Engineering, Delft University of Technology, where he continued his research in the field of adaptive and predictive control.

Acknowledgements

The work described in this thesis could not have been carried out without the help of many people.

In the first place I wish to thank my supervisors Prof. ir. H.B. Verbruggen and Prof. dr. ir. P.P.J. van den Bosch for the fruitful discussions on discrete time controllers and on optimization methods.

Further, I want to thank Addy J. Koster of the Royal Gist-brocades N.V. in Delft, who helped me doing the experiments on the distillation column and Arthur F. Pels of the National Aerospace Laboratory in Amsterdam for his collaboration in doing the experiments on the wind tunnel.

Also, I wish to thank Hans Butler for the many fruitful discussions we had (and still have) on model reference adaptive control versus predictive control.

I am obliged to Mrs. J.B. Zaat-Jones for the correction of my English and

to the staff and students of the Control Laboratory of the Department of Electrical Engineering for providing a pleasant working atmosphere.

Finally, I want to thank Nick who forced me to finish this thesis in time and Anita who hardly ever complained me sitting in my study behind a computer doing my thesis work.

Index

U

W