E.F. Camacho and C. Bordons

Model Predictive control

Second Edition

Springer

2

Model Predictive Controllers

This chapter describes the elements that are common to all Model Predictive controllers, showing the various alternatives used in the different implementations. Some of the most popular methods will later be reviewed to demonstrate their most outstanding characteristics.

2.1 MPC Elements

All the MPC algorithms possess common elements, and different options can be chosen for each element giving rise to different algorithms. These elements are:

- prediction model,
- objective function and
- obtaining the control law.

2.1.1 Prediction Model

The model is the cornerstone of MPC; a complete design should include the necessary mechanisms for obtaining the best possible model, which should be complete enough to fully capture the process dynamics and allow the predictions to be calculated, and at the same time to be intuitive and permit theoretic analysis. The use of the process model is determined by the necessity to calculate the predicted output at future instants $\hat{y}(t+k \mid t)$. The different strategies of MPC can use various models to represent the relationship between the outputs and the measurable inputs, some of which are manipulated variables and others can be considered to be measurable disturbances which can be compensated for by feedforward action. A disturbance model can also be taken into account to describe the behaviour not reflected by the process model, including the effect of nonmeasurable inputs,

E. F. Camacho et al., *Model Predictive Control*

noise and model errors. The model can be separated into two parts: the actual process model and the disturbances model. Both parts are needed for the prediction.

Process Model

Practically every possible form of modelling a process appears in a given MPC formulation, the following being the most commonly used:

- Impulse response. Also known as weighting sequence or convolution model, it appears in MAC and as a special case in GPC and EPSAC. The output is related to the input by the equation

$$y(t) = \sum_{i=1}^{\infty} h_i u(t-i)$$

where h_i is the sampled output when the process is excited by a unitary impulse (see Figure 2.1a). This sum is truncated and only N values are considered (thus only stable processes without integrators can be represented), having

$$y(t) = \sum_{i=1}^{N} h_i u(t-i) = H(z^{-1})u(t) \tag{2.1}$$

where $H(z^{-1}) = h_1 z^{-1} + h_2 z^{-2} + \cdots + h_N z^{-N}$, where z^{-1} is the backward shift operator. Another inconvenience of this method is the large number of parameters necessary, as N is usually a high value (on the order of 40 to 50). The prediction will be given by:

$$\hat{y}(t+k \mid t) = \sum_{i=1}^{N} h_i u(t+k-i \mid t) = H(z^{-1})u(t+k \mid t)$$

This method is widely accepted in industrial practice because it is very intuitive and clearly reflects the influence of each manipulated variable on a determined output. Note that if the process is multivariable, the different outputs will reflect the effect of the m inputs in the following way:

$$y_j(t) = \sum_{k=1}^{m} \sum_{i=1}^{N} h_i^{kj} u^k(t-i)$$

One great advantage of this method is that no prior information about the process is needed, so that the identification process is simplified, and at the same time it allows complex dynamics such as nonminimum phase or delays to be described easily.

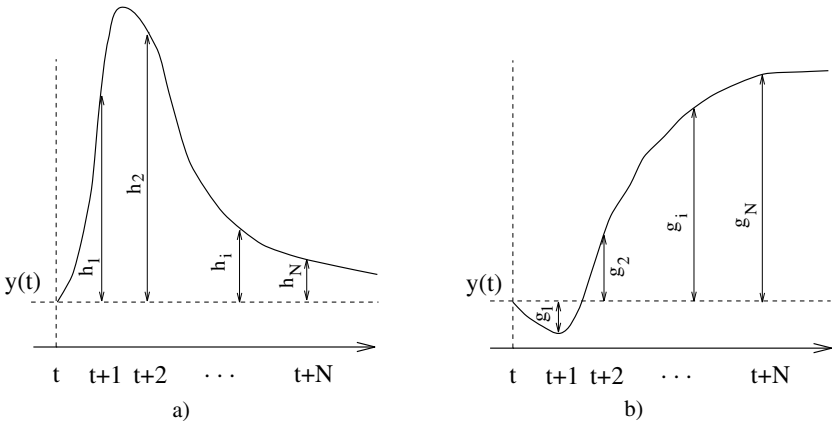


Fig. 2.1. Impulse and step response

- Step response. Used by DMC and its variants, this is very similar to impulse response except that the input signal is a step. For stable systems, the truncated response is given by:

$$y(t) = y_0 + \sum_{i=1}^{N} g_i \triangle u(t-i) = y_0 + G(z^{-1})(1-z^{-1})u(t) \tag{2.2}$$

where g_i are the sampled output values for the step input and $\triangle u(t) = u(t) - u(t-1)$ as shown in Figure 2.1b. The value of y_0 can be taken to be 0 without loss of generality, so that the predictor will be:

$$\hat{y}(t+k \mid t) = \sum_{i=1}^{N} g_i \triangle u(t+k-i \mid t)$$

As an impulse can be considered as the difference between two steps with a lag of one sampling period, it can be written for a linear system that:

$$h_i = g_i - g_{i-1} \qquad\qquad g_i = \sum_{j=1}^{i} h_j$$

This method has the same advantages and disadvantages as the impulse response method.

- Transfer function. Used by GPC, UPC, EPSAC, EHAC, MUSMAR or MURHAC (amongst others), this uses the concept of transfer function $G = B/A$ so that the output is given by:

$$A(z^{-1})y(t) = B(z^{-1})u(t)$$

with

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{na} z^{-na}$$
$$B(z^{-1}) = b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{nb} z^{-nb}$$

Thus the prediction is given by

$$\hat{y}(t+k \mid t) = \frac{B(z^{-1})}{A(z^{-1})} u(t+k \mid t)$$

This representation is also valid for unstable processes and has the advantage that it only needs a few parameters, although *a priori* knowledge of the process is fundamental, especially that of the order of the A and B polynomials.

- State space. Used in PFC, for example, it has the following representation:

$$x(t) = Ax(t-1) + Bu(t-1)$$
$$y(t) = Cx(t)$$

where x is the state and A, B and C are the matrices of the system, input and output respectively. The prediction for this model is given by [11]

$$\hat{y}(t+k \mid t) = C\hat{x}(t+k \mid t) = C[A^k x(t) + \sum_{i=1}^{k} A^{i-1} Bu(t+k-i \mid t)]$$

It has the advantage that it can be used for multivariable processes in a straightforward manner. The control law is simply the feedback of a linear combination of the state vector, although sometimes the state basis chosen has no physical meaning. The calculations may be complicated with the additional necessity of including an observer if the states are not accessible.

- Others. Nonlinear models can also be used to represent the process, but they cause the optimization problem to be more complicated. Neural nets [198] and fuzzy logic [192] are other forms of representation used in some applications.

Disturbances Model

The choice of the model used to represent the disturbances is as important as the choice of the process model. A model widely used is the Controlled Auto-Regressive and Integrated Moving Average (CARIMA) in which the disturbances, that is, the differences between the measured output and the output calculated by the model, are given by

$$n(t) = \frac{C(z^{-1})e(t)}{D(z^{-1})}$$

where the polynomial $D(z^{-1})$ explicitly includes the integrator $\triangle = 1 - z^{-1}$, $e(t)$ is a white noise of zero mean and the polynomial C is normally considered to equal one. This model is considered appropriate for two types of disturbances, random changes occurring at random instants (for example, changes in the quality of the material) and "Brownian motion" and it is used directly in GPC, EPSAC, EHAC UPC and with slight variations in other methods. Note that by including an integrator an offset-free steady-state control is achieved.

Using the Diophantine equation

$$1 = E_k(z^{-1})D(z^{-1}) + z^{-k}F_k(z^{-1}) \tag{2.3}$$

one has

$$n(t) = E_k(z^{-1})e(t) + z^{-k}\frac{F_k(z^{-1})}{D(z^{-1})}e(t) \qquad n(t+k) = E_k(z^{-1})e(t+k) + F_k(z^{-1})n(t)$$

and the prediction will be

$$\hat{n}(t+k \mid t) = F_k(z^{-1})n(t) \tag{2.4}$$

If equation (2.4) is combined with a transfer function model (like the one used in GPC), making $D(z^{-1}) = A(z^{-1})(1 - z^{-1})$, the output prediction can be obtained:

$$\hat{y}(t+k \mid t) = \frac{B(z^{-1})}{A(z^{-1})}u(t+k \mid t) + F_k(z^{-1})(y(t) - \frac{B(z^{-1})}{A(z^{-1})}u(t))$$

$$\hat{y}(t+k \mid t) = F_k(z^{-1})y(t) + \frac{B(z^{-1})}{A(z^{-1})}(1 - z^{-k}F_k(z^{-1}))u(t+k \mid t)$$

and using (2.3) the following expression is obtained for the k-step ahead predictor

$$\hat{y}(t+k \mid t) = F_k(z^{-1})y(t) + E_k(z^{-1})B(z^{-1}) \triangle u(t+k \mid t)$$

In the particular case of ARIMA the constant disturbance

$$n(t) = \frac{e(t)}{1 - z^{-1}}$$

can be included whose best predictions will be $\hat{n}(t+k \mid t) = n(t)$. This disturbance model, together with the step response model is the one used on DMC and related methods.

An extension of this is the drift disturbance used in PFC

$$n(t) = \frac{e(t)}{(1 - z^{-1})^2}$$

with $\hat{n}(t+k \mid t) = n(t) + (n(t) - n(t-1))k$ being the optimum prediction. Other polynomial models of high order can likewise be used.

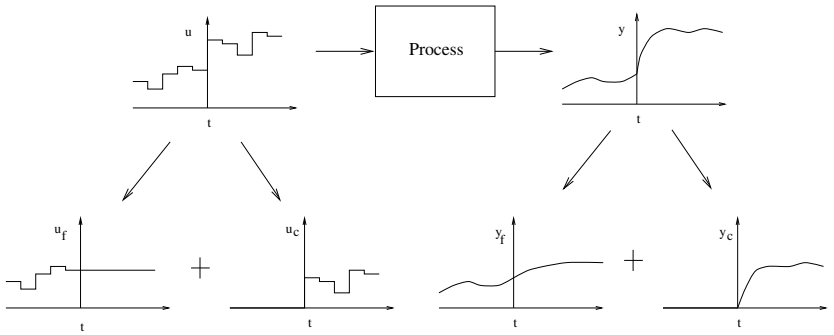


Fig. 2.2. Free and forced responses

Free and Forced Response

A typical characteristic of most MPC is the use of *free* and *forced* response concepts. The idea is to express the control sequence as the addition of the two signals:

$$u(t) = u_f(t) + u_c(t)$$

The signal $u_f(t)$ corresponds to the past inputs and is kept constant and equal to the last value of the manipulated variable in future time instants. That is,

$$u_f(t-j) = u(t-j) \text{ for } j = 1, 2, \cdots$$
$$u_f(t+j) = u(t-1) \text{ for } j = 0, 1, 2, \cdots$$

The signal $u_c(t)$ is made equal to zero in the past and equal to the next control moves in the future. That is,

$$u_c(t-j) = 0 \quad \text{for } j = 1, 2, \cdots$$
$$u_c(t+j) = u(t+j) - u(t-1) \quad \text{for } j = 0, 1, 2, \cdots$$

The prediction of the output sequence is separated into two parts, as can be seen in Figure 2.2. One of them ($y_f(t)$), the *free* response, corresponds to the prediction of the output when the process manipulated variable is made equal to $u_f(t)$, and the other, the *forced* response ($y_c(t)$), corresponds to the prediction of the process output when the control sequence is made equal to $u_c(t)$. The *free* response corresponds to the evolution of the process due to its present state, while the forced response is due to future control moves.

2.1.2 Objective Function

The various MPC algorithms propose different cost functions for obtaining the control law. The general aim is that the future output (y) on the considered horizon should follow a determined reference signal (w) and, at the

same time, the control effort ($\triangle u$) necessary for doing so should be penalized. The general expression for such an objective function will be:

$$J(N_1, N_2, N_u) = \sum_{j=N_1}^{N_2} \delta(j)[\hat{y}(t+j \mid t) - w(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\triangle u(t+j-1)]^2 \tag{2.5}$$

In some methods the second term, which considers the control effort, is not taken into account, whilst in others (UPC) the values of the control signal (not its increments) also appear directly. In the cost function it is possible to consider:

- parameters: N_1 and N_2 are the minimum and maximum prediction horizons and N_u is the control horizon, which does not necessarily have to coincide with the maximum horizon, as will be seen later. The meaning of N_1 and N_2 is rather intuitive. They mark the limits of the instants in which it is desirable for the output to follow the reference. Thus, if a high value of N_1 is taken, it is because it is of no importance if there are errors in the first instants. This will originate a smooth response of the process. Note that in processes with dead time d there is no reason for N_1 to be less than d because the output will not begin to evolve until instant $t + d$. Also if the process is nonminimum phase, this parameter will allow the first instants of inverse response to be eliminated from the objective function. The coefficients $\delta(j)$ and $\lambda(j)$ are sequences that consider the future behaviour; usually constant values or exponential sequences are considered. For example, it is possible to obtain an exponential weight of $\delta(j)$ along the horizon by using:

  $$\delta(j) = \alpha^{N_2 - j}$$

  If α is given a value between 0 and 1, the errors farthest from instant t are penalized more than those nearer to it, giving rise to smoother control with less effort. If, on the other hand, $\alpha > 1$, the first errors are more penalized, provoking tighter control. In PFC the error is only counted at certain points (coincidence points); this is easily achieved in the objective function giving value one to the elements of sequence $\delta(j)$ at said points and zero at the others. All these values can be used as tuning parameters to cover an ample scope of options, from standard control to a made-to-measure design strategy for a particular process.
- reference trajectory: One of the advantages of predictive control is that if the future evolution of the reference is known a priori, the system can react before the change has effectively been made, thus avoiding the effects of delay in the process response. The future evolution of reference $r(t + k)$ is known beforehand in many applications, such as robotics, servos or batch processes; in other applications a noticeable improvement in performance can be obtained even though the reference is constant by

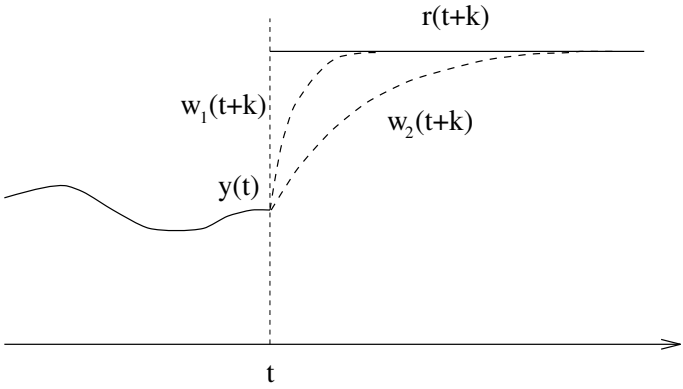


Fig. 2.3. Reference trajectory

simply knowing the instant when the value changes and getting ahead of this circumstance. In minimization (2.5), the majority of methods usually use a reference trajectory $w(t+k)$ which does not necessarily have to coincide with the real reference. It is normally a smooth approximation from the current value of the output $y(t)$ towards the known reference by means of the first-order system:

$$w(t) = y(t) \qquad w(t+k) = \alpha w(t+k-1) + (1-\alpha) r(t+k) \qquad k = 1 \dots N \quad (2.6)$$

α is a parameter between 0 and 1 (the closer to 1 the smoother the approximation) that constitutes an adjustable value that will influence the dynamic response of the system. In Figure 2.3 the form of trajectory is shown from when the reference $r(t+k)$ is constant and for two different values of α; small values of this parameter provide fast tracking (w_1), if it is increased then the reference trajectory becomes w_2, giving rise to a smoother response.

Another strategy is the one used in PFC, which is useful for variable setpoints:

$$w(t+k) = r(t+k) - \alpha^k (y(t) - r(t))$$

The reference trajectory can be used to specify closed-loop behaviour; this idea is used in GPC or EPSAC defining an auxiliary output

$$\psi(t) = P(z^{-1}) y(t)$$

where the error in the objective function is given by $\psi(t+k) - w(t+k)$. The filter $P(z^{-1})$ has unit static gain and the generation of a reference trajectory with dynamics defined by $1/P(z^{-1})$ and an initial value of that of the measured output is achieved. In [57] it is demonstrated that if a *deadbeat* control in $\psi(t)$ is achieved so that

$$\psi(t) = B(z^{-1}) w(t)$$

$B(z^{-1})$ being a determined polynomial with unit gain, the closed-loop response of the process will clearly be:

$$y(t) = \frac{B(z^{-1})}{P(z^{-1})} w(t)$$

In short, that is equivalent to placing the closed-loop poles at the zeros of design polynomial $P(z^{-1})$.

- constraints: In practice all processes are subject to constraints. The actuators have a limited field of action and a determined slew rate, as is the case of the valves, limited by the positions of totally open or closed and by the response rate. Constructive reasons, safety or environmental ones, or even the sensor scopes themselves can cause limits in the process variables such as levels in tanks, flows in piping, or maximum temperatures and pressures; moreover, the operational conditions are normally defined by the intersection of certain constraints for basically economic reasons, so that the control system will operate close to the boundaries. All of this makes the introduction of constraints in the function to be minimized necessary. Many predictive algorithms intrinsically take into account constraints (MAC, DMC) and have therefore been very successful in industry, whilst others can incorporate them *a posteriori* (GPC)[38]. Normally, bounds in the amplitude and in the slew rate of the control signal and limits in the output will be considered

$$\begin{aligned} u_{\min} \leq \quad & u(t) \quad \leq u_{\max} \quad \forall t \\ du_{\min} \leq u(t) - u(t-1) \leq & du_{\max} \quad \forall t \\ y_{\min} \leq \quad & y(t) \quad \leq y_{\max} \quad \forall t \end{aligned}$$

By adding these constraints to the objective function, the minimization becomes more complex, so that the solution cannot be obtained explicitly as in the unconstrained case.

2.1.3 Obtaining the Control Law

In order to obtain values $u(t+k \mid t)$ it is necessary to minimize the functional J of Equation (2.5). To do this the values of the predicted outputs $\hat{y}(t+k \mid t)$ are calculated as a function of past values of inputs and outputs and future control signals, making use of the model chosen and substituted in the cost function, obtaining an expression whose minimization leads to the looked-for values. An analytical solution can be obtained for the quadratic criterion if the model is linear and there are not constraints, otherwise an iterative method of optimization should be used. Whatever the method, obtaining the solution is not easy because there will be $N_2 - N_1 + 1$ independent variables, a value which can be high (on the order of 10 to 30). In order to reduce this degree of freedom a certain structure may be imposed on the control law.

Furthermore, it has been found [105] that this structuralizing of the control law produces an improvement in robustness and in the general behaviour of the system, basically due to the fact that allowing the free evolution of the manipulated variables (without being structured) may lead to undesirable high-frequency control signals and at the worst to instability. This control law structure is sometimes imposed by the use of the control horizon concept (N_u) used in DMC, GPC, EPSAC and EHAC, that consists of considering that after a certain interval $N_u < N_2$ there is no variation in the proposed control signals, that is:

$$\triangle u(t+j-1) = 0 \qquad j > N_u$$

which is equivalent to giving infinite weights to the changes in the control from a certain instant. The extreme case would be to consider N_u equal to 1 with which all future actions would be equal to $u(t)$[1]. Another way of structuring the control law is by using base functions, a procedure used in PFC which consists of representing the control signal as a linear combination of certain predetermined base functions:

$$u(t+k) = \sum_{i=1}^{n} \mu_i(t) B_i(k) \tag{2.7}$$

The B_i are chosen according to the nature of the process and the reference, they are normally polynomial type

$$B_0 = 1 \qquad B_1 = k \qquad B_2 = k^2 \ldots$$

As has been indicated previously, an explicit solution does not exist in the presence of constraints, so that quadratic programming methods have to be used (these methods will be studied in Chapter 7). However, an explicit solution does exist for certain types of constraints, for example, when the condition that the output attains the reference value at a determined instant is imposed, this method is used in Constrained Receding Horizon Predictive Control (CRHPC) [61], which is very similar to GPC and which guarantees stability results.

2.2 Review of Some MPC Algorithms

Some of the most popular methods will now be reviewed in order to demonstrate their most outstanding characteristics. Comparative studies can be found in [73],[108], [113] and [170]. The methods considered to be most representative, DMC, MAC, GPC, PFC, EPSAC and EHAC will briefly be dealt with. Some of them will be studied in greater detail in following chapters. Chapter 3 is devoted to DMC, MAC and PFC while GPC and its derivations are treated in Chapter 4.

[1] Remember that due to the receding horizon, the control signal is recalculated in the following sample.

Dynamic Matrix Control

Dynamic Matrix Control uses the step response (2.2) to model the process, only taking into account the first N terms, therefore assuming the process to be stable and without integrators. As regards the disturbances, their value will be considered to be the same as at instant t all along the horizon, that is, to be equal to the measured value of the output (y_m) minus the one estimated by the model ($\hat{y}(t \mid t)$).

$$\hat{n}(t+k \mid t) = \hat{n}(t \mid t) = y_m(t) - \hat{y}(t \mid t)$$

and therefore the predicted value of the output will be

$$\hat{y}(t+k \mid t) = \sum_{i=1}^{k} g_i \triangle u(t+k-i) + \sum_{i=k+1}^{N} g_i \triangle u(t+k-i) + \hat{n}(t+k \mid t)$$

where the first term contains the future control actions to be calculated, the second contains past values of the control actions and is therefore known, and the last represents the disturbances. The cost function may consider future errors only, or it can include the control effort, in which case it presents the generic form (2.5). One of the characteristics of this method that makes it very popular in the industry is the addition of constraints, in such a way that equations of the form

$$\sum_{i=1}^{N} C_{yi}^{j} \hat{y}(t+k \mid t) + C_{ui}^{j} u(t+k-i) + c^{j} \leq 0 \qquad j = 1 \ldots N_c$$

must be added to the minimization. Optimization (numerical because of the presence of constraints) is carried out at each sampling instant and the value of $u(t)$ is sent to the process as is normally done in all MPC methods. The inconveniences of this method are, on one hand, the size of the process model required and, on the other hand, the inability to work with unstable processes.

Model Algorithmic Control

Also known as Model Predictive Heuristic Control, Model Algorithmic Control is marketed under the name of IDCOM (Identification-Command). It is very similar to the previous method with a few differences. Firstly, it uses an impulse response model (2.1) valid only for stable processes, in which the value of $u(t)$ appears instead of $\triangle u(t)$. Furthermore, it makes no use of the control horizon concept so that in the calculations as many control signals as future outputs appear. It introduces a reference trajectory as a first-order system which evolves from the actual output to the setpoint according to a determined time constant, following Expression (2.6). The variance of the error between this trajectory and the output is what one aims at minimizing

in the objective function. The disturbances can be treated as in DMC or their estimations can be carried out by the following recursive expression:

$$\hat{n}(t+k \mid t) = \alpha \hat{n}(t+k-1 \mid t) + (1-\alpha)(y_m(t) - \hat{y}(t \mid t))$$

with $\hat{n}(t \mid t) = 0$. α is an adjustable parameter $(0 \leq \alpha < 1)$ closely related to the response time, the bandwidth and the robustness of the closed-loop system [73]. It also takes into account constraints in the actuators and in the internal variables or secondary outputs. Various algorithms can be used for optimizing in the presence of constraints, from the ones presented initially by Richalet *et al.* that can also be used for identifying the impulse response, to others that are shown in [187].

Predictive Functional Control

This controller was developed by Richalet [178] for the case of fast processes. It uses a state space model of the process and allows for nonlinear and unstable linear internal models. Nonlinear dynamics can be entered in the form of a nonlinear state space model. PFC has two distinctive characteristics: the use of *coincidence points* and *basis functions*.

The concept of coincidence points is used to simplify the calculation by considering only a subset of points in the prediction horizon. The desired and the predicted future outputs are required to coincide at these points, not in the whole prediction horizon.

The controller parameterizes the control signal using a set of polynomial basis functions, as given by Equation (2.7). This allows a relatively complex input profile to be specified over a large horizon using a small number of parameters. Choosing the family of basis functions establishes many of the features of the computed input profile. These functions can be selected to follow a polynomial setpoint with no lag, an important feature for mechanical servo control applications.

The cost function to be minimized is:

$$J = \sum_{j=1}^{n_H} [\hat{y}(t+h_j) - w(t+h_j)]^2$$

where $w(t+j)$ is usually a first-order approach to the known reference.

The PFC algorithm can also accommodate maximum and minimum input acceleration constraints which are useful in mechanical servo control applications.

Extended Prediction Self Adaptive Control

The implementation of EPSAC is different to the previous methods. For predicting, the process is modelled by the transfer function

$$A(z^{-1})y(t) = B(z^{-1})u(t-d) + v(t)$$

where d is the delay and $v(t)$ the disturbance. The model can be extended by a term $D(z^{-1})d(t)$, with $d(t)$ being a measurable disturbance in order to include *feedforward* effect. Using this method the prediction is obtained as shown in [108]. One characteristic of the method is that the control law structure is very simple, as it is reduced to considering that the control signal is going to stay constant from instant t, that is, $\triangle u(t+k) = 0$ for $k > 0$. In short, the control horizon is reduced to 1 and therefore the calculation is reduced to one single value: $u(t)$. To obtain this value a cost function is used of the form:

$$\sum_{k=d}^{N} \gamma(k)[w(t+k) - P(z^{-1})\hat{y}(t+k \mid t)]^2$$

where $P(z^{-1})$ is a design polynomial with unit static gain and factor $\gamma(k)$ being a weighting sequence, similar to those appearing in (2.5). The control signal can be calculated analytically (which is an advantage over the previous methods) in the form:

$$u(t) = \frac{\sum_{k=d}^{N} h_k \gamma(k)[w(t+k) - P(z^{-1})\hat{y}(t+k \mid t)]}{\sum_{k=d}^{N} \gamma(k) h_k^2}$$

where h_k is the discrete impulse response of the system.

Extended Horizon Adaptive Control

This formulation considers the process modelled by its transfer function without taking a model of the disturbances into account:

$$A(z^{-1})y(t) = B(z^{-1})u(t-d)$$

It aims at minimizing the discrepancy between the model and the reference at instant $t+N$: $\hat{y}(t+N \mid t) - w(t+N)$, with $N \geq d$. The solution to this problem is not unique (unless $N = d$)[207]; a possible strategy is to consider that the control horizon is 1, that is,

$$\triangle u(t+k-1) = 0 \qquad 1 < k \leq N-d$$

or to minimize the control effort:

$$J = \sum_{k=0}^{N-d} u^2(t+k)$$

There is an incremental version of EHAC that allows the disturbances in the load to be dealt with easily; it consists of considering

$$J = \sum_{k=0}^{N-d} \triangle u^2(t+k)$$

In this formulation a predictor of N steps is used as follows

$$\hat{y}(t+N \mid t) = y(t) + F(z^{-1}) \triangle y(t) + where E(z^{-1}) B(z^{-1}) \triangle u(t+N-d)$$

$E(z^{-1})$ and $F(z^{-1})$ are polynomials satisfying the equation

$$(1-z^{-1}) = A(z^{-1})E(z^{-1})(1-z^{-1}) + z^{-N}F(z^{-1})(1-z^{-1})$$

with the degree of E being equal to $N-1$. One advantage of this method is that a simple explicit solution can easily be obtained, resulting in

$$u(t) = u(t-1) + \frac{\alpha_0(w(t+N) - \hat{y}(t+N \mid t))}{\sum_{k=0}^{N-d} \alpha_i^2}$$

where α_k is the coefficient corresponding to $\triangle u(t+k)$ in the prediction equation. Thus the control law only depends on the process parameters and can therefore easily be made self-tuning if it has an online identifier. As can be seen the only parameter of adjustment is the horizon of prediction N, which simplifies its use but provides little freedom for the design. One sees that the reference trajectory cannot be used because the error is only considered at one instant $(t+N)$, neither is it possible to ponder the control efforts at each point, so that certain frequencies in the performance cannot be eliminated.

Generalized Predictive Control

The output predictions of the Generalized Predictive Controller are based on using a CARIMA model:

$$A(z^{-1})y(t) = B(z^{-1})z^{-d}\, u(t-1) + C(z^{-1})\frac{e(t)}{\triangle}$$

where the unmeasurable disturbance is given by a white noise coloured by $C(z^{-1})$. As its true value is difficult to know, this polynomial can be used for optimal disturbance rejection, although its role in robustness enhancement is more convincing.

The derivation of the optimal prediction is done by solving a Diophantine equation whose solution can be found by an efficient recursive algorithm.

This algorithm, as with all algorithms using transfer function models, can easily be implemented in an adaptive mode using an online estimation algorithm such as recursive least squares.

GPC uses a quadratic cost function of the form:

$$J(N_1, N_2, N_u) = \sum_{j=N_1}^{N_2} \delta(j)[\hat{y}(t+j \mid t) - w(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\triangle u(t+j-1)]^2$$

where the weighting sequences $\delta(j)$ and $\lambda(j)$ are usually chosen constant or exponentially increasing and the reference trajectory $w(t+j)$ can be generated by a simple recursion which starts at the current output and tends exponentially to the setpoint.

The theoretical basis of the GPC algorithm has been widely studied, and it has been shown [57] that, for limiting cases of parameter choices, this algorithm is stable and also that well-known controllers such as mean level and deadbeat control are inherent in the GPC structure.

2.3 State Space Formulation

State space models can be used to formulate the predictive control problem. The main theoretical results of MPC related to stability come from a state space formulation, which can be used for both monovariable and multivariable processes and can easily be extended to nonlinear processes. The following equations are used in the linear case to capture process dynamics:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{2.8}$$

In the single-input single-output (SISO) case, $y(t)$ and $u(t)$ are scalars and $x(t)$ is the state vector. A multiple-input multiple-output (MIMO) process has the same description but with input vectors u of dimension m and y of dimension n. In this section, for notation simplicity, only the SISO case is considered. The MIMO case is addressed in Chapter 6 together with other formulations of MPC for MIMO processes.

An incremental state space model can also be used if the model input is the control increment $\triangle u(t)$ instead of the control signal $u(t)$. This model can be written in the general state space form taking into account that $\triangle u(t) = u(t) - u(t-1)$. The following representation is obtained combining this expression with (2.8):

$$\begin{bmatrix} x(t+1) \\ u(t) \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ u(t-1) \end{bmatrix} + \begin{bmatrix} B \\ I \end{bmatrix} \triangle u(t)$$
$$y(t) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ u(t-1) \end{bmatrix}$$

Defining a new state vector as $\overline{x}(t) = [x(t) \quad u(t-1)]^T$, the incremental model takes the general form (2.8):

$$\begin{aligned} \overline{x}(t+1) &= M\overline{x}(t) + N \triangle u(t) \\ y(t) &= Q\overline{x}(t) \end{aligned} \tag{2.9}$$

where the relationship between (M, N, Q) and the nonincremental form matrices (A, B, C) can easily be obtained by comparing (2.8) and (2.9).

In order to minimize the objective function (2.5), output predictions over the horizon must be computed. If the incremental model is used, predictions can be obtained using (2.9) recursively, resulting in:

$$\hat{y}(t+j) = QM^j\hat{x}(t) + \sum_{i=0}^{j-1} QM^{j-i-1}N \triangle u(t+i)$$

Notice that the prediction needs an unbiased estimation of the state vector $x(t)$. If the state vector is not accessible an observer must be included, which calculates the estimation by means of

$$\hat{x}(t \mid t) = \hat{x}(t \mid t-1) + K(y_m(t) - y(t \mid t-1))$$

where $y_m(t)$ is the measured output. If the plant is subject to white noise disturbances affecting the process and the output with known covariance matrices, the observer becomes a Kalman filter [11] and the gain K is calculated solving a Riccati equation.

Now, the predictions along the horizon are given by

$$\mathbf{y} = \begin{bmatrix} \hat{y}(t+1|t) \\ \hat{y}(t+2|t) \\ \vdots \\ \hat{y}(t+N_2|t) \end{bmatrix} = \begin{bmatrix} QM\hat{x}(t) + QN \triangle u(t) \\ QM^2\hat{x}(t) + \sum_{i=0}^{1} QM^{1-i}N \triangle u(t+i) \\ \vdots \\ QM^{N_2}\hat{x}(t) + \sum_{i=0}^{N_2-1} QM^{N_2-1-i}N \triangle u(t+i) \end{bmatrix}$$

which can be expressed in vector form as

$$\mathbf{y} = \mathbf{F}\hat{x}(t) + \mathbf{H}\mathbf{u} \tag{2.10}$$

where $\mathbf{u} = [\triangle u(t)\ \triangle u(t+1)\ \ldots\ \triangle u(t+N_u-1)]^T$ is the vector of future control increments, $\mathbf{H}$ is a block lower triangular matrix with its nonnull elements defined by $\mathbf{H}_{ij} = QM^{i-j}N$ and matrix $\mathbf{F}$ is defined as

$$\mathbf{F} = \begin{bmatrix} QM \\ QM^2 \\ \vdots \\ QM^{N_2} \end{bmatrix}$$

Notice that (2.10) is composed of two terms: the first depends on the current state and therefore is known at instant t, while the second depends on

the vector of future control actions, which is the decision variable that must be calculated. The control sequence $\mathbf{u}$ is calculated minimizing the objective function (2.5), that (in the case of $\delta(j) = 1$ and $\lambda(j) = \lambda$) can be written as:

$$J = (\mathbf{Hu} + \mathbf{F}\hat{x}(t) - \mathbf{w})^T(\mathbf{Hu} + \mathbf{F}\hat{x}(t) - \mathbf{w}) + \lambda \mathbf{u}^T\mathbf{u}$$

If the are no constraints, an analytical solution exists that provides the optimum as:

$$\mathbf{u} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T(\mathbf{w} - \mathbf{F}\hat{x}(t))$$

As a receding horizon strategy is used, only the first element of the control sequence, $\triangle u(t)$, is sent to the plant and all the computation is repeated at the next sampling time. Notice that a state observer is needed, since the control law depends on $\hat{x}(t)$.

It must be noted that when the control and the maximum prediction horizons approach infinity and there are no constraints, the predictive controller becomes the well-known linear quadratic regulator (LQR) problem (see Appendix B). The optimal control sequence is generated by a static state feedback law where the feedback gain matrix is computed via the solution of an algebraic Riccati equation. This equivalence allows the theoretical study of MPC problems based on results coming from the optimal control field, as in the case of closed-loop stability (see Section 9.5).

If the state space model of Equation (2.8) is used, the predictions are computed in a slightly different manner, as shown by Maciejowski [131]. Now

$$\mathbf{y} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{N_2} \end{bmatrix} \hat{\mathbf{x}}(t) + \begin{bmatrix} CB \\ CA^2B \\ \vdots \\ \sum_{i=0}^{N_2-1} CA^iB \end{bmatrix} u(t-1)$$
$$+ \begin{bmatrix} B & \dots & 0 \\ C(AB+B) & \dots & 0 \\ \vdots & \ddots & \vdots \\ \sum_{i=0}^{N_2-1} CA^iB & \dots & \sum_{i=0}^{N_2-N_u} CA^iB \end{bmatrix} \mathbf{u}$$

which can be expressed in vector form as

$$\mathbf{y} = \mathbf{\Psi}\hat{x}(t) + \mathbf{\Upsilon}u(t-1) + \mathbf{\Theta u}$$

Notice that a new term has appeared that depends on $u(t-1)$ and does not affect the optimization since it does not depend on the decision variable $\mathbf{u}$. Therefore, the control action is calculated as

$$\mathbf{u} = (\mathbf{\Theta}^T\mathbf{\Theta} + \lambda I)^{-1}\mathbf{\Theta}^T(\mathbf{w} - \mathbf{\Psi}\hat{x}(t) - \mathbf{\Upsilon}u(t-1))$$

Whatever kind of model is used, the control law is a static state feedback law that needs a state observer. In the case where constraints must be taken into account, the solution must be obtained by a Quadratic Programming (QP) algorithm, as will be studied in Chapter 7.

3

Commercial Model Predictive Control Schemes

As has been shown in previous chapters, there is a wide family of predictive controllers, each member of which is defined by the choice of the common elements such as the prediction model, the objective function and obtaining the control law.

This chapter is dedicated to an overview of some MPC algorithms widely used in industry. The first two belong to a major category of predictive control approaches, those employing convolutional models, also called non-parametric methods. These approaches are based on step response or impulse response models; the most representative formulations are Dynamic Matrix Control (DMC) and Model Algorithmic Control (MAC). The third MPC algorithm presented in this chapter is Predictive Functional Control (PFC), which uses a set of basis functions to form the future control sequence.

It should be clear that the descriptions given here are necessarily incomplete, since only the general characteristics of each method are presented and each controller has proprietary features which are not known.

3.1 Dynamic Matrix Control

DMC was developed at the end of the seventies by Cutler and Ramaker [62] of Shell Oil Co. and has been widely accepted in the industrial world, mainly by petrochemical industries [170].

Nowadays DMC is something more than an algorithm, and part of its success is due to the fact that the commercial product covers topics such as model identification and global plant optimization. In this section only the *standard* algorithm is analyzed, without addressing technical details such as software and hardware compatibilities, user interface requirements, personnel training or configuration and maintenance issues. The great success of DMC in industry comes from its ability to deal with multivariable processes. In this chapter, only the Single-Input Single-Output (SISO) case is addressed,

E. F. Camacho et al., *Model Predictive Control*

leaving the Multiple-Input Multiple-Output (MIMO) case for Section 6.5. Fundamentals of this controller can be more easily understood in the SISO; the extension of the method to MIMO plants is basically a matter of notation.

3.1.1 Prediction

The process model employed in this formulation is the step response of the plant, while the disturbance is considered to be constant along the horizon. The procedure to obtain the predictions is as follows.

As a step response model is employed:

$$y(t) = \sum_{i=1}^{\infty} g_i \triangle u(t-i)$$

the predicted values along the horizon will be:

$$\hat{y}(t+k \mid t) = \sum_{i=1}^{\infty} g_i \triangle u(t+k-i) + \hat{n}(t+k \mid t) =$$

$$\sum_{i=1}^{k} g_i \triangle u(t+k-i) + \sum_{i=k+1}^{\infty} g_i \triangle u(t+k-i) + \hat{n}(t+k \mid t)$$

Disturbances are considered to be constant, that is, $\hat{n}(t+k \mid t) = \hat{n}(t \mid t) = y_m(t) - \hat{y}(t \mid t)$. Then it can be written that:

$$\hat{y}(t+k \mid t) = \sum_{i=1}^{k} g_i \triangle u(t+k-i) + \sum_{i=k+1}^{\infty} g_i \triangle u(t+k-i) + y_m(t)$$

$$-\sum_{i=1}^{\infty} g_i \triangle u(t-i) = \sum_{i=1}^{k} g_i \triangle u(t+k-i) + f(t+k)$$

where $f(t+k)$ is the free response of the system, that is, the part of the response that does not depend on the future control actions, and is given by:

$$f(t+k) = y_m(t) + \sum_{i=1}^{\infty} (g_{k+i} - g_i) \triangle u(t-i) \tag{3.1}$$

If the process is asymptotically stable, the coefficients g_i of the step response tend to a constant value after N sampling periods, so it can be considered that

$$g_{k+i} - g_i \approx 0, \qquad i > N$$

and therefore the free response can be computed as:

$$f(t+k) = y_m(t) + \sum_{i=1}^{N} (g_{k+i} - g_i) \triangle u(t-i)$$

Notice that if the process is not asymptotically stable, then N does not exist and $f(t+k)$ cannot be computed (although a generalization exists in the case of the instability being produced by pure integrators).

Now the predictions can be computed along the prediction horizon ($k = 1, \ldots, p$), considering m control actions:

$$\begin{aligned}
\hat{y}(t+1 \mid t) &= g_1 \triangle u(t) + f(t+1) \\
\hat{y}(t+2 \mid t) &= g_2 \triangle u(t) + g_1 \triangle u(t+1) + f(t+2) \\
&\vdots \\
\hat{y}(t+p \mid t) &= \sum_{i=p-m+1}^{p} g_i \triangle u(t+p-i) + f(t+p)
\end{aligned}$$

Defining the system's *dynamic matrix* $\mathbf{G}$ as

$$\mathbf{G} = \begin{bmatrix}
g_1 & 0 & \cdots & 0 \\
g_2 & g_1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
g_m & g_{m-1} & \cdots & g_1 \\
\vdots & \vdots & \ddots & \vdots \\
g_p & g_{p-1} & \cdots & g_{p-m+1}
\end{bmatrix}$$

it can be written that

$$\hat{\mathbf{y}} = \mathbf{G}\mathbf{u} + \mathbf{f} \tag{3.2}$$

Observe that $\mathbf{G}$ is made up of m (the control horizon) columns of the system's step response appropriately shifted down in order. $\hat{\mathbf{y}}$ is a p-dimensional vector containing the system predictions along the horizon, $\mathbf{u}$ represents the m-dimensional vector of control increments and $\mathbf{f}$ is the free response vector.

This is the expression that relates the future outputs with the control increments, so it will be used to calculate the necessary action to achieve a specific system behaviour.

Notice that $\mathbf{f}$ depends on the state vector $x(t)$, which in this case is given by $x(t)^T = [y_m(t)\ u(t-1)\ u(t-2) \ldots\ u(t-N-1)]$ and can be expressed as $\mathbf{f} = \mathbf{F}x(t)$, and consequently the prediction can be written as:

$$\hat{\mathbf{y}} = \mathbf{G}\mathbf{u} + \mathbf{F}x(t)$$

3.1.2 Measurable Disturbances

Measurable disturbances can easily be added to the prediction equations, since they can be treated as system inputs. Expression (3.2) can be used to calculate the predicted disturbances

$$\hat{\mathbf{y}}_d = \mathbf{D}\,\mathbf{d} + \mathbf{f}_d$$

where $\hat{\mathbf{y}}_d$ is the contribution of the measurable disturbance to the system output, $\mathbf{D}$ is a matrix similar to $\mathbf{G}$ containing the coefficients of the system response to a step in the disturbance, $\mathbf{d}$ is the vector of disturbance increment and $\mathbf{f}_d$ is the part of the response that does not depend on the disturbance.

In the most general case of measurable and nonmeasurable disturbances, the complete free response of the system (the fraction of the output that does not depend on the manipulated variable) can be considered as the sum of four effects: the response to the input $u(t)$, to the measurable disturbance $d(t)$, to the nonmeasurable disturbance and to the actual process state:

$$\mathbf{f} = \mathbf{f}_u + \mathbf{D}\,\mathbf{d} + \mathbf{f}_d + \mathbf{f}_n$$

Therefore the prediction can be computed by the general known expression:

$$\hat{\mathbf{y}} = \mathbf{G}\mathbf{u} + \mathbf{f}$$

3.1.3 Control Algorithm

The industrial success of DMC has mainly come from its application to high-dimension multivariable systems with the use of constraints. This section describes the control algorithm starting from the simpler case of a monovariable system without constraints; later it is extended to the general multivariable and constrained cases.

The objective of a DMC controller is to drive the output as close to the setpoint as possible in a least-squares sense with the possibility of the inclusion of a penalty term on the input moves. Therefore, the manipulated variables are selected to minimize a quadratic objective that can consider the minimization of future errors alone

$$J = \sum_{j=1}^{p} [\hat{y}(t+j \mid t) - w(t+j)]^2$$

or it can include the control effort, in which case it presents the generic form

$$J = \sum_{j=1}^{p} [\hat{y}(t+j \mid t) - w(t+j)]^2 + \sum_{j=1}^{m} \lambda [\triangle u(t+j-1)]^2$$

If there are no constraints, the solution to the minimization of the cost function $J = \mathbf{e}\mathbf{e}^T + \lambda \mathbf{u}\mathbf{u}^T$, where $\mathbf{e}$ is the vector of future errors along the

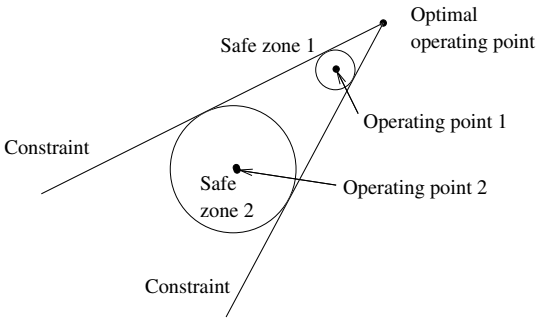


Fig. 3.1. Economic operating point of a typical process

prediction horizon and $\mathbf{u}$ is the vector composed of the future control increments $\triangle u(t), \ldots, \triangle u(t+m)$, can be obtained analytically by computing the derivative of J and making it equal to 0, which provides the general result:

$$\mathbf{u} = (\mathbf{G}^T\mathbf{G} + \lambda I)^{-1}\mathbf{G}^T(\mathbf{w} - \mathbf{f})$$

Remember that, as in all predictive strategies, only the first element of vector $\mathbf{u}$ ($\triangle u(t)$) is really sent to the plant. It is not advisable to implement the entire sequence over the next m intervals in automatic succession. This is because is impossible to perfectly estimate the disturbance vector, and therefore it is impossible to anticipate precisely the unavoidable disturbances that cause the actual output to differ from the predictions used to compute the sequence of control actions. Furthermore, the setpoint can also change over the next m intervals.

The Constrained Problem

Though computationally more involved than simpler algorithms, the flexible constraint-handling capabilities of the method (and MPC in general) are very attractive for practical applications, since the economic operating point of a typical process unit often lies at the intersection of constraints [168], as shown in Figure 3.1. It can be seen that, due to safety reasons, it is necessary to keep a safe zone around the operating point, since the effect of perturbations can make the process violate constraints. This zone can be reduced, and therefore the economic profit improved, if the controller is able to handle constraints (operating point 1).

Constraints in both inputs and outputs can be posed in such a way that equations of the form

$$\sum_{i=1}^{N} C_{yi}^{j}\hat{y}(t+k \mid t) + C_{ui}^{j}u(t+k-i) + c^{j} \leq 0 \qquad j = 1 \ldots N_c$$

must be added to the minimization. As future projected outputs can be related directly back to the control increment vector through the dynamic matrix, all input and output constraints can be collected into a matrix inequality involving the input vector, $\mathbf{Ru} \leq \mathbf{c}$ (for further details see Chapter 7). Therefore the problem takes the form of a standard Quadratic Programming (QP) formulation. The optimization is now numerical because of the presence of constraints and is carried out by means of standard commercial optimization QP code at each sampling instant, and then the value of $u(t)$ is sent to the process, as is normally done in all MPC methods. In this case the method is known as QDMC, due to the Quadratic Programming algorithm employed.

3.2 Model Algorithmic Control

Maybe the simplest and most intuitive formulation of Predictive Control is the one based on the key ideas of Richalet *et al.* [182], known as MAC and Model Predictive Heuristic Control (MPHC), whose software is called IDCOM (Identification-Command). This method is very similar to DMC with a few differences. It makes use of a truncated step response of the process and provides a simple explicit solution in the absence of constraints. This method has clearly been accepted by practitioners and is extensively used in many control applications [70] where most of its success is due to the process model used. It is known that transfer function models can give results with large errors when there is a mismatch in the model order. On the other hand, the impulse response representation is a good choice, since the identification of impulse responses is relatively simple.

3.2.1 Process Model and Prediction

The system output at instant t is related to the inputs by the coefficients of the truncated impulse response as follows:

$$y(t) = \sum_{j=1}^{N} h_j u(t-j) = H(z^{-1})u(t)$$

This model predicts that the output at a given time depends on a linear combination of past input values; the weights h_i are the impulse response coefficients. As the response is truncated to N elements, the system is assumed to be stable and causal. Using this internal model, a k-step ahead predictor can be written as

$$\hat{y}(t+k \mid t) = \sum_{j=1}^{N} h_j u(t+k-j) + \hat{n}(t+k \mid t)$$

where the sum can be divided into two terms

$$f_r(t+k) = \sum_{j=k+1}^{N} h_j u(t+k-j) \qquad f_o(t+k) = \sum_{j=1}^{k} h_j u(t+k-j)$$

such that f_r represents the free response, being the expected value of $y(t+j)$ assuming zero future control actions, and f_o is the forced response, that is, the additional component of output response due to the proposed set of future control actions. It is now assumed that the disturbances will remain constant in the future with the same value as at instant t, that is, $\hat{n}(t+k \mid t) = \hat{n}(t \mid t)$, which is the measured output minus the output predicted by the nominal model:

$$\hat{n}(t+k \mid t) = \hat{n}(t \mid t) = y(t) - \sum_{j=1}^{N} h_j u(t-j)$$

Then the prediction is given by:

$$\hat{y}(t+k \mid t) = f_r + f_o + \hat{n}(t \mid t)$$

If M is the horizon and $\mathbf{u}_+$ the vector of proposed control actions (not increments), $\mathbf{u}_-$ of past control actions, $\mathbf{y}$ the predicted outputs, $\mathbf{n}$ the disturbances, and the reference vector $\mathbf{w}$ is a smooth approach to the current setpoint

$$\mathbf{u}_+ = \begin{bmatrix} u(t) \\ u(t+1) \\ \vdots \\ u(t+M-1) \end{bmatrix} \qquad \mathbf{u}_- = \begin{bmatrix} u(t-N+1) \\ u(t-N+2) \\ \vdots \\ u(t-1) \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} \hat{y}(t+1) \\ \hat{y}(t+2) \\ \vdots \\ \hat{y}(t+M) \end{bmatrix}$$

$$\mathbf{n} = \begin{bmatrix} \hat{n}(t+1) \\ \hat{n}(t+2) \\ \vdots \\ \hat{n}(t+M) \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w(t+1) \\ w(t+2) \\ \vdots \\ w(t+M) \end{bmatrix}$$

and defining the matrices

$$\mathbf{H}_1 = \begin{bmatrix} h_1 & 0 & \cdots & 0 \\ h_2 & h_1 & \cdots & 0 \\ \cdots & \cdots & \ddots & \cdots \\ h_M & h_{M-1} & \cdots & h_1 \end{bmatrix} \mathbf{H}_2 = \begin{bmatrix} h_N & \cdots & h_i & \cdots & h_2 \\ 0 & \cdots & h_j & \cdots & h_3 \\ \cdots & \ddots & \cdots & \cdots & \cdots \\ 0 & \cdots & h_N & \cdots & h_{M+1} \end{bmatrix}$$

the predictor can be written as

$$\mathbf{y} = \mathbf{H}_1\, \mathbf{u}_+ + \mathbf{H}_2\, \mathbf{u}_- + \mathbf{n}$$

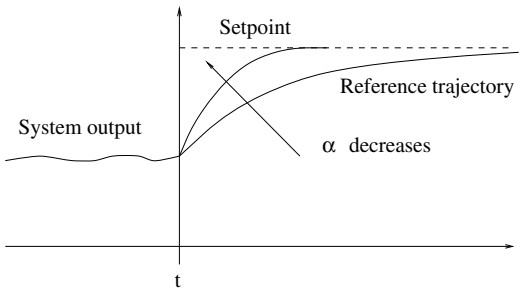


Fig. 3.2. Influence of α on the reference tracking

3.2.2 Control Law

The primary objective of the controller is to determine the sequence of control moves that will minimize the sum of the squared deviations of the predicted output from the reference trajectory.

The reference trajectory used in MAC is normally a smooth approximation from the current value of the system output towards the known reference by means of a first-order system of the form:

$$w(t+k) = \alpha w(t+k-1) + (1-\alpha) r(t+k) \qquad k = 1 \ldots N, \qquad \text{with } w(t) = y(t)$$

It is important to note that the shape of the reference trajectory (which depends on the choice of α) determines the desired speed of approach to the setpoint. This is of great interest in practice because it provides a natural way to control the aggressiveness of the algorithm: increasing the time constant leads to a slower but more robust controller (see Figure 3.2). Therefore this is an important tuning parameter for this controller, and its choice is very closely linked to the robustness of the closed-loop system [159]. Parameter α is a more direct and more intuitive tuning parameter than factors such as weighting sequences or horizon lengths employed by other formulations.

The objective function minimizes the error as well as the control effort. If future errors are expressed as

$$\mathbf{e} = \mathbf{w} - \mathbf{y} = \mathbf{w} - \mathbf{H_2 u_-} - \mathbf{n} - \mathbf{H_1 u_+} = \mathbf{w} - \mathbf{f} - \mathbf{H_1 u_+}$$

where vector $\mathbf{f}$ contains the terms depending on known values (past inputs, current output and references). Then the cost function can be written as

$$J = \mathbf{e}^T \mathbf{e} + \lambda \mathbf{u_+}^T \mathbf{u_+}$$

where λ is the penalization factor for the input variable variations. If no constraints are considered, the solution can be obtained explicitly, giving:

$$\mathbf{u_+} = (\mathbf{H_1}^T \mathbf{H_1} + \lambda \mathbf{I})^{-1} \mathbf{H_1}^T (\mathbf{w} - \mathbf{f}) \tag{3.3}$$

As it is a receding horizon strategy, only the first element of this vector $u(t)$ is used, rejecting the rest and repeating the calculations at the next sampling time.

The calculation of the control law (3.3) is relatively simple compared to other formulations, although it requires the inversion of an $M \times M$ matrix. Notice that if the number of future inputs to be calculated is chosen as a value $P < M$, then this matrix is of dimension $P \times P$, since $\mathbf{H}_1$ is of dimension $M \times P$, hence reducing the necessary calculations.

The simplicity of the algorithm, as well as the possibility of including constraints, has converted this formulation into one of the most frequently used in industry nowadays.

3.3 Predictive Functional Control

The Predictive Functional Controller PFC was proposed by Richalet [178] for fast processes and is characterized by two distinctive features: the control signal is structured as a linear combination of certain predetermined *basis functions* and the concept of *coincidence points* to evaluate the cost function along the horizon.

3.3.1 Formulation

Consider the following state space model

$$\begin{aligned} x(t) &= Mx(t-1) + Nu(t-1) \\ y(t) &= Qx(t) \end{aligned}$$

as representing the process behaviour. The prediction is obtained adding an autocompensation term calculated as a function of the observed differences between the model and the past outputs:

$$\hat{y}(t+k \mid t) = y(t+k) + \hat{e}(t+k \mid t)$$

The predicted plant-model error is assumed to have the form

$$\hat{e}(t+k \mid t) = y_m(t) - \hat{y}(t \mid t-1) + \sum_{j=1}^{r} e_j k^j$$

where $y_m(t)$ is the output measured value and coefficients e_j are obtained by a least-squares fit to previous errors.

The future control signal is structured as a linear combination of the basis functions B_i, which are chosen according to the nature of the process and the reference:

$$u(t+k) = \sum_{i=1}^{n_B} \mu_i(t) B_i(k) \tag{3.4}$$

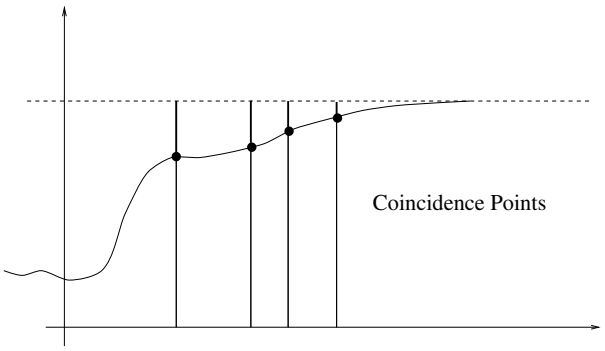


Fig. 3.3. Coincidence points

Normally these functions are polynomial type: steps ($B_1(k) = 1$), ramps ($B_2(k) = k$), or parabolas ($B_3(k) = k^2$), as the majority of references can be expressed as combinations of these functions. With this strategy a complex input profile can be specified using a small number of unknown parameters.

The choice of basis functions defines the input profile and can assure a predetermined behaviour (smooth signal, for instance). This can result in an advantage when controlling nonlinear systems. An important feature for mechanical servo applications is that if a polynomial basis is chosen, then the order can be selected to follow a polynomial setpoint.

The cost function to be minimized is:

$$J = \sum_{j=1}^{n_H} [\hat{y}(t + h_j) - w(t + h_j)]^2$$

where $w(t + j)$ is usually a first-order approach to the known reference, as in (2.6) or

$$w(t + k) = r(t + k) - \alpha^k (r(t) - y(t))$$

In order to smooth the control signal, a quadratic factor of the form $\lambda[\triangle u(k)]^2$ can be added to the cost function.

The predicted error is not considered all along the horizon, only in certain instants h_j, $j = 1, \ldots, n_H$ called *coincidence points* (see figure 3.3). These points can be considered tuning parameters and must be chosen taking into account their influence on the stability and robustness of the control system. Their number must be at least equal to the selected number of basis functions.

Calculation of the Control Law

The calculation of the control law implies computing the values of $\mu_i(t)$ of Equation (3.4). Notice that these coefficients are chosen to be optimal at each instant t and therefore they are different at each step.

In the case of SISO processes without constraints, the control law can be obtained as follows. First, the output is decomposed into free and forced outputs, and the structure of the control signal is employed to give

$$y(t+k) = QM^k x(t) + \sum_{i=1}^{n_B} y_{B_i}(k)\mu_i(t)$$

where y_{B_i} is the system response to the basis function B_i.

Now the cost function can be written as

$$J = \sum_{j=1}^{n_H} [\hat{y}(t+h_j) - w(t+h_j)]^2 = \sum_{j=1}^{n_H} [\mathbf{y}_B(h_j)\,\mu - d(t+h_j)]^2$$

where

$$\mu = [\mu_1(t) \dots \mu_{nB}(t)]^T$$
$$\mathbf{y}_B(h_j) = [y_{B_1}(h_j) \dots y_{B_{n_B}}(h_j)]$$
$$d(t+h_j) = \omega(t+h_j) - QM^j x(t) - e(t+h_j)$$

The cost function can be written in vector form defining $\mathbf{d} = [d(t+h_1) \dots d(t+h_{n_H})]^T$ as the term which contains values that are known at time t and where $\mathbf{Y}_B$ is a matrix whose rows are the vectors $\mathbf{y}_B$ at the coincidence points h_j, $j = 1, \dots n_h$, giving:

$$J = (\mathbf{Y}_B\mu - \mathbf{d})^T(\mathbf{Y}_B\mu - \mathbf{d})$$

Minimizing J with respect to the coefficients μ:

$$\frac{\partial J}{\partial \mu} = 0 \Rightarrow \mathbf{Y}_B^T\mathbf{Y}_B\mu - \mathbf{Y}_B^T\mathbf{d} = 0$$

and therefore the vector of the weights of the basis functions is given by the solution of

$$\mathbf{Y}_B\mu = \mathbf{d}$$

Then the control signal, taking into account the receding horizon strategy, is given by:

$$u(t) = \sum_{i=1}^{n_B} \mu_i(t) B_i(0)$$

This algorithm can only be used for stable models, since the pole cancellations can lead to stability problems when unstable or high-oscillatory modes appear. In this case, a procedure that decomposes the model into two stable ones can be employed [178]. The method can be used for nonlinear processes using nonlinear state space models.

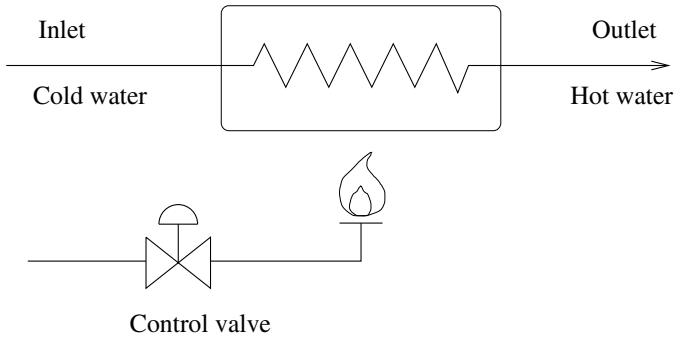


Fig. 3.4. Water heater

3.4 Case Study: A Water Heater

This example shows the design of a DMC to control the outlet temperature of a water heater. Notice that a MAC can be designed following the same steps.

Consider a water heater where the cold water is heated by means of a gas burner. The outlet temperature depends on the energy added to the water through the gas burner (see Figure 3.4). Therefore this temperature can be controlled by the valve which manipulates the gas flow to the heater.

The step response model of this process must be obtained to design the controller. The step response is obtained by applying a step in the control valve. Coefficients g_i can be obtained directly from the response shown in figure 3.5. It can be seen that the output stabilizes after 30 periods, so the model is given by

$$y(t) = \sum_{i=1}^{30} g_i \triangle u(t-i),$$

where the coefficients g_i are shown in the following table:

g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9	g_{10}
0	0	0.271	0.498	0.687	0.845	0.977	1.087	1.179	1.256
g_{11}	g_{12}	g_{13}	g_{14}	g_{15}	g_{16}	g_{17}	g_{18}	g_{19}	g_{20}
1.320	1.374	1.419	1.456	1.487	1.513	1.535	1.553	1.565	1.581
g_{21}	g_{22}	g_{23}	g_{24}	g_{25}	g_{26}	g_{27}	g_{28}	g_{29}	g_{30}
1.592	1.600	1.608	1.614	1.619	1.623	1.627	1.630	1.633	1.635

The response shown in Figure 3.5 corresponds to a system with a transfer function given by:

$$G(z) = \frac{0.2713z^{-3}}{1 - 0.8351z^{-1}}$$

Notice that although the g_i coefficients are obtained in practice from real plant tests, for this example, where the response has been generated with a simple model, the step response coefficients can easily be obtained from the transfer function by the expression

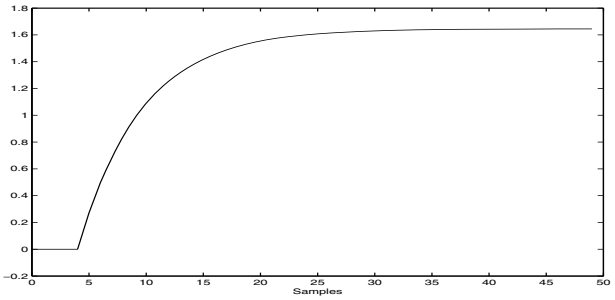


Fig. 3.5. Step response

$$g_j = -\sum_{i=1}^{j} a_i g_{j-i} + \sum_{i=0}^{j-1} b_i \qquad g_k = 0 \quad \text{for} \quad k \leq 0 \tag{3.5}$$

where a_i and b_i are the coefficients of the denominator and numerator of the discrete transfer function respectively (starting from $i = 0$).

In this example the first two coefficients of the step model are zero since the system has a dead time of two sampling periods.

Considering a prediction horizon of 10 and a control horizon of 5, the dynamic matrix is obtained from the coefficients of the step response and is given by:

$$\mathbf{G} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.271 & 0 & 0 & 0 & 0 \\ 0.498 & 0.271 & 0 & 0 & 0 \\ 0.687 & 0.498 & 0.271 & 0 & 0 \\ 0.845 & 0.687 & 0.498 & 0.271 & 0 \\ 0.977 & 0.845 & 0.687 & 0.498 & 0.271 \\ 1.087 & 0.977 & 0.845 & 0.687 & 0.498 \\ 1.179 & 1.087 & 0.977 & 0.845 & 0.687 \\ 1.256 & 1.179 & 1.087 & 0.977 & 0.845 \end{bmatrix}$$

Taking $\lambda = 1$, matrix $(\mathbf{G}^T\mathbf{G} + \lambda I)^{-1}\mathbf{G}^T$ is calculated and therefore the control law is given by the product of the first row of this matrix ($\mathbf{K}$) times the vector that contains the difference between the reference trajectory and the free response

$$\triangle u(t) = \mathbf{K}(\mathbf{w} - \mathbf{f})$$

with

$$\mathbf{K} = [0 \;\; 0 \;\; 0.1465 \;\; 0.1836 \;\; 0.1640 \;\; 0.1224 \;\; 0.0780 \;\; 0.0410 \;\; 0.0101 \;\; -0.0157]$$

where the free response is easily calculated using equation (3.1):

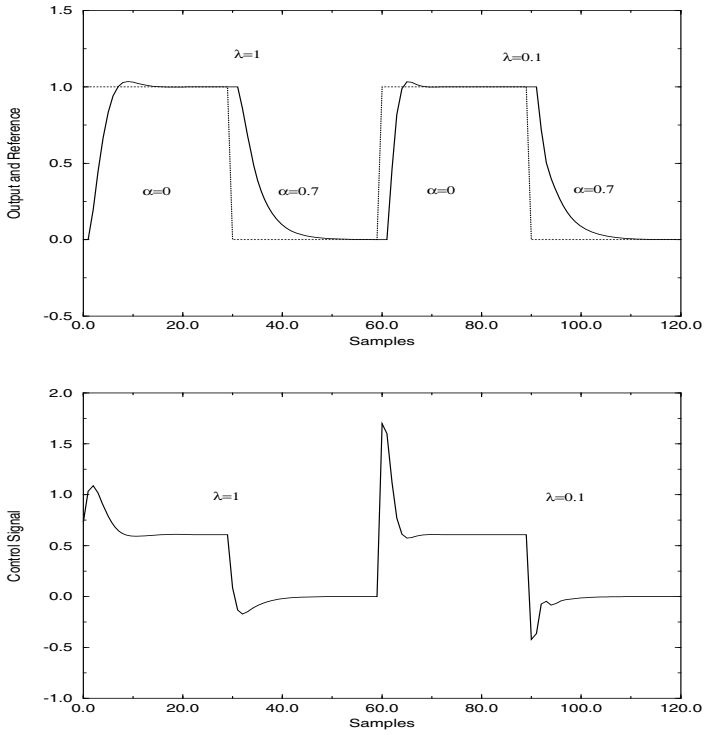


Fig. 3.6. Controller behaviour

$$f(t+k) = y_m(t) + \sum_{i=1}^{30} (g_{k+i} - g_i) \triangle u(t-i)$$

Figure 3.6 shows the system response to a change in the outlet temperature setpoint for different shapes of the control weighting factor and the reference trajectory. The first setpoint change is made with a value of $\lambda = 1$ and $\alpha = 0$. In the second change α is changed to 0.7 and later the control weighting factor is changed to 0.1 for the same values of α. It can be seen that a small value of α makes the system response faster with a slight oscillation, while a small value of λ gives bigger control actions. The combination $\lambda = 0.1$, $\alpha = 0$ provides the fastest response, but the control effort seems to be too vigorous.

The inlet temperature can become a disturbance, since any change in its value will disturb the process from its steady-state operating point. This temperature can be measured and the controller can take into account its value in order to reject its effect before it appears in the system output. That is, it can be treated by DMC as a measurable disturbance and explicitly incorporated into the formulation. To do this, a model of the effect of the inlet temperature changes on the outlet temperature can easily be obtained by a step test.

In this example the disturbance is modelled by

$$y(t) = \sum_{i=1}^{30} d_i \triangle u(t-i)$$

with

d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
0	0	0.050	0.095	0.135	0.172	0.205	0.234	0.261	0.285
d_{11}	d_{12}	d_{13}	d_{14}	d_{15}	d_{16}	d_{17}	d_{18}	d_{19}	d_{20}
0.306	0.326	0.343	0.359	0.373	0.386	0.397	0.407	0.417	0.425
d_{21}	d_{22}	d_{23}	d_{24}	d_{25}	d_{26}	d_{27}	d_{28}	d_{29}	d_{30}
0.433	0.439	0.445	0.451	0.456	0.460	0.464	0.468	0.471	0.474

which corresponds to the transfer function:

$$G(z) = \frac{0.05z^{-3}}{1 - 0.9z^{-1}}$$

Notice that the first ten d_i coefficients are used to build matrix $\mathbf{D}$ in the same way as matrix $\mathbf{G}$.

Figure 3.7 shows a simulation where a disturbance occurs from $t = 20$ to $t = 60$. In case the controller explicitly considers the measurable disturbances, it is able to reject them, since the controller starts acting when the disturbance appears, not when its effect appears in the outlet temperature. On the other hand, if the controller does not take into account the measurable disturbance, it reacts later, when the effect on the output is considerable.

3.5 Exercises

3.1. If the cost function of a predictive controller is $J = \mathbf{e}\mathbf{e}^T + \lambda\mathbf{u}\mathbf{u}^T$, with $\mathbf{e} = \mathbf{G}\mathbf{u} + \mathbf{f} - \mathbf{w}$, demonstrate that the minimum is given by $\mathbf{u} = (\mathbf{G}^T\mathbf{G} + \lambda I)^{-1}\mathbf{G}^T(\mathbf{w} - \mathbf{f})$ in the unconstrained case.

3.2. For the water heater of Section 3.4:

1. Obtain the impulse response model.
2. Create matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ for the MAC controller with $P = 3$ and $M = 5$.
3. Calculate the control law and simulate the same experiments as in the example.

3.3. Given a process described by

$$x(t+1) = \begin{bmatrix} 0.8 & 0.1 \\ 0.1 & 0.9 \end{bmatrix} x(t) + \begin{bmatrix} 0.1 \\ 0 \end{bmatrix} u(t), \quad y(t) = \begin{bmatrix} 0.1 & 0.9 \end{bmatrix} x(t)$$

simulate a PFC to move the process output from 0 to 1, with $N = 30$. Compare the effect of using one or three basis functions.

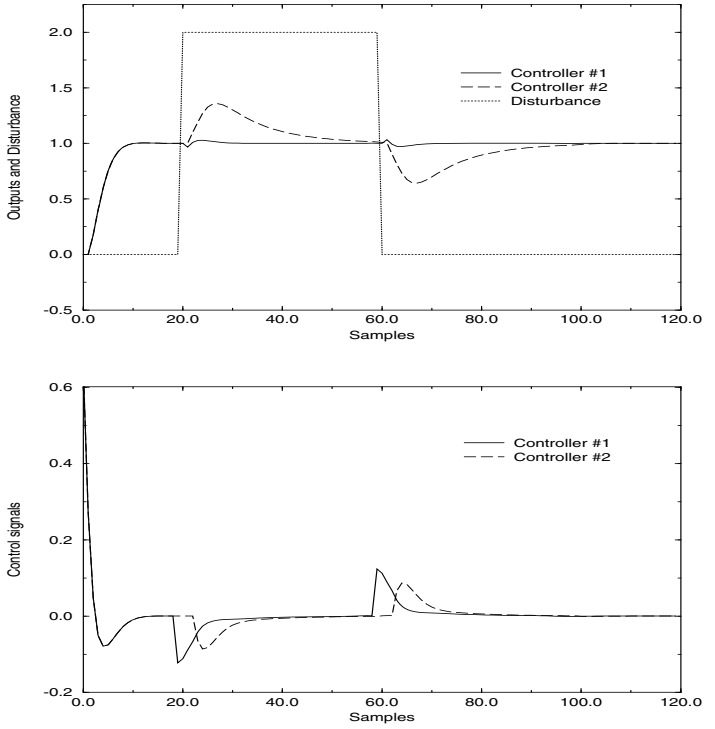


Fig. 3.7. Disturbance rejection with (Controller #1) and without (Controller #2) considering measurable disturbances

3.4. Change the number of model elements of the water heater from $N = 30$ to $N = 10$ and simulate the effect on the closed-loop behaviour.

3.5. For the same example, change the values of m, p and λ and see the results.

3.6. Obtain a state space model for the water heater, design a PFC controller and compare the results with those obtained with DMC.

3.7. Design a DMC for the process $G(s) = \frac{225(s-1)}{(s+1)(s^2+30s+225)}$. In order to do this:

1. Obtain the discrete equivalent when sampling at $T = 0.2$ second.
2. Compute the step response model by performing a unitary step at the input.
3. Calculate the control law with $N = 40$, $p = 15$, $m = 2$ and $\lambda = 1$.
4. Simulate the response to a setpoint change of 1.

4

Generalized Predictive Control

This chapter describes one of the most popular predictive control algorithms: Generalized Predictive Control (GPC). The method is developed in detail, showing the general procedure to obtain the control law and its most outstanding characteristics. The original algorithm is extended to include the cases of measurable disturbances and change in the predictor. Close derivations of this controller such as CRHPC and Stable GPC are also treated here, illustrating the way they can be implemented.

4.1 Introduction

The GPC method was proposed by Clarke *et al.* [58] and has become one of the most popular MPC methods in both industry and academia. It has been successfully implemented in many industrial applications [54], showing good performance and a certain degree of robustness. It can handle many different control problems for a wide range of plants with a reasonable number of design variables, which have to be specified by the user depending upon prior knowledge of the plant and control objectives.

The basic idea of GPC is to calculate a sequence of future control signals in such a way that it minimizes a multistage cost function defined over a prediction horizon. The index to be optimized is the expectation of a quadratic function measuring the distance between the predicted system output and some predicted reference sequence over the horizon plus a quadratic function measuring the control effort. This approach was used in [118] and [119] to obtain a generalized pole placement controller which is an extension of the well-known pole placement controllers [4], [205] and belongs to the class of extended horizon controllers.

Generalized Predictive Control has many ideas in common with the predictive controllers previously mentioned since it is based upon the same concepts but it has some differences. As will be seen, it provides an analytical

E. F. Camacho et al., *Model Predictive Control*

solution (in the absence of constraints), it can deal with unstable and non-minimum phase plants and it incorporates the concept of control horizon as well as the consideration of weighting control increments in the cost function. The general set of choices available for GPC leads to a greater variety of control objectives compared to other approaches, some of which can be considered as subsets or limiting cases of GPC.

4.2 Formulation of Generalized Predictive Control

Most single-input single-output (SISO) plants, when considering operation around a particular setpoint and after linearization, can be described by

$$A(z^{-1})y(t) = z^{-d}B(z^{-1})u(t-1) + C(z^{-1})e(t)$$

where $u(t)$ and $y(t)$ are the control and output sequences of the plant and $e(t)$ is a zero mean white noise. A, B and C are the following polynomials in the backward shift operator z^{-1}:

$$\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_{na} z^{-na} \\ B(z^{-1}) &= b_0 + b_1 z^{-1} + b_2 z^{-2} + ... + b_{nb} z^{-nb} \\ C(z^{-1}) &= 1 + c_1 z^{-1} + c_2 z^{-2} + ... + c_{nc} z^{-nc} \end{aligned}$$

where d is the dead time of the system. This model is known as a Controller Auto-Regressive Moving-Average (CARMA) model. It has been argued [58] that for many industrial applications in which disturbances are non-stationary an integrated CARMA (CARIMA) model is more appropriate. A CARIMA model is given by

$$A(z^{-1})y(t) = B(z^{-1})z^{-d}\, u(t-1) + C(z^{-1})\frac{e(t)}{\triangle} \tag{4.1}$$

with

$$\triangle = 1 - z^{-1}$$

For simplicity in the following, the C polynomial is chosen to be 1. Notice that if C^{-1} can be truncated it can be absorbed into A and B. The general case of a coloured noise will be treated later.

The Generalized Predictive Control (GPC) algorithm consists of applying a control sequence that minimizes a multistage cost function of the form

$$J(N_1, N_2, N_u) = \sum_{j=N_1}^{N_2} \delta(j)[\hat{y}(t+j \mid t) - w(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\triangle u(t+j-1)]^2 \tag{4.2}$$

where $\hat{y}(t+j \mid t)$ is an optimum j step ahead prediction of the system output on data up to time t, N_1 and N_2 are the minimum and maximum costing horizons, N_u is the control horizon, $\delta(j)$ and $\lambda(j)$ are weighting sequences and $w(t+j)$ is the future reference trajectory, which can be calculated as shown in (2.6).

The objective of predictive control is to compute the future control sequence $u(t), u(t+1), \ldots$ in such a way that the future plant output $y(t+j)$ is driven close to $w(t+j)$. This is accomplished by minimizing $J(N_1, N_2, N_u)$.

In order to optimize the cost function the optimal prediction of $y(t+j)$ for $j \geq N_1$ and $j \leq N_2$ will be obtained. Consider the following Diophantine equation:

$$1 = E_j(z^{-1})\tilde{A}(z^{-1}) + z^{-j}F_j(z^{-1}) \quad \text{with} \quad \tilde{A}(z^{-1}) = \triangle A(z^{-1}) \tag{4.3}$$

The polynomials E_j and F_j are uniquely defined with degrees $j-1$ and na, respectively. They can be obtained by dividing 1 by $\tilde{A}(z^{-1})$ until the remainder can be factorized as $z^{-j}F_j(z^{-1})$. The quotient of the division is the polynomial $E_j(z^{-1})$.

If Equation (4.1) is multiplied by $\triangle E_j\ (z^{-1})\ z^j$,

$$\begin{aligned}\tilde{A}(z^{-1})E_j(z^{-1})y(t+j) = {} & E_j(z^{-1})B(z^{-1}) \triangle u(t+j-d-1) \\ & + E_j(z^{-1})e(t+j)\end{aligned} \tag{4.4}$$

Considering (4.3), Equation (4.4) can be written as

$$(1-z^{-j}F_j(z^{-1}))y(t+j) = E_j(z^{-1})B(z^{-1}) \triangle u(t+j-d-1) + E_j(z^{-1})e(t+j)$$

which can be rewritten as:

$$y(t+j) = F_j(z^{-1})y(t) + E_j(z^{-1})B(z^{-1}) \triangle u(t+j-d-1) + E_j(z^{-1})e(t+j) \tag{4.5}$$

As the degree of polynomial $E_j(z^{-1}) = j-1$, the noise terms in Equation (4.5) are all in the future. The best prediction of $y(t+j)$ is therefore

$$\hat{y}(t+j \mid t) = G_j(z^{-1}) \triangle u(t+j-d-1) + F_j(z^{-1})y(t)$$

where $G_j(z^{-1}) = E_j(z^{-1})B(z^{-1})$.

It is very simple to show that the polynomials E_j and F_j can be obtained recursively. The recursion of the Diophantine equation has been demonstrated in [58]. A simpler demonstration is given in the following. There are other formulations of GPC not based on the recursion of the Diophantine equation [2].

Consider that polynomials E_j and F_j have been obtained by dividing 1 by $\tilde{A}(z^{-1})$ until the remainder of the division can be factorized as $z^{-j}F_j(z^{-1})$. These polynomials can be expressed as:

$$\begin{aligned}F_j(z^{-1}) &= f_{j,0} + f_{j,1}z^{-1} + \ldots + f_{j,na}z^{-na} \\ E_j(z^{-1}) &= e_{j,0} + e_{j,1}z^{-1} + \ldots + e_{j,j-1}z^{-(j-1)}\end{aligned}$$

Suppose that the same procedure is used to obtain E_{j+1} and F_{j+1}, that is, dividing 1 by $\tilde{A}(z^{-1})$ until the remainder of the division can be factorized as $z^{-(j+1)}F_{j+1}(z^{-1})$ with

$$F_{j+1}(z^{-1}) = f_{j+1,0} + f_{j+1,1}z^{-1} + \ldots + f_{j+1,na}z^{-na}$$

It is clear that only another step of the division performed to obtain the polynomials E_j and F_j has to be taken in order to obtain the polynomials E_{j+1} and F_{j+1}. The polynomial E_{j+1} will be given by

$$E_{j+1}(z^{-1}) = E_j(z^{-1}) + e_{j+1,j}z^{-j}$$

with $e_{j+1,j} = f_{j,0}$

The coefficients of polynomial F_{j+1} can then be expressed as:

$$f_{j+1,i} = f_{j,i+1} - f_{j,0}\,\tilde{a}_{i+1} \quad i = 0 \ldots na-1$$

The polynomial G_{j+1} can be obtained recursively as follows:

$$G_{j+1} = E_{j+1}B = (E_j + f_{j,0}z^{-j})B$$
$$G_{j+1} = G_j + f_{j,0}z^{-j}B$$

That is, the first j coefficient of G_{j+1} will be identical to those of G_j and the remaining coefficients will be given by:

$$g_{j+1,j+i} = g_{j,j+i} + f_{j,0}\,b_i \qquad i = 0 \ldots nb$$

To solve the GPC problem the set of control signals $u(t)$, $u(t+1)$, ..., $u(t+N)$ has to be obtained in order to optimize Expression (4.2). As the system considered has a dead time of d sampling periods, the output of the system will be influenced by signal $u(t)$ after sampling period $d+1$. The values N_1, N_2 and N_u defining the horizon can be defined by $N_1 = d+1$, $N_2 = d+N$ and $N_u = N$. Notice that there is no point in making $N_1 < d+1$ as terms added to expression (4.2) will only depend on the past control signals. On the other hand, if $N_1 > d+1$ the first points in the reference sequence, being the ones guessed with most certainty, will not be taken into account.

Now consider the following set of j ahead optimal predictions:

$$\begin{aligned}
\hat{y}(t+d+1 \mid t) &= G_{d+1} \triangle u(t) + F_{d+1}y(t) \\
\hat{y}(t+d+2 \mid t) &= G_{d+2} \triangle u(t+1) + F_{d+2}y(t) \\
&\vdots \\
\hat{y}(t+d+N \mid t) &= G_{d+N} \triangle u(t+N-1) + F_{d+N}y(t)
\end{aligned}$$

which can be written as:

$$\mathbf{y} = \mathbf{G}\mathbf{u} + \mathbf{F}(z^{-1})y(t) + \mathbf{G'}(z^{-1}) \triangle u(t-1) \tag{4.6}$$

where

$$\mathbf{y} = \begin{bmatrix} \hat{y}(t+d+1 \mid t) \\ \hat{y}(t+d+2 \mid t) \\ \vdots \\ \hat{y}(t+d+N \mid t) \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} \triangle u(t) \\ \triangle u(t+1) \\ \vdots \\ \triangle u(t+N-1) \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} g_0 & 0 & \dots & 0 \\ g_1 & g_0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ g_{N-1} & g_{N-2} & \cdots & g_0 \end{bmatrix}$$

$$\mathbf{G}'(z^{-1}) = \begin{bmatrix} (G_{d+1}(z^{-1}) - g_0)z \\ (G_{d+2}(z^{-1}) - g_0 - g_1 z^{-1})z^2 \\ \vdots \\ (G_{d+N}(z^{-1}) - g_0 - g_1 z^{-1} - \cdots - g_{N-1} z^{-(N-1)})z^N \end{bmatrix}$$

$$\mathbf{F}(z^{-1}) = \begin{bmatrix} F_{d+1}(z^{-1}) \\ F_{d+2}(z^{-1}) \\ \vdots \\ F_{d+N}(z^{-1}) \end{bmatrix}$$

Notice that the last two terms in Equation (4.6) only depend on the past and can be grouped into $\mathbf{f}$ leading to:

$$\mathbf{y} = \mathbf{G}\mathbf{u} + \mathbf{f}$$

Notice that if all initial conditions are zero, the free response $\mathbf{f}$ is also zero. If a unit step is applied to the input at time t; that is,

$$\triangle u(t) = 1, \triangle u(t+1) = 0, \dots, \triangle u(t+N-1) = 0$$

the expected output sequence $[\hat{y}(t+1), \hat{y}(t+2), \dots, \hat{y}(t+N)]^T$ is equal to the first column of matrix $\mathbf{G}$. That is, the first column of matrix $\mathbf{G}$ can be calculated as the step response of the plant when a unit step is applied to the manipulated variable. The free response term can be calculated recursively by

$$\mathbf{f}_{j+1} = z(1 - \tilde{A}(z^{-1}))\mathbf{f}_j + B(z^{-1}) \triangle u(t-d+j)$$

with $\mathbf{f}_0 = y(t)$ and $\triangle u(t+j) = 0$ for $j \geq 0$.

Expression (4.2) can be written as

$$J = (\mathbf{G}\mathbf{u} + \mathbf{f} - \mathbf{w})^T(\mathbf{G}\mathbf{u} + \mathbf{f} - \mathbf{w}) + \lambda \mathbf{u}^T \mathbf{u} \tag{4.7}$$

where

$$\mathbf{w} = \begin{bmatrix} w(t+d+1) & w(t+d+2) & \cdots & w(t+d+N) \end{bmatrix}^T$$

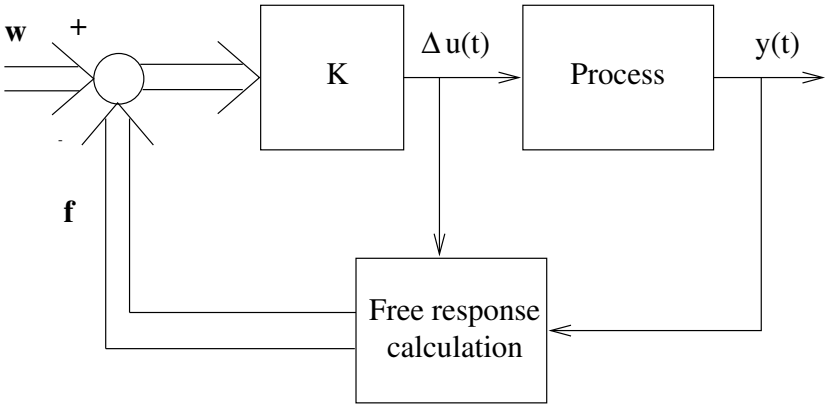


Fig. 4.1. GPC control law

Equation (4.7) can be written as

$$J = \frac{1}{2}\mathbf{u}^T\mathbf{H}\mathbf{u} + \mathbf{b}^T\mathbf{u} + f_0 \tag{4.8}$$

where

$$\begin{aligned}\mathbf{H} &= 2(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}) \\ \mathbf{b}^T &= 2(\mathbf{f} - \mathbf{w})^T\mathbf{G} \\ f_0 &= (\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w})\end{aligned}$$

The minimum of J, assuming there are no constraints on the control signals, can be found by making the gradient of J equal to zero, which leads to:

$$\mathbf{u} = -\mathbf{H}^{-1}\mathbf{b} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T(\mathbf{w} - \mathbf{f}) \tag{4.9}$$

Notice that the control signal that is actually sent to the process is the first element of vector $\mathbf{u}$, given by:

$$\triangle u(t) = \mathbf{K}(\mathbf{w} - \mathbf{f}) \tag{4.10}$$

where $\mathbf{K}$ is the first row of matrix $(\mathbf{G}^T\mathbf{G}+\lambda\mathbf{I})^{-1}\mathbf{G}^T$. This has a clear meaning that can easily be derived from Figure 4.1: if there are no future predicted errors, that is, if $\mathbf{w}-\mathbf{f} = \mathbf{0}$, then there is no control move, since the objective will be fulfilled with the free evolution of the process. However, in the other case, there will be an increment in the control action proportional (with a factor $\mathbf{K}$) to that future error. Notice that the action is taken with respect to *future* errors, not *past* errors, as is the case in conventional feedback controllers.

Notice that only the first element of $\mathbf{u}$ is applied and the procedure is repeated at the next sampling time. The solution to the GPC given involves the

inversion (or triangularization) of an $N \times N$ matrix which requires a substantial amount of computation. In [58] the concept of control horizon is used to reduce the amount of computation needed, assuming that the projected control signals are going to be constant after $N_u < N$. This leads to the inversion of an $N_u \times N_u$ matrix which reduces the amount of computation (in particular, if $N_u = 1$ it is reduced to a scalar computation, as in EPSAC), but restricts the optimality of the GPC. A fast algorithm to implement self-tuning GPC for processes that can be modelled by the reaction curve method is presented in the next chapter. The use of Hopfield neural networks has also been proposed [172] to obtain fast GPCs.

4.3 The Coloured Noise Case

When the noise polynomial $C(z^{-1})$ of Equation (4.1) is not equal to 1 the prediction changes slightly. In order to calculate the predictor in this situation, the following Diophantine equation is solved:

$$C(z^{-1}) = E_j(z^{-1})\tilde{A}(z^{-1}) + z^{-j}F_j(z^{-1}) \tag{4.11}$$

with $\delta(E_j(z^{-1})) = j - 1$ and $\delta(F_j(z^{-1})) = \delta(\tilde{A}(z^{-1})) - 1$.

Multiplying equation (4.1) by $\triangle E_j(z^{-1})z^j$ and using (4.11)

$$C(z^{-1})(y(t+j) - E_j(z^{-1})e(t+j)) = E_j(z^{-1})B(z^{-1})\triangle u(t+j-1) + F_j(z^{-1})y(t)$$

As the noise terms are all in the future, the expected value of the left-hand side of this equation is:

$$E[C(z^{-1})(y(t+j) - E_j(z^{-1})e(t+j))] = C(z^{-1})\hat{y}(t+j|t)$$

The expected value of the output can be generated by the equation:

$$C(z^{-1})\hat{y}(t+j|t) = E_j(z^{-1})B(z^{-1}) \triangle u(t+j-1) + F_j(z^{-1})y(t) \tag{4.12}$$

Notice that this prediction equation could be used to generate the predictions in a recursive way. An explicit expression for the optimal j step ahead prediction can be obtained by solving the Diophantine equation

$$1 = C(z^{-1})M_j(z^{-1}) + z^{-k}N_j(z^{-1}) \tag{4.13}$$

with $\delta(M_j(z^{-1})) = j - 1$ and $\delta(N_j(z^{-1})) = \delta(C(z^{-1})) - 1$.

Multiplying Equation (4.12) by $M_j(z^{-1})$ and using (4.13),

$$\hat{y}(t+j|t) = M_j E_j(z^{-1})B(z^{-1})\triangle u(t+j-1) + M_j(z^{-1})F_j(z^{-1})y_(t) + N_j(z^{-1})y(t)$$

which can be expressed as

$$\hat{y}(t+j|t) = G(z^{-1}) \triangle u(t+j-1) + G_p(z^{-1}) \triangle u(t+j-1) + (M_j(z^{-1})F_j(z^{-1}) + N_j(z^{-1}))y(t)$$

with $\delta(G(z^{-1})) < j$. These predictions can be used in the cost function which can be minimized as in the white noise case.

Another way of computing the prediction is by considering the filtered signals from the plant input/output data

$$y^f(t) = \frac{1}{C(z^{-1})}y(t) \qquad u^f(t) = \frac{1}{C(z^{-1})}u(t)$$

so that the resulting overall model becomes

$$A(z^{-1})y^f(t) = B(z^{-1})u^f(t) + \frac{e(t)}{\triangle}$$

and the white noise procedure for computing the prediction can be used. The predicted signal $\hat{y}^f(t+j|t)$ obtained this way has to be filtered by $C(z^{-1})$ in order to get $\hat{y}(t+j|t)$.

4.4 An Example

In order to show how a Generalized Predictive Controller can be implemented, a simple example is presented. The controller will be designed for a first-order system for the sake of clarity.

The following discrete equivalence can be obtained when a first-order continuous plant is discretized

$$(1 + az^{-1})y(t) = (b_0 + b_1 z^{-1})u(t-1) + \frac{e(t)}{\triangle}$$

In this example the delay d is equal to 0 and the noise polynomial $C(z^{-1})$ is considered to be equal to 1.

The algorithm to obtain the control law described in the previous section will be used on the preceding system, obtaining numerical results for the parameter values $a = -0.8$, $b_0 = 0.4$ and $b_1 = 0.6$, the horizons being $N_1 = 1$ and $N_2 = N_u = 3$. As has been shown, predicted values of the process output over the horizon are first calculated and rewritten in the form of Equation (4.6), and then the control law is computed using Expression (4.9).

Predictor polynomials $E_j(z^{-1})$, $F_j(z^{-1})$ from $j = 1$ to $j = 3$ will be calculated solving the Diophantine Equation (4.3), with

$$\tilde{A}(z^{-1}) = A(z^{-1})(1 - z^{-1}) = 1 - 1.8z^{-1} + 0.8z^{-2}$$

In this simple case where the horizon is not too long, the polynomials can be directly obtained by dividing 1 by $\tilde{A}(z^{-1})$ with simple calculations. As has

been explained earlier, they can also be computed recursively, starting with the values obtained at the first step of the division, that is:

$$E_1(z^{-1}) = 1 \qquad F_1(z^{-1}) = 1.8 - 0.8z^{-1}$$

Whatever the procedure employed, the values obtained are:

$$E_2 = 1 + 1.8z^{-1} \qquad F_2 = 2.44 - 1.44z^{-1}$$
$$E_3 = 1 + 1.8z^{-1} + 2.44z^{-2} \; F_3 = 2.952 - 1.952z^{-1}$$

With these values and the polynomial $B(z^{-1}) = 0.4 + 0.6z^{-1}$, the values of $G_i(z^{-1})$ are

$$\begin{aligned} G_1 &= 0.4 + 0.6z^{-1} \\ G_2 &= 0.4 + 1.32z^{-1} + 1.08z^{-2} \\ G_3 &= 0.4 + 1.32z^{-1} + 2.056z^{-2} + 1.464z^{-3} \end{aligned}$$

and so the predicted outputs can be written as:

$$\begin{bmatrix} \hat{y}(t+1 \mid t) \\ \hat{y}(t+2 \mid t) \\ \hat{y}(t+3 \mid t) \end{bmatrix} = \begin{bmatrix} 0.4 & 0 & 0 \\ 1.32 & 0.4 & 0 \\ 2.056 & 1.32 & 0.4 \end{bmatrix} \begin{bmatrix} \triangle u(t) \\ \triangle u(t+1) \\ \triangle u(t+2) \end{bmatrix} + \underbrace{\begin{bmatrix} 0.6 \triangle u(t-1) + 1.8y(t) - 0.8y(t-1) \\ 1.08 \triangle u(t-1) + 2.44y(t) - 1.44y(t-1) \\ 1.464 \triangle u(t-1) + 2.952y(t) - 1.952y(t-1) \end{bmatrix}}_{\mathbf{f}}$$

The following step is to calculate $\mathbf{H}^{-1}\mathbf{b}$. If λ is taken as equal to 0.8

$$(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T = \begin{bmatrix} 0.133 & 0.286 & 0.147 \\ -0.154 & -0.165 & 0.286 \\ -0.029 & -0.154 & 0.1334 \end{bmatrix}$$

As only $\triangle u(t)$ is needed for the calculations, only the first row of the matrix is used, obtaining the following expression for the control law:

$$\begin{aligned} \triangle u(t) = &-0.604 \triangle u(t-1) - 1.371y(t) + 0.805y(t-1) \\ &+ 0.133w(t+1) + 0.286w(t+2) + 0.147w(t+3) \end{aligned}$$

where $w(t+i)$ is the reference trajectory which can be considered constant and equal to the current setpoint or a first-order approach to the desired value. Then the control signal is a function of this desired reference and of past inputs and outputs and is given by:

$$\begin{aligned} u(t) = &\, 0.396u(t-1) + 0.604u(t-2) - 1.371y(t) + 0.805y(t-1) \\ &+ 0.133w(t+1) + 0.286w(t+2) + 0.147w(t+3) \end{aligned} \tag{4.14}$$

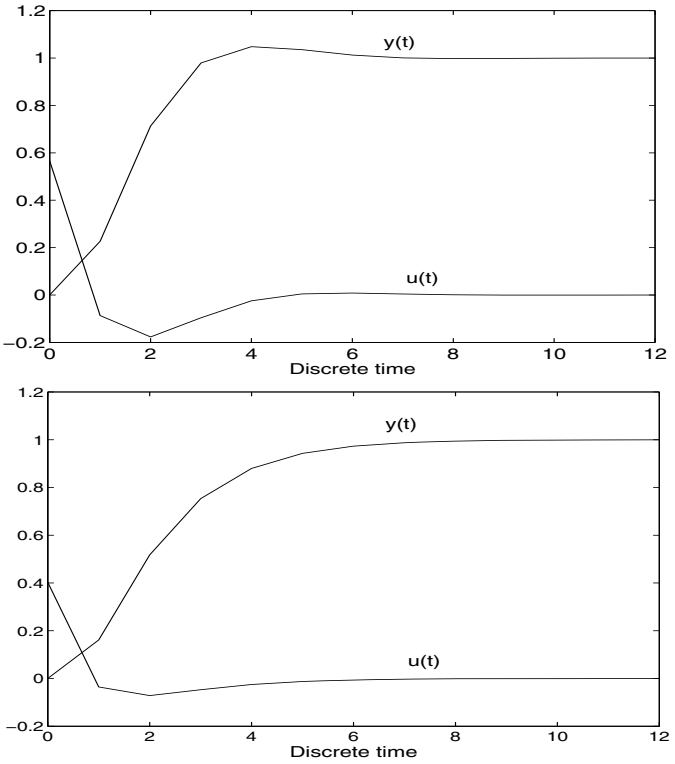


Fig. 4.2. System response

Simulation results show the behaviour of the closed-loop system. In the first graph of Figure 4.2 the reference is constant and equal to 1, and in the second one there is a smooth approach to the same value, obtaining a slightly different response, slower but without overshoot.

The GPC control law can also be calculated without the use of the Diophantine equation.

To obtain the control law it is necessary to know matrix $\mathbf{G}$ and the free response $\mathbf{f}$, to compute $\mathbf{u} = (\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T(\mathbf{w} - \mathbf{f})$. Matrix $\mathbf{G}$ is composed of the plant step response coefficients, so that the elements of the first column of this matrix are the first N coefficients, that can be computed as

$$g_j = -\sum_{i=1}^{j} a_i g_{j-i} + \sum_{i=0}^{j-1} b_i \text{ with } g_k = 0 \quad \forall k < 0$$

where b_i and a_i are the parameters of the numerator and denominator of the transfer function.

Therefore, as the prediction horizon is 3, $A = 1 - 0.8z^{-1}$ and $B = 0.4 + 0.6z^{-1}$

$$g_0 = b_0 = 0.4$$
$$g_1 = -a_1 g_0 + b_0 + b_1 = 1.32$$
$$g_2 = -a_1 g_1 - a_2 g_0 - a_3 g_0 + b_0 + b_1 = 2.056$$

and the matrix is given by

$$\mathbf{G} = \begin{bmatrix} 0.4 & 0 & 0 \\ 1.32 & 0.4 & 0 \\ 2.056 & 1.32 & 0.4 \end{bmatrix}$$

which logically coincides with the one obtained by the previous method.

The free response can also be calculated without the use of the Diophantine equation, just noting that it is the response of the plant assuming that future controls equal the previous control $u(t-1)$ and that the disturbance is constant. Thus, using the transfer function

$$y(t) = 0.8y(t-1) + 0.4u(t-1) + 0.6u(t-2)$$
$$y(t+1) = 0.8y(t) + 0.4u(t) + 0.6u(t-1)$$

If both equations are added and $y(t+1)$ is extracted

$$y(t+1) = 1.8y(t) - 0.8y(t-1) + 0.4 \triangle u(t) + 0.6 \triangle u(t-1)$$

Now, considering that in the free response only the control increments before instant t appear:

$$f(t+1) = 1.8y(t) - 0.8y(t-1) + 0.6 \triangle u(t-1)$$
$$f(t+2) = 1.8f(t+1) - 0.8y(t) = 2.44y(t) - 1.44y(t-1) + 1.08 \triangle u(t-1)$$
$$f(t+3) = 1.8f(t+2) - 0.8f(t+1)$$
$$= 2.952y(t) - 1.952y(t-1) + 1.464 \triangle u(t-1)$$

Vector **f** obtained this way is the same as the one previously obtained, so the control law is the one given by Equation (4.14).

4.5 Closed-Loop Relationships

Closed-loop relations can be obtained for the unconstrained GPC. The closed-loop system can be posed in the classical pole-placement structure of Figure 4.3

The control law can be stated as

$$R(z^{-1}) \triangle u(t) = T(z^{-1})w(t) - S(z^{-1})y(t) \tag{4.15}$$

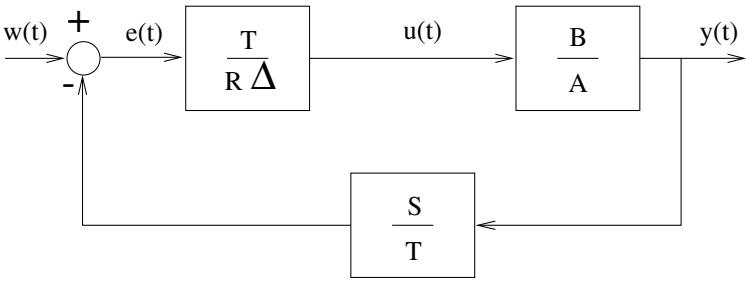


Fig. 4.3. Classical pole-placement structure

where R, S and T are polynomials in the backward shift operator. This control law can be considered as composed of a feedforward term (T/R) and a feedback part (S/R). In this situation it is possible to obtain the closed-loop transfer function and derive some properties such as stability and robustness. First, the general GPC control scheme of figure 4.3 must be rearranged to take the form of Equation (4.15).

The control law of Equation (4.9) gives the future control sequence **u**. As a receding strategy is being used, only the first element of that sequence $\triangle u(t \mid t)$ is actually sent to the process, therefore the control action is given by

$$\triangle u(t) = \mathbf{K}(\mathbf{w} - \mathbf{f}) = \sum_{i=N_1}^{N_2} k_i[w(t+i) - f(t+i)] \tag{4.16}$$

where $\mathbf{K}$ is the first row of matrix $(\mathbf{G}^T\mathbf{G} + \lambda\mathbf{I})^{-1}\mathbf{G}^T$.

The general case in which the $C(z^{-1})$ polynomial is not equal to zero will be considered to obtain the free response. In many situations this polynomial is not identified, since identification is not easy due to its time-varying characteristics and the difficulty of the CARIMA model to describe general deterministic disturbances. In these cases it is substituted by the so-called T polynomial that can be regarded as a fixed observer or a prefilter, as will be discussed later.

Then the plant model is given by:

$$A(z^{-1})y(t) = B(z^{-1})\, u(t-1) + T(z^{-1})\frac{e(t)}{\triangle}$$

The Diophantine equation that must be solved now includes the T polynomial:

$$T(z^{-1}) = E_j(z^{-1}) \triangle A(z^{-1}) + z^{-j}F_j(z^{-1}) \tag{4.17}$$

Using this equation and the plant model, the future output value is given by

$$y(t+j) = \frac{B(z^{-1})}{A(z^{-1})}u(t+j-1) + E_j(z^{-1})e(t+j) + \frac{F_j(z^{-1})}{A(z^{-1})\triangle}e(t)$$

replacing the $e(t)$ from the plant model and using (4.17):

$$y(t+j) = \frac{F_j}{T}y(t) + \frac{E_jB}{T} \triangle u(t+j-1) + E_je(t+j)$$

The best prediction is obtained by replacing $e(t+j)$ by its expected value (zero):

$$\hat{y}(t+j \mid t) = \frac{F_j}{T}y(t) + \frac{E_jB}{T} \triangle u(t+j-1)$$

This expression is a function of known values and future control actions. The control actions can be separated into past ones (those taken before instant t) and future ones (which must be calculated by the controller) using the Diophantine equation[1]

$$E_j(z^{-1})B(z^{-1}) = H_j(z^{-1})T(z^{-1}) + z^{-j}I_j(z^{-1}) \tag{4.18}$$

that leads to the prediction equation:

$$\begin{aligned}\hat{y}(t+j \mid t) &= H_j \triangle u(t+j) + \frac{I_j}{T} \triangle u(t-1) + \frac{F_j}{T}y(t) \\ &= H_j \triangle u(t+j) + I_j \triangle u^f(t-1) + F_jy^f(t)\end{aligned} \tag{4.19}$$

Using the filtered variables $y^f(t) = \frac{y(t)}{T}$ and $\triangle u^f(t-1) = \frac{\triangle u(t-1)}{T}$, this equation provides the same prediction along the horizon as given by (4.6) when $T(z^{-1}) = 1$, where the coefficients of H_j are the elements of matrix $\mathbf{G}$ and I_j are the rows of vector $\mathbf{G'}$.

Now the free response of the system (the one needed for the control law) is given by:

$$\mathbf{f} = \mathbf{I}(z^{-1}) \triangle u^f(t-1) + \mathbf{F}(z^{-1})y^f(t) = \mathbf{I}(z^{-1})\frac{\triangle u(t-1)}{T(z^{-1})} + \mathbf{F}(z^{-1})\frac{y(t)}{T(z^{-1})}$$

Once the free response has been obtained when the T polynomial is considered, it can be included in the expression of the control law given by (4.16):

$$\begin{aligned}\triangle u(t) &= \mathbf{K}(\mathbf{w} - \mathbf{f}) = \sum_{i=N_1}^{N_2} k_i[w(t+i) - f(t+i)] \\ &= \sum_{i=N_1}^{N_2} k_iw(t+i) - \sum_{i=N_1}^{N_2} k_i\frac{I_i(z^{-1})}{T(z^{-1})} \triangle u(t-1) - \sum_{i=N_1}^{N_2} k_i\frac{F_i(z^{-1})}{T(z^{-1})}y(t)\end{aligned}$$

Omitting the term z^{-1} and reordering the last equation

$$\left[T + z^{-1}\sum_{i=N_1}^{N_2} k_iI_i\right] \triangle u(t) = T\sum_{i=N_1}^{N_2} k_iw(t) - \sum_{i=N_1}^{N_2} k_iF_iy(t)$$

[1] Notice that this equation with $T(z^{-1}) = 1$ is implicitly used to derive $\mathbf{G}$ and $\mathbf{G'}$ in (4.6).

where it has been considered that the future reference trajectory keeps constant along the horizon or its evolution is unknown and therefore $w(t+i)$ is taken as equal to $w(t)$. In the other case, the first term of the right hand side should be expressed as $T\sum_{i=N_1}^{N_2} k_i z^i w(t)$ and therefore the following relations could change slightly.

The values of polynomials R and S can be obtained by comparing the last equation with (4.15), and are given by:

$$R(z^{-1}) = \frac{T(z^{-1}) + z^{-1}\sum_{i=N_1}^{N_2} k_i I_i}{\sum_{i=N_1}^{N_2} k_i}$$

$$S(z^{-1}) = \frac{\sum_{i=N_1}^{N_2} k_i F_i}{\sum_{i=N_1}^{N_2} k_i}$$

The closed-loop characteristic equation comes from inclusion of the control action given by (4.15) in the plant model expressed as:

$$A \triangle y(t) = B \triangle u(t-1) + Te(t)$$

Therefore, if the control action

$$\triangle u(t) = \frac{T}{R} w(t) - \frac{S}{R} y(t)$$

is replaced in the plant model, the following expression is obtained:

$$A \triangle y(t) = Bz^{-1}(\frac{T}{R} w(t) - \frac{S}{R} y(t)) + Te(t)$$

Extracting $y(t)$ from this equation provides the closed-loop relation that gives the output as a function of the reference and the disturbance

$$y(t) = \frac{BTz^{-1}}{RA \triangle + BSz^{-1}} w(t) + \frac{TR}{RA \triangle + BSz^{-1}} e(t) \tag{4.20}$$

and consequently the characteristic equation is given by:

$$RA \triangle + BSz^{-1} = 0$$

With a few manipulations and using (4.18), the characteristic polynomial can be decomposed as:

$$RA \triangle + BSz^{-1} = \frac{1}{\sum_{i=N_1}^{N_2} k_i}(T\tilde{A} + T\sum_{i=N_1}^{N_2} k_i z^{i-1}(B - \tilde{A}H_i)) = TP_c$$

Therefore Equation (4.20) turns to

$$y(t) = \frac{Bz^{-1}}{P_c} w(t) + \frac{R}{P_c} e(t)$$

where it is shown that the T polynomial is cancelled in the closed-loop transfer function between output and reference, as is the case in any observer, and that stability and performance are driven by the roots of polynomial P_c. However, it is difficult to establish clear dependencies of these roots on the tuning parameters N_1, N_2, N_u and λ. It is interesting to note that $P_c(1) = B(1)$, which guarantees offset-free response since the static gain of the transfer function between output and reference is always one.

When the GPC is written in the general pole-placement structure, some stability properties can be derived from the transfer function. A paper by Clarke and Mohtadi [57] presents some properties related to stability. It is proven that stability can be guaranteed if the tuning parameters (horizons and control-weighting factor) are correctly chosen. In the following sections, two formulations related to GPC with guaranteed stability will be treated in more detail.

4.6 The Role of the T Polynomial

Although the T polynomial does not appear in the transfer function between the output and the reference, this is not the case for the transfer function between the output and the disturbance. From Equation (4.19) it can be seen that both the output $y(t)$ and the control move $\triangle u(t)$ appear in the prediction, and therefore in the control law, filtered by $1/T$. Thus, from a practical point of view, it means that the T polynomial can be treated as a filter. By ensuring that the degree of T is big enough, the roll-off of the filter attenuates the component of prediction error caused by model mismatch, which is particularly important at high frequencies. Notice that low-frequency disturbances can be removed by the $\triangle$ term that appears in the prediction.

The high-frequency disturbances are mainly due to the presence of high-frequency unmodelled dynamics and unmeasurable load disturbances. If there are no unmodelled dynamics, the effect of T is the rejection of disturbances, with no influence on reference tracking. In this case T can be used to *detune* the response to unmeasurable high-frequency load disturbances, preventing excessive control actions.

On the other hand, T is used as a design parameter that can influence robust stability. In this case the predictions will not be optimal but robustness in the face of uncertainties can be achieved, in a similar interpretation to that used by Ljung [128]. Then this polynomial can be considered as a prefilter as well as an observer. The effective use of observers is known to play an essential role in the robust realization of predictive controllers (see [57] for the effect of prefiltering on robustness).

4.6.1 Selection of the T Polynomial

The selection of the filter polynomial T is not a trivial matter. Although some guidelines are given in [183] for mean-level and deadbeat GPC, a systematic

design strategy for the T filter has not been completely established. Usually it is assumed that the stronger filtering (considered as stronger that filtering with smaller bandwidth or bigger slope if the bandwidth is the same) has better robustness properties against high-frequency uncertainties. But this is not always true, as is shown with some counterexamples in [210]. In this paper and in [209] Yoon and Clarke present guidelines for the selection of T. These guidelines are based upon the robustness margin improvement at high frequencies and conclude stating that, for open-loop stable processes, the best choice is

$$T(z^{-1}) = A(z^{-1})(1 - \beta z^{-1})^{N_1 - \delta(P)}$$

Where β is close to the dominant root of A, N_1 is the minimum prediction horizon and $\delta(P)$ is the degree of polynomial P (the filter used to generate a reference trajectory with specified dynamics, see Section 2.1.2).

Notice that this idea of filtering for improving robustness also lies in Internal Model Control (IMC)[141] where, once the controller that provides the desired performance is obtained, it is detuned with a filter to improve robustness.

4.6.2 Relationships with Other Formulations

The prefiltering with T can be compared to $\mathcal{H}_\infty$ optimization based on the Q parameterization [141], obtaining equivalent robustness results [208]. It implies that prefiltering with polynomial T is an alternative to the optimal Q whose computation is demanding, especially in the adaptive case.

Robustness is improved by the introduction of polynomial T but, on the other hand, this fact implies that the prediction is no longer optimal. In a certain way, this idea is similar to the one used in the Linear Quadratic Gaussian (LQG) regulator to recover the good robustness properties that the Linear Quadratic Regulator (LQR) loses with the inclusion of the observer. This recovery is achieved by means of the LQG/LTR method, that consists of the Loop Transfer Recovery, (LTR), in such a way that it approaches the open-loop transfer function of the LQR method. This can be done by acting on the Kalman filter parameters, working with fictitious covariances (see [81]). In this way robustness is gained although prediction deteriorates. In both cases, the loss of optimality in the prediction or estimation is not considered a problem, since the controller works and is robust.

4.7 The P Polynomial

In the presentation of Generalized Predictive Control Clarke [58] points out the possibility of the use of an additional polynomial $P(z^{-1})$ as a design element in a similar way as employed in the Minimum Variance Controller [55] as a weighting polynomial that would be used for model following.

The $P(z^{-1})$ polynomial can appear when defining an auxiliary output as happens, for instance, in EPSAC (see Chapter 2)

$$\psi(t) = P(z^{-1})y(t)$$

in such a way that it affects the output. This polynomial allows the control objectives to expand by using its roots as design parameters. In this way *deadbeat*, pole-placement or LQ control can be achieved.

This can easily be incorporated into the standard GPC formulation by considering the augmented plant

$$A \triangle P(z^{-1})y(t) = BA \triangle P(z^{-1})u(t-1)$$

with P(1)=1 to guarantee $\psi(t) = y(t)$ in steady state.

This is equivalent to defining filtered auxiliary signals $\nu(t) = P(z^{-1})u(t)$ and $\psi(t) = P(z^{-1})y(t)$. Thus the plant is given by:

$$A \triangle \psi(t) = BA \triangle \nu(t-1)$$

Now the error that appears in the cost function is defined by $w(t+j) - \psi(t+j)$, which is equivalent to considering a reference trajectory generated by $1/P(z^{-1})$. A *deadbeat* control of $\psi(t)$ can be achieved by acting on $\nu(t)$, whose closed-loop transfer function is given by:

$$\psi(t) = \frac{B(z^{-1})}{B(1)}w(t)$$

This means that

$$y(t) = \frac{B(z^{-1})}{B(1)P(z^{-1})}w(t) \tag{4.21}$$

and the GPC algorithm is solved to provide the auxiliary control increment $\triangle\nu(t)$, from which the system input is calculated as:

$$u(t) = u(t-1) + \frac{\triangle\nu(t)}{P(z^{-1})}$$

As can be observed (4.21) is the typical response of a pole-placement method, with poles placed at zeros of the chosen $P(z^{-1})$. That is, the output is made to track the dynamics specified by $P(z^{-1})$.

4.8 Consideration of Measurable Disturbances

Many processes are affected by external disturbances caused by the variation of variables that can be measured. This situation is typical in processes whose outputs are affected by variations of the load regime. Consider, for instance, a

cooled jacket continuous reactor where the temperature is controlled by manipulating the water flow entering the cooling jacket. Any variation of the reactive flows will influence the reactor temperature. These types of perturbations, also known as load disturbances, can easily be handled by the use of feedforward controllers. Known disturbances can be taken explicitly into account in MPC, as will be seen in the following.

Consider a process described by the following In this case the CARIMA model must be changed to include the disturbances:

$$A(z^{-1})y(t) = B(z^{-1})u(t-1) + D(z^{-1})v(t) + \frac{1}{\triangle}C(z^{-1})e(t) \tag{4.22}$$

where the variable $v(t)$ is the measured disturbance at time t and $D(z^{-1})$ is a polynomial defined as:

$$D(z^{-1}) = d_0 + d_1 z^{-1} + d_2 z^{-2} + \ldots + d_{n_d} z^{-n_d}$$

Multiplying Equation (4.22) by $\triangle E_j(z^{-1})z^j$:

$$\begin{aligned} E_j(z^{-1})\tilde{A}(z^{-1})y(t+j) = {} & E_j(z^{-1})B(z^{-1}) \triangle u(t+j-1) \\ & + E_j(z^{-1})D(z^{-1}) \triangle v(t+j) + E_j(z^{-1})e(t+j) \end{aligned}$$

By using (4.3), and after some manipulation, we get:

$$\begin{aligned} y(t+j) = {} & F_j(z^{-1})y(t) + E_j(z^{-1})B(z^{-1}) \triangle u(t+j-1) \\ & + E_j(z^{-1})D(z^{-1}) \triangle v(t+j) + E_j(z^{-1})e(t+j) \end{aligned}$$

Notice that because the degree of $E_j(z^{-1})$ is $j-1$, the noise terms are all in the future. By taking the expectation operator and considering that $E[e(t)] = 0$, the expected value for $y(t+j)$ is given by:

$$\begin{aligned} \hat{y}(t+j|t) = E[y(t+j)] = {} & F_j(z^{-1})y(t) + E_j(z^{-1})B(z^{-1}) \triangle u(t+j-1) \\ & + E_j(z^{-1})D(z^{-1}) \triangle v(t+j) \end{aligned}$$

By making the polynomial $E_j(z^{-1})D(z^{-1}) = H_j(z^{-1}) + z^{-j}H'_j(z^{-1})$, with $\delta(H_j(z^{-1})) = j-1$, the prediction equation can now be written as:

$$\begin{aligned} \hat{y}(t+j|t) = {} & G_j(z^{-1}) \triangle u(t+j-1) + H_j(z^{-1}) \triangle v(t+j) \\ & + G'_j(z^{-1}) \triangle u(t-1) + H'_j(z^{-1}) \triangle v(t) + F_j(z^{-1})y(t) \end{aligned} \tag{4.23}$$

Notice that the last three terms of the right-hand side of this equation depend on past values of the process output, measured disturbances and input variables and correspond to the free response of the process considered if the control signals and measured disturbances are kept constant; while the first term only depends on future values of the control signal and can be interpreted as the forced response, that is, the response obtained when the initial conditions are zero $y(t-j) = 0$, $\triangle u(t-j-1) = 0$, $\triangle v(t-j)$ for $j > 0$.

The second term of Equation (4.23) depends on the future deterministic disturbances. In some cases, when they are related to the process load, future disturbances are known. In other cases, they can be predicted using trends or other means. If this is the case, the term corresponding to future deterministic disturbances can be computed. If the future load disturbances are supposed to be constant and equal to the last measured value (i.e., $v(t+j) = v(t)$), then $\triangle v(t+j) = 0$ and the second term of this equation vanishes.

Equation (4.23) can be rewritten as

$$\hat{y}(t+j|t) = G_j(z^{-1}) \triangle u(t+j-1) + H_j(z^{-1}) \triangle v(t+j) + f_j$$

with $f_j = G'_j(z^{-1}) \triangle u(t-1) + H'_j(z^{-1}) \triangle v(t) + F_j(z^{-1})y(t)$.

Let us now consider a set of N j ahead predictions:

$$\begin{aligned}
\hat{y}(t+1|t) &= G_1(z^{-1}) \triangle u(t) + H_j(z^{-1}) \triangle v(t+1) + f_1 \\
\hat{y}(t+2|t) &= G_2(z^{-1}) \triangle u(t+1) + H_j(z^{-1}) \triangle v(t+2) + f_2 \\
&\vdots \\
\hat{y}(t+N|t) &= G_N(z^{-1}) \triangle u(t+N-1) + H_j(z^{-1}) \triangle v(t+N) + f_N
\end{aligned}$$

Because of the recursive properties of the E_j polynomial, these expressions can be rewritten as

$$\begin{bmatrix} \hat{y}(t+1|t) \\ \hat{y}(t+2|t) \\ \vdots \\ \hat{y}(t+j|t) \\ \vdots \\ \hat{y}(t+N|t) \end{bmatrix} = \begin{bmatrix} g_0 & 0 & \cdots & 0 & \cdots & 0 \\ g_1 & g_0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ g_{j-1} & g_{j-2} & \cdots & g_0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N-1} & g_{N-2} & \cdots & \cdots & \cdots & g_0 \end{bmatrix} \begin{bmatrix} \triangle u(t) \\ \triangle u(t+1) \\ \vdots \\ \triangle u(t+j-1) \\ \vdots \\ \triangle u(t+N-1) \end{bmatrix}$$

$$+ \begin{bmatrix} h_0 & 0 & \cdots & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ h_{j-1} & \cdots & h_1 & h_0 & \vdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ h_{N-1} & \cdots & \cdots & \cdots & h_1 & h_0 \end{bmatrix} \begin{bmatrix} \triangle v(t+1) \\ \triangle v(t+2) \\ \vdots \\ \triangle v(t+j-1) \\ \vdots \\ \triangle v(t+N) \end{bmatrix} + \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_j \\ \vdots \\ f_N \end{bmatrix}$$

where $H_j(z^{-1}) = \sum_{i=1}^{j} h_i z^{-i}$ and h_i are the coefficients of the system step response to the disturbance.

By making $\mathbf{f}' = \mathbf{Hv} + \mathbf{f}$, the prediction equation is now

$$\mathbf{y} = \mathbf{Gu} + \mathbf{f}'$$

which has the same shape as the general prediction equation used in the case of zero measured disturbances. The future control signal can be found

in the same way, simply using as free response the process response due to initial conditions (including external disturbances) and future "known" disturbances.

4.9 Use of a Different Predictor in GPC

In this section it is shown that a GPC is equivalent to a structure based on an optimal predictor plus a classical two-degree-of-freedom controller. If the optimal predictor is replaced by a Smith predictor [193], a new controller with similar nominal performance and more robust properties is obtained. This has great interest in the case of time-delay systems. The controller uses the same procedure to compute the control signal, although future outputs are calculated using the Smith predictor instead of the optimal predictor.

4.9.1 Equivalent Structure

In order to show the equivalence between the GPC and a structure composed of an optimal predictor and a classical controller, a CARIMA model with white integrated noise is used to compute the prediction. Let us consider a process with a dead time d, $T(z^{-1}) = 1$, $N_1 = d + 1$, $N_2 = d + N$, $N_u = N$, and the weighting sequences $\delta(j) = 1$, $\lambda(j) = \lambda$. Thus it is possible to write:

$$\begin{aligned}&\hat{y}(t+d+j \mid t) = (1-a_1)\hat{y}(t+d+j-1 \mid t)\\&+(a_1-a_2)\hat{y}(t+d+j-2 \mid t) + \ldots + a_{na}\hat{y}(t+d+j-n_a-1 \mid t)\\&+b_0 \triangle u(t+j-1) + \ldots + b_{nb} \triangle u(t+j-1-nb)\end{aligned} \tag{4.24}$$

If this equation is applied recursively for $j = 1, 2, \cdots, N$ we get

$$\begin{bmatrix} \hat{y}(t+d+1 \mid t) \\ \hat{y}(t+d+2 \mid t) \\ \vdots \\ \hat{y}(t+d+N \mid t) \end{bmatrix} = \mathbf{G} \begin{bmatrix} \triangle u(t) \\ \triangle u(t+1) \\ \vdots \\ \triangle u(t+N-1) \end{bmatrix} + \mathbf{H} \begin{bmatrix} \triangle u(t-1) \\ \triangle u(t-2) \\ \vdots \\ \triangle u(t-nb) \end{bmatrix} + \mathbf{S} \begin{bmatrix} \hat{y}(t+d \mid t) \\ \hat{y}(t+d-1 \mid t) \\ \vdots \\ \hat{y}(t+d-na \mid t) \end{bmatrix}$$

where $\mathbf{G}$, $\mathbf{H}$ and $\mathbf{S}$ are constant matrices of dimension $N \times N$, $N \times n_b$ and $N \times n_a + 1$, respectively. This equation can be written in a vector form as follows:

$$\hat{\mathbf{y}} = \mathbf{G}\mathbf{u} + \mathbf{H}\mathbf{u}' + \mathbf{S}\mathbf{y}' = \mathbf{G}\mathbf{u} + \mathbf{f}$$

where it is clear that $\mathbf{f} = \mathbf{H}\mathbf{u}' + \mathbf{S}\mathbf{y}'$ is composed of the terms in the past and correspond to the free response of the system.

If $\hat{\mathbf{y}}$ is introduced in the cost function, $J(N)$ is a function of $\mathbf{y}'$, $\mathbf{u}$, $\mathbf{u}'$ and the reference sequence. Minimizing $J(N)$ with respect to $\mathbf{u}$, that is, $\triangle u(t)$, $\triangle u(t+1)$ ... $\triangle u(t+N-1)$) leads to

$$\mathbf{M}\begin{bmatrix} \triangle u(t) \\ \triangle u(t+1) \\ \vdots \\ \triangle u(t+N-1) \end{bmatrix} = \mathbf{P}_0 \begin{bmatrix} \hat{y}(t+d \mid t) \\ \hat{y}(t+d-1 \mid t) \\ \vdots \\ \hat{y}(t+d-na \mid t) \end{bmatrix} + \mathbf{P}_1 \begin{bmatrix} \triangle u(t-1) \\ \triangle u(t-2) \\ \vdots \\ \triangle u(t-nb) \end{bmatrix} + \mathbf{P}_2 \begin{bmatrix} w(t+d+1) \\ w(t+d+2) \\ \vdots \\ w(t+d+N) \end{bmatrix}$$

where $\mathbf{M} = \mathbf{G}^T\mathbf{G} + \lambda\mathbf{I}$ and $\mathbf{R} = \mathbf{G}^T$ are of dimension $N \times N$, $\mathbf{P}_0 = -\mathbf{G}^T S$ of dimension $N \times n_a + 1$ and $\mathbf{P}_1 = -\mathbf{G}^T\mathbf{H}$ of dimension $N \times n_b$. As in a receding horizon algorithm only the value of $\triangle u(t)$ is computed, if $\mathbf{q}$ is the first row of matrix $\mathbf{M}^{-1}$, $\triangle u(t)$ is given by:

$$\triangle u(t) = \mathbf{q}\mathbf{P}_0 \begin{bmatrix} \hat{y}(t+d \mid t) \\ \hat{y}(t+d-1 \mid t) \\ \vdots \\ \hat{y}(t+d-na \mid t) \end{bmatrix} + \mathbf{q}\mathbf{P}_1 \begin{bmatrix} \triangle u(t-1) \\ \triangle u(t-2) \\ \vdots \\ \triangle u(t-nb) \end{bmatrix} + \mathbf{q}\mathbf{P}_2 \begin{bmatrix} w(t+d+1) \\ w(t+d+2) \\ \vdots \\ w(t+d+N) \end{bmatrix}$$

Therefore the control increment $\triangle u(t)$ can be written as:

$$\triangle u(t) = \mathbf{q}\mathbf{P}_0\mathbf{y}' + \mathbf{q}\mathbf{P}_1\mathbf{u}' + \mathbf{q}\mathbf{P}_2\mathbf{w}$$

The resulting control scheme (see Figure 4.4) is a linear feedback of predictions $\hat{y}(t+d \mid t)$, ..., $\hat{y}(t+d-na \mid t)$ generated by an optimal predictor. That is, prediction over a time horizon equal to the process dead time. The controller coefficients are computed for each choice of N and λ.

The block diagram presented in Figure 4.4 can easily be transformed into the one shown in Figure 4.5, which shows that a GPC is equivalent to a classical controller plus an optimal predictor. This classical controller is composed of a reference filter and a cascade block. If the plant can be modelled by a first-order system with a dead time then the classical controller results in a

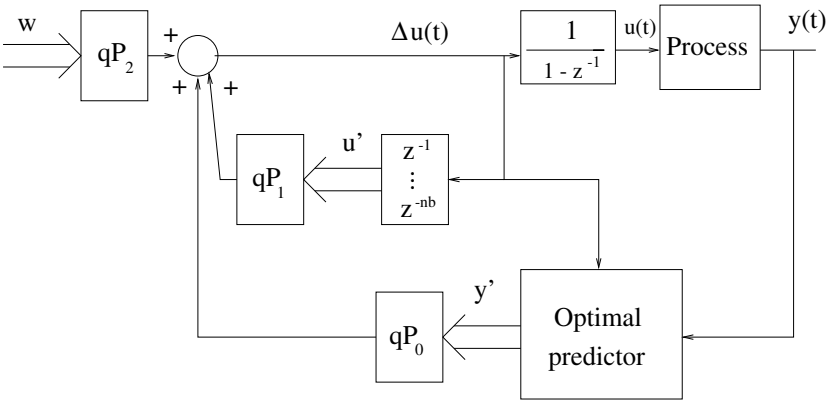


Fig. 4.4. Control scheme

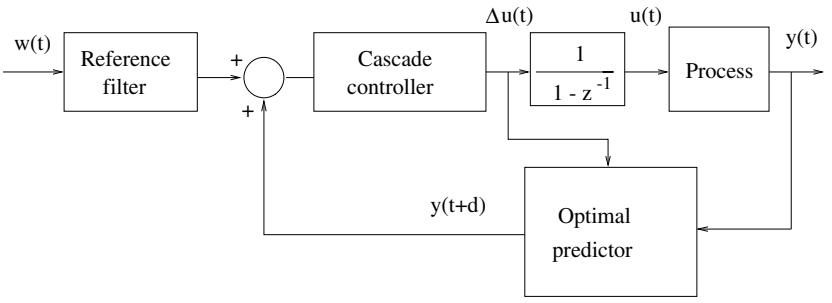


Fig. 4.5. Equivalent control structure of the GPC

simple PI [153]. In general, the primary controller is of the same order as the model of the plant.

This relation can be used to study how to improve the robustness of the GPC using a different predictor structure. Note that the computation of the classical controller is done independently of the predictor structure, so different predictors can be compared using the same controller parameters of the GPC.

The prediction can be obtained directly from polynomials A and B and the delay d considering Equation (4.24) for $j = 1$:

$$\hat{y}(t+1 \mid t) = (1-\tilde{A}(z^{-1}))zy(t)+\tilde{B}(z^{-1})u(t-d) \quad \text{with } \tilde{B}(z^{-1}) = (1-z^{-1})B(z^{-1})$$

Using the same procedure for $j = 2, \ldots,$ $\hat{y}(t+j \mid t)$ is computed as:

$$\hat{y}(t+j \mid t) = ((1-\tilde{A}(z^{-1}))z)^j y(t) + \sum_{i=1}^{j}(1-\tilde{A}(z^{-1}))^{i-1}\tilde{B}(z^{-1})u(t-d+j-1)$$

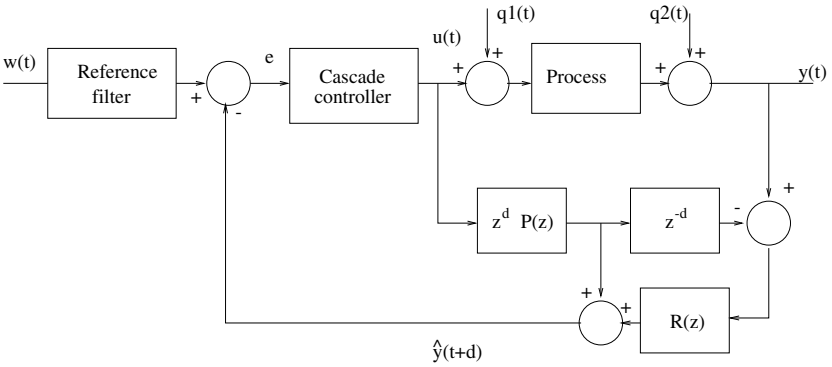


Fig. 4.6. Equivalent structure for the OP in the GPC

The predicted output at instant $t + d$ is therefore

$$\hat{y}(t + d \mid t) = ((1 - \tilde{A}(z^{-1}))z)^d y(t) + (1 - (1 - \tilde{A}(z^{-1}))^d)z^d P u(t)$$

where $P(z) = \frac{B(z^{-1})z^{-1}}{A(z^{-1})} z^{-d}$ is the plant model.

Defining $R(z) = (1 - \tilde{A}(z^{-1}))^d z^d$, the prediction can be written as:

$$\hat{y}(t + d \mid t) = R(z)y(t) + (z^d - R(z))P(z)u(t)$$

The closed-loop block diagram of the whole control system is shown in Figure 4.6. Now it is clear that the GPC has a structure similar to the well-known dead-time compensators like the Smith predictor (obtained when $R(z) = 1$) and that in the absence of a dead time the final control law is a classical two-degree-of-freedom controller.

The theoretical comparative results about the robustness and performance of GPC and the GPC which uses a Smith predictor can be found in [155], where it is shown that for stable processes, the Smith predictor (SP) based control structure has similar performance and more robustness than the one based on the optimal predictor (OP) when using the same primary controller. An application of this controller to mobile robot path tracking can be found in [157].

It is easy to show that the norm-bound uncertainty of the controller is:

$$\mid \delta P \mid \leq \left| \frac{1 + C(z)z^d P(z)}{C(z)R(z)} \right|$$

Notice that the norm-bound uncertainty is inversely proportional to $\mid R(z) \mid$ and that for the GPC the block $R(z)$ has high pass characteristics, so the controller has low values of the norm-bound uncertainty at high frequencies. On the other hand, the Smith predictor based controller has $R(z) = 1$ and consequently a higher robustness index.

As has been mentioned in this chapter, the robustness of the GPC can be improved by the use of a prefilter in the prediction equations. The effect of this filter (the T polynomial) can also be analyzed in Figure 4.6 because, when T is included in the predictor $R(z)$ is a function of $T(z)$. If T is chosen appropriately then $R(z)$ could have low pass characteristic and the robustness of the controller could be increased. On the other hand, the disturbance rejection response deteriorates when the robustness increases [8]. Note that for the Smith predictor based controller a filter $F(z)$ could also be included, but in this case the tuning of F is simpler than the tuning of T in the GPC case as for the new structure $R(z) = F(z)$. Note that to improve the robustness F must be chosen as a low pass filter [156].

For the computation of the controller the following steps must be taken:

- compute the prediction of the output (from t to $t+d$) using the open-loop model of the plant without considering disturbances
- correct each open-loop prediction adding the mismatch between the output and the prediction:

$$\hat{y}(t+d-i \mid t) = \hat{y}(t+d-i \mid t) + y(t-i) - \hat{y}(t-i)$$

- compute the control law as in the normal GPC using the coefficients of $\mathbf{q}$, $\mathbf{P}_0$, $\mathbf{P}_1$ and $\mathbf{P}_2$.

Note that to compute the control law it is also possible to use the forced and free response concepts used in standard GPC. In this case the forced response can be computed as done in GPC but the free response must be computed using the Smith predictor from t to $t+d$ and the optimal predictor from $t+d+1$ to $t+N$.

4.9.2 A Comparative Example

In order to illustrate the robustness properties of the proposed GPC an example comparing the SPGPC and the GPC is presented. It corresponds to a temperature control of a heat exchanger, where the model used in the predictor is a first-order system with dead time, such as the one presented in [154]. This system represents a typical industrial process, and because of this it is normally used to evaluate the performance of industrial controllers. In the example ARIMA disturbances are considered.

The relation between the output temperature and the input flow in the heat exchanger was obtained using the reaction curve method and is given by:

$$P(s) = \frac{0.12e^{-3s}}{1+6s}$$

Using a sample time $T_s = 0.6s$ the obtained discrete plant is

$$P(z) = \frac{bz^{-1}}{1 - az^{-1}} z^{-d}$$

where the nominal values of the parameters are $d_n = 5, a_n = 0.905$ and $b_n = 0.0114$. In practice there are errors in the estimation of all parameters but for this example only dead time uncertainty is considered, with a maximum value of $\delta d = 2$.

The GPC is computed in order to obtain (for the nominal case) a step response faster than the open-loop one. Thus, the GPC parameters were chosen as $N = 15$, $\lambda = 0.8$. The SPGPC has the same parameters.

The closed-loop behaviour of the GPC and the SPGPC for the nominal case are shown in Figure 4.7(a). At $t = 0$ a step change in the reference is performed and at $t = 100$ samples a 10% step load disturbance is applied to the system. The noise is generated with an ARIMA model with uniform distribution in ± 0.005. As can be observed, both systems have similar setpoint tracking and disturbance rejection behaviour for the nominal case.

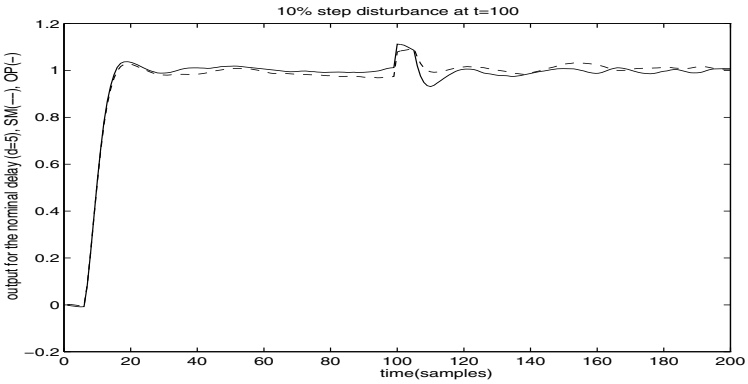
In the next simulation the delay of the plant is set to $d = 7$ and again a step change in the reference is considered at $t = 0$. Figure 4.7(b) shows that the SPGPC based control system is stable and the GPC one is unstable.

It must be stated that the comparison has been made with a GPC without T polynomial and the introduction of this in the formulation could improve its response when the dead time uncertainty appears, but at the same time the nominal disturbance rejection response will be deteriorated.

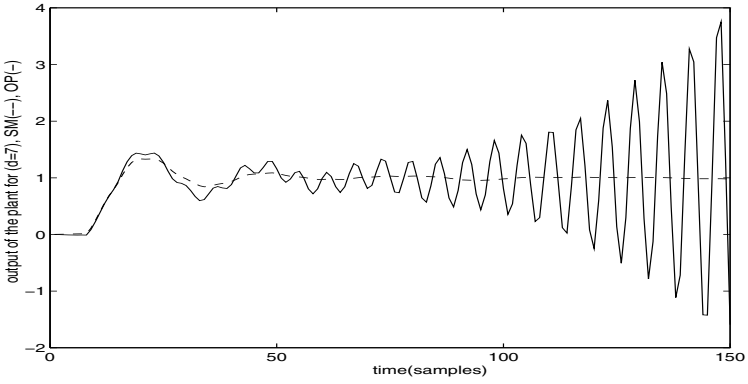
4.10 Constrained Receding Horizon Predictive Control

In spite of the great success of GPC in industry, there was an original lack of theoretic results about the properties of predictive control and an initial gap in important questions such as stability and robustness. In fact, the majority of stability results are limited to the infinite horizon case and there is a lack of a clear theory relating the closed-loop behaviour to design parameters, such as horizons and weighting sequences.

Bearing in mind the need to solve some of these drawbacks, a variation of the standard formulation of GPC appears developed by Clarke and Scattolini called Constrained Receding-Horizon Predictive Control (CRHPC) [61], [144],[151], which allows stability and robustness results to be obtained for small horizons. The idea basically consists of deriving a future control sequence so that the predicted output over some future time range is constrained to be at the reference value exactly, as shown in Figure 4.8. Some degrees of freedom of the future control signals are employed to force the



(a)



(b)

Fig. 4.7. Behaviour of the GPC(solid) and SPGPC (dashed) based control systems: (a) nominal case, (b) dead time uncertainty case

output, whilst the rest is available to minimize the cost function over a certain interval.

4.10.1 Computation of the Control Law

The computation of the control law is performed in a similar way to GPC, although the calculations become a little more complicated. Control signals must minimize the objective function

$$J(N_y, N_u) = \sum_{i=1}^{N_y} \mu(i)[\hat{y}(t+i \mid t) - y^o(t+i)]^2 + \sum_{i=0}^{N_u} \lambda(i)[\triangle u(t+j)]^2 \quad (4.25)$$

where $y^o(t+i)$ is the reference, subject to constraints

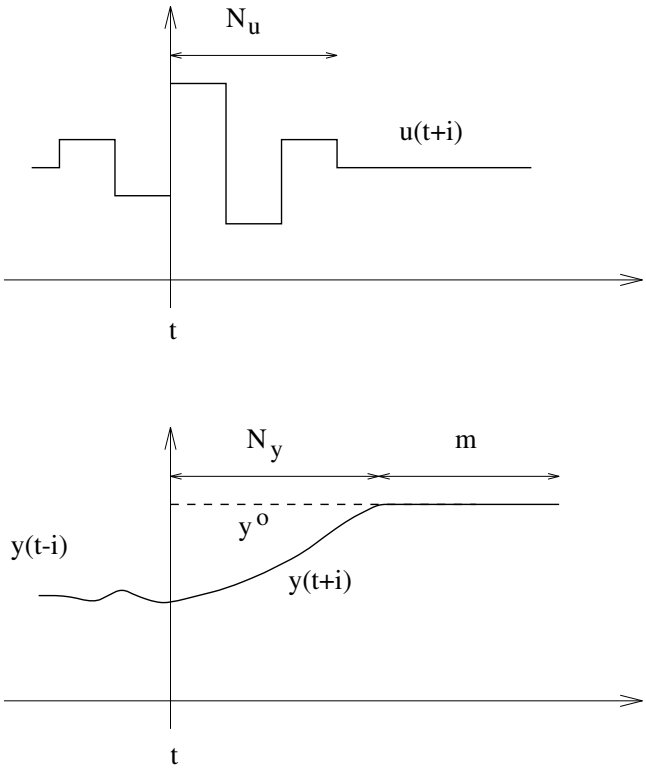


Fig. 4.8. Constrained Receding Horizon Predictive Control

$$\begin{array}{rl} y(t+N_y+i) = y^o(t+N_y+1) & i = 1 \ldots m \\ \triangle u(t+N_u+j) = 0 & j > 0 \end{array} \tag{4.26}$$

The design parameters are the values of the horizons, the weighting sequences $\mu(i)$ and $\lambda(i)$ and the value of m, which defines the instants of coincidence between output and reference. The system is modelled by:

$$\triangle A(z^{-1})y(t) = B(z^{-1}) \triangle u(t-d) \tag{4.27}$$

In order to solve the optimization problem, it is necessary to calculate the output prediction, which is done by defining polynomials $E_i(z^{-1})$, $F_i(z^{-1})$ and $G_i(z^{-1})$ where E_i is of degree $i-1$ such that

$$1 = E_i(z^{-1}) \triangle A(z^{-1}) + z^{-i}F_i(z^{-1})$$

$$z^{-d}E_i(z^{-1})B(z^{-1}) = \sum_{h=1}^{i} s_h z^{-h} + z^{-i+1}G_i(z^{-1})$$

where s_h are the coefficients of the process step response.

Multiplying Equation (4.27) by $E_i(z^{-1})$ and taking the last equations into account leads to:

$$y(t+i) = \sum_{h=1}^{i} s_h \triangle u(t+i-h) + f(t+i)$$
$$f(t+i) = F_i(z^{-1})y(t) + G_i(z^{-1}) \triangle u(t-1)$$

Now defining the following sequences of future variables

$$\begin{aligned}
Y(t) &= [\ y(t+1)\ y(t+2)\ \ldots\ y(t+N_y)\ ]^T \\
Y^o(t) &= [\ y^o(t+1)\ y^o(t+2)\ \ldots\ y^o(t+N_y)\ ]^T \\
\triangle U(t) &= [\ \triangle u(t)\ \triangle u(t+1)\ \ldots\ \triangle u(t+N_u)\ ]^T \\
F(t) &= [\ f(t+1)\ f(t+2)\ \ldots\ f(t+N_y)\ ]^T \\
\overline{Y}(t) &= [\ y(t+N_y+1)\ y(t+N_y+2)\ \ldots\ y(t+N_y+m)\ ]^T \\
\overline{Y}^o(t) &= [\ y^o(t+N_y+1)\ y^o(t+N_y+2)\ \ldots y^o(t+N_y+m)\ ]^T \\
\overline{F}^o(t) &= [\ f(t+N_y+1)\ f(t+N_y+2)\ \ldots\ f(t+N_y+m)\ ]^T
\end{aligned}$$

the following matrices, of dimension $N_y \times (N_u+1)$ and $m \times (N_u+1)$, respectively;

$$G = \begin{bmatrix} s_1 & 0 & 0 & \ldots & 0 \\ s_2 & s_1 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{Ny} & s_{Ny-1} & s_{Ny-2} & \cdots & s_{Ny-Nu} \end{bmatrix}$$

$$\overline{G} = \begin{bmatrix} s_{Ny+1} & s_{Ny} & \cdots & s_{Ny-Nu+1} \\ s_{Ny+2} & s_{Ny+1} & \cdots & s_{Ny-Nu+2} \\ \vdots & \vdots & \vdots & \vdots \\ s_{Ny+m} & s_{Ny+m-1} & \cdots & s_{Ny-Nu+m} \end{bmatrix}$$

and the weighting sequences

$$\begin{aligned}
M(t) &= \text{ diag } \{\mu(1), \mu(2), \ldots \mu(N_y)\} \\
\Lambda(t) &= \text{ diag } \{\lambda(0), \lambda(1), \ldots \lambda(N_u)\}
\end{aligned}$$

the following relations hold:

$$\begin{aligned}
Y(t) &= G\ \triangle U(t) + F(t) \\
\overline{Y}(t) &= \overline{G}\ \triangle U(t) + \overline{F}(t)
\end{aligned}$$

Then the cost function (Equation 4.25) and the constraints (4.26) can be written as:

$$J = [Y(t) - Y^o(t)]^T \, M(t) \, [Y(t) - Y^o(t)] + \triangle U^T(t) \, \Lambda(t) \; \triangle U(t)$$
$$\overline{G} \triangle U(t) + \overline{F}(t) = \overline{Y}^o(t)$$

The solution can be obtained by the use of Lagrange multipliers. If the common case of constant weighting sequence is considered, that is, $M(t) = \mu I$, $\Lambda(t) = \lambda I$, the solution can be written as:

$$\triangle U(t) = (\mu G^T G + \lambda I)^{-1}[\mu G^T (Y^o(t) - F(t)) + \overline{G}^T (\overline{G}(\mu G^T G + \lambda I)^{-1}\overline{G}^T)^{-1} \times (\overline{Y}^o(t) - \overline{F}(t) - \mu \overline{G}(\mu G^T G + \lambda I)^{-1} G^T (Y^o(t) - F(t)))] \quad (4.28)$$

As it is a receding horizon strategy, only the first element of vector $\triangle U(t)$ is used, repeating the calculation at the next sampling time. This method provides an analytical solution that, as can be observed in (4.28) proves to be more complex than the standard GPC solution. Computational burden can be considerable since various matrix operations, including inversion, must be made, although some calculations can be optimized knowing that G is triangular and the matrices to be inverted are symmetrical. This factor can be decisive in the adaptive case, where all vectors and matrices can change at every sampling time.

Obtaining the control signal requires inversion of matrix $\overline{G}(\mu G^T G + \lambda I)^{-1}\overline{G}^T)$. ¿From the definition of matrix $\overline{G}$, it can be derived that condition $m \leq N_u + 1$ must hold, which can be interpreted as the number m of output constraints cannot be bigger than the number of control signal variations $N_u + 1$. Another condition for invertibility is that $m \leq n + 1$, since the coefficient s_i of the step response is a linear combination of the previous $n + 1$ values (where n is the system order). Notice that this last condition constrains the order of the matrix to invert ($m \times m$) to relative small values, as in the majority of situations the value of m will not be bigger than two or three.

4.10.2 Properties

As has been stated, one of the advantages of this method is the availability of stability results for finite horizons, compensating in a certain way the computational burden it carries. The following results present the outstanding properties of CRHPC, whose demonstration can be found in [151], based upon a state-space formulation of the control law (4.28). The fact that the output follows the reference over some range guarantees the monotonicity of the associated Riccati equation and in consequence stability, based upon the results by Kwon and Pearson [115].

Property 1. If $N_y = N_u > n + d + 1$ and $m = n + 1$ then the closed-loop system is asymptotically stable.

Property 2. If $N_u = n + d$ and $m = n + 1$ the control law results in a stable dead-beat control.

Property 3. If the system is asymptotically stable, $\mu = 0$, $m = 1$ and there exists ν such that either

$$s_\nu \leq s_{\nu+1} \leq \ldots \leq s_\infty, \;\; s_\nu > s_\infty/2, \;\; s_\infty > 0$$

or

$$s_\nu \geq s_{\nu+1} \geq \ldots \geq s_\infty, \;\; s_\nu < s_\infty/2, \;\; s_\infty < 0$$

Then, for any $N_y = N_u \geq \nu - 1$ the closed-loop system is asymptotically stable.

Property 4. Under the latter conditions, the closed-loop system is asymptotically stable for $N_u = 0$ and $N_y \geq \nu - 1$.

Property 5. If the open-loop system is asymptotically stable, $\mu = 0$, $m = 1$, $N_u = 0$ and constants $K > 0$ and $0 < \eta < 1$ exist such that

$$\mid s_i - s_\infty \mid \leq K\eta^i, \;\; i \geq 0$$

Then, for any N_y such that

$$\mid s_{Ny+1} \mid > K\frac{1+\eta}{1-\eta}\eta^{Ny+1}$$

the closed-loop system is asymptotically stable.

So it is seen that the method is able to stabilize any kind of process, such as unstable or nonminimum phase ones. As in GPC, the use of filtered inputs and outputs can result in a pole-placement control. Using the $P(z^{-1})$ polynomial introduced in [59] shows the close relationship between predictive control and pole placement [61]. This polynomial appears when defining an auxiliary output $\psi(t) = P(z^{-1})y(t)$, which substitutes $y(t)$ in (4.25) and (4.26) in such a way that the desired closed-loop poles are the roots of $z^{n+1}P(z^{-1})$.

4.11 Stable GPC

To overcome the lack of stability results for GPC, a new formulation has been developed by Rossiter and Kouvaritakis which ensures that the associated cost function is monotonically decreasing, guaranteeing closed-loop stability. For this reason, this algorithm is called Stable Generalized Predictive Control SGPC [112], [186] and it is based on stabilizing the loop before the application of the control strategy. Now the control actions that must be calculated are the future values of the reference that are sent to the closed loop, instead of the system inputs, which are functions of these. The stabilizing inner-loop controller is designed to obtain a finite closed-loop impulse response; this fact simplifies the implementation of the algorithm at the same time as ensuring the monotonicity of the cost function.

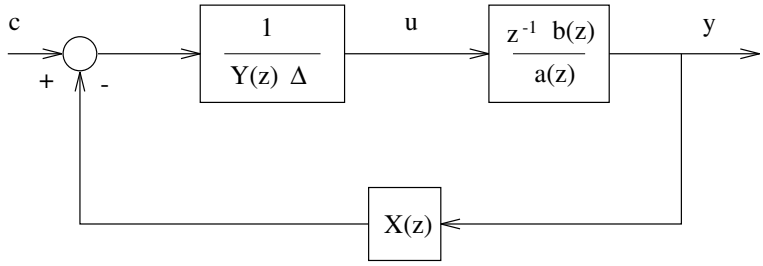


Fig. 4.9. Stable loop

4.11.1 Formulation of the Control Law

The model considered for obtaining the control law is:

$$G(z) = \frac{z^{-1}b(z)}{a(z)} = \frac{b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n}}{1 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}} \tag{4.29}$$

where, for the sake of simplicity and without any loss of generality, the order of numerator and denominator is considered to be the same; if the delay is bigger than one, it is enough to set the first terms of $b(z)$ to zero. As has been stated, before optimizing the cost function, the loop is stabilized by means of polynomials $X(z)$ and $Y(z)$ as shown in Figure 4.9. Signal c is the closed-loop reference and is the value that will appear in the cost function.

Polynomial $X(z)$ and $Y(z)$ satisfy the following relations:

$$a(z)Y(z)\triangle(z) + z^{-1}b(z)X(z) = 1$$
$$K(z) = \frac{X(z)}{Y(z)\triangle(z)}$$

with $\triangle(z) = 1 - z^{-1}$ and $K(z)$ the overall feedback controller. With these relations, it can be deduced that:

$$y(z) = z^{-1}b(z)c(z)$$
$$\triangle u(z) = A(z)c(z)$$

where $A(z) = a(z)\triangle$.

The cost function that, like standard GPC, measures the discrepancies between future outputs and reference as well as the necessary control effort can be obtained making use of these relations. The cost must be expressed as a function of the future values of c in such a way that the reference signals for the stabilized system can be obtained from its minimization. In order to achieve this objective, the following vectors are considered:

$$y^+ = [\, y(t+1)\; y(t+2)\; \ldots\; y(t+n_y)\,]^T$$
$$c^+ = [\, c(t+1)\; c(t+2)\; \ldots\; c(t+n_c)\,]^T$$
$$\triangle u^+ = [\, \triangle u(t)\;\; \triangle u(t+1)\; \ldots\; \triangle u(t+n_y-1)\,]^T$$
$$c^- = [\, c(t)\; c(t-1)\; \ldots\; c(t-n)\,]^T$$

where n_y and n_c are the output and reference horizons.

Simulating Equation (4.29) forward in time it can be written

$$\begin{aligned} y^+ &= \Gamma_b c^+ + H_b c^- + M_b c^\infty \\ \triangle u^+ &= \Gamma_a c^+ + H_A c^- + M_A c^\infty \end{aligned}$$

where the last terms of each equation represent the free response y_f and $\triangle u_f$ of y and $\triangle u$, respectively. Matrices Γ_A, Γ_b, H_A, H_b, M_A and M_b can easily be derived as shown in [112] while c^∞ is the vector of $n_y - n_c$ rows that contains the desired future values of the command input c. The elements of this vector are chosen to ensure offset-free steady state.

Usually, if $r(t+i)$, $i = 1, \ldots, n_y$ is the setpoint to be followed by the system output, c^∞ is chosen to be $[1, 1, \ldots, 1]^T \times r(t+n_y)/b(1)$. If step setpoint changes are assumed, c^∞ can be chosen as the vector of future references premultiplied by a matrix E formed out of the last $n_y - n_c$ rows of I_{ny}.

Then the cost function can be written as:

$$J = \|r^+ - y^+\|_2 + \lambda \| \triangle u^+ \|_2 = [c^+ - c_o]^T S^2 [c^+ - c_o] + \gamma$$

where

$$\begin{aligned} S^2 &= \Gamma_b^T \Gamma_b + \lambda \Gamma_A^T \Gamma_A \\ c_o &= S^{-2}[\Gamma_b^T (r^+ - y_f) - \lambda \Gamma_A^T \triangle u_f] \\ \gamma &= \|r^+ - y_f\|_2 + \lambda \| \triangle u_f \|_2 - \|c_o\|_2 \end{aligned}$$

As γ is a known constant, the control law of SGPC comes from the minimization of cost $J = \|S(c^+ - c_o)\|$, and is defined by

$$\triangle u(t) = A(z)c(t) \qquad c(t) = \frac{p_r(z)}{p_c(z)} r(t + ny)$$

where $p_r(z)$ and $p_c(z)$ are polynomials defined as:

$$\begin{aligned} p_r &= e^T T [\Gamma_b^T (I - M_b E) - \Gamma_b^T \Gamma_A E] \\ p_c &= e^T T [\Gamma_b^T H_b + \lambda \Gamma_b H_A] \end{aligned}$$

and $T = (\Gamma_b^T \Gamma_b + \lambda \Gamma_A \Gamma_A)^{-1}$; e is the first standard basis vector and the polynomials are in descending order of z.

The stability of the algorithm is a consequence of the fact that y and $\triangle u$ are related to c by means of finite impulse response operators ($z^{-1}b(z)$ and $A(z)$), which can be used to establish that the cost is monotonically decreasing [186] and can therefore be interpreted as a Lyapunov function guaranteeing stability.

4.12 Exercises

4.1. Given the process

$$y(t) - 1.5y(t-1) + 0.56y(t-2) = 0.9u(t-1) - 0.6u(t-2)$$

and $N_1 = 1$ and $N_2 = 3$:

1. Solve the Diophantine equation.
2. Compute $G_i(z^{-1})$ for $i = 1$, 2 and 3.
3. Write the elements of the prediction vector as functions of $\triangle u(t-1)$, $y(t-1)$, $y(t-2)$ and the future control signals.
4. Simulate the response when the setpoint changes from 0 to 1.

4.2. Given the system $G(s) = \frac{0.5}{1+10s}e^{-2s}$:

1. Obtain the discrete model for a sampling time of 1 second.
2. Use a GPC to steer the output from 0 to 1.5.
3. Consider that there is a model mismatch and the model used by the controller has a time constant of 12 seconds (that is, the first model is used to simulate the process and the second is used to design the controller). Simulate the results with the nominal controller.
4. Add a T polynomial and simulate the results.
5. Use a Smith predictor and simulate the results.

4.3. Design a GPC for the system described by

$$\frac{(0.6445 - 0.7176z^{-1} - 0.0906z^{-2})z^{-2}}{1 - 0.9183z^{-1} + 0.084z^{-2} - 0.002z^{-3}}$$

Try different values for the tuning parameters (horizons and weights) and simulate the results for a setpoint change of 1 unit and a step disturbance at the output of 0.2.

4.4. A DC motor can be modelled by

$$\frac{0.00489z^{-1} + 0.00473z^{-2}}{(1 - z^{-1})(1 - 0.935z^{-1})}$$

where the input is the voltage applied to the motor and the output is the shaft angle.

1. Design a GPC and simulate a setpoint move from 0 to 3.
2. Try to obtain the same results with a DMC. Justify the results.

4.5. Design a GPC for the system described by

$$G(z) = \frac{-1 + 1.2592z^{-1}}{1 - 0.7408z^{-1}}$$

with $N_1 = 1$, $N_2 = 30$, $N_u = 10$ and $\lambda = 0.1$. Change the value of λ to 0 and see the results (for the justification see Section 6.9).

4.6. Given an oscillatory system $G(s) = \frac{50}{s^2+25}$, find the parameters of a GPC such that the overshoot is less than 20%. Is it possible to control this process with a MAC?

5

Simple Implementation of GPC for Industrial Processes

One of the reasons for the success of the traditional PID controllers in industry is that PID are very easy to implement and tune using heuristic tuning rules such as the Ziegler-Nichols rules frequently used in practice. A Generalized Predictive Controller, as shown in the previous chapter, results in a linear control law which is very easy to implement once the controller parameters are known. The derivation of the GPC parameters requires, however, some mathematical complexities such as recursively solving the Diophantine equation, forming the matrices $\mathbf{G}, \mathbf{G}'$ and $\mathbf{f}$ and then solving a set of linear equations. Although this is not a problem for people in the research control community where mathematical packages are normally available, it may be discouraging for practitioners used to much simpler ways of implementing and tuning controllers.

The previously mentioned computation has to be carried out only once when dealing with processes with fixed parameters, but if the process parameters change, the GPC's parameters have to be derived again, perhaps in real time, at every sampling time if a self-tuning control is used. This may be difficult because, on one hand, some distributed control equipment has only limited mathematical computation capabilities for the controllers, and, on the other hand, the computation time required for the derivation of the GPC parameters may be excessive for the sampling time required by the process and the number of loops implemented.

The goal of this chapter is to show how a GPC can very easily be implemented and tuned for a wide range of processes in industry. It will be shown that a GPC can be implemented with a limited set of instructions, available in most control systems, and that the computation time required, even for tuning, is very short. The method to implement the GPC is based on the fact that a wide range of processes in industry can be described by a few parameters and that a set of simple Ziegler-Nichols type of functions relating GPC parameters to process parameters can be obtained. By using these functions the implementation and tuning of a GPC is almost as simple as the implementation and tuning of a PID.

E. F. Camacho et al., *Model Predictive Control*

The influence of modelling errors is also analyzed in this chapter, with a section dedicated to performing a robustness analysis of the method when modelling errors are taken into account.

5.1 Plant Model

Most processes in industry, when considering small changes around an operating point can be described by a linear model of, normally, very high order. This is because most industrial processes are composed of many dynamic elements, usually first order, so the full model is of an order equal to the number of elements. In fact, each mass or energy storage element in the process provides a first-order element in the model. Consider, for instance, a long pipe used for heat exchanging purposes, as the case of a cooler or a steam generator. The pipe can be modelled by breaking it into a set of small pieces, each of which can be considered a first-order system. The resulting model will have an order equal to the number of pieces used to model the pipe, that is, a very high-order model. These very high-order models would be difficult to use for control purposes but, fortunately, it is possible to approximate the behaviour of such high-order processes by a system with one time constant and a dead time.

As shown in [63], we may consider a process having N first-order elements in series, each having a time constant τ/N. That is, the resulting transfer function will be

$$G(s) = \frac{1}{(1 + \frac{\tau}{N}s)^N}$$

Changing the value of N from 1 to ∞ the response shifts from exact first order to pure dead time (equal to τ). When a time constant is much larger than the others (as in many processes) the smaller time constants work together to produce a lag that acts as pure dead time. In this situation the dynamical effects are mainly due to this larger time constant, as can be seen in Figure 5.1. It is therefore possible to approximate the model of a very high-order, complex, dynamic process with a simplified model consisting of a first-order process combined with a dead time element. This type of system can then be described by the following transfer function:

$$G(s) = \frac{K}{1 + \tau s} e^{-s^{\tau_d}} \tag{5.1}$$

where K is the process static gain, τ is the time constant or process lag, and τ_d is the dead time or delay.

5.1.1 Plant Identification: The Reaction Curve Method

Once the model structure is defined, the next step is to choose the correct value for the parameters. In order to identify these parameters, a suitable

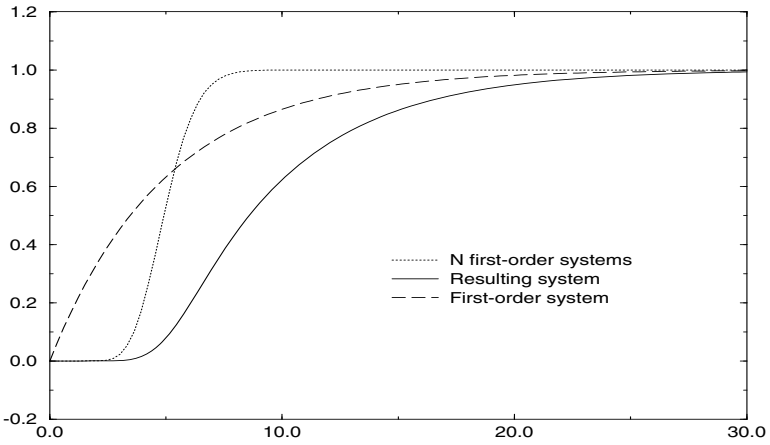


Fig. 5.1. System response

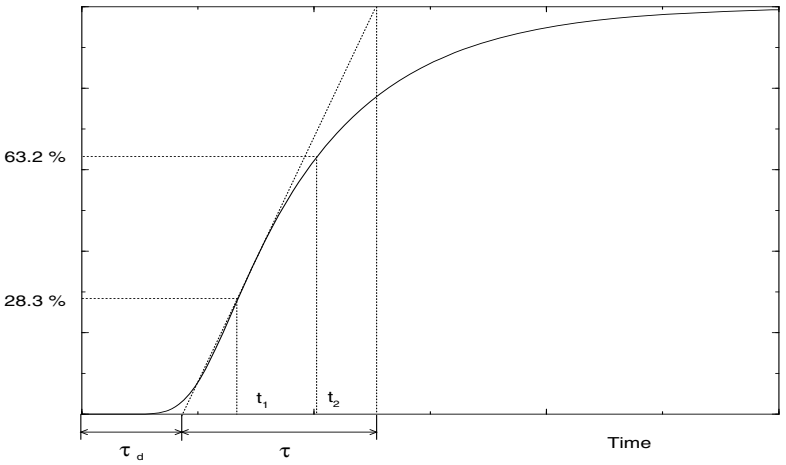


Fig. 5.2. Reaction curve

stimulus must be applied to the process input. In the Reaction Curve Method a step perturbation, that is, an input with a wide frequency content, is applied to the process and the output is recorded to fit the model to the data.

The step response or reaction curve of the process looks like Figure 5.2. Here, a step of magnitude $\triangle u$ is produced in the manipulated variable and the time response of the process variable $y(t)$ is shown. The process parameters of Equation (5.1) can be obtained by measuring two times: t_1, the time when the output reaches 28.3 % of its steady-state value $\triangle y$, and t_2, when the output reaches 63.2 %. Using these values, the process parameters are given by:

$$\begin{aligned} K &= \frac{\triangle y}{\triangle u} \\ \tau &= 1.5(t_2 - t_1) \\ \tau_d &= 1.5(t_1 - \tfrac{1}{3}t_2) \end{aligned}$$

A similar and perhaps more intuitive way of obtaining the process parameters consists of finding the inflection point of the response and drawing the line that represents the slope at that point [95]. The static gain is obtained by the former expression and the times τ and τ_d come directly from the response, as can be seen in the figure. The values obtained are similar in the two approaches.

The Reaction Curve method is probably one of the most popular methods used in industry for tuning regulators, as in the Ziegler-Nichols method for tuning PIDs, and it is used in the pretune stage of some commercial adaptive and auto-tuning regulators.

5.2 The Dead Time Multiple of the Sampling Time Case

5.2.1 Discrete Plant Model

When the dead time τ_d is an integer multiple of the sampling time T ($\tau_d = dT$), the corresponding discrete transfer function of Equation (5.1) has the form

$$G(z^{-1}) = \frac{bz^{-1}}{1 - az^{-1}} z^{-d} \tag{5.2}$$

where discrete parameters a, b and d can easily be derived from the continuous parameters by discretization of the continuous transfer function, resulting in the following expressions:

$$a = e^{-\frac{T}{\tau}} \qquad b = K(1 - a) \qquad d = \frac{\tau_d}{T}$$

If a CARIMA (Controlled Auto-Regressive and Integrated Moving Average) model is used to model the random disturbances in the system and the noise polynomial is chosen to be 1 , the following equation is obtained

$$(1 - az^{-1})y(t) = bz^{-d}u(t-1) + \frac{\varepsilon(t)}{\triangle}$$

where $u(t)$ and $y(t)$ are the control and output sequences of the plant, $\varepsilon(t)$ is a zero mean white noise and $\triangle = 1 - z^{-1}$. This equation can be transformed into:

$$y(t+1) = (1+a)y(t) - ay(t-1) + b \triangle u(t-d) + \varepsilon(t+1) \tag{5.3}$$

5.2.2 Problem Formulation

As was shown in the previous chapter, the Generalized Predictive Control (GPC) algorithm consists of applying a control sequence that minimizes a multistage cost function of the form

$$J(N_1, N_2) = \sum_{j=N_1}^{N_2} \delta(j)[\hat{y}(t+j \mid t) - w(t+j)]^2 + \sum_{j=1}^{N_2-d} \lambda(j)[\triangle u(t+j-1)]^2 \tag{5.4}$$

Notice that the minimum output horizon N_1 should be set to a value greater than the dead time d as the output for smaller time horizons cannot be affected by the first action $u(t)$. In the following N_1 and N_2 will be considered to be $N_1 = d+1$ and $N_2 = d+N$, where N is the control horizon.

If $\hat{y}(t+d+j-1 \mid t)$ and $\hat{y}(t+d+j-2 \mid t)$ are known, it is clear, from Equation (5.3) that the best expected value for $\hat{y}(t+d+j \mid t)$ is given by:

$$\hat{y}(t+d+j \mid t) = (1+a)\hat{y}(t+d+j-1 \mid t) - a\hat{y}(t+d+j-2 \mid t) + b \triangle u(t+j-1) \tag{5.5}$$

If Equation (5.5) is applied recursively for $j = 1, 2, \ldots, i$, we get

$$\hat{y}(t+d+i \mid t) = G_i(z^{-1})\hat{y}(t+d \mid t) + D_i(z^{-1}) \triangle u(t+i-1) \tag{5.6}$$

where $G_i(z^{-1})$ is of degree 1 and $D_i(z^{-1})$ is of degree $i-1$. Notice that when $\delta(i) = 1$ and $\lambda(i) = \lambda$, the polynomials $D_i(z^{-1})$ are equal to the polynomials $G_i(z^{-1})$ given in [58] for the case of $d = 0$ and the terms $f(t+i)$ given in that reference are equal to $G_i(z^{-1})y(t)$ of equation (5.6).

If $\hat{y}(t+d+i \mid t)$ is introduced in Equation (5.4), $J(N)$ is a function of $\hat{y}(t+d \mid t)$, $\hat{y}(t+d-1 \mid t)$, $\triangle u(t+N_2-d-1)$, $\triangle u(t+N_2-d-2)$ ... $\triangle u(t)$ and the reference sequence.

Minimizing $J(N)$ with respect to $\triangle u(t)$, $\triangle u(t+1)$ ... $\triangle u(t+N-1)$ leads to

$$\mathbf{M}\,\mathbf{u} = \mathbf{P}\,\mathbf{y} + \mathbf{R}\,\mathbf{w} \tag{5.7}$$

where

$$\mathbf{u} = [\triangle u(t) \ \triangle u(t+1) \ \cdots \ \triangle u(t+N-1)]^T$$
$$\mathbf{y} = [\hat{y}(t+d \mid t) \ \hat{y}(t+d-1 \mid t)]^T$$
$$\mathbf{w} = [w(t+d+1) \ w(t+d+2) \ \cdots \ w(t+d+N)]^T$$

$\mathbf{M}$ and $\mathbf{R}$ are matrices of dimension $N \times N$ and $\mathbf{P}$ of dimension $N \times 2$. Let us call $\mathbf{q}$ the first row of matrix $\mathbf{M}^{-1}$. Then $\triangle u(t)$ is given by

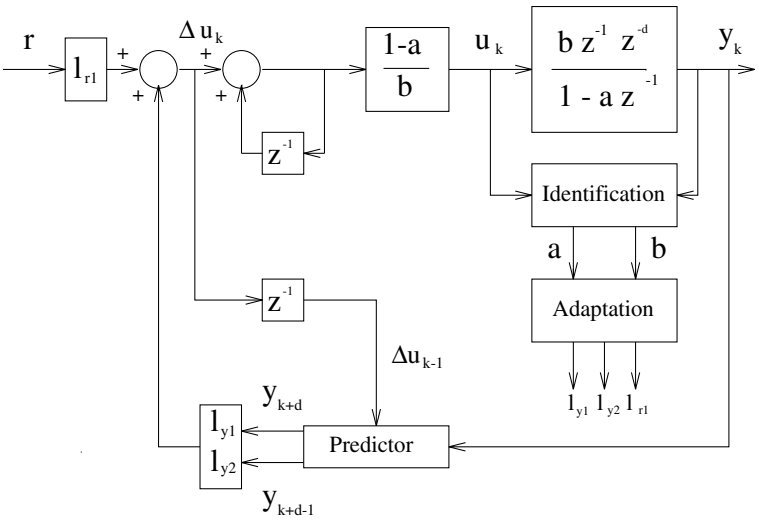


Fig. 5.3. Control Scheme

$$\triangle u(t) = \mathbf{q}\,\mathbf{P}\,\mathbf{y} + \mathbf{q}\,\mathbf{R}\,\mathbf{w} \tag{5.8}$$

When the future setpoints are unknown, $w(t+d+i)$ is supposed to be equal to the current reference $r(t)$. The reference sequence can be written as:

$$\mathbf{w} = [1 \cdots 1] r(t)$$

The control increment $\triangle u(t)$ can be written as

$$\triangle u(t) = l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{r1} r(t) \tag{5.9}$$

where $\mathbf{q}\,\mathbf{P} = [l_{y1} l_{y2}]$ and $l_{r1} = \sum_{i=1}^{N} q_i \sum_{j=1}^{N} r_{ij}$. The coefficients l_{y1}, l_{y2} and l_{r1} are functions of a, b, $\delta(i)$ and $\lambda(i)$. If the GPC is designed considering the plant to have a unit static gain, the coefficients in (5.9) will only depend on $\delta(i)$ and $\lambda(i)$ (which are supposed to be fixed) and on the pole of the plant which will change for the adaptive control case. Notice that by doing this, a normalized weighting factor λ is used and it should be corrected for systems with different static gains.

The resulting control scheme is shown in Figure 5.3. The estimated plant parameters are used to compute the controller coefficients (l_{y1}, l_{y2}, l_{r1}). The values $\hat{y}(t+d \mid t)$, $\hat{y}(t+d-1 \mid t)$ are obtained by the use of the predictor given by equation (5.5). The control signal is divided by the process static gain in order to get a system with a unitary static gain.

Notice that the controller coefficients do not depend on the dead time d and for fixed values of $\delta(i)$ and $\lambda(i)$ they will be a function of the estimated

pole ($\hat{a}$). The standard way of computing the controller coefficients would be by computing the matrices $\mathbf{M}$, $\mathbf{P}$ and $\mathbf{R}$ and solving Equation (5.7) followed by the generation of the control law of Equation (5.9). This involves the triangularization of an $N \times N$ matrix, which could be prohibitive for some real-time applications.

As suggested in [43], the controller coefficients can be obtained by interpolating in a set of previously computed values as shown in Figure 5.4. Notice that this can be accomplished in this case because the controller coefficients only depend on one parameter. The number of points of the set used depends on the variability of the process parameters and on the accuracy needed. The set does not need to be uniform and more points can be computed in regions where the controller parameters vary substantially in order to obtain a better approximation or to reduce the computer memory needed.

The predictor needed in the algorithm to calculate $\hat{y}(t+d \mid t)$, $\hat{y}(t+d-1 \mid t)$ is obtained by applying Equation (5.5) sequentially for $j = 1-d \cdots 0$. Notice that it basically consists of a model of the plant which is projected towards the future with the values of past inputs and outputs, and it only requires straightforward computation.

5.2.3 Computation of the Controller Parameters

The algorithm just described can be used to compute controller parameters of GPC for plants which can be described by Equation (5.2) (most industrial plants can be described this way) over a set covering the region of interest.

Notice that the sampling time of a digital controller is chosen in practice according to the plant time response. Sampling time between $1/15$ and $1/4$ of T_{95} (the time needed by the system to reach 95 % of the final output value) is recommended in [95]. The pole of the discrete form of the plant transfer function is therefore going to vary between 0.5 and 0.95 for most industrial processes when sampled at appropriate rates.

The curves shown in Figure 5.4 correspond to the controller parameters (l_{y1}, l_{y2}, l_{r1}) obtained for $\delta(i) = \delta^i$ and $\lambda(i) = \lambda^i$ with $\delta = 1$, $\lambda = 0.8$ and $N = 15$. The pole of the system has been changed with a 0.0056 step from 0.5 to 0.99. Notice that due to the fact that the closed-loop static gain must be equal to 1, the sum of the three parameters equals zero. This result implies that only two of the three parameters need to be known.

By looking at Figure 5.4 it can be seen that the functions relating the controller parameters to the process pole can be approximated by functions of the form:

$$l_{yi} = k_{1i} + k_{2i}\frac{a}{k_{3i} - a} \qquad i = 1, 2 \tag{5.10}$$

The coefficients k_{ji} can be calculated by a least squares adjustment using the set of known values of l_{yi} for different values of a. Equation (5.10) can be written as:

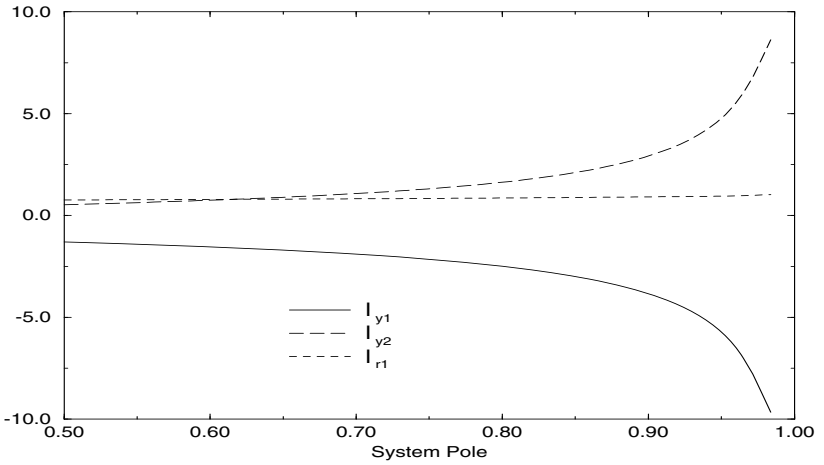


Fig. 5.4. Controller parameters

$$al_{yi} = l_{yi}k_{3i} - k_{1i}k_{3i} + a(k_{i1}a - k_{2i})$$

Repeating this equation for the N_p points used to obtain the approximation, we get

$$\begin{bmatrix} a^1 l_{yi}^1 \\ a^2 l_{yi}^2 \\ \vdots \\ a^{N_p} l_{yi}^{N_p} \end{bmatrix} = \begin{bmatrix} l_{yi}^1 & 1 & a^1 \\ l_{yi}^2 & 1 & a^2 \\ \vdots & \vdots & \vdots \\ l_{yi}^{N_p} & 1 & a^{N_p} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} e^1 \\ e^2 \\ \vdots \\ e^{N_p} \end{bmatrix} \tag{5.11}$$

where l_i^j, a^j, e^j, $j = 1...N_p$ are the N_p values of the system pole, the precalculated parameters, and the approximation errors, $x_1 = k_{3i}$, $x_2 = -k_{1i}k_{3i}$ and $x_3 = k_{1i} - k_{2i}$.

Equation (5.11) can be written in matrix form

$$\mathbf{Y} = \mathbf{M}\,\mathbf{X} + \mathbf{E}$$

In order to calculate the optimum values of X we minimize $E^T E$ obtaining

$$\mathbf{X} = (\mathbf{M}^T\,\mathbf{M})^{-1}\,\mathbf{M}^T\,\mathbf{Y}$$

The desired coefficients can now be evaluated as:

$$\begin{aligned} k_{3i} &= \quad x_1 \\ k_{1i} &= -x_2/k_{3i} \\ k_{2i} &= k_{1i} - x_3 \end{aligned}$$

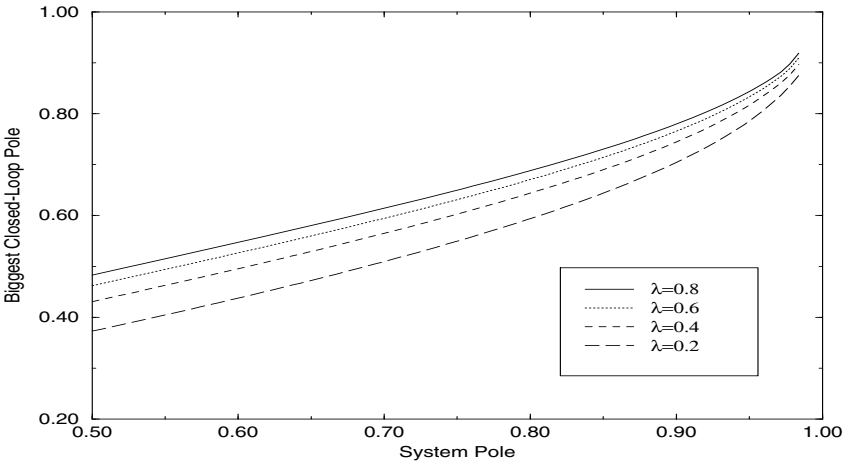


Fig. 5.5. Influence of the control-weighting factor

In the case of $\lambda = 0.8$ and for a control horizon of 15, the controller coefficients are given by:

$$\begin{aligned} l_{y1} &= -0.845 - 0.564\frac{a}{1.05 - a} \\ l_{y2} &= 0.128 + 0.459\frac{a}{1.045 - a} \\ l_{r1} &= -l_{y1} - l_{y2} \end{aligned}$$

These expressions give a very good approximation to the true controller parameters and fit the set of computed data with a maximum error of less than 1 % of the nominal value for the range of interest of the open-loop pole.

5.2.4 Role of the Control-weighting Factor

The control-weighting factor λ affects the control signal in Equation (5.9). The bigger this value is, the smaller the control effort is allowed to be. If it is given a small value, the system response will be fast since the controller tends to minimize the error between the output and the reference, forgetting about control effort. The controller parameters l_{y1}, l_{y2} and l_{r1} and therefore the closed-loop poles depend on the values of λ.

Figure 5.5 shows the value of the modulus of the biggest closed-loop pole when changing λ from 0 to 0.8 by increments of 0.2. As can be seen, the modulus of the biggest closed-loop pole decreases with λ, indicating faster systems. For a value of λ equal to 0, the closed-loop poles are zero, indicating deadbeat behaviour.

A set of functions was obtained by making λ change from 0.3 to 1.1 by increments of 0.1. It was found that the values of the parameters $k_{ij}(\lambda)$ of

Equation (5.10) obtained could be approximated by functions with the form: $\text{sgn}(k_{ij})e^{c_0+c_1\lambda+c_2\lambda^2}$.

By applying logarithm the coefficients c_1, c_2 and c_3 can be adjusted using a polynomial fitting procedure. The following expressions were obtained for the grid of interest:

$$\begin{aligned} k_{11} &= -e^{0.3598-0.9127\lambda+0.3165\lambda^2} \\ k_{21} &= -e^{0.0875-1.2309\lambda+0.5086\lambda^2} \\ k_{31} &= 1.05 \\ k_{12} &= e^{-1.7383-0.40403\lambda} \\ k_{22} &= e^{-0.32157-0.8192\lambda+0.3109\lambda^2} \\ k_{32} &= 1.045 \end{aligned} \tag{5.12}$$

The values of the control parameters l_{y1} and l_{y2} obtained when introducing the k_{ij} given in Equation (5.10) are a very good approximation to the real ones. The maximum relative error for $0.55 < a < 0.95$ and $0.3 \leq \lambda \leq 1.1$ is less than 3 %.

5.2.5 Implementation Algorithm

Once the λ factor has been decided, the values k_{ij} can very easily be computed by Expressions (5.12) and the approximate adaptation laws given by Equation (5.10) can easily be employed. The proposed algorithm in the adaptive case is:

0. Compute k_{ij} with Expressions (5.12).
1. Perform an identification step.
2. Make $l_i = k_{1i} + k_{2i}\frac{\hat{a}}{k_{3i}-\hat{a}}$ for $i = 1, 2$ and $l_{r1} = -l_{y1} - l_{y2}$.
3. Compute $\hat{y}(t+d \mid t)$ and $\hat{y}(t+d-1 \mid t)$ using equation (5.5) recursively.
4. Compute control signal $u(t)$ with $\triangle u(t) = l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{r1}r(t)$
5. Divide the control signal by the static gain.
6. Go to step 1.

Notice that in a fixed-parameter case the algorithm is simplified since the controller parameters need to be computed only once (unless the control-weighting factor λ is changed) and only steps 3 and 4 have to be carried out at every sampling time.

5.2.6 An Implementation Example

In order to show the straightforwardness of this method, an application to a typical process such as a simple furnace is presented. First, identification

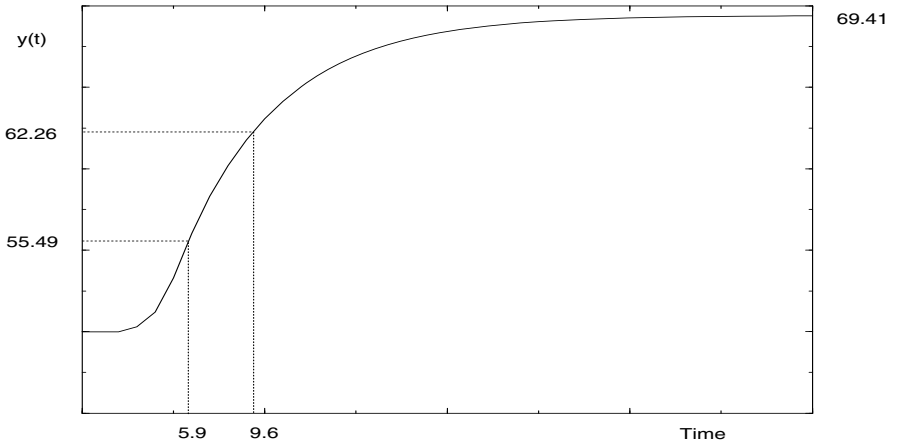


Fig. 5.6. Outlet temperature response

by means of the Reaction Curve method is performed and then the precalculated GPC is applied. The process basically consists of a water flow being heated by fuel which can be manipulated by a control valve. The output variable is the coil outlet temperature whereas the manipulated variable is the fuel flow.

The stationary values of the variables are: inlet temperature $20\ ^oC$, outlet temperature $50\ ^oC$ and fuel valve at 18.21 %. Under these conditions, the fuel rate is changed to a value of 30 % and the outlet temperature response is shown in Figure 5.6, reaching a final value of $y = 69.41\ ^oC$. The plant parameters can be obtained directly from this response as was explained in Section 3.1.1. First, the times t_1 (when the response reaches 28.3 % of its final value) and t_2 (63.2 %) are obtained, resulting in $t_1 = 5.9$ and $t_2 = 9.6$ seconds. Then the plant parameters are

$$\begin{aligned} K &= \frac{\triangle y}{\triangle u} = \frac{69.411 - 50}{30 - 18.21} = 1.646 \\ \tau &= 1.5(t_2 - t_1) = 1.5(9.6 - 5.9) = 5.55 \\ \tau_d &= 1.5(t_1 - \tfrac{1}{3}t_2) = 1.5(5.9 - \tfrac{9.6}{3}) = 4.05 \end{aligned}$$

with these values, and a sampling time of one second, the equivalent discrete transfer function results in

$$G(z^{-1}) = \frac{0.2713z^{-1}}{1 - 0.8351z^{-1}} z^{-4}$$

As the process is considered to have fixed parameters, the controller coefficients l_{y1}, l_{y2} and l_{r1} can be calculated offline. In this case, choosing $\lambda = 0.8$, the coefficients are obtained from Equation (5.12).

```
/* Predictor */ for (i=2; i<=5; i++)
y[i]=1.8351*y[i-1]-0.8351*y[i-2]+0.2713*(u[5-i]-u[6-i]);

/* Control Law */
u[0]=u[1]+(-3.0367*y[5]+1.9541*y[4]+1.08254*r)/1.646;

/* Updating */ for (i=5; i>0; i--)   u[i] = u[i-1];
y[0] = y[1];
```

Fig. 5.7. C code of implementation algorithm

$$
\begin{aligned}
k_{11} &= -0.845 \\
k_{21} &= -0.564 \\
k_{31} &= \ \ 1.05 \\
k_{12} &= \ \ 0.128 \\
k_{22} &= \ \ 0.459 \\
k_{32} &= \ \ 1.045
\end{aligned}
$$

In consequence the controller coefficients are:

$$l_{y1} = -3.0367 \qquad l_{y2} = 1.9541 \qquad l_{r1} = 1.0826$$

Therefore, at every sampling time it will only be necessary to compute the predicted outputs and the control law. The predictions needed are $\hat{y}(t+4 \mid t)$ and $\hat{y}(t+3 \mid t)$, that will be calculated from the next equation with $i = 1 \cdots 4$

$$\hat{y}(t+i \mid t) = (1+a)\hat{y}(t+i-1 \mid t) - a\hat{y}(t+i-2 \mid t) + b \triangle u(t+i-5)$$

and the control law, where G is the static gain

$$u(t) = u(t-1) + (l_{y1}\ \hat{y}(t+4 \mid t) + l_{y2}\ \hat{y}(t+3 \mid t) + l_{r1}\ r)/G$$

The implementation of the controller in a digital computer will result in a simple program, a part of whose code written in *C* is shown in Figure 5.7. Two arrays, u and y, are used. The first is used to store the past values of the control signal and the second to store the values of the outputs. In this process with a dead time of 4, the arrays are:

$$
\begin{aligned}
\mathbf{y} &= [\ y(t-1), y(t), \hat{y}(t+1), \hat{y}(t+2), \hat{y}(t+3), \hat{y}(t+4)\ ] \\
\mathbf{u} &= [\ u(t), u(t-1), u(t-2), u(t-3), u(t-4), u(t-5)\ ]
\end{aligned}
$$

Notice that $\mathbf{y}$ contains the predicted outputs and the outputs in t and $t-1$, because these last two values are needed in the first two predictions $\hat{y}(t+1), \hat{y}(t+2)$.

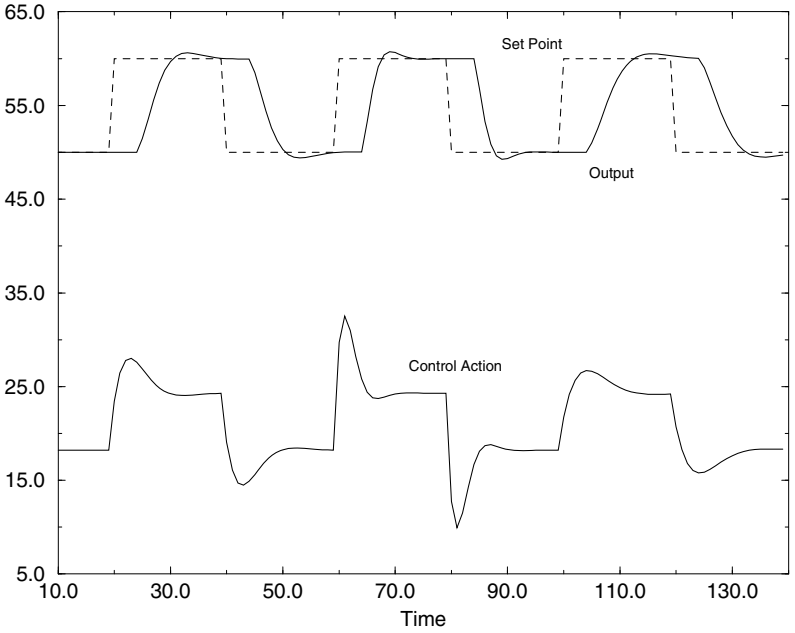


Fig. 5.8. System response

The closed-loop response to a setpoint change of $+10\ ^0$C is plotted in Figure 5.8, where the evolution of the control signal can also be seen . Additionally, the control-weighting factor λ was changed at $t = 60$ from the original value of 0.8 to a smaller 0.3; notice that the control effort increases and the output tends to be faster. On the other hand, if λ takes a bigger value, such as 1.3, the behaviour tends to be slower; this change was performed at $t = 100$.

5.3 The Dead Time Nonmultiple of the Sampling Time Case

5.3.1 Discrete Model of the Plant

When the dead time τ_d of the process is not an integer multiple of the sampling time T ($dT \leq \tau_d \leq (d+1)T$), Equation (5.2) cannot be employed. In this case the fractional delay time can be approximated [66] by the first two terms of the Padé expansion and the plant discrete transfer function can be written as:

$$G(z^{-1}) = \frac{b_0 z^{-1} + b_1 z^{-2}}{1 - a z^{-1}} z^{-d} \tag{5.13}$$

As can be seen this transfer function is slightly different from the previous model (Equation(5.2)), presenting an additional zero; a new parameter

appears in the numerator. Using the same procedure as in the previous case, a similar implementation of GPC can be obtained for this family of processes.

To obtain the discrete parameters a, b_0 and b_1, the following relations can be used [66]: first, the dead time is decomposed as $\tau_d = dT + \epsilon T$ with $0 < \epsilon < 1$. Then the parameters are:

$$a = e^{-\frac{T}{\tau}} \qquad b_0 = K(1-a)(1-\alpha) \qquad b_1 = K(1-a)\alpha \qquad \alpha = \frac{a(a^{-\epsilon}-1)}{1-a}$$

Since the derivation of the control law is very similar in this case to the case in the previous section, some steps will be omitted here for simplicity.

The function J to be minimized is also that of Equation (5.4). Using the CARIMA model with the noise polynomial equal to 1, the system can be written as

$$(1 - az^{-1})y(t) = (b_0 + b_1 z^{-1})z^{-d}u(t-1) + \frac{\varepsilon(t)}{\triangle}$$

which can be transformed into:

$$y(t+1) = (1+a)y(t) - ay(t-1) + b_0 \triangle u(t-d) + b_1 \triangle u(t-d-1) + \varepsilon(t+1) \quad (5.14)$$

If $\hat{y}(t+d+i-1 \mid t)$ and $\hat{y}(t+d+i-2 \mid t)$ are known, it is clear, from Equation (5.14), that the best expected value for $\hat{y}(t+d+i \mid t)$ is given by:

$$\begin{aligned}\hat{y}(t+d+i \mid t) = (1+a)\hat{y}(t+d+i-1 \mid t) - a\hat{y}(t+d+i-2 \mid t) \\ + b_0 \triangle u(t+i-1) + b_1 \triangle u(t+i-2)\end{aligned}$$

If $\hat{y}(t+d+i \mid t)$ is introduced in the function to be minimized, $J(N)$ is a function of $\hat{y}(t+d \mid t)$, $\hat{y}(t+d-1 \mid t)$, $\triangle u(t+N_2-d-1)$, $\triangle u(t+N_2-d-2)$ ... $\triangle u(t)$, $\triangle u(t-1)$ and the reference sequence.

Minimizing $J(N)$ with respect to $\triangle u(t)$, $\triangle u(t+1)$ ... $\triangle u(t+N-1)$ leads to

$$\mathbf{M}\,\mathbf{u} = \mathbf{P}\,\mathbf{y} + \mathbf{R}\,\mathbf{w} + \mathbf{Q} \triangle u(t-1) \quad (5.15)$$

where

$$\begin{aligned}\mathbf{u} &= [\, \triangle u(t) \;\; \triangle u(t+1) \;\cdots\; \triangle u(t+N-1) \,]^T \\ \mathbf{y} &= [\, \hat{y}(t+d \mid t) \; \hat{y}(t+d-1 \mid t) \,]^T \\ \mathbf{w} &= [\, w(t+d+1) \; w(t+d+2) \;\cdots\; w(t+d+N) \,]^T\end{aligned}$$

$\mathbf{M}$ and $\mathbf{R}$ are matrices of dimension $N \times N$, $\mathbf{P}$ of dimension $N \times 2$ and $\mathbf{Q}$ of $N \times 1$.

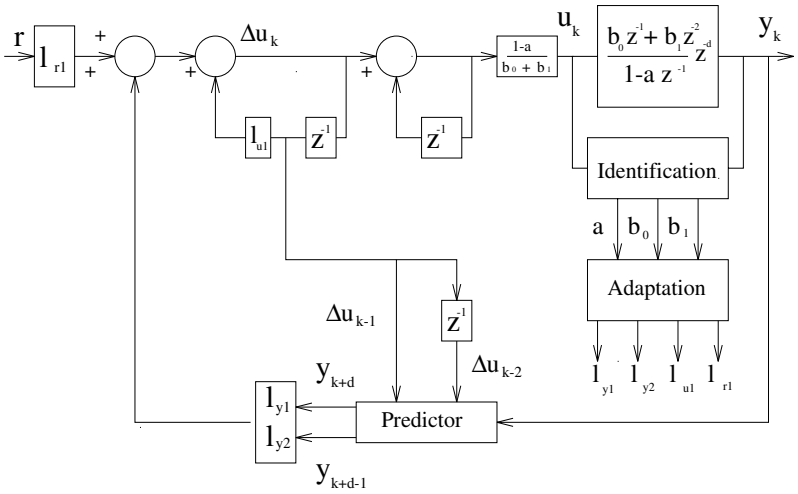


Fig. 5.9. Control scheme

Notice that the term $\mathbf{Q} \triangle u(t-1)$ did not appear in the simpler plant because of the different plant parameters; hence the control law will not be the same. Let us call $\mathbf{q}$ the first row of matrix $\mathbf{M}^{-1}$. Then $\triangle u(t)$ is given by

$$\triangle u(t) = \mathbf{q}\,\mathbf{P}\,\mathbf{y} + \mathbf{q}\,\mathbf{R}\,\mathbf{w} + \mathbf{q}\,\mathbf{Q} \triangle u(t-1) \tag{5.16}$$

If the reference sequence is considered to be $\mathbf{w} = [1 \cdots 1]r(t)$, the control increment $\triangle u(t)$ can be written as:

$$\triangle u(t) = l_{y1}\,\hat{y}(t+d \mid t) + l_{y2}\,\hat{y}(t+d-1 \mid t) + l_{r1}\,r(t) + l_{u1} \triangle u(t-1) \tag{5.17}$$

where $\mathbf{q}\,\mathbf{P} = [l_{y1}\ l_{y2}]$, $l_{r1} = \sum_{i=1}^{N} q_i \sum_{j=1}^{N} r_{ij}$ and $l_{u1} = \mathbf{q}\,\mathbf{Q}$. The resulting control scheme is shown in Figure 5.9, where the values $\hat{y}(t+d \mid t)$, $\hat{y}(t+d-1 \mid t)$ are obtained by use of the predictor previously described. Notice that the predictor basically consists of a model of the plant projected towards the future with the values of past inputs and outputs.

5.3.2 Controller Parameters

The plant estimated parameters can be used to compute the controller coefficients (l_{y1}, l_{y2}, l_{r1} and l_{u1}). These coefficients are functions of the plant parameters a, b_0, b_1, $\delta(i)$ and $\lambda(i)$. If the GPC is designed considering the plant to have a unit static gain, there exists a relationship between the plant parameters

$$1 - a = b_0 + b_1$$

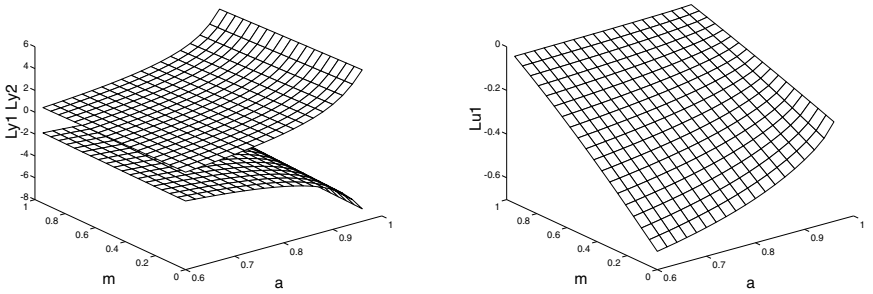


Fig. 5.10. Controller parameters l_{y1}, l_{y2} and l_{u1} as functions of a and m

so that only two of the three parameters will be needed to calculate the coefficients in (5.17). One parameter will be the system pole a and the other will be:

$$m = \frac{b_0}{b_0 + b_1}$$

Parameter m indicates how close the true dead time is to parameter d in the model used in Equation (5.13). That is, if $m = 1$ the plant dead time is d and if $m = 0$ it is $d + 1$; so a range of m between 1 and 0 covers the fractional dead times between d and $d + 1$.

Once the values of $\delta(i)$ and $\lambda(i)$ have been chosen and the plant parameters are known, the controller coefficients can easily be derived. The control signal is divided by the process static gain in order to get a system with a unitary static gain and reduce the number of parameters.

In order to avoid the heavy computational requirements needed to compute matrices $\mathbf{M}$, $\mathbf{P}$, $\mathbf{R}$ and $\mathbf{Q}$ and solve Equation (5.15), the coefficients can be obtained by interpolating in a set of previously computed values as shown in Figure 5.10. Notice that this can be accomplished in this case because the controller coefficients only depend on two parameters. As they have been obtained considering a unitary static gain, they must be corrected dividing the coefficients l_{y1}, l_{y2} and l_{r1} by this value.

The algorithm just described can be used to compute controller parameters of GPC for plants which can be described by Equation (5.13) over a set covering the region of interest. This region is defined by values of the pole in the interval $[0.5, 0.95]$ and the other plant parameter m that will vary between 0 and 1.

The curves shown in Figure 5.10 correspond to the controller parameters l_{y1}, l_{y2} and l_{u1} for $\delta(i) = \delta^i$ and $\lambda(i) = \lambda^i$ with $\delta = 1$, $\lambda = 0.8$ and $N = 15$. Notice that because the closed-loop static gain must equal the value 1, the sum of parameters l_{y1}, l_{y2} and l_{r1} equals zero. This result implies that only three of the four parameters need to be known.

The expressions relating the controller parameters to the process parameters can be approximated by functions of the form:

$$k_{1i}(m) + k_{2i}(m)\frac{a}{k_{3i}(m) - a} \tag{5.18}$$

The coefficients $k_{ji}(m)$ depend on the value of m and can be calculated by a least squares fitting using the set of known values of l_{yi} for different values of a and m. Low-order polynomials that give a good approximation for $k_{ji}(m)$ have been obtained by Bordons [30].

In the case of $m = 0.5$, λ = 0.8 and for a control horizon of 15, the controller coefficients are given by:

$$\begin{aligned} l_{y1} &= -0.9427 - 0.5486\frac{a}{1.055 - a} \\ l_{y2} &= 0.1846 + 0.5082\frac{a}{1.0513 - a} \\ l_{u1} &= -0.3385 + 0.0602\frac{a}{1.2318 - a} \\ l_{r1} &= -l_{y1} - l_{y2} \end{aligned}$$

These expressions give a very good approximation of the true controller parameters and fit the set of computed data with a maximum error of less than 2 % of the nominal values for the range of interest of plant parameters.

The influence of the control-weighting factor λ on the controller parameters can also be taken into account. For small values of λ, the parameters are bigger so that they produce a bigger control effort, thus this factor has to be considered in the approximate functions. With a procedure similar to that in previous sections, the values of $k_{ij}(m)$ in Expressions (5.18) can be approximated as functions of λ, obtaining a maximum error of around 3 % (with the worst cases at the limits of the region).

The algorithm in the adaptive case will consider the plant parameters and the control law and can be seen here:

1. Perform an identification step.
2. Compute $k_{ij}(m, \lambda)$.
3. Calculate l_{y1}, l_{y2} and l_{u1}. Make $l_{r1} = -l_{y1} - l_{y2}$.
4. Compute $\hat{y}(t + d \mid t)$ and $\hat{y}(t + d - 1 \mid t)$ using equation (5.5) recursively.
5. Compute $u(t)$ with:
   $\triangle u(t) = (l_{y1}\ \hat{y}(t + d \mid t) + l_{y2}\ \hat{y}(t + d - 1 \mid t)$
   $+ l_{r1}\ r(t))/G + l_{u1} \triangle u(t - 1)$
6. Go to step 1.

5.3.3 Example

This example is taken from [196] and corresponds to the distillate composition loop of a binary distillation column. The manipulated variable is the reflux flow rate and the controlled variable is the distillate composition. Although the process is clearly nonlinear, it can be modelled by a first-order model plus a dead time of the form $G(s) = Ke^{-s^{\tau_d}}/(1+\tau s)$ at different operating points. Notice that this is a reasonable approach since, in fact, a distillation column is composed of a number of plates, each being a first-order element.

As the process is nonlinear, the response varies for different operating conditions, so different values for the parameters K, τ and τ_d were obtained, changing the reflux flow rate from 3.5 to 4.5 mol/min (see Table 5.1). By these tests, it was seen that variations in the process parameters as $0.107 \leq K \leq 0.112$, $15.6 \leq \tau_d \leq 16.37$, and $40.49 \leq \tau \leq 62.8$ should be considered (where τ and τ_d are in minutes).

Considering a sample time of five minutes the dead time is not an integer and the discrete transfer function must be that of equation (5.13). The discrete parameters can be seen in the same table. The system pole can vary between 0.8838 and 0.9234, while parameter m is going to move between 0.7381 and 0.8841.

Table 5.1. Process parameters for different operating conditions

Flow	K	τ	τ_d	a	$b_0(\times 10^{-3})$	$b_1(\times 10^{-3})$	d
4.5	0.107	62.8	15.6	0.9234	7.2402	0.9485	3
4	0.112	46.56	15.65	0.8981	9.9901	1.4142	3
3.5	0.112	40.49	16.37	0.8838	9.6041	3.4067	3

To cope with the variations of the system dynamics depending on the operating point, a gain-scheduling predictive controller can be developed. To do this, a set of controller parameters is obtained for each operating point, and depending on the reflux flow, the parameters in each condition will be calculated by interpolating in these sets.

For a fixed value of λ, the coefficients $k_{ij}(m)$ are used to calculate the values $l_i(a, m)$. For $\lambda = 0.8$, the following expressions can be used:

$$
\begin{aligned}
k_{11} &= 0.141 * m^2 - 0.125 * m - 0.920 \\
k_{21} &= -0.061 * m^2 + 0.202 * m - 0.625 \\
k_{31} &= -0.015 * m + 1.061 \\
k_{12} &= -0.071 * m^2 + 0.054 * m + 0.180 \\
k_{22} &= -0.138 * m + 0.575 \\
k_{32} &= -0.015 * m + 1.058 \\
k_{14} &= -0.115 * m^2 + 0.847 * m - 0.729 \\
k_{24} &= -0.113 * m + 0.112 \\
k_{34} &= -0.071 * m^2 + 0.091 * m + 1.181
\end{aligned}
$$

The GPC parameters for the different operating conditions are given in Table 5.2.

Table 5.2. GPC parameters for different operating conditions

Flow	l_{y1}	l_{y2}	l_{r1}	l_{u1}
4.5	-4.577	3.629	0.948	-0.035
4	-3.881	2.954	0.927	-0.039
3.5	-3.637	2.745	0.892	-0.091

As the dead time is of three sampling periods, the predictor is

$$\hat{y}(t+i \mid t) = (1+a)\hat{y}(t+i-1 \mid t) - a\hat{y}(t+i-2 \mid t) + b_0 \triangle u(t+i-3) + b_1 \triangle u(t+i-4)$$

and the control law is

$$u(t) = u(t-1) + (l_{y1}\hat{y}(t+3 \mid t) + l_{y2}\hat{y}(t+2 \mid t) + l_{r1}r)/G + l_{u1} \triangle u(t-1)$$

where G is the static gain and the controller and predictor parameters are obtained by interpolating the flow, once filtered by a low-pass filter, in Table 5.2.

To show the system behaviour, changes of the setpoint covering the complete region are produced, and it can be observed in Figure 5.11 that the output follows the reference, in spite of the changes in the system dynamics due to the changing operating point.

5.4 Integrating Processes

In industrial practice it is easy to find some processes including an integral effect. The output of one of these processes grows infinitely when excited by a step input. This is the case of a tank where the level increases, provided there is an input flow and a constant output. Also the angle of an electrical

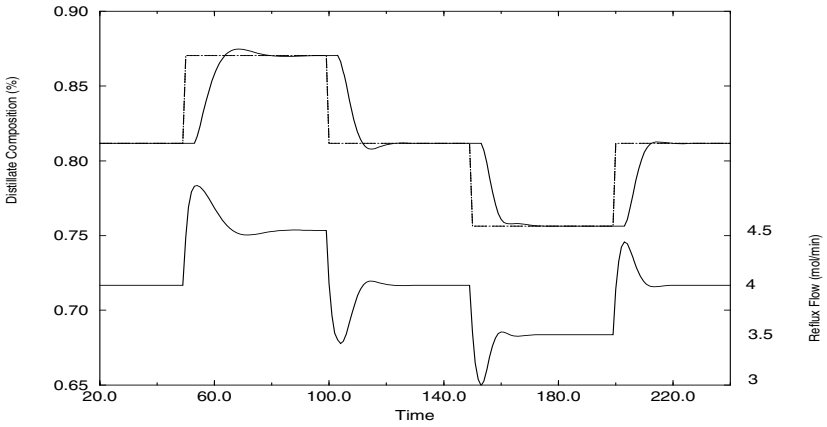


Fig. 5.11. Changes in operating conditions

motor shaft grows while being powered until the torque equals the load. The behaviour of these processes differs drastically from that of those considered up to now in this chapter.

These processes cannot be modelled by a first-order-plus-delay transfer function, but they need the addition of an $1/s$ term to model the integrating effect. Hence, the transfer function for this kind of process will be:

$$G(s) = \frac{K}{s(1+\tau s)} e^{-\tau_d s} \tag{5.19}$$

In the general case of dead time being nonmultiple of the sampling time the equivalent discrete transfer function when a zero-order hold is employed is given by:

$$G(z) = \frac{b_0 z^{-1} + b_1 z^{-2} + b_2 z^{-3}}{(1-z^{-1})(1-az^{-1})} z^{-d} \tag{5.20}$$

In the simpler case of the dead time being an integer multiple of the sampling time, the term b_2 disappears.

The GPC control law for processes described by (5.19) will be calculated in this section. Notice that some formulations of MPC are unable to deal with these processes since they use the truncated impulse or step response, which is not valid for unstable processes. As GPC makes use of the transfer function, there is no problem about unstable processes.

5.4.1 Derivation of the Control Law

The procedure for obtaining the control law is analogous to the one used in previous sections, although logically the predictor will be different and the final expression will change slightly.

Using a CARIMA model with the noise polynomial equal to 1, the system can be written as

$$(1-z^{-1})(1-az^{-1})y(t) = (b_0+b_1z^{-1}+b_2z^{-2})z^{-d}u(t-1)+\frac{\varepsilon(t)}{\triangle}$$

which can be transformed into:

$$\begin{aligned} y(t+1) &= (2+a)y(t)-(1+2a)y(t-1)+ay(t-2) \\ &\quad + b_0\triangle u(t-d)+b_1\triangle u(t-d-1)+b_2\triangle u(t-d-2)+\varepsilon(t+1) \end{aligned}$$

If the values of $\hat{y}(t+d+i-1\mid t)$, $\hat{y}(t+d+i-2\mid t)$ and $\hat{y}(t+d+i-3\mid t)$ are known, then the best predicted output at instant $t+d+i$ will be:

$$\hat{y}(t+d+i\mid t) = (2+a)\hat{y}(t+d+i-1\mid t)-(1+2a)\hat{y}(t+d+i-2\mid t)+$$
$$a\hat{y}(t+d+i-3\mid t)+b_0\triangle u(t+i-1)+b_1\triangle u(t+i-2)+b_2\triangle u(t+i-3)$$

With these expressions of the predicted outputs, the cost function to be minimized will be a function of $\hat{y}(t+d\mid t)$, $\hat{y}(t+d-1\mid t)$ and $\hat{y}(t+d-2\mid t)$, as well as the future control signals $\triangle u(t+N-1)$, $\triangle u(t+N-2)$ ... $\triangle u(t)$, and past inputs $\triangle u(t-1)$ and $\triangle u(t-2)$ and, of course, of the reference trajectory.

Minimizing $J(N_1,N_2,N_3)$ leads to the following matrix equation for calculating $\mathbf{u}$

$$\mathbf{M}\,\mathbf{u} = \mathbf{P}\,\mathbf{y} + \mathbf{R}\,\mathbf{w} + \mathbf{Q}_1\,\triangle u(t-1) + \mathbf{Q}_2\,\triangle u(t-2)$$

where $\mathbf{M}$ and $\mathbf{R}$ are matrices of dimension $N\times N$, $\mathbf{P}$ of dimension $N\times 2$ and $\mathbf{Q}_1$ and $\mathbf{Q}_2$ of $N\times 1$. As in the previous section, $\mathbf{u}$ are the future input increments and $\mathbf{y}$ the predicted outputs.

The first element of vector $\mathbf{u}$ can be obtained by

$$\triangle u(t) = \mathbf{q}\,\mathbf{P}\,\mathbf{y}+\mathbf{q}\,\mathbf{R}\,\mathbf{w}+\mathbf{q}\,\mathbf{Q}_1\triangle u(t-1)+\mathbf{q}\,\mathbf{Q}_2\triangle u(t-2)$$

where $\mathbf{q}$ is the first row of matrix $\mathbf{M}^{-1}$.

If the reference is considered to be constant over the prediction horizon and equal to the current setpoint

$$\mathbf{w} = [1\ldots 1]r(t+d)$$

the control law results as

$$\begin{aligned} \triangle u(t) &= l_{y1}\hat{y}(t+d\mid t)+l_{y2}\hat{y}(t+d-1\mid t)+l_{y3}\hat{y}(t+d-2\mid t) \\ &\quad +l_{r1}r(t+d)+l_{u1}\triangle u(t-1)+l_{u2}\triangle u(t-2) \end{aligned} \tag{5.21}$$

where $\mathbf{q}\,\mathbf{P} = [l_{y1}\ l_{y2}\ l_{y3}]$, $l_{r1} = \sum_{i=1}^{N}(q_i\sum_{j=1}^{N}r_{ij})$, $l_{u1} = \mathbf{q}\,\mathbf{Q}_1$ and $l_{u2} = \mathbf{q}\,\mathbf{Q}_2$.

Therefore the control law results in a linear expression depending on six coefficients which depend on the process parameters (except on the dead time) and on the control-weighting factor λ. Furthermore, one of these coefficients is a linear combination of the others, since the following relation must hold to get a closed loop with unitary static gain:

$$l_{y1}+l_{y2}+l_{y3}+l_{r1} = 0$$

5.4.2 Controller Parameters

The control law (5.21) is very easy to implement provided the controller parameters l_{y1}, l_{y2}, l_{y3}, l_{r1}, l_{u1} and l_{u2} are known. The existence of available relationships of these parameters with process parameters is of crucial importance for a straightforward implementation of the controller. In a similar way to the previous sections, simple expressions for these relationships will be obtained.

As the process can be modelled by (5.20) four parameters (a, b_0, b_1 and b_2) are needed to describe the plant. Expressions relating the controller coefficients to these parameters can be obtained as earlier, although the resulting functions are not as simple, due to the number of plant parameters involved. As the dead time can often be considered as a multiple of the sampling time, simple functions will be obtained for this case from now on. Then b_2 will be considered equal to 0.

In a similar way to the process without integrator case, the process can be considered to have $(b_0+b_1)/(1-a)=1$ in order to work with normalized plants. Then the computed parameters must be divided by this value that will not equal 1 in general.

The controller coefficients will be obtained as a function of the pole a and a parameter:

$$n=\frac{b_0}{b_0+b_1}$$

This parameter has a short range of variability for any process. As b_0 and b_1 are related to the continuous parameters by (see [11])

$$b_0=K(T+\tau(-1+e^{-\frac{T}{\tau}})) \qquad b_1=K(\tau-e^{-\frac{T}{\tau}}(T+\tau))$$

then

$$n=\frac{a-1-\log a}{(a-1)\log a}$$

that for the usual values of the system pole is going to vary between $n=0.5$ and $n=0.56$. Therefore the controller parameters can be expressed as functions of the system pole, and n for a fixed value of λ.

The shape of the parameters is displayed in Figure 5.12 for a fixed value of $\lambda=1$. It can be seen that the coefficients depend mainly on the pole a, being almost independent of n except in the case of l_{u1}. Functions of the form

$$f(a,n,\lambda)=k_1(n,\lambda)+k_2(n,\lambda)\frac{a}{k_3(n,\lambda)-a}$$

where k_i can be approximated by

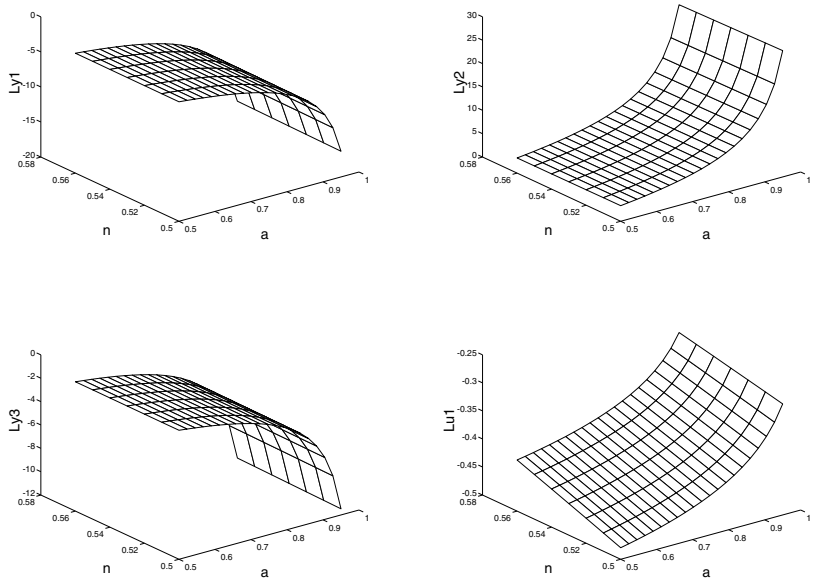


Fig. 5.12. Controller coefficients l_{y1}, l_{y2}, l_{y3} and l_{u1}

$$
\begin{aligned}
k_{y1,1} &= -e^{0.955-0.559\lambda+0.135\lambda^2} \\
k_{y1,2} &= -e^{0.5703-0.513\lambda+0.138\lambda^2} \\
k_{y1,3} &= 1.0343 \\
k_{y2,1} &= e^{0.597-0.420\lambda+0.0953\lambda^2} \\
k_{y2,2} &= e^{1.016-0.4251\lambda+0.109\lambda^2} \\
k_{y2,3} &= 1.0289 \\
k_{y3,1} &= -e^{-1.761-0.422\lambda+0.071\lambda^2} \\
k_{y3,2} &= -e^{0.103-0.353\lambda+0.089\lambda^2} \\
k_{y3,3} &= 1.0258 \\
k_{u1,1} &= 1.631n - 1.468 + 0.215\lambda - 0.056\lambda^2 \\
k_{u1,2} &= -0.124n + 0.158 - 0.026\lambda + 0.006\lambda^2 \\
k_{u1,3} &= 1.173 - 0.019\lambda
\end{aligned}
\tag{5.22}
$$

provide good approximations for l_{y1}, l_{y2}, l_{y3}, l_{r1} and l_{u1} in the usual range of the plant parameter variations. Notice that an approximate function for l_{r1} is not supplied since it is linearly dependent on the other coefficients. The functions fit the set of computed data with a maximum error of less than 1.5 % of the nominal values. Notice that closer approximations can be obtained if developed for a concrete case where the range of variability of the process parameters is smaller.

5.4.3 Example

The control law (5.21) will be implemented in an extensively used system as a direct-current motor. When the input of the process is the voltage applied to the motor (U) and the output is the shaft angle (θ) it is obvious that the process has an integral effect, given that the position grows indefinitely whilst it is fed by a certain voltage. In order to obtain a model that describes the behaviour of the motor the inertia load (proportional to the angular acceleration) and the dynamic friction load (proportional to angular speed) are taken into account. Their sum is equal to the torque developed by the motor, which depends on the voltage applied to it. It is a first-order system with regards to speed but a second-order one if the angle is considered as the output of the process:

$$J\frac{d^2\theta}{dt^2} + f\frac{d\theta}{dt} = M_m$$

and the transfer function will be:

$$\frac{\theta(s)}{U(s)} = \frac{K}{s(1+\tau s)}$$

where K and τ depend on electromechanical characteristics of the motor.

The controller is going to be implemented on a real motor with a feed voltage of 24 V and nominal current of 1.3 A, subjected to a constant load. The Reaction Curve Method is used to obtain experimentally the parameters of the motor, applying a step in the feed voltage and measuring the evolution of the angular speed (which is a first-order system). The parameters obtained are

$$K = 2.5 \qquad \tau = 0.9 \text{ second}$$

and zero dead time. Taking a sampling time of $T = 0.06$ second one gets the discrete transfer function:

$$G(z) = \frac{0.004891z^{-1} + 0.004783z^{-2}}{(1-z^{-1})(1-0.935507z^{-1})}$$

If a high value of the control-weighting factor is taken to avoid overshooting ($\lambda = 2$) the control parameters (5.21) can be calculated using expressions (5.22):

$$\begin{aligned} l_{y1} &= -11.537 \\ l_{y2} &= 19.242 \\ l_{y3} &= -8.207 \\ l_{u1} &= -0.118 \\ l_{r1} &= 0.502 \end{aligned}$$

The evolution of the shaft angle when some steps are introduced in the reference can be seen in Figure 5.13. It can be observed that there is no overshooting due to the high value of λ chosen. The system has a dead zone such

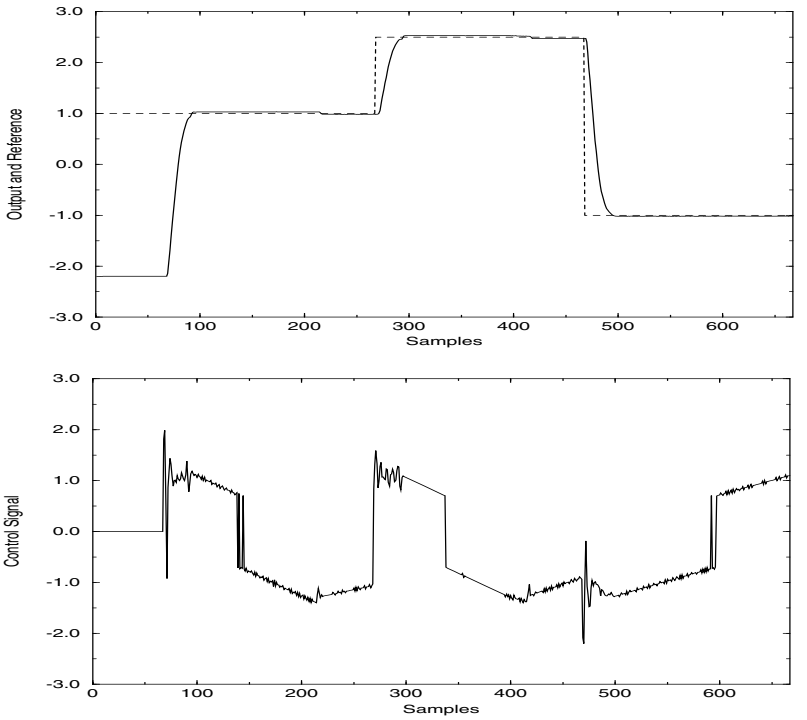


Fig. 5.13. Motor response for setpoint changes

that it is not sensitive to control signals less than 0.7 V; in order to avoid this a nonlinearity is added.

It is important to remember that the sampling time is very small (0.06 second) which could make the implementation of the standard GPC algorithm impossible. However, due to the simple formulation used here, the implementation is reduced to the calculation of Expression (5.21) and hardly takes any time in a computer.

The process is disturbed by the addition of an electromagnetic break that changes the load and the friction constant. The model parameters used for designing the GPC do not coincide with the process parameters, but in spite of this, as can be seen in Figure 5.14, GPC is able to control the motor reasonably well even though a slight overshoot appears.

5.5 Consideration of Ramp Setpoints

It is usual for a process reference signal to keep a certain constant value for a time and to move to other constant values by step changes during normal plant operation. This is what has been considered up to now, that is,

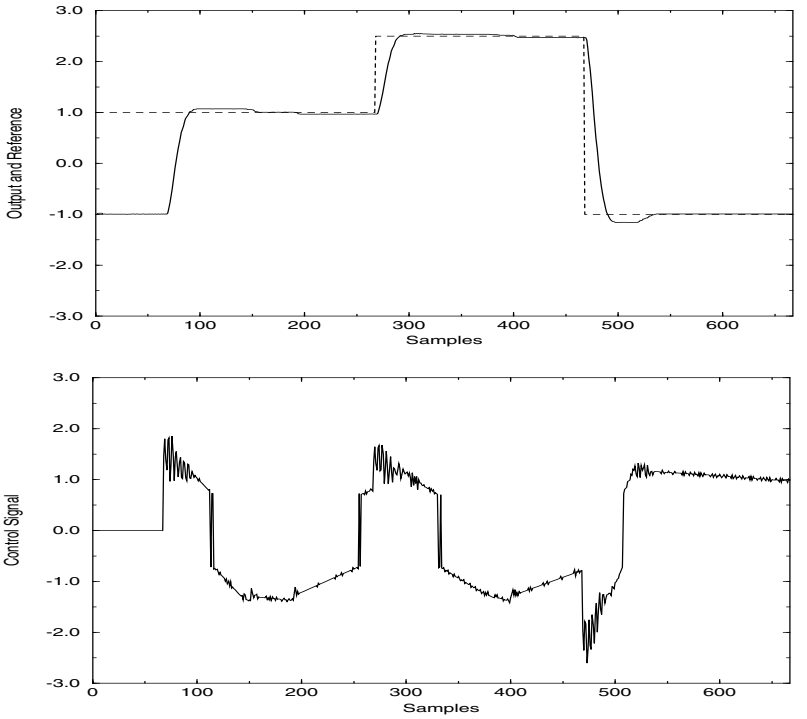


Fig. 5.14. Motor response with electromagnetic break

$w(t+d+1) = w(t+d+2) \ldots = r(t)$, where $r(t)$ is the setpoint at instant t which is going to maintain a fixed value.

But the reference evolution will not behave like this in all circumstances. On many occasions it can evolve as a ramp, which changes smoothly to another constant setpoint. In general it would be desirable for the process output to follow a mixed trajectory composed of steps and ramps.

This situation frequently appears in different industrial processes. In the food and pharmaceutical industries some thermal processes require the temperature to follow a profile given by ramps and steps. It is also of interest that in the control of motors and in robotics applications, the position or velocity follows evolutions of this type.

GPC will be reformulated when the reference is a ramp, defined by a parameter α indicating the increment at each sampling time. The reference trajectory is therefore:

$$\begin{aligned} w(t+d+1) &= r(t+d) + \alpha \\ w(t+d+2) &= r(t+d) + 2\alpha \\ \cdots &\quad \cdots \\ w(t+d+N) &= r(t+d) + N\alpha \end{aligned}$$

Employing the procedure used throughout this chapter, and for first-order systems with dead time, we get

$$\mathbf{M}\,\mathbf{u} = \mathbf{P}\,\mathbf{y} \;+\; \mathbf{R}\,\mathbf{w} + \;\mathbf{Q}\;\triangle u(t-1)$$

If $\mathbf{q}$ is the first row of matrix $\mathbf{M}^{-1}$ then $\triangle u(t)$ can be expressed as

$$\triangle u(t) = \mathbf{q}\,\mathbf{P}\,\mathbf{y} + \mathbf{q}\,\mathbf{R}\,\mathbf{w} + \mathbf{q}\,\mathbf{Q}\;\triangle u(t-1)$$

By making $\mathbf{h} = \mathbf{q}\,\mathbf{R}$ the term of the preceding expression including the reference ($\mathbf{h}\,\mathbf{w}$) takes the form:

$$\mathbf{h}\;\;\mathbf{w} = \sum_{i=1}^{N} h_i\;\; r(t+d+i) = \sum_{i=1}^{N} h_i\;\; (r(t+d)+\alpha\;\; i) = \sum_{i=1}^{N} h_i\;\; r(t+d)+\alpha \sum_{i=1}^{N} h_i\;\; i$$

Therefore

$$\mathbf{h}\;\;\mathbf{w} = l_{r1}\;\; r(t+d) + \alpha\;\; l_{r2}$$

The control law can now be written as

$$\triangle u(t) = l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{r1}r(t+d) + \alpha\, l_{r2} + l_{u1} \triangle u(t-1) \tag{5.23}$$

where $\mathbf{q}\,\mathbf{P} = [l_{y1}\;\; l_{y2}]$, $l_{u1} = \mathbf{q}\,\mathbf{Q}$, $l_{r1} = \sum_{i=1}^{N}(q_i \sum_{j=1}^{N} r_{ij})$ and $l_{r2} = \alpha \sum_{i=1}^{N} h_i\; i$.

The control law is therefore linear. The new coefficient l_{r2} is due to the ramp. It can be noticed that when the ramp becomes a constant reference, the control law coincides with the one developed for the constant reference case. The only modification that needs to be made because of the ramps is the term $l_{r2}\alpha$. The predictor is the same and the resolution algorithm does not differ from the one used for the constant reference case. The new parameter l_{r2} is a function of the process parameters (a, m) and of the control weighting factor (λ). As in the previous cases an approximating function can easily be obtained. Notice that the other parameters are exactly the same as in the constant reference case, meaning that the previously obtained expressions can be used.

In what has been seen up to now (nonintegrating processes, integrating processes, constant reference, ramp reference), a new coefficient appeared in the control law with each new situation. All these situations can be described by the following control law:

Table 5.3. Coefficients that may appear in the control law. The $\times$ indicates that the coefficient exists

Process	Reference	l_{y1}	l_{y2}	l_{y3}	l_{u1}	l_{u2}	l_{r1}	l_{r2}
$\frac{k}{1+\tau s}e^{-\tau_d s}$	Constant	$\times$	$\times$	0	0	0	$\times$	0
τ_d integer	Ramp	$\times$	$\times$	0	0	0	$\times$	$\times$
$\frac{k}{1+\tau s}e^{-\tau_d s}$	Constant	$\times$	$\times$	0	$\times$	0	$\times$	0
τ_d non integer	Ramp	$\times$	$\times$	0	$\times$	0	$\times$	$\times$
$\frac{k}{s(1+\tau s)}e^{-\tau_d s}$	Constant	$\times$	$\times$	$\times$	$\times$	0	$\times$	0
τ_d integer	Ramp	$\times$	$\times$	$\times$	$\times$	0	$\times$	$\times$
$\frac{k}{s(1+\tau s)}e^{-\tau_d s}$	Constant	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	0
τ_d non integer	Ramp	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$	$\times$

$$\begin{aligned}\triangle u(t) = {} & l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{y3}\hat{y}(t+d-2 \mid t) \\ & + l_{r1}r(t+d) + \alpha\, l_{r2} + l_{u1} \triangle u(t-1) + l_{u2} \triangle u(t-2)\end{aligned}$$

Table 5.3 shows which coefficients of this control law may be zero depending on the particular situation.

5.5.1 Example

As an application example, a GPC with ramp following capability is going to be designed for the motor described earlier. The reference trajectory is composed of a series of steps and ramps defined by the value of α ($\alpha = 0$ for the case of constant reference).

The same controller parameters as in the previous example are used, with the addition of the new parameter $l_{r2} = 2.674$. Considering that $(b_0 + b_1)/(1 - a) = 0.15$, the control law is given by:

$$\begin{aligned}\triangle u(t) = {} & -76.92\, y(t) + 128.29\, y(t-1) - 54.72\, y(t-2) \\ & + 3.35 r(t) + 17.82\, \alpha - 0.12 \triangle u(t-1)\end{aligned}$$

As the dead time is zero, the predicted outputs are known at instant t.

The results obtained are shown in Figure 5.15, where it can be seen that the motor is able to follow the ramp reference quite well.

5.6 Comparison with Standard GPC

The approximations made in the method can affect the quality of the controlled performance. Some simulation results are presented that compare the results obtained with the proposed method with those when the standard GPC algorithm as originally proposed by Clarke *et al.* [58] is used.

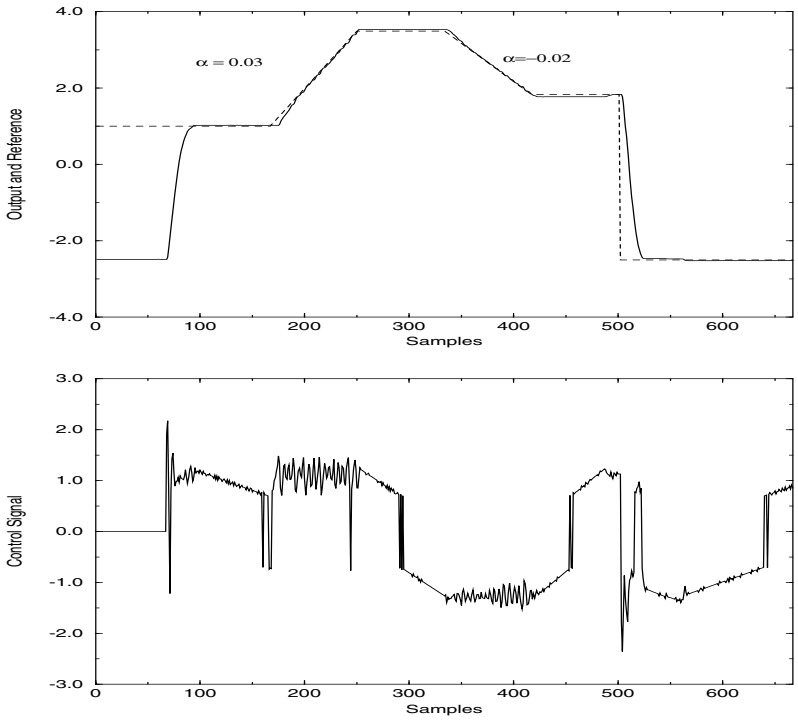


Fig. 5.15. Combined steps and ramps setpoint

Two indices are used to measure the performance: ISE (sum of the square errors during the transient) and ITAE (sum of the absolute error multiplied by discrete time). Also the number of floating-point operations and the computing time needed to calculate the control law are analyzed.

First, the performance of the proposed algorithm is compared with that of the standard GPC with no modelling errors. In this situation the error is only caused by the approximative functions of the controller parameters. For the system $G(s) = \frac{1.5}{1+10s}e^{-4s}$ with a sampling time of one second, the values for the proposed algorithm when the process is perturbed by a white noise uniformly distributed in the interval ± 0.015 are ISE= 7.132, ITAE= 101.106 and for the standard controller ISE= 7.122, ITAE=100.536. The plot comparing the two responses is not shown because there is practically no difference.

The plant model is supposed to be first-order plus deadtime. If the process behaviour can be reasonably described by this model, there will not be a substantial loss of performance. Consider, for instance, the process modelled by:

$$G_p(z) = \frac{0.1125z^{-1} - 0.225z^{-2}}{1 - z^{-1} + 0.09z^{-2}}z^{-3}$$

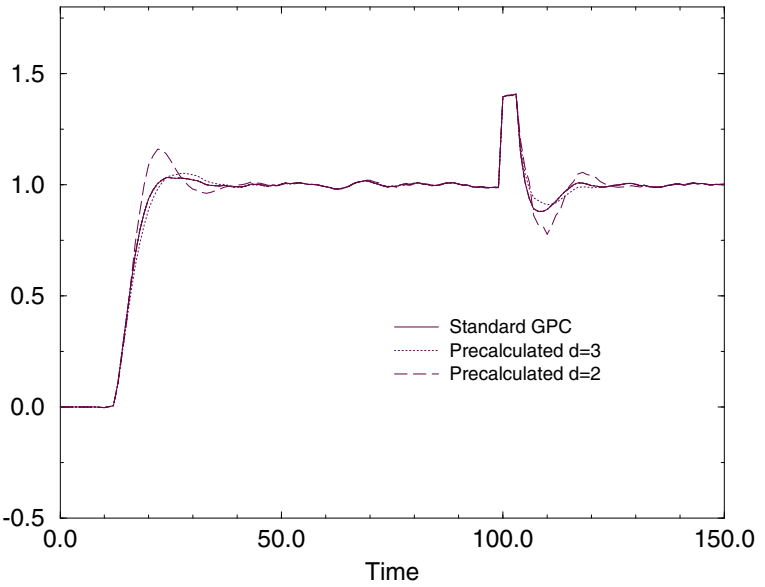


Fig. 5.16. System performance. Standard GPC: ISE = 6.809, ITAE = 395.608; proposed algorithm $d = 3$ ISE = 6.987, ITAE = 402.544; proposed algorithm $d = 2$ ISE = 7.064, ITAE = 503.687

For control purposes, it is approximated by the following first-order model, obtained from data generated by the process G_p:

$$G_m(z) = \frac{0.1125z^{-1}}{1 - 0.8875z^{-1}} z^{-3}$$

That is, the precomputed GPC is working in the presence of unmodelled dynamics. From the previous studies of robust stability, it can be deduced that the closed-loop system is going to be stable. The performance in this situation is shown in Figure 5.16 where the system response for both controllers is shown; notice that for $t = 100$ a disturbance is added to the output. The figure also shows the behaviour of the precomputed GPC when an additional deadtime mismatch is included, that is, the controller uses a model with $d = 2$ instead of the true value $d = 3$.

Logically, there is a slight loss of performance due to the uncertainties, that must be considered in conjunction with the benefits in the calculation. Besides, consider that in a real case the uncertainties (such as deadtime mismatch) can also affect the standard GPC since high-frequency effects are usually very difficult to model.

The computational requirements of the method are compared with the standard in Table 5.4 for this example, working with a control horizon of $N = 15$ and $\lambda = 0.8$. The table shows the computation needed for the calcu-

lation of the control law in both floating-point operations and CPU time on a personal computer.

Table 5.4. Computational requirements for the standard and precalculated GPC

Algorithm	Calculation	Operations (flops)
Standard	Build matrices	1057
	Compute $G^T G + \lambda I$	10950
	Inversion	7949
	Rest	1992
	TOTAL	**21948**
Proposed	**TOTAL**	**79**

As can be seen, these examples show that although a little performance is lost, there is a great improvement in real-time implementability, reaching a computing effort around 275 times smaller. This advantage can represent a crucial factor for the implementation of this strategy in small controllers with low computational facilities, considering that the impact on the performance is negligible. The simulations also show the robustness of the controller in the presence of structured uncertainties.

5.7 Stability Robustness Analysis

The elaboration of mathematical models of processes in real life requires simplifications to be adopted. In practice no mathematical model capable of exactly describing a physical process exists. It is always necessary to bear in mind that modelling errors may adversely affect the behaviour of the control system. The aim is that the controller should be insensitive to these uncertainties in the model, that is, that it should be robust. The aim here is to deal with the robustness of the controller presented. In any case, developments of predictive controller design using robust criteria can be found, for instance, in [44] and [208].

The modelling errors, or uncertainties, can be represented in different forms, reflecting in certain ways the knowledge of the physical mechanisms which cause the discrepancy between the model and the process as well as the capacity to formalize these mechanisms so that they can be handled. Uncertainties can, in many cases, be expressed in a structured way, as expressions in function of determined parameters which can be considered in the transfer function [69]. However, there are usually residual errors particularly dominant at high frequencies which cannot be modelled in this way, which

constitute unstructured uncertainties [65]. In this section a study of the precalculated GPC stability in the presence of both types of uncertainties is made; that is, the stability robustness of the method will be studied.

This section aims to study the influence of uncertainties on the behaviour of the process working with a controller which has been developed for the nominal model. That is, both the predictor and the controller parameters are calculated for a model which does not exactly coincide with the real process to be controlled. The following question is asked: what discrepancies are permissible between the process and the model for the controlled system to be stable?

The controller parameters l_{y1}, l_{y2}, l_{r1} and l_{u1} that appear in the control law

$$\triangle u(t) = l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{r1}r(t) + l_{u1} \triangle u(t-1)$$

have been precalculated for the model (not for the process as this is logically unknown). Likewise the predictor works with the parameters of the model, although it keeps up to date with the values taken from the output produced by the real process.

5.7.1 Structured Uncertainties

A first-order model with pure delay, in spite of its simplicity, describes the dynamics of most plants in the process industry. However, it is fundamental to consider the case where the model is unable to completely describe all the dynamics of the real process. Two types of structured uncertainties are considered: parametric uncertainties and unmodelled dynamic uncertainties. In the first case, the order of the control model is supposed to be identical to the order of the plant but the parameters are considered to be within an uncertainty region around the nominal parameters (these parameters will be the pole, the gain, and the coefficient $m = b_0/(b_0 + b_1)$ that measures the fractional delay between d and $d+1$). The other type of uncertainty will take into account the existence of process dynamics not included in the control model as an additional unmodelled pole and delay estimation error. This will be reflected in differences between the plant and model orders.

The uncertainty limits have been obtained numerically for the range of variation of the process parameters ($0.5 < a < 0.98$, $0 \leq m \leq 1$) with a delay $0 \leq d \leq 10$ obtaining the following results (for more details, see [44]):

- **uncertainty at the pole:** for a wide working zone ($a < 0.75$) and for normal values of the delay an uncertainty of more than $\pm 20\%$ is allowed. For higher poles the upper limit decreases due almost exclusively to the fact that the open loop would now be unstable. The stable area only becomes narrower for very slow systems with large delays. Notice that this uncertainty refers to the time constant (τ) uncertainty of the continuous process ($a = \exp(-T/\tau)$) and thus the time constant can vary around 500% of the nominal one in many cases.

- **gain uncertainty:** When the gain of the model is G_e and that of the process is $\gamma \times G_e$, γ will be allowed to move between 0.5 and 1.5, that is, uncertainties in the value of the gain of about 50% are permitted. For small delays $(1, 2)$ the upper limit is always above the value $\gamma = 2$ and only comes close to the value 1.5 for delays of about 10. It can thus be concluded that the controller is very robust when faced with this type of error.
- **uncertainty in m:** The effect of this parameter can be ignored since a variation of 300% is allowed without reaching instability.
- **unmodelled pole:** The real process has another less dominant pole $(k \times a)$ apart from the one appearing in the model (a), and the results show that the system is stable even for values of k close to 1; stability is only lost for systems with very large delays.
- **delay estimation error:** From the results obtained in a numerical study, it is deduced that for small delays stability is guaranteed for errors of up to two units through the range of the pole, but when bigger poles are dealt with this only happens for small values of a, and even for delay 10 only a delay mismatch of one unit is permitted. It can be concluded, therefore, that a good delay estimation is fundamental to GPC, because for errors of more than one unit the system can become unstable if the process delay is high.

5.7.2 Unstructured Uncertainties

In order to consider unstructured uncertainties, it will be assumed from now on that the dynamic behaviour of a determined process is described not by an invariant time linear model but by a family of linear models. Thus the real possible processes (G) will be in a vicinity of the nominal process $(\widetilde{G})$, which will be modelled by a first-order-plus-delay system.

A family $\mathcal{F}$ of processes in the frequency domain will therefore be defined which in the Nyquist plane will be represented by a region about the nominal plant for each ω frequency. If this family is defined as

$$\mathcal{F} = \{G :\mid G(i\omega) - \widetilde{G}(i\omega) \mid \leq \overline{l}_a(\omega)\}$$

the region consists of a disc with its centre at $G(i\omega)$ and radius $\overline{l}_a(\omega)$. Therefore any member of the family fulfils the condition

$$G(i\omega) = \widetilde{G}(i\omega) + l_a(i\omega) \qquad \mid l_a(i\omega) \mid \leq \overline{l}_a(\omega)$$

This region will change with ω because l_a does and, therefore, in order to describe family $\mathcal{F}$ we will have a zone formed by the discs at different frequencies. If one wishes to work with multiplicative uncertainties the family of processes can be described by

$$\mathcal{F} = \{G : \left| \frac{G(i\omega) - \widetilde{G}(i\omega)}{\widetilde{G}(i\omega)} \right| \leq \overline{l}_m(\omega)\} \tag{5.24}$$

simply considering

$$l_m(i\omega) = l_a(i\omega)/\widetilde{G}(i\omega) \qquad \overline{l}_m(i\omega) = \overline{l}_a(\omega)/ \mid \widetilde{G}(i\omega) \mid$$

Therefore any member of family $\mathcal{F}$ satisfies

$$G(i\omega) = \widetilde{G}(i\omega)(1 + l_m(i\omega)) \qquad \mid l_m(i\omega) \mid \leq \overline{l}_m(i\omega)$$

This representation of uncertainties in the Nyquist plane as a disc around the nominal process can encircle any set of structured uncertainties, although some times it can result in a rather conservative attitude [141].

The measurement of the robustness of the method can be tackled using the robust stability theorem [141], that for discrete systems states:

Suppose that all processes G of family $\mathcal{F}$ have the same number of unstable poles, which do not become unobservable for the sampling, and that a controller $C(z)$ stabilizes the nominal process $\widetilde{G}(s)$. Then the system has robust stability with controller C if and only if the complementary sensitivity function for the nominal process satisfies the following relation:

$$\mid \widetilde{T}(e^{i\omega T}) \mid \overline{l}_m(\omega) < 1 \qquad 0 \leq \omega \leq \pi/T \tag{5.25}$$

Using this condition the robustness limits will be obtained for systems that can be described by (5.13), and for all the values of the parameters that describe the system (a, m and d). For each value of the frequency ω the limits can be calculated as:

$$\overline{l}_m = \left| \frac{1 + \widetilde{G}C}{\widetilde{G}C} \right| \qquad \overline{l}_a = \overline{l}_m \mid \widetilde{G} \mid$$

In Figure 5.17 the form taken by the limits in function of ωT can be seen for some values of a, and fixed values of m and d. Both limits are practically constant and equal to unity at low frequencies and change (the additive limit $\overline{l}_a$ decreases and the multiplicative $\overline{l}_m$ increases) at a certain point. Notice that these curves show the great degree of robustness that the GPC possesses since $\overline{l}_m$ is relatively big at high frequencies, where multiplicative uncertainties are normally smaller than unity, and increases with frequency as uncertainties do. The small value of $\overline{l}_a$ at high frequencies is due to the fact that the process itself has a small gain at those frequencies; remember that both limits are dependent and related by $\overline{l}_a = \overline{l}_m \mid \widetilde{G} \mid$.

Figure 5.18 shows the frequency response of the nominal process alone and with the controller, as well as the discs of radius $\overline{l}_a$ and $\overline{l}_m \mid \widetilde{G}C \mid$ for a certain frequency. All the G processes belonging to the $\mathcal{F}$ family maintaining

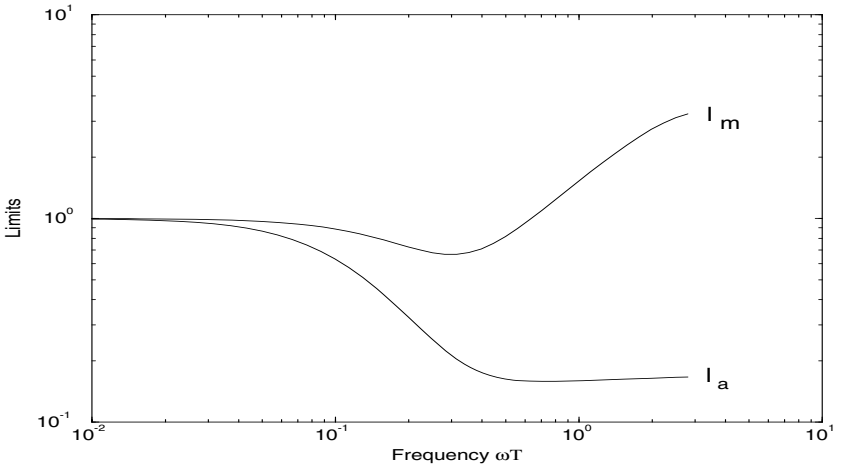


Fig. 5.17. General shape of $\bar{l}_a$ and $\bar{l}_m$

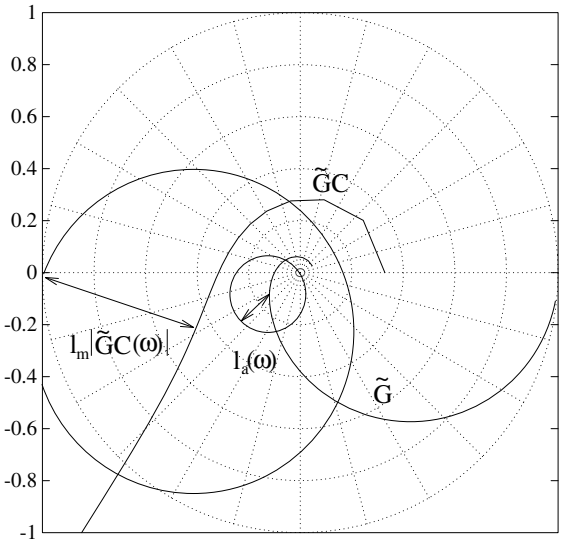


Fig. 5.18. Polar diagram of the process $\widetilde{G}$ and $\widetilde{G}C$ showing the limits for a given frequency

the stability of the closed loop can be found inside the disc of radius $\bar{l}_a$. The shape of the frequency response leads to limits $\bar{l}_a$ and $\bar{l}_m$. Thus, $\widetilde{G}C(\omega)$ has a big modulus (due to the integral term) at low frequencies, leading to a value

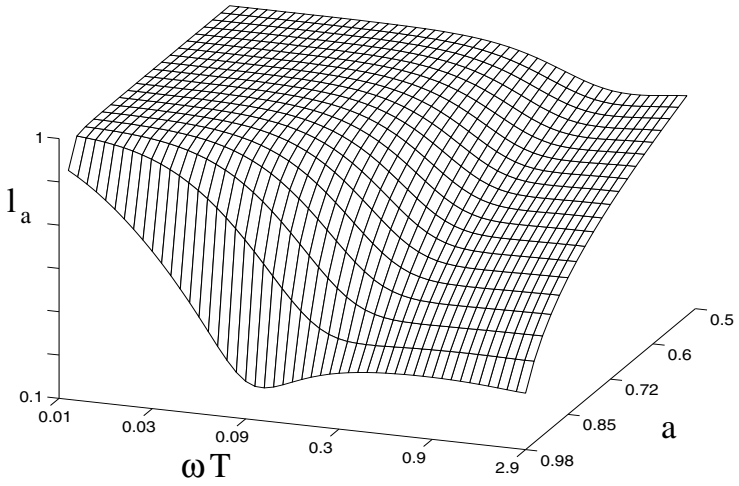


Fig. 5.19. Limit $\bar{l}_a$ for dead time 1

of $\bar{l}_m$ close to unity. When ω increases, $\widetilde{GC}(\omega)$ separates from -1 (without decreasing in modulus) and therefore the limit can safely grow.

It can be seen that the most influential parameters are pole a and delay d. The evolution of limit $\bar{l}_a$ with frequency (ωT) for parameter a changing between 0.5 and 0.98 is presented in Figure 5.19 for a concrete value of delay, $d = 1$, and for an average value of m, $m = 0.5$. As was to be expected, the limit decreases for greater poles because with open-loop poles near the limit of the unit circle the uncertainty allowed is smaller, as it would be easier to enter the open-loop unstable zone.

5.7.3 General Comments

The results obtained for both types of uncertainties are qualitatively the same. It can be concluded that the factor that mainly affects robustness is delay uncertainty, because of its effect at high frequencies. The robustness zone decreases when the open-loop pole increases whilst the parameter m hardly has any influence. As the analysis has been performed based on a particular choice of parameters in the GPC formulation the conclusions depend on these values. The influence of the choice of these parameters on the closed-loop stability is studied in [153].

In any case, the GPC algorithm presented has shown itself to be very robust against the types of uncertainties considered. For small delays the closed loop is stable for static gain mismatch of more than 100% and time constant mismatch of more than 200%.

The stability robustness of GPC can be improved with the use of an observer polynomial, the so-called $T(z^{-1})$ polynomial. In [57] a reformulation of the standard GPC algorithm including this polynomial can be found. In order to do this, the CARIMA model is expressed in the form:

$$A(z^{-1})y(t) = B(z^{-1})u(t-1) + \frac{T(z^{-1})}{\triangle}\xi(t)$$

Up to now the $T(z^{-1})$ has been considered equal to 1, describing the most common disturbances or as the colouring polynomial $C(z^{-1})$. But it can also be considered as a design parameter. In consequence the predictions will not be optimal but on the other hand robustness in the face of uncertainties can be achieved, in a similar interpretation as that used by Ljung [128]. Then this polynomial can be considered as a prefilter as well as an observer. The effective use of observers is known to play an essential role in the robust realization of predictive controllers (see [57] for the effect of prefiltering on robustness and [209] for guidelines for the selection of T).

This polynomial can easily be added to the proposed formulation, computing the prediction with the values of inputs and outputs filtered by $T(z^{-1})$. Then, the predictor works with $y^f(t) = y(t)/T(z^{-1})$ and $u^f(t) = u(t)/T(z^{-1})$. The actual prediction for the control law is computed as $\hat{y}(t+d) = T(z^{-1})\hat{y}^f(t+d)$.

5.8 Composition Control in an Evaporator

This chapter ends with an application of the method to a typical process. An evaporator, a very common process in industry, has been chosen as a testing bed for the GPC. This process involves a fair number of interrelated variables and although it may appear to be rather simple compared to other processes of greater dimension, it allows the performance of any control technique to be checked. The results presented in this section have been obtained by simulation on a nonlinear model of the process.

5.8.1 Description of the Process

The process in question is a forced circulation evaporator in which the raw material is mixed with an extraction of the product and pumped through a vertical heat exchanger through which water steam is circulating, which condenses in the tubes. The mix evaporates and passes through a separating vessel where the liquid and the vapour are separated. The former is made to recycle and a part is extracted as the final product whilst the vapour is condensed with cooling water. Figure 5.20 shows the diagram of this process, used in many production sectors, such as the sugar industry.

The process behaviour can be modelled by a series of equations obtained from the mass and energy balance equations, as well as by making some

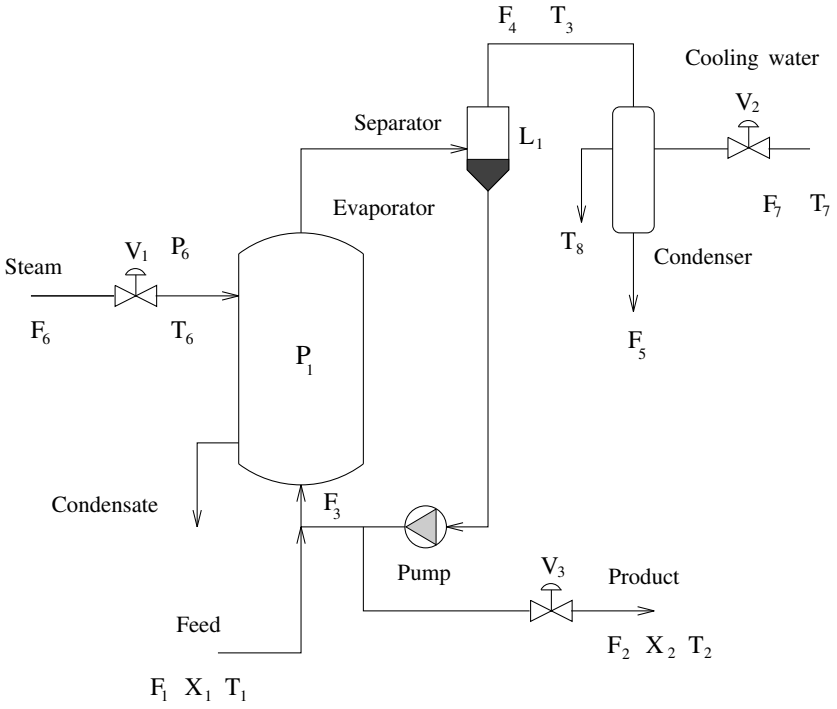


Fig. 5.20. Diagram of the evaporator

realistic assumptions. The equations describing the process behaviour can be found in [149]. The main variables, together with their values at the point of operation, are grouped in table 5.5.

Table 5.5. Process variables and values at operating point

Variable	Description	Value	Units
F_1	Feed flow rate	10.0	kg/min
F_2	Product flow rate	2.0	kg/min
F_4	Vapour flow rate	8.0	kg/min
F_5	Condensate flow rate	8.0	kg/min
X_1	Feed composition	5.0	%
X_2	Product composition	25.0	%
L_1	Separator level	1.0	m
P_1	Operating pressure	50.5	kPa
P_6	Steam pressure	194.7	kPa
F_6	Steam flow rate	9.3	kg/min
F_7	Cooling water flow rate	208.0	kg/min

The system dynamics are mainly dictated by the differential equations modelling the mass balances:

- mass balance in the liquid:

$$\rho A \frac{dL_1}{dt} = F_1 - F_4 - F_2 \tag{5.26}$$

  where ρ is the density of the liquid and A the section of the separator, whose product can be considered constant.
- mass balance in the solute:

$$M \frac{dX_2}{dt} = F_1 X_1 - F_2 X_2 \tag{5.27}$$

  where M is the total quantity of liquid in the evaporator.
- mass balance in the process vapour, the total amount of water vapour can be expressed in a function of the pressure existing in the system according to

$$C \frac{dP_1}{dt} = F_4 - F_5 \tag{5.28}$$

  where C is a constant that converts the steam mass into an equivalent pressure.

The dynamics of the interchanger and the condenser can be considered very fast compared to previous ones.

Degrees of Freedom

Twelve equations can be found for twenty variables so there are eight degrees of freedom. Eight more equations must therefore be considered to close the problem; these will be the ones which provide the values of the manipulated variables and the disturbances:

- three manipulated variables: the steam pressure P_6, which depends on the opening of valve V_1, the cooling water flow rate F_7, controlled by valve V_2 and the product flow rate F_2 with V_3.
- five disturbances: feed flow rate F_1, circulating flow rate F_3, composition and temperature of feed X_1 and T_1, and cooling water temperature T_7.

A single solution can be obtained with these considerations which allows the value of the remaining variables to be calculated.

5.8.2 Obtaining the Linear Model

As can be deduced from Equations (5.26)-(5.28) the process is a nonlinear system with a strong interrelationship amongst the variables. Even so, a linear model with various independent loops will be used to design the controller.

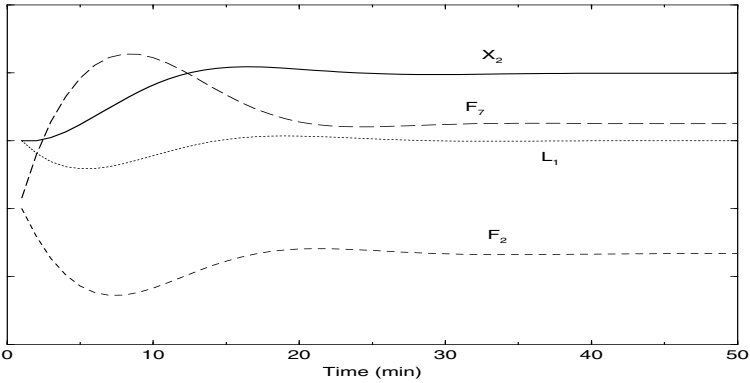


Fig. 5.21. Evaporator response to step input

It is clear that these hypotheses of work are incorrect, as will be made obvious when putting the control to work. The control of the evaporator includes maintaining certain stable working conditions as well as obtaining a product of a determined quality. To achieve the first objective it is necessary to control the mass and energy of the system, which can be achieved by keeping the level in the separator L_1 and the process pressure P_1 constant . In order to do this, two PI-type local controllers will be used, so that the level L_1 is controlled by acting on the product flow rate F_2 and the process pressure P_1 is controlled by the cooling water F_7. The justification of the choice of the couplings and the tuning of these loops can be found in [149].

The other objective is to obtain a determined product composition. This is achieved by acting on the remaining manipulated variable, the steam pressure which supplies the energy to the evaporator, P_6. The interaction amongst the variables is very strong, as can be seen by using Bristol method [36] and it would even be possible to have coupled X_2 with F_7 and P_1 with P_6. To obtain the linear model of the composition loop, a step is applied at the input (P_6) and the effect on the output (X_2) is studied. As was expected and as can be seen from Figure 5.21 which shows the evolution of the more significant variables for a 10% step, the interaction amongst the variables is considerable.

The evolution of the composition does not therefore follow the pattern of a first-order system, due mainly to the fact that the experiment was not done in open loop because the level and pressure regulators are functioning which, as has been indicated, need to be activated for stable functioning of the evaporator and indirectly affect the composition. The approximation of loop $X_2 - P_6$ to that of a first-order model with delay of the form

$$G(s) = \frac{K}{1+\tau s} e^{-\tau_d s}$$

that, as is known, in spite of its simplicity is much used in practice, will be attempted. The reaction curve method will be used to obtain the model parameters; this provides the values of K, τ and τ_d starting from the graph of the system response at a step input. Due to the nonlinearity the system behaviour will be different for inputs of different value and different sign. By conducting various experiments for steps of different signs and magnitudes the following parameters can be considered to be appropriate:

$$K = 0.234\ \%/\text{KPa} \qquad \tau = 4.5\ \text{min} \qquad \tau_d = 3.5\ \text{min}$$

By taking a sampling time of one minute the delay is not integer so that the discrete transfer function about the working point will be (see conversion expressions in Chapter 3):

$$G(z^{-1}) = \frac{0.02461z^{-1} + 0.02202z^{-2}}{1 - 0.8007374z^{-1}} z^{-3}$$

This transfer function will be used for the design of the controller in spite of its limitations because of the existence of the previously mentioned phenomena.

5.8.3 Controller Design

Once a linear model of the process is obtained, the design of the controller is direct if the precalculated GPC is used. It is only necessary to calculate the parameters which appear in the control law:

$$\triangle u(t) = (l_{y1}\hat{y}(t+d \mid t) + l_{y2}\hat{y}(t+d-1 \mid t) + l_{r1}r(t))/K + l_{u1} \triangle u(t-1) \quad (5.29)$$

If one wants to design a fixed (nonadaptive) regulator, it is only necessary to calculate these parameters once. In the case of the evaporator with $a = 0.8007374$, $m = b_0/(b_0 + b_1) = 0.528$ and for a value of λ of 1.2 one has:

$$\begin{aligned} l_{y1} &= -2.2748 \\ l_{y2} &= 1.5868 \\ l_{r1} &= 0.6879 \\ l_{u1} &= -0.1862 \end{aligned}$$

The control signal at each instant is therefore

$$\begin{aligned} u(t) = 0.814u(t-1) + 0.186u(t-2) - 9.721\hat{y}(t+d \mid t) \\ +6.781\hat{y}(t+d-1 \mid t) + 2.939r(t) \end{aligned}$$

In order to complete the computations of the control law (5.29) the predicted values of the output at instants $t + d$ and $t + d - 1$ are necessary. This

computation is easy to do given the simplicity of the model. It is enough to project the equation of the model towards the future

$$\begin{aligned}\hat{y}(t+i) = (1+a)\hat{y}(t+i-1) - a\hat{y}(t+i-2) \\ +b_0(u(t-d+i-1) - u(t-d+i-2)) \\ +b_1(u(t-d+i-2) - u(t-d+i-3)) \qquad i = 1 \ldots d\end{aligned}$$

where the elements $\hat{y}(t) = y(t)$ and $\hat{y}(t-1) = y(t-1)$ are known values at instant t. Note that if one wanted to make the controller adaptive it would be enough to just calculate the new value of l_i when the parameters of the system change. The simplicity of the control law obtained is obvious, being comparable to that of a digital PID, and it is therefore easy to implant in any control system.

5.8.4 Results

In the following, some results of applying the previous control law to the evaporator are presented (simulated to a nonlinear model). Even though the simplifications which were employed in the design phase (monovariable system, first-order linear model) were not very realistic, a reasonably good behaviour of the closed-loop system is obtained.

In Figure 5.22 the behaviour of the process in the presence of changes in the reference of the composition is shown. It can be observed that the output clearly follows the reference although with certain initial overshoot. It should be taken into account that the loops considered to be independent are greatly interrelated amongst themselves and in particular that the composition is very disturbed by the variations in the cooling water flow rate F_7, which is constantly changing to keep the process pressure constant.

In spite of the overshoot, the behaviour can be considered good. It can be compared to that obtained with a classical controller such as PI. Some good values for adjusting this controller are those calculated in [149]:

$$K = 1.64 \text{ kPa}/\% \qquad T_I = 3.125 \text{ min}$$

In Figure 5.23 both regulators are compared for a change in the reference from 28 to 25 %. The GPC is seen to be faster and overshoots less than the PI and does not introduce great complexity in the design, as was seen in the previous section. The responses of both controllers in the presence of changes in the feed flow rate are reflected in Figure 5.24, where at instant $t = 50$ the feed flow rate changes from 10 to 11 kg/min and at $t = 200$ the composition at the input brusquely changes from 5 to 6 %. It can be seen that these changes considerably affect the composition of the product and although both controllers return the output to the reference value, the GPC does it sooner and with less overshoot, reducing the peaks by about 30%.

Tests can also be made with regard to the study of robustness. It is already known that the model used does not correspond to the real one (which

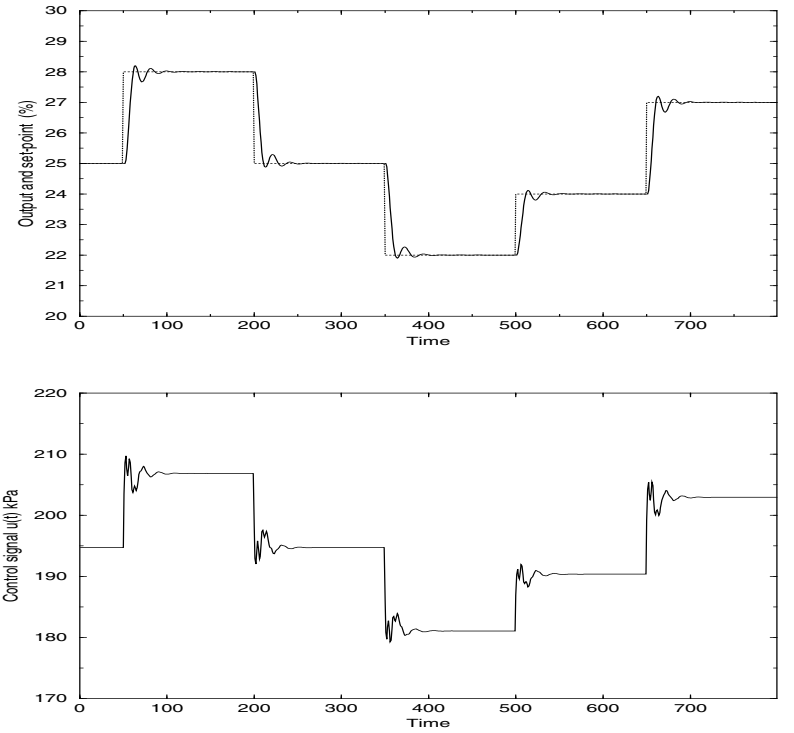


Fig. 5.22. GPC behaviour in the evaporator

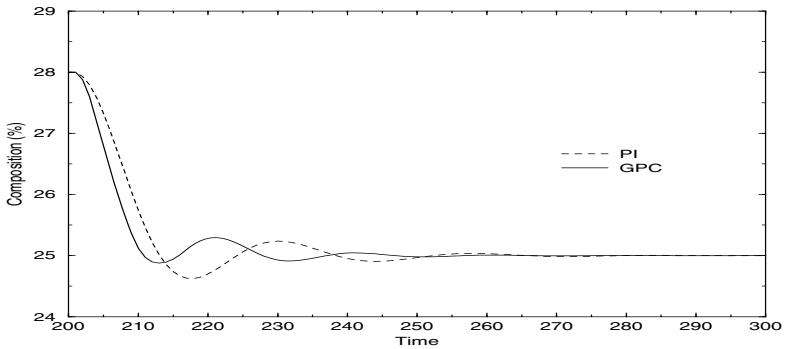


Fig. 5.23. Comparison of GPC and PI for a setpoint change

is neither first order nor linear) and therefore the controller already possesses certain robustness. However, to corroborate the robustness results previously presented, instead of using the linear model that best fits the nonlinear process as a control model , a model with estimation errors is going to be used. For example, when working in a model with an error on the pole estimation of the form that $\hat{a} = \alpha \times a$ with $\alpha = 0.9$, that is

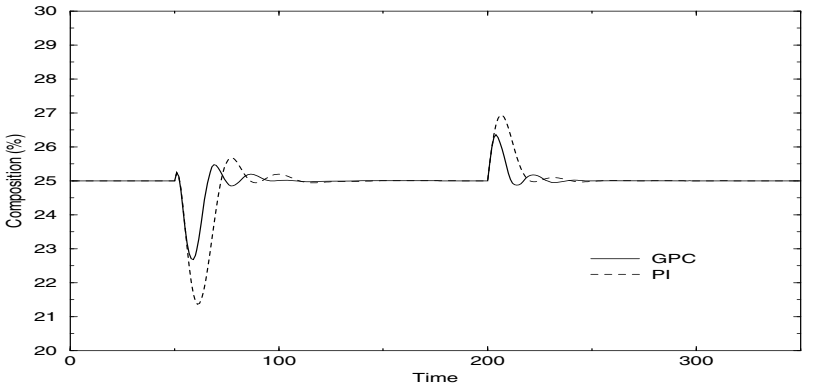


Fig. 5.24. Comparison of GPC and PI for feed changes

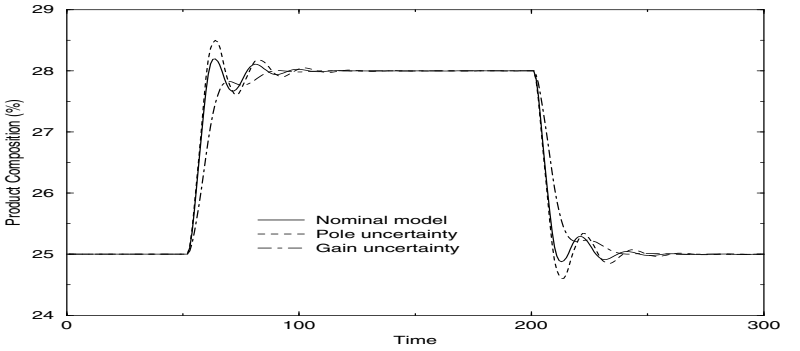


Fig. 5.25. Influence of errors on the estimation of gain and delay

an error of 10%, it is necessary to calculate the new l_i parameters using $\hat{a} = 0.9 \times a = 0.9 \times 0.8007374 = 0.7206$ (supposing that $a = 0.8007374$ is the *real* value). Thus, by recalculating the control law (including the predictor) for this new value and leaving the gain unaltered, the response shown in Figure 5.25 is obtained. As can be seen, the composition is hardly altered by this modelling error and similar response to the initial model is obtained. The same can be done by changing the system gain. For the model values considered to be *good*, uncertainties of up to 100% in the gain can be seen to be permissible without any problem. Thus by doubling the gain of the model used and calculating the new control law, slower but not less satisfactory behaviour is obtained, as is shown in the same figure.

Knowledge of delay is a fundamental factor of model based predictive methods, to such an extent that large errors in estimation can give rise to instability. Whilst for a difference of one unit the response hardly varies, the same is not true if the discrepancy is two or more units. Figure 5.26 shows the effect of using a model with delay 1 the *real* being equal to 3. The response is

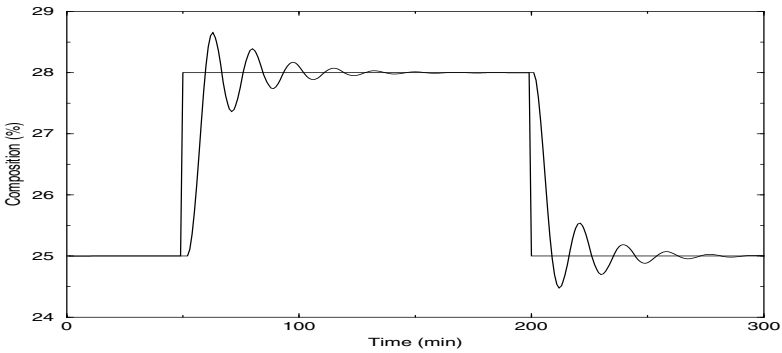


Fig. 5.26. Influence of error on the estimation of delay

defective but does not reach instability. It should be noted that once a mistake in the value of the delay is detected it is very easy to correct as it is enough to calculate higher or lower values of $\hat{y}(t+d)$ using the model equation whose other parameters are unchanged. Furthermore, it is not necessary to change the values of the l_i coefficients as they are independent of the delay.

5.9 Exercises

5.1. Given a system described by a static gain $K = 0.25$, a time constant $\tau = 10.5$ and a dead time of $\tau_d = 10$, compute l_{y1}, l_{y2} and l_{r1} with the given formula and simulate the process output to a step setpoint change:

1. Compare the results with those obtained by the *standard* algorithm.
2. Simulate the response to a setpoint composed of a ramp with slope 0.5 unit/second until time $t = 50$ and constant from this time to $t = 100$.
3. Add a white noise of mean 0.005 and compute the results.

5.2. Use the general procedure described in Chapter 3 to compute the values of l_{y1}, l_{y2} and l_{r1} for the system $G(z) = \frac{0.4z^{-1}}{1-0.8z^{-1}}$ with $N = 3$ and $\lambda = 0.8$.

5.3. Use the method described in this chapter to simulate the response of the process $G(s) = \frac{0.41e^{-50s}}{s(1+50s)}$ to a setpoint change from 0 to 2 when the sampling time is 10 seconds. Try different values for λ.

5.4. Control the following process $G(s) = \frac{1.12e^{-45s}}{1+87s}$ when the sampling time is 10 seconds and when it is sampled every 5 seconds. Compare the control law and results in the two cases.

5.5. Consider that the distillation column in Example 5.3.3 can work in a new operating regime at 5 mol/min. In this situation, $K = 0.102$, $\tau = 65.3$ and $\tau_d = 16.7$ (time in minutes).

1. Obtain the discrete model when $T = 5$ minutes.
2. Obtain the controller parameters for $\lambda = 0.8$.
3. Simulate the response to a setpoint that takes the value 0.1 at time $t = 20$, 0 at time $t = 120$ and -0.1 at time $t = 220$ (values taken around the nominal values of the variables).