Proceedings of 3rd
European Control Conference
Rome, Italy, September 1995

IMPLEMENTATION OF GPC FOR INTEGRATING PROCESSES WITH LOW COMPUTATIONAL REQUIREMENTS *

Carlos Bordons and Eduardo F. Camacho
Dpto. Ing. de Sistemas y Automática
Escuela Superior de Ingenieros
Univ. de Sevilla
Av. Reina Mercedes s/n
41012 Sevilla
Spain
bordons@esi.us.es

Keywords: Predictive control, adaptive control, process control, implementation issues, real-time control.

Abstract

This paper presents a straightforward method for implementing generalized predictive self-tuning controllers with low computational requirements. The method makes use of the fact that a generalized predictive controller results in a control law that can be described with few parameters.
The controller has been developed for processes having an integral effect. A set of simple functions relating the controller parameters to the process parameters has been obtained. With this set of functions either a fixed or a selftuning GPC can be implemented in a straightforward manner. An application to the control of a DC motor is given.

1 Introduction

One of the reasons for the success of the traditional PID controllers in industry is that they are very easy to implement and tune by using heuristic tuning rules such as the Ziegler-Nichols rules frequently used in practice. A Generalized Predictive Controller (GPC) 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 solving recursively a Diophantine equation, forming the prediction equation matrices 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 those 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 selftuning control is used. This again may be a difficulty 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.

Fast implementation methods for processes that can be modelled by the reaction curve method (most plants in the process industry) have been proposed in [6], [4], where some Ziegler-Nichols type of rules were given for implementing adaptive GPC requiring only a few multiplications. Some applications can be found in [2], [3] and [5].

This paper extends those results and presents a method for the easy implementation and tuning of GPC for a wide range of processes in industry, such as processes with integral effect. It will be shown that a GPC can be implemented with a limited set of instructions, available in most distributed 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 results almost as simple as the implementation and tuning of a PID.

The paper is organized as follows: first a short review of GPC is given in section 2. Section 3 presents the

*Work supported in part by CICYT Contract #TAP-95-0370

plant model used to model the integral effect and the control law obtained in section 4. The controller parameters are calculated and some approximation formulas are obtained in section 5 whilst the implementation algorithm is presented in section 6. The consideration of ramp setpoints is dealt with in section 7 and an illustrative application to a DC motor is presented in section 8. The paper ends with some concluding remarks.

2 Generalized Predictive Control

The GPC method was proposed by Clarke *et al.* [8] and has become one of the most popular Model-Based Predictive Control (MPC) methods both in industry and academia. It has been successfully implemented in many industrial applications [7], showing good performance and a certain degree of robustness with respect to overparametrization or poorly known delays. 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 a 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.

Generalized Predictive Control provides an explicit solution (in the absence of constraints), it can deal with unstable and non-minimum phase plants and incorporates the concept of control horizon as well as the consideration of weighting of 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 [8].

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

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

where E $\{.\}$ is the expectation operator and $\hat{y}(t+j \mid t)$ 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, Nu is the control horizon, $\delta(j)$ and $\lambda(j)$ are weighting sequences and $w(t+j)$ is the future reference trajectory.

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

The standard algorithm is given in [8]. This algorithm involves the optimal prediction of $y(t+j)$ for $N_1 \leq j \leq N_2$, which is obtained by the recursion of a Diophantine equation and the triangularization of an $(N_2 - d) \times (N_2 - d)$ matrix.

The goal of this paper is to develop a fast algorithm to implement self-tuning GPCs for processes that can be modelled by a first order integrating system plus a pure dead time.

3 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 motor shaft which grows while being powered until the torque equals the load.

These processes can be modelled by a first order plus delay transfer function, with the inclusion of an $1/s$ term in order to model the integrating effect. Hence, the transfer function for this kind of processes will be:

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

In the case of dead time being multiple of the sampling time the equivalent discrete transfer function when a zero-order hold is employed is given by:

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

In this section the GPC control law for processes described by (1) will be calculated. Notice that some formulations of MPC such as Dynamic Matrix Control (DMC [9]) or Model Algorithmic Control (MAC [11]) 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.

4 Derivation of the Control Law

The procedure for obtaining the control law is described below:

Using a CARIMA model with the noise polynomial equal to 1, the system can be written as

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

which can be transformed into:

$$y(t+1) = (2+a)y(t) - (1+2a)y(t-1) + ay(t-2) + b_0 \Delta u(t-d) + b_1 \Delta u(t-d-1) + \varepsilon(t+1)$$

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:

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

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 $\Delta u(t+Nu-1)$, $\Delta u(t+Nu-2)$ ... $\Delta u(t)$, and past inputs $\Delta u(t-1)$ and, of course, of the reference trajectory. If the horizons are: $N_1 = d+1$, $N_2 = N+d$ and $Nu = N$, the minimization of $J(N_1, N_2, Nu)$ leads to the following matrix equation for calculating u:

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

where $\mathbf{M}$ and $\mathbf{R}$ are matrices of dimension $N \times N$, $\mathbf{P}$ of dimension $N \times 3$ and $\mathbf{Q}$ of $N \times 1$. Vector $\mathbf{u}$ contains the future input increments and $\mathbf{y}$ the predicted outputs.

The first element of vector $\mathbf{u}$ can be obtained by:

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

being q 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:

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

the control law results as:

$$\Delta u(t) = l_{y1}\ \hat{y}(t+d \mid t) + l_{y2}\ \hat{y}(t+d-1 \mid t) \\ +l_{y3}\ \hat{y}(t+d-2 \mid t) + l_{r1}\ r(t) + l_{u1}\ \Delta u(t-1) \quad (4)$$

Being $q\ \mathbf{P} = [l_{y1}\ l_{y2}\ l_{y3}]$, $l_{r1} = \sum_{i=1}^{N}(q_i \sum_{j=1}^{N} r_{ij})$ and $l_{u1} = q\ \mathbf{Q}$.

Therefore the control law results in a linear expression depending on five 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 so as to get a closed loop with unitary static gain:

$$l_{y1} + l_{y2} + l_{y3} + l_{r1} = 0$$

5 Controller Parameters

The control law (4) is very easy to implement provided the controller parameters l_{y1}, l_{y2}, l_{y3}, l_{r1} and l_{u1} 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 [4] for processes without integrators, simple expressions for these relationships will be obtained.

As the process can be modelled by (2) three parameters (a, b_0 and b_1) are needed to describe the plant. Expressions relating the controller coefficients with these parameters can be obtained, although the resulting functions are not as simple, due to the number of plant parameters involved. In order to reduce the number of parameters involved, the process can be considered to have $(b_0+b_1)/(1-a) = 1$ so as to work with normalized plants. Then the computed parameters must be divided by this value that will not be equal to 1 in general.

The controller coefficients will be obtained as a function of the system 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 [1]):

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

then

$$n = \frac{a-1-\log a}{(a-1)\log a}$$

that for the usual values of the system pole (in the range from 0.5 to 0.99 when sampling at an appropriate rate [10]) 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. Notice that the use of a predictor makes the parameters independent of the dead time.

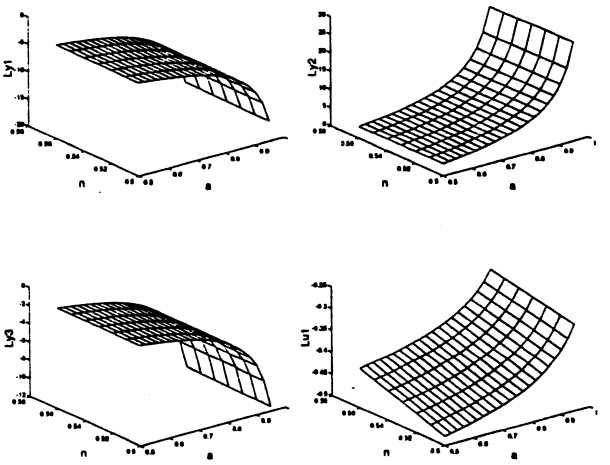

Figure 1: Controller coefficients l_{y1}, l_{y2}, l_{y3} and l_{u1}

The shape of the parameters is displayed in figure 1 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

$$f(a,n,\lambda) = k_1(n,\lambda) + k_2(n,\lambda)\frac{a}{k_3(n,\lambda) - a} \quad (5)$$

where (for $N = 15$) k_i can be approximated by:

$$\begin{aligned}
k_{y1,1} &= -\exp(0.955 - 0.559\lambda + 0.135\lambda^2) \\
k_{y1,2} &= -\exp(0.5703 - 0.513\lambda + 0.138\lambda^2) \\
k_{y1,3} &= 1.0343 \\
k_{y2,1} &= \exp(0.597 - 0.420\lambda + 0.0953\lambda^2) \\
k_{y2,2} &= \exp(1.016 - 0.4251\lambda + 0.109\lambda^2) \\
k_{y2,3} &= 1.0289 \\
k_{y3,1} &= -\exp(-1.761 - 0.422\lambda + 0.071\lambda^2) \\
k_{y3,2} &= -\exp(0.103 - 0.353\lambda + 0.089\lambda^2) \\
k_{y3,3} &= 1.0258 \\
k_{u1,1} &= 1.631n - 1.468 + 0.215\lambda - 0.056\lambda^2 \\
k_{u1,2} &= -0.124n + 0.158 - 0.026\lambda + 0.006\lambda^2 \\
k_{u1,3} &= 1.173 - 0.019\lambda
\end{aligned} \tag{6}$$

provide good approximations for l_{y1}, l_{y2}, l_{y3} 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 percent 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. For other values of N and N_u similar expressions can be obtained.

6 Implementation Algorithm

Once the λ factor has been decided, the values k_{ij} can very easily be computed by expressions (6) and the approximate adaptation laws given by equation (5) can easily be employed.

The algorithm in the adaptive case will consider the plant parameters (a, b_0, b_1, d and the factor $G = (b_0 + b_1)/(1 - a)$) and the control law can be seen below.

1. Perform an identification step.
2. Compute $k_{ij}(n, \lambda)$.
3. Calculate l_{y1}, l_{y2}, l_{y3} and l_{u1}
   Make $l_{r1} = -l_{y1} - l_{y2} - l_{y3}$
4. Compute $\hat{y}(t + d \mid t)$, $\hat{y}(t + d - 1 \mid t)$ and $\hat{y}(t + d - 2 \mid t)$ using (3) recursively.
5. Compute $u(t)$ with: $\Delta u(t) = (l_{y1}\ \hat{y}(t + d \mid t) + l_{y2}\ \hat{y}(t + d - 1 \mid t) + l_{y3}\ \hat{y}(t + d - 2 \mid t) + l_{r1}\ r(t))/G + l_{u1}\ \Delta u(t - 1)$
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 4 and 5 have to be carried out at every sampling time.

7 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, $w(t + d + 1) = w(t + d + 2) \ldots = r(t)$, $r(t)$ being 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 follow 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:

$$\begin{aligned}
w(t + d + 1) &= r(t + d) + \alpha \\
w(t + d + 2) &= r(t + d) + 2\alpha \\
&\ldots \quad\quad \ldots \\
w(t + d + N) &= r(t + d) + N\alpha
\end{aligned}$$

Employing the previously used procedure we get

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

If q is the first row of matrix $\mathbf{M}^{-1}$ then $\Delta u(t)$ can be expressed as

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

By making $\mathbf{h}^T = q\ \mathbf{R}$ the term of the above expression including the reference($\mathbf{h}^T\mathbf{w}$) takes the form:

$$\begin{aligned}
\mathbf{h}^T\mathbf{w} = \sum_{i=1}^{N} h_i r(t + d + i) &= \sum_{i=1}^{N} h_i (r(t + d) + \alpha i) \\
&= \sum_{i=1}^{N} h_i r(t + d) + \alpha \sum_{i=1}^{N} h_i i
\end{aligned}$$

therefore

$$\mathbf{h}^T\mathbf{w} = l_{r1}\ r(t + d) + \alpha\ l_{r2}$$

The control law can now be written as

$$\begin{aligned}
\Delta u(t) = l_{y1}\hat{y}(t + d \mid t) + l_{y2}\hat{y}(t + d - 1 \mid t) + \\
l_{y3}\hat{y}(t + d - 2 \mid t) + l_{r1} r(t + d) + \\
\alpha\ l_{r2} + l_{u1}\ \Delta u(t - 1)
\end{aligned} \tag{7}$$

Where $q\ \mathbf{P} = [l_{y1} \quad l_{y2} \quad l_{y3}]$, $l_{u1} = q\ \mathbf{Q}$, $l_{r1} = \sum_{i=1}^{N}(q_i \sum_{j=1}^{N} r_{ij})$ 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, n) 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.

8 Applications

8.1 Step Setpoint

The control law (4) 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 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, that 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 are constructive parameters 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, provoking a step in the feed voltage and measuring the evolution of the angular speed (which is a first order system). The parameters obtained are:

$$K = 2.5 \qquad \tau = 0.9 \text{ seconds}$$

and zero dead time. Taking a sampling time of $T = 0.06$ seconds 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 in order to avoid overshooting $(\lambda = 2)$ the control parameters in (4) can be calculated using expressions (6) with $a = 0.935507$ and $n = 0.50558$:

$$\begin{aligned} l_{y1} &= -11.537 \\ l_{y2} &= 19.242 \\ l_{y3} &= -8.207 \\ l_{u1} &= -0.118 \\ l_{r1} &= 0.502 \end{aligned}$$

The evolution of the shaft angle when some steps are introduced in the reference can be seen in figure 2. It can be observed that there is no overshooting due to the high value of λ chosen. The system has a dead zone such that it is not sensitive to control signals less than 0.7 V; in order to avoid this a non-linearity is added.

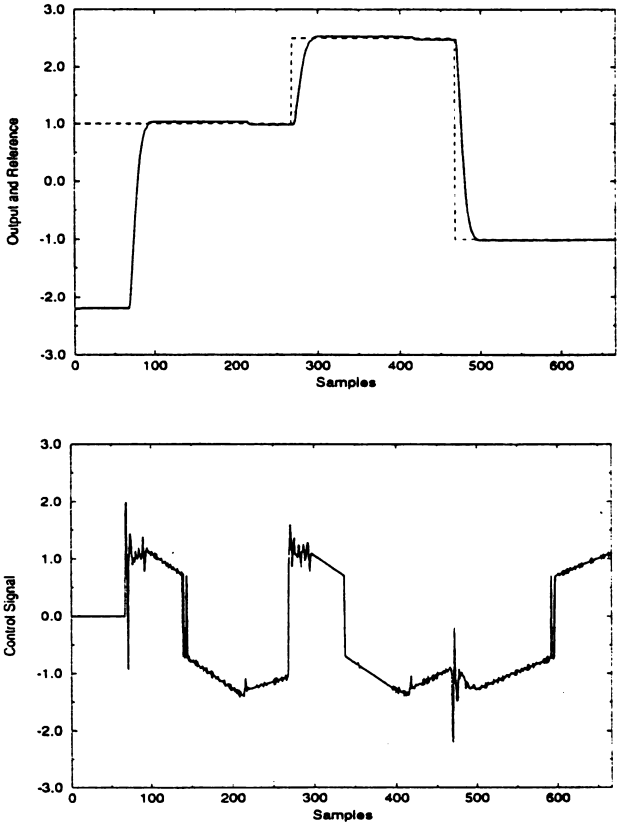

Figure 2: Motor evolution for setpoint changes

It is important to remember that the sampling time is very small (0.06 seconds) 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 (4) and hardly takes any time in any computer.

8.2 Ramp Setpoint

As an application example, a GPC with ramp following capability is going to be designed for the motor described above. 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:

$$\Delta u(t) = -76.921\ y(t) + 128.29\ y(t-1)$$
$$-54.72\ y(t-2) + 3.351 r(t) + 17.82\ \alpha - 0.118\ \Delta u(t-1)$$

As the dead time is zero, the predicted outputs are known at instant t.

The results obtained are shown in figure 3 where it can be seen that the motor is able to follow the ramp reference quite well.

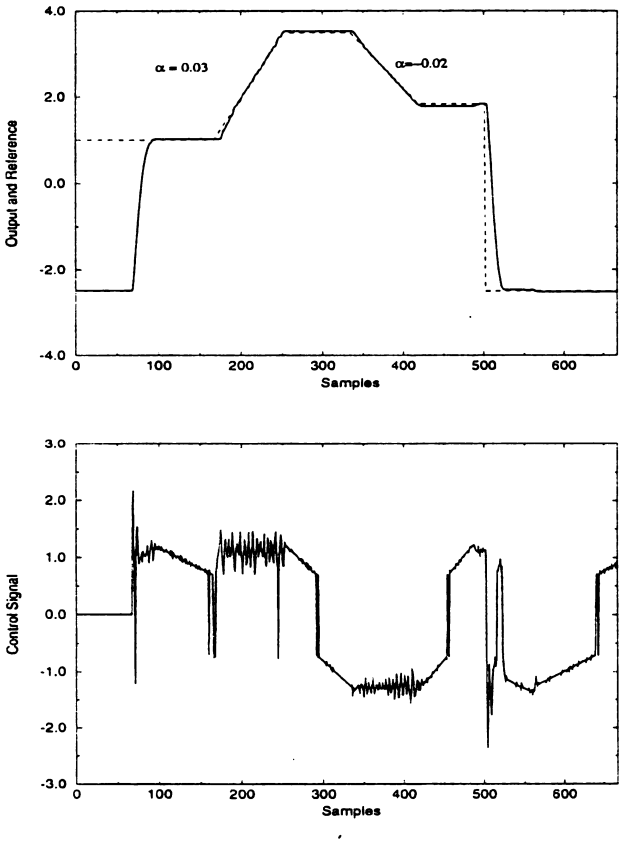

Figure 3: Combined steps and ramps setpoint

9 Concluding Remarks

A method for approximating GPC parameters for processes that can be modelled by a time lag plus one integrator has been presented.

Very simple formulas have been obtained to approximate the GPC parameters. With the help of this formulas, the GPC can be implemented and tuned very easily allowing and adaptive policy even in fast systems with small sampling times.

This straightforward formulation not only reduces calculations, but poses GPC implementation and tuning in an intuitive form similar to that usually employed by practitioners used to PID controllers and Ziegler-Nichols tuning rules.

References

[1] K.J. Aström and B. Wittenmark. *Computer Controlled Systems. Theory and Design.* Prentice-Hall. Englewood Cliffs, NJ, 1984.

[2] E.F. Camacho, M. Berenguel, and C. Bordons. Adaptive Generalized Predictive Control of a Distributed Collector Field. *IEEE Trans. on Systems Technology*, 2(4):462–467, 1994.

[3] E.F. Camacho, M. Berenguel, and F.R. Rubio. Application of a Gain Scheduling Generalized Predictive Controller to a Solar Power Plant. *Control Engineering Practice*, 2(2):227–238, 1994.

[4] E.F. Camacho and C. Bordons. Implementation of Self Tuning Generalized Predictive Controllers for the Process Industry. *Int. Journal of Adaptive Control and Signal Processing*, 7:63–73, 1993.

[5] E.F. Camacho and C. Bordons. *Model Predictive Control in the Process Industry.* Springer-Verlag, 1995.

[6] E.F. Camacho and J.M. Quero. Precomputation of Generalized Predictive Self-tuning Controllers. *IEEE Trans. on Automatic Control*, 36(7):852–859, 1991.

[7] D. W. Clarke. Application of Generalized Predictive Control to Industrial Processes. *IEEE Control Systems Magazine*, 122:49–55, 1988.

[8] D.W. Clarke, C. Mohtadi, and P.S. Tuffs. Generalized Predictive Control - Part I. The Basic Algorithm. *Automatica*, 23(2):137–148, 1987.

[9] C.R. Cutler and B.C. Ramaker. Dynamic Matrix Control- A Computer Control Algorithm. In *Automatic Control Conference, San Francisco*, 1980.

[10] R. Isermann. *Digital Control Systems.* Springer-Verlag, 1981.

[11] J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model Predictive Heuristic Control: Application to Industrial Processes. *Automatica*, 14(2):413–428, 1978.