A Candidate to Replace PID Control: SISO-Constrained LQ Control

Gabriele Pannocchia
Dept. of Chemical Engineering, University of Pisa, 56126 Pisa, Italy

Nabil Laachi and James B. Rawlings
Dept. of Chemical and Biological Engineering, University of Wisconsin, Madison, WI 53706

DOI 10.1002/aic.10373
Published online March 3, 2005 in Wiley InterScience (www.interscience.wiley.com).

It is commonly believed that for single-input/single-output (SISO) systems, well-tuned proportional, integral, derivative (PID) controllers work as well as model-based controllers and that PID controllers are more robust to model errors. In this paper we present a novel offset-free constrained linear quadratic (LQ) controller for SISO systems, which is implemented in an efficient way so that the total controller execution time is similar to that of a PID. The proposed controller has three modules: a state and disturbance estimator, a target calculation, and a constrained dynamic optimization. It is shown that the proposed controller outperforms PID both in setpoint changes and disturbance rejection, it is robust to model errors, it is insensitive to measurement noise, and it handles constraints better than common anti-windup PID. Tuning the proposed controller is simple. In principle there are three tuning parameters to choose, but in all examples presented only one was actually varied, obtaining a clear and intuitive effect on the closed-loop performance. © 2005 American Institute of Chemical Engineers *AIChE J,* 51: 1178–1189, 2005

Keywords: model predictive control (MPC), linear quadratic regulation (LQR), PID control, tuning, constraints

Preamble: Six myths of PID and LQ Control

To stimulate discussion and set the stage for the results section of the paper, we present first what we call six myths of PID (proportional, integral, derivative) and LQ (linear quadratic) control of SISO (single-input/single-output) systems.

Myth 1. A PID controller is simpler to implement and tune than an LQ controller.

Myth 2. A PID controller with model-based tuning is as good as model-based control for simple processes such as SISO, first-order plus time delay.

Myth 3. A well-tuned PID controller is more robust to plant/model mismatch than an LQ controller.

Myth 3 (Alternate Version). LQ controllers are not very robust to plant/model mismatch.

Myth 4. Integrating the tracking error as in PID control is necessary to remove steady-state offset. Applying some anti-windup strategy for this integrator is thus necessary when an input saturates.

Myth 5. For simple processes (SISO, first-order plus time delay) in the presence of input saturation, a PID controller with a simple anti-windup strategy is as good as model predictive control.

Myth 6. PID controllers are omnipresent because they work well on most processes.

Like all myths, it is impossible to assign a single author or pin down a precise history of these statements. Again, like all

Correspondence concerning this article should be addressed to G. Pannocchia at g.pannocchia@ing.unipi.it.

myths, many slightly different variations of these same basic ideas are in currency. One may argue with the particular details of any one of these statements, but these basic ideas have been repeated by many people over many years. This particular selection provides ample motivation for the rest of the article, and we revisit these statements, each in turn, at the conclusion of the paper.

Introduction

PID control for SISO systems shows up everywhere in chemical process applications and process control education. Tuning rules are presented in numerous texts and, surprisingly, remain a topic of current control (research).[1,2] In this article we raise the issue of whether this popularity is attributable to any concrete technological advantage of PID controllers, or whether the popularity of PIDs is simply a historical accident stemming from the success of analog PID controllers. The main technical advantages ascribed to PID control are: PID is simple, fast, and easy to implement in hardware and software; it is easy to tune; it provides good nominal control performance; and it is robust to model errors.

Model-based control methods, such as LQ control of unconstrained systems, and model predictive control (MPC) of constrained systems, on the other hand, are regarded by many in process control as complex to implement and tune. The robustness of LQ control to model error has been a topic of debate.[3] Some claim that PID controllers outperform MPC controllers in the rejection of unmeasured load disturbances.[4] MPC has become the advanced controller of choice by industry mainly for the economically important, large-scale, multivariable processes in the plant. The rationale for MPC in these applications is that the complexity of implementing MPC is justified only for the important loops with large payoffs.

To address this perception of complexity, we propose a constrained, SISO constrained, linear quadratic (CLQ) controller with the following features: it is essentially as fast to execute as PID (within a factor of 5 regardless of system order), it is easy to implement in software and hardware, and it displays both higher performance and better robustness than those of PID controllers.

Other researchers have explored the following, related topics. A SISO model predictive controller based on a first-order plus time delay model, with input horizon of one, is proposed in Mukati and Ogunnaike.[5] Soroush and Muske[6] show that a particular MPC algorithm with input horizon of one, has PI or PID form when the system is first order or second order (without delay), respectively.

The rest of this article is organized as follows. In the third section we present the proposed offset-free CLQ regulator for single-input/single-output systems, and we discuss its properties. In the fourth section we discuss how the proposed controller can be implemented in an efficient manner for applications on simple hardware and programming languages. In the fifth section we present a number of examples to show the advantages with respect to PID controllers, and in the last section we summarize the main achievements of the proposed controller. An abbreviated form of this paper was presented at the DYCOPS 7 meeting.[7]

Notation. I_N denotes the identity matrix of dimension N and $\mathbf{1}_N$ denotes a column vector of length N with all elements equal to 1. When the dimension is clear from the context the subscript is omitted.

Offset-Free Constrained LQ Controller for SISO Systems

Because most tuning rules for PID controllers require simple transfer function process models, we assume such a model is available. The offset-free CLQ control algorithm has three main modules that use a state-space realization of the system model:

- A state and disturbance estimator
- A constrained target calculation
- A constrained dynamic optimization

Model and estimator

We assume that a state-space discrete-time model of the system to be regulated is known

$$x_{k+1} = Ax_k + Bu_{k-m}$$

$$y_k = Cx_k \tag{1}$$

in which $x \in \mathbb{R}^n$ is the state vector; $u \in \mathbb{R}$ is the input; $y \in \mathbb{R}$ is the output; and m is a nonnegative integer, the time delay.

Assumption 1 (General). The pair (A, B) is controllable, the pair (C, A) is observable, and the following condition holds

$$\operatorname{rank}\begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix} = n + 1 \tag{2}$$

The input u is assumed to be constrained as follows

$$u_{\min} \le u \le u_{\max} \tag{3}$$

in which $u_{\min} < u_{\max}$.

Remark 1. It is important to note that any minimal discrete-time state-space realization of a proper nonsingular transfer function model satisfies Assumption 1. The input rate of change constraints can be directly included in the proposed framework, but are omitted for simplicity of presentation.

To guarantee offset-free control of the output y in the presence of plant/model mismatch and/or unmeasured integrating disturbances, the system model expressed in Eq. 1 is augmented with an integrating disturbance according to the general methodology proposed in Pannocchia and Rawlings.[8] This methodology requires one to add a number of integrating disturbances equal to the number of measured variables (in this SISO case, one) in a way that the resulting augmented system is detectable. To this aim infinitely many choices are available, and in this work we choose the so-called *input disturbance model*, that is, we add an integrating state d that enters the system at the same place as the input u. The resulting augmented system is as follows

$$\begin{bmatrix} x \\ d \end{bmatrix}_{k+1} = \begin{bmatrix} A & B \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ d \end{bmatrix}_k + \begin{bmatrix} B \\ 0 \end{bmatrix} u_{k-m}$$

$$y_k = [C \quad 0] \begin{bmatrix} x \\ d \end{bmatrix}_k \tag{4}$$

Remark 2. The detectability condition for the augmented system (Eq. 4), stated in Lemma 1 of Pannocchia and Rawlings,[8] is guaranteed by Assumption 1. Several studies have pointed out that such a disturbance model is an appropriate choice for efficiently rejecting unmeasured disturbances,[9,10] and it provides good robustness to plant/model mismatch.[11]

The state x and the disturbance d are estimated from the plant measurement y by means of a steady-state Kalman filter. At each sampling time, an estimate of the state $\hat{x}_{k \mid k-1}$ and of the disturbance $\hat{d}_{k|k-1}$ based on previous measurements and inputs are available. Thus, the current filtered state and disturbance, respectively, are

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + L_x(y_k - C\hat{x}_{k|k-1})$$

$$\hat{d}_{k|k} = \hat{d}_{k|k-1} + L_d(y_k - C\hat{x}_{k|k-1}) \tag{5}$$

in which the filter gains $L_x \in \mathbb{R}^n$ and $L_d \in \mathbb{R}$ are computed offline as described later in this paragraph. Given the input u_{k-m} (stored if $m > 0$, or computed as described in the next paragraph, if $m = 0$), the state and disturbance estimates, respectively, for the next sampling time are

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} + Bu_{k-m} + B\hat{d}_{k|k}$$

$$\hat{d}_{k+1|k} = \hat{d}_{k|k} \tag{6}$$

To compute the filter gains L_x and L_d, let

$$\hat{A} = \begin{bmatrix} A & B \\ 0 & 1 \end{bmatrix} \qquad \hat{Q} = \begin{bmatrix} q_x I_n & 0 \\ 0 & 1 \end{bmatrix} \qquad \hat{C} = [C \quad 0]$$

in which q_x is a nonnegative scalar. Also let R_v be a positive scalar; then, the estimator steady-state Riccati equation is[12]

$$\Pi = \hat{Q} + \hat{A}\Pi\hat{A}^T - \hat{A}\Pi\hat{C}^T(\hat{C}\Pi\hat{C}^T + R_v)^{-1}\hat{C}\Pi\hat{A}^T$$

in which $\Pi \in \mathbb{R}^{(n+1)\times(n+1)}$ is a symmetric semidefinite matrix. Finally, the filter gain is

$$L = \begin{bmatrix} L_x \\ L_d \end{bmatrix} = \Pi\hat{C}^T(\hat{C}\Pi\hat{C}^T + R_v)^{-1} \tag{7}$$

Strictly speaking, q_x and R_v should be regarded as the estimator tuning parameters. The scalar q_x represents the ratio between the variance of the state noise and the variance of the disturbance noise, and an increase in q_x makes the estimator less "aggressive" in estimating the disturbance. The scalar R_v represents, instead, the ratio between the variance of output noise and the variance of the disturbance noise, and an increase in R_v makes the estimator less "sensitive" to output noise. In this paper, fixed values for q_x (0.05) and R_v (0.01) are used for all examples. As a general rule, we recommend varying R_v to ensure low sensitivity to output noise.

Constrained target calculation

At each sampling time, given the current disturbance estimate $\hat{d}_{k|k}$, $\bar{y}$, we compute the steady-state targets for input and state such that the output ultimately reaches the set point. If the input is unconstrained, these targets are simply the solution to the following square linear system

$$\begin{bmatrix} I - A & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_k \\ \bar{u}_k \end{bmatrix} = \begin{bmatrix} B\hat{d}_{k|k} \\ \bar{y} \end{bmatrix} \tag{8}$$

which exists because of Assumption 1. However, the solution to this system may be such that the input target violates Eq. 3. Moreover, for integrating processes it is possible that a steady state does not exist because of the input constraints in Eq. 3. For these reasons, we compute the targets from the following quadratic program

$$(\bar{x}_k, \bar{u}_k) = \arg\min_{(\bar{x},\bar{u})} \frac{1}{2}\{(C\bar{x} - \bar{y})^2 + \eta[(I - A)\bar{x} - B(\bar{u} + \hat{d}_{k|k})]^T[(I - A)\bar{x} - B(\bar{u} + \hat{d}_{k|k})]\} \tag{9a}$$

subject to

$$u_{\min} \le \bar{u} \le u_{\max} \tag{9b}$$

in which η is a large positive number.

Remark 3. The large penalty η used in the second term of Eq. 9a guarantees that $(\bar{x}_k, \bar{u}_k)$ denote a steady state, that is, such that $\bar{x}_k = A\bar{x}_k + B\bar{u}_k + B\hat{d}_{k|k}$ holds, whenever a steady state exists. It is important to notice that, if the $(\bar{x}_k, \bar{u}_k)$ solution to Eq. 8 satisfies $u_{\min} \le \bar{u}_k \le u_{\max}$, the quadratic program expressed in Eq. 9 returns the same values as those in Eq. 8.

Constrained dynamic optimization problem: case of reachable setpoint

Given the estimates of the current state and disturbance $\hat{x}_{k|k}$ and $\hat{d}_{k|k}$, respectively, and given the current steady-state targets $(\bar{x}_k, \bar{u}_k)$, for the cases in which the desired set point is reachable (that is, $C\bar{x}_k = \bar{y}$), we compute the control input by means of the following constrained dynamic optimization problem

$$\min_{\{v_j\}_{j=0}^{N-1}} \frac{1}{2}\left\{\sum_{j=0}^{N-1} w_j^T Q w_j + s(v_j - v_{j-1})^2\right\} + \frac{1}{2}\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix}^T P \begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix} \tag{10a}$$

subject to

$$w_0 = \hat{x}_{k+m|k} - \bar{x}_k = \left[A^m\hat{x}_{k|k} + \sum_{i=1}^{m} A^{i-1}B(u_{k-i} + \hat{d}_{k|k})\right] - \bar{x}_k$$

$$v_{-1} = u_{k-1} - \bar{u}_k \tag{10b}$$

$$w_{j+1} = Aw_j + Bv_j \tag{10c}$$

$$u_{\min} - \bar{u}_k \le v_j \le u_{\max} - \bar{u}_k \tag{10d}$$

in which N is a positive integer, s a positive scalar, and $Q = C^TC$. The matrix $P \in \mathbb{R}^{(n+1)\times(n+1)}$ is chosen as the symmetric positive semidefinite solution of the following Riccati equation

$$P = \tilde{Q} + \tilde{A}^TP\tilde{A} - \tilde{A}^TP\tilde{B}(\tilde{B}^TP\tilde{B} + s)^{-1}\tilde{B}P\tilde{A} \tag{11}$$

in which

$$\tilde{A} = \begin{bmatrix} A & B \\ 0 & 1 \end{bmatrix} \qquad \tilde{B} = \begin{bmatrix} B \\ 1 \end{bmatrix} \qquad \tilde{Q} = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \tag{12}$$

Let $v := (v_0, v_1, \ldots, v_{N-1})$ be a column vector of length N. We can write Eq. 10 as a strictly convex QP

$$\min_v \frac{1}{2} v^THv + v^T\tilde{c} \tag{13a}$$

subject to

$$\mathbf{1} \cdot (u_{\min} - \bar{u}_k) \le v \le \mathbf{1} \cdot (u_{\max} - \bar{u}_k) \tag{13b}$$

in which

$$H = \mathcal{B}^T\mathcal{Q}\mathcal{B} + \mathcal{D}^T\mathcal{R}\mathcal{D} \qquad \tilde{c} = \mathcal{B}^T\mathcal{Q}\mathcal{A}w_0 + \mathcal{D}^T\mathcal{R}\mathcal{C}v_{-1} \tag{14}$$

and

$$\mathcal{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \qquad \mathcal{B} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix}$$

$$\mathcal{C} = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \mathcal{D} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -1 & 1 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & 0 & -1 & 1 \end{bmatrix} \tag{15a}$$

$$\mathcal{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & Q & 0 \\ 0 & \cdots & 0 & P \end{bmatrix} \qquad \mathcal{R} = \begin{bmatrix} s & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & s \end{bmatrix} \tag{15b}$$

Let $v^* = (v_0^*, \ldots, v_{N-1}^*)$ denote the optimal solution to Eq. 13. Then, the current control input is defined by using a receding horizon implementation, that is

$$u_k = \bar{u}_k + v_0^* \tag{16}$$

Remark 4. It can be shown that the matrix P solution to Eq. 11 is such that the quadratic terminal penalty in Eq. 10 corresponds to the following infinite-horizon unconstrained cost-to-go

$$\frac{1}{2}\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix}^T P \begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix} = \min_{\{v_j\}_{j=N}^{\infty}} \frac{1}{2}\left\{ \sum_{j=N}^{\infty} w_j^TQw_j + s(v_j - v_{j-1})^2 \right\} \tag{17a}$$

subject to

$$w_{j+1} = Aw_j + Bv_j \tag{17b}$$

Furthermore, the corresponding optimal unconstrained control law associated to this cost-to-go is

$$v_j = K\begin{bmatrix} w_j \\ v_{j-1} \end{bmatrix} \tag{18}$$

in which

$$K = \tilde{K} + [0_{1\times n} \quad 1] \qquad \tilde{K} = -(s + \tilde{B}^TP\tilde{B})^{-1}\tilde{B}^TP\tilde{A} \tag{19}$$

Constrained dynamic optimization problem: case of unreachable setpoint

When the setpoint is not reachable, that is, when the feasible steady-state targets are such that

$$\bar{y}_k = C\bar{x}_k \ne \bar{y} \tag{20}$$

the optimization problem (Eq. 10) needs to be modified because the corresponding optimal input would drive the controlled variable to the reachable target $\bar{y}_k$ as quickly as possible. There are important cases in which this behavior is undesirable. These situations occur when a "large" disturbance enters the system, and the input constraints are such that the input asymptotically saturates without completely rejecting the disturbance, and thus offset occurs. It is clear that, if the disturbance continues to affect the system, steady-state offset is unavoidable, although even in such cases it is desirable to keep the controlled variable close to the desired setpoint $\bar{y}$ as long as possible. This goal can be achieved by modifying the optimization problem (Eq. 10) with a linear penalty,[13] as follows

$$\min_{\{v_j\}_{j=0}^{N-1}} \frac{1}{2}\left\{ \sum_{j=0}^{N-1} w_j^T(Qw_j + 2q) + s(v_j - v_{j-1})^2 \right\} + \frac{1}{2}\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix}^T \left(P\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix} + 2p \right) \tag{21a}$$

subject to (10*b*)–(10*d*) (21b)

in which P is given in Eq. 11 and

$$q = C^T(\bar{y}_k - \bar{y}) \qquad p = (I - (\tilde{A} + \tilde{B}\tilde{K})^T)^{-1}\begin{bmatrix} q \\ 0 \end{bmatrix} \tag{22}$$

Notice that if $\bar{y}_k = \bar{y}$ (that is, the setpoint is reachable), we obtain the same formulation as in Eq. 10.

Similarly to the previous case, let $v := (v_0, v_1, \ldots, v_{N-1})$ be a column vector of length N. We can write Eq. 21 as the same strictly convex QP in Eq. 13, in which H is still given in Eq. 14, whereas $\tilde{c}$ is

$$\tilde{c} = \mathcal{B}^T\mathcal{Q}\mathcal{A}w_0 + \mathcal{B}^T\mathcal{P} + \mathcal{D}^T\mathcal{R}\mathcal{C}v_{-1} \tag{23}$$

in which $\mathcal{P} = [q^T \cdots q^T\ p^T]^T$. Let $v^* = (v_0^*, \ldots, v_{N-1}^*)$ denote the optimal solution to Eq. 13 with $\tilde{c}$ given in Eq. 23. Then, the current control input is still defined by Eq. 16.

Remark 5. It is possible to show that the chosen linear and quadratic terminal penalties in Eq. 21 correspond to the following infinite-horizon unconstrained cost-to-go

$$\frac{1}{2}\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix}^T\left(P\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix} + 2p\right) = \frac{1}{2}\left\{\sum_{j=N}^{\infty} w_j^T(Qw_j + 2q) + s(v_j - v_{j-1})^2\right\} \tag{24a}$$

subject to

$$w_{j+1} = Aw_j + Bv_j \qquad v_j = K\begin{bmatrix} w_j \\ v_{j-1} \end{bmatrix} \tag{24b}$$

in which K is given in Eq. 19. Also notice that Eq. 24 is equivalent to Eq. 17 if $\bar{y}_k = \bar{y}$.

Properties

If constraints are not present, the proposed controller reduces to an infinite-horizon LQ controller with target calculation and origin "shifting" (see Kwakernaak and Sivan,[12] p. 504). Thus, it is easy to show that it is nominally stable for any choice of the tuning parameters.

It is also possible to derive a simple sufficiency test for nominal constrained stability using ellipsoid invariant set theory.[14] This test, described below, can be used online to detect whether the terminal state is not in the output admissible set and to flag a warning for the operator. If one wishes to have a guarantee of nominal stability, we can easily formulate the regulator with the terminal constraint $w_N = 0$. However, because N is chosen fairly small for computational speed, we find the terminal state constraint controller not as robust as the one presented here, and therefore do not recommend it for industrial practice.

Suppose that P solution to Eq. 11 is positive definite[1] and consider the following ellipsoid region

$$\mathbb{O}_{\text{ell}} = \{\xi | \xi^T P \xi \le \alpha^2\} \tag{25}$$

in which α is a positive scalar to be determined. It immediately follows from the definition of P that if $\xi \in \mathbb{O}_{\text{ell}}$, then $(\tilde{A} + \tilde{B}\tilde{K})^i\xi \in \mathbb{O}_{\text{ell}}$ for any $i = 0, 1, \cdots$, that is $\mathbb{O}_{\text{ell}}$ is positively invariant for the system $\xi_{k+1} = (\tilde{A} + \tilde{B}\tilde{K})\xi_k$. Thus, we want to compute α such that

$$\xi^T P \xi \le \alpha^2 \Rightarrow u_{\min} - \bar{u}_k \le K\xi \le u_{\max} - \bar{u}_k \tag{26}$$

in which K is given in Eq. 19. To compute α, consider the symmetric Schur decomposition of P (see Golub and Van Loan,[15] p. 393)

$$P = V\Lambda V^T \tag{27}$$

in which $V \in \mathbb{R}^{(n+1)\times(n+1)}$ is orthogonal and $\Lambda \in \mathbb{R}^{(n+1)\times(n+1)}$ is diagonal with strictly positive elements. Let

$$H = [h_1 \quad \cdots \quad h_{n+1}] = KV\Lambda^{-1/2} \qquad \beta = \frac{1}{\sqrt{h_1^2 + \cdots + h_{n+1}^2}} \tag{28}$$

It is possible to show that the largest value of α (that is, the largest ellipsoid), such that Eq. 26 holds, is given by

$$\alpha = \beta \min(\bar{u}_k - u_{\min}, u_{\max} - \bar{u}_k) \tag{29}$$

Hence, at each sampling time, after having computed the optimal input sequence v^*, we can perform a sufficiency test for nominal constrained stability by checking whether or not

$$\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix}^T P\begin{bmatrix} w_N \\ v_{N-1} \end{bmatrix} \le \alpha^2 \tag{30}$$

holds.

Remark 6. Notice that, in general, the value of α is to be computed at each sampling time because the input target $\bar{u}_k$ may change. To this aim, one can simply use Eq. 29, in which β is computed offline by means of Eq. 28. Also notice that this sufficiency test is meaningful only when constraints are not active at steady state because in such a case we would have that either $\bar{u}_k = \bar{u}_{\min}$ or $\bar{u}_k = \bar{u}_{\max}$, and thus $\alpha = 0$.

Another important property of the proposed CLQ controller is that it guarantees offset-free control whenever the closed-loop system reaches a steady state in which the input is not saturated. This property holds independently of the plant dynamics and is attributed to the presence of the integrating state d in Eq. 4 (see Pannocchia and Rawlings,[8] Theorem 1). It is important to remark that, unlike PID, the proposed controller does not integrate the tracking error (that is, $\bar{y} - y_k$). In fact from Eqs. 5 and 6 one can write

$$\hat{d}_{k+1|k} = \hat{d}_{k|k-1} + L_d(y_k - C\hat{x}_{k|k-1})$$

from which it is clear that there is integration of the prediction error (that is, $y_k - C\hat{x}_{k|k-1}$). This approach is significantly

[1] If P is only semidefinite, a simple modification to this test is to use P as the solution to the following Lyapunov equation: $P = R + (\tilde{A} + \tilde{B}\tilde{K})^T P(\tilde{A} + \tilde{B}\tilde{K})$, where R is positive definite.

different from integration of the tracking error as in PID control, and it does not require any anti-windup strategy when the input saturates.

Furthermore, unlike PID control, CLQ is a "two-degrees-of-freedom" controller and it can simultaneously provide both efficient setpoint tracking and disturbance rejection. This feature is ascribed to the state and disturbance estimator, which can also be tuned to be insensitive to measurement noise by adjusting R_v.

Efficient Implementation of CLQ

Two modules of CLQ—the target calculation and the constrained dynamic optimization—require one to solve a quadratic program at each sampling time. For the proposed method to be applicable to simple hardware and programming languages, we have developed efficient methods for solving these QPs.

Constrained target calculation

The first step is to compute the solution to the unconstrained problem (Eq. 8)

$$\begin{bmatrix} \bar{x}^* \\ \bar{u}^* \end{bmatrix} = \begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} B\hat{d}_{k|k} \\ \bar{y} \end{bmatrix} = \begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} B & 0_{n\times 1} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{d}_{k|k} \\ \bar{y} \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} \hat{d}_{k|k} \\ \bar{y} \end{bmatrix} \tag{31}$$

in which $M_{11} \in \mathbb{R}^n$, $M_{12} \in \mathbb{R}$, $M_{21} \in \mathbb{R}$, and $M_{22} \in \mathbb{R}^n$ are computed offline. If $u_{\min} \le \bar{u}^* \le u_{\max}$, then we set $\bar{u}_k = \bar{u}^*$ and $\bar{x}_k = \bar{x}^*$. Otherwise, we set

$$\bar{u}_k = \text{sat}(\bar{u}^*) = \begin{cases} u_{\min} & \text{if } \bar{u}^* < u_{\min} \\ u_{\max} & \text{if } \bar{u}^* > u_{\max} \end{cases} \tag{32}$$

and we find $\bar{x}_k$ from the following unconstrained problem

$$\bar{x}_k = \arg\min_{\bar{x}} \frac{1}{2} \{ (C\bar{x} - \bar{y})^2 + \eta [(I-A)\bar{x} - B(\bar{u}_k + \hat{d}_{k|k})]^T [(I-A)\bar{x} - B(\bar{u}_k + \hat{d}_{k|k})] \} \tag{33}$$

Notice that the QP in Eq. 33 has an analytical solution because it is unconstrained. In fact, we can rewrite it as follows

$$\bar{x}_k = \arg\min_{\bar{x}} \frac{1}{2} \bar{x}^T [C^T C + \eta (I-A)^T (I-A)] \bar{x} - \bar{x}^T [C^T \bar{y} + \eta (I-A)^T B(\bar{u}_k + \hat{d}_{k|k})]$$

the solution of which is

$$\bar{x}_k = [C^T C + \eta (I-A)^T (I-A)]^{-1} [C^T \bar{y} + \eta (I-A)^T B(\bar{u}_k + \hat{d}_{k|k})] = G_1 \hat{d}_{k|k} + G_2 \bar{y} + G_3 \bar{u}_k \tag{34}$$

in which $G_1 \in \mathbb{R}^n$, $G_2 \in \mathbb{R}^n$, and $G_3 \in \mathbb{R}^n$ are computed offline.

Given the discussion above we present the following algorithm for computing the state and input steady-state feasible targets.

Algorithm 1 (Constrained target calculation). *Given* $\hat{d}_{k|k}$ *and* $\bar{y}$.

1. *Compute the unconstrained input target* $\bar{u}^* = M_{21}\hat{d}_{k|k} + M_{22}\bar{y}$.
2. *If* $u_{\min} \le \bar{u}^* \le u_{\max}$, *set* $\bar{u}_k = \bar{u}^*$, *compute* $\bar{x}_k = M_{11}\hat{d}_{k|k} + M_{12}\bar{y}$ *and stop. Otherwise,*
3. *Set* $\bar{u}_k = \text{sat}(\bar{u}^*)$ *and compute* $\bar{x}_k = G_1\hat{d}_{k|k} + G_2\bar{y} + G_3\bar{u}_k$.

Constrained dynamic optimization problem

Introduction. As shown in the previous section, the constrained dynamic optimization problem is written as a QP (Eq. 13) that needs to be solved at each sampling time. It is important to notice that in Eq. 13 the matrix H does not change at each sampling time, whereas the vector $\tilde{c}$ changes with time because w_0 and v_{-1} (and also q and therefore $\mathcal{P}$ for the case of unreachable setpoint) change. Moreover, the bounds on v may change at each sampling time (because $\bar{u}_k$ may change), although this problem can be overcome by defining a new vector

$$u = v + \mathbf{1}\bar{u}_k \tag{35}$$

which allows one to rewrite Eq. 13 as follows

$$\min_u \frac{1}{2} u^T H u + u^T c \tag{36a}$$

subject to

$$\mathbf{1}u_{\min} \le u \le \mathbf{1}u_{\max} \tag{36b}$$

in which $c = \tilde{c} - H\bar{u}_k$. Notice that in Eq. 36 only c varies at each sampling time. Moreover, it is clear from Eqs. 16 and 35 that the control input u_k is defined as the first element of the optimal solution to Eq. 36.

For large multivariable systems the use of on-line optimization seems unavoidable, whereas for relatively small systems one can choose a multiparametric quadratic programming (mp-QP) approach, as proposed in Ref. 16. This algorithm can be applied to Eq. 36 to build a solution table and compute the control input as an affine function of c. However, an alternative and simpler method, specifically tailored to SISO systems with constraints on the input only, is developed here. This new method has two basic steps:

(1) The offline generation of a solution table using H, $u_{\min}$, and $u_{\max}$: this step involves solving linear equations (that is, matrix inversions), multiplications, and additions.

(2) The online table scanning given the current value of c: this step involves only multiplications and additions and checking conditionals. These same operations are required in PID control.

The problem (Eq. 36) can be rewritten in a standard optimization notation as follows

$$\min_u \frac{1}{2} u^T H u + u^T c \tag{37a}$$

subject to

$$Du \geq d \tag{37b}$$

in which

$$D = \begin{bmatrix} -I_N \\ I_N \end{bmatrix} \qquad d = \begin{bmatrix} -\mathbf{1}_N u_{\max} \\ \mathbf{1}_N u_{\min} \end{bmatrix} \tag{38}$$

Because H is positive definite, the quadratic program (Eqs. 37a and 37b) has a unique solution u^*, which must satisfy the following KKT first-order conditions (see Nocedal and Wright,[17] p. 454)

$$Hu^* + c - A_a^T \lambda^* = 0 \tag{39a}$$

$$A_a u^* = b_a \tag{39b}$$

$$A_r u^* \geq b_r \tag{39c}$$

$$\lambda^* \geq 0 \tag{39d}$$

in which $\lambda^* \in \mathbb{R}^l$; $A_a \in \mathbb{R}^{l \times N}$, and $b_a \in \mathbb{R}^l$ are formed by stacking the rows of D and d corresponding to the active constraints at the optimal point, and $A_r \in \mathbb{R}^{(2N-l)\times N}$ and b_r^{2N-l} are formed by stacking the remaining rows of D and d, respectively. Notice that the following properties hold

$$l \leq N \qquad \text{rank}(A_a) = l \tag{40}$$

Also notice that if $l > 0$, then

$$A_a A_a^T = I_l \tag{41}$$

Building the Table. Each component of u can be at the lower bound, at the upper bound or somewhere in between. Thus, we can construct all possible combinations of active and inactive constraints, and such combinations are in number 3^N. For each $i = 1, 2, \ldots, 3^N$ we denote with A_i and b_i the rows of D and d corresponding to the active constraints of the ith possible solution. We denote with u_i the solution of the following equality-constrained quadratic program

$$\min_u \frac{1}{2} u^T H u + u^T c \tag{42a}$$

subject to

$$A_i u = b_i \tag{42b}$$

which is given by the solution of the following square linear system (see Nocedal and Wright,[17] p. 444)

$$\begin{bmatrix} H & -A_i^T \\ A_i & 0 \end{bmatrix} \begin{bmatrix} u_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} -c \\ b_i \end{bmatrix} \tag{43}$$

The solution u_i has the following form

$$u_i = K_i c + B_i \tag{44}$$

in which $K_i \in \mathbb{R}^{N \times N}$ and $B_i \in \mathbb{R}^N$ are computed from the corresponding KKT system, using the "null space" method (see Nocedal and Wright,[17] p. 450), as described below.

Given the ith active set matrix and vector (A_i, b_i), we start by building a vector $\bar{u}_i$ such that

$$A_i \bar{u}_i = b_i \tag{45}$$

Because of Eq. 41, one such vector is

$$\bar{u}_i = A_i^T b_i \tag{46}$$

Then we express the solution of Eq. 42, u_i, as $u_i = \bar{u}_i + p_i$ and rewrite Eq. 43 as follows

$$\begin{bmatrix} H & -A_i^T \\ A_i & 0 \end{bmatrix} \begin{bmatrix} p_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} -(H\bar{u}_i + c) \\ 0 \end{bmatrix} \tag{47}$$

From the second row block of Eq. 47 we have that $A_i p_i = 0$, that is, p_i is in the null space of A_i (notice that if A_i is square, then $p_i = 0$ and thus the solution u_i is equal to $\bar{u}_i$). Let $Z_i \in \mathbb{R}^{N \times (N-l)}$ be a matrix whose columns form a basis for the null space of A_i; then we can write p_i as

$$p_i = Z_i r_i \tag{48}$$

in which $r_i \in \mathbb{R}^{N-l}$ is to be computed. From the first row block of Eq. 47 we can write

$$HZ_i r_i - A_i^T \lambda_i = -(H\bar{u}_i + c)$$

and then we multiply by Z_i^T on the left, obtaining

$$Z_i^T H Z_i r_i = -Z_i^T (H\bar{u}_i + c)$$

which can be solved to compute r_i

$$r_i = -(Z_i^T H Z_i)^{-1} Z_i^T (H\bar{u}_i + c) \tag{49}$$

Thus, the solution u_i is written as

$$u_i = \bar{u}_i + p_i = \bar{u}_i + Z_i r_i = \bar{u}_i - Z_i (Z_i^T H Z_i)^{-1} Z_i^T (H\bar{u}_i + c) \tag{50}$$

Finally, the form in Eq. 44 is obtained by letting

$$K_i = -Z_i (Z_i^T H Z_i)^{-1} Z_i^T \qquad B_i = \bar{u}_i - Z_i (Z_i^T H Z_i)^{-1} Z_i^T H \bar{u}_i \tag{51}$$

Finding the Solution. We now assume that a table contain-

ing K_i, B_i (and A_i) for $i = 1, 2, \ldots, 3^N$ is known. We can now compute u^*, the optimal solution to Eq. 37 by "scanning" the table in the following way.

Algorithm 2 (Constrained dynamic optimization). *Given c and starting with $i = 1$, repeat the following steps.*

1. *Compute the ith solution, $u_i = K_i c + B_i$.*
2. *Check whether $Du_i \geq d$. If this is not satisfied, increase $i \leftarrow i + 1$ and go to* 1. *Otherwise,*
3. *Compute the Lagrange multipliers for the active constraints as*

$$\lambda_i = A_i(Hu_i + c) \tag{52}$$

4. *If all elements of λ_i are nonnegative, set $u^* = u_i$ and stop. Otherwise, increase $i \leftarrow i + 1$ and go to* 1.

Remark 7. Notice that given the particular structure of A_i (each row of A_i contains only a nonzero element, which is either 1 or -1), computing from Eq. 52 simply requires one to compute the elements of the gradient vector $g_i = Hu_i + c$ corresponding to the active constraints. Also notice that Algorithm 2 has a finite termination, because Eq. 37 is always feasible, and thus one of the 3^N possible solutions u_i is the optimal solution.

Remark 8. One could also solve the regulation problem using the mp-QP approach presented in Bemporad et al.,[16] which generates offline the regions in state space corresponding to different active sets, and searches online to determine in which region the current state resides. The valuable insight provided by this method is that it reveals the structure of the regulator, a piecewise affine function of the state. Because the state regions are not required for the control calculation, however, we dispense with the offline overhead of generating the state regions in the proposed CLQ controller. Simply enumerating all active sets and testing for optimality is simple and does not require the use of standard efficient QP and LP solvers for even the offline construction of the table. Because of its simplicity, the proposed method generates 3^N table entries for every model, which may be larger than the minimum number of state regions required for the same problem. For SISO applications, however, the number of table entries is small and the online execution is fast, that is, close to the PID execution time.

Illustrative Examples

In this section we present a number of examples of common processes to show that the proposed CLQ controller is simpler to tune than a PID controller, is robust to model errors, is insensitive to noise measurements, and guarantees superior performance both for setpoint changes and disturbance rejections.

Because constraints are present, the common anti-windup "velocity" algorithm for PID is used.[18]

Algorithm 3 (PID with anti-windup). *Initialize the controller with u_{-1}, e_{-1}, e_{-2}.*

1. *Data at sampling time k: y_k, u_{k-1}, e_{k-1}, e_{k-2}.*
2. *Compute the tracking error.*

$$e_k = \bar{y} - y_k$$

3. *Compute the unconstrained input*

$$\Delta u_k = K_c(e_k - e_{k-1}) + \frac{K_c T_s}{T_i} e_k + \frac{K_c T_d}{T_s}(e_k - 2e_{k-1} + e_{k-2})$$

$$u_k = u_{k-1} + \Delta u_k$$

4. *Check saturation*

$$\begin{cases} u_k \leftarrow u_k & \text{if } u_{\min} \leq u_k \leq u_{\max} \\ u_k \leftarrow u_{\max} & \text{if } u_k > u_{\max} \\ u_k \leftarrow u_{\min} & \text{if } u_k < u_{\min} \end{cases}$$

5. *Inject u_k into the plant as input, update time $k \leftarrow k + 1$ and go to* 1.

Notice that if $T_d \neq 0$, the derivative action is suppressed when a setpoint change occurs, which is the common industrial practice to avoid the "derivative kick." As a performance index we choose the following quadratic cost that weighs the tracking error as well as the input movement

$$\Phi = \sum_{k=0}^{\infty} (y_k - \bar{y})^2 + (u_k - u_{k-1})^2 \tag{53}$$

First-order plus time delay

The first example is a first-order plus time delay (FOPTD) system

$$G_1(s) = \frac{e^{-2s}}{10s + 1}$$

sampled with $T_s = 0.25$. The input is assumed to be constrained $|u| \leq 1.5$, a horizon of $N = 4$ is used, and in all simulations the setpoint is changed from 0 to 1 at time zero. At time 25 a load disturbance (that is, a disturbance passed through the same dynamics as the plant) of magnitude -0.25 enters the system; then at time 50 the disturbance magnitude becomes -1 (which makes the setpoint 1 unreachable); finally at time 75 the disturbance magnitude becomes -0.25 again. Figure 1 shows the simulation results in the nominal case for two CLQ controllers and two PID controllers. The estimator is designed with $q_x = 0.05$ and $R_v = 0.01$ for both CLQ controllers, whereas the regulator input penalty is $s = 5$ for CLQ 1 and $s = 50$ for CLQ 2. The tuning parameters for PID 1 are chosen according to Luyben's rules (see Luyben and Luyben,[19] p. 97): $K_c = 2.51$, $T_i = 17.3$, $T_d = 0$. The tuning parameters for PID 2 are chosen according to Skogestad's IMC rules[20]: $K_c = 2.35$, $T_i = 10$, $T_d = 0$. Figure 2 shows the simulation results for CLQ 1 and PID 1 in the presence of random output noise (with variance $\sigma^2 = 0.001$). Figure 3 shows a comparison of the performance index Φ vs. the gain and delay relative plant/model mismatch, respectively, for CLQ 1 and PID 1.

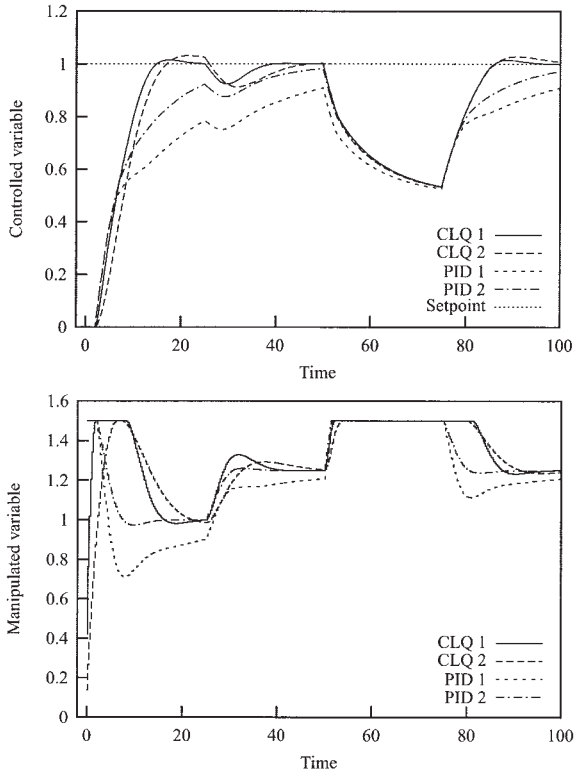

Figure 1. FOPTD system: nominal case.

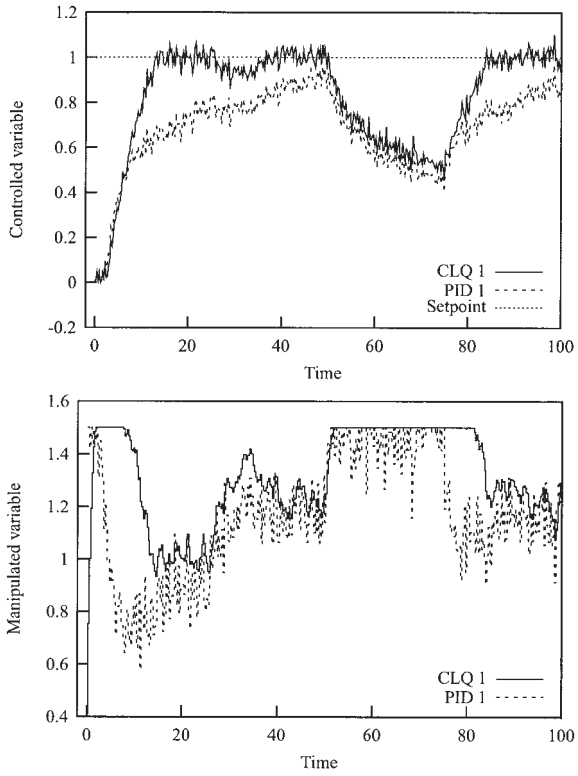

Figure 2. FOPTD system: noisy case.

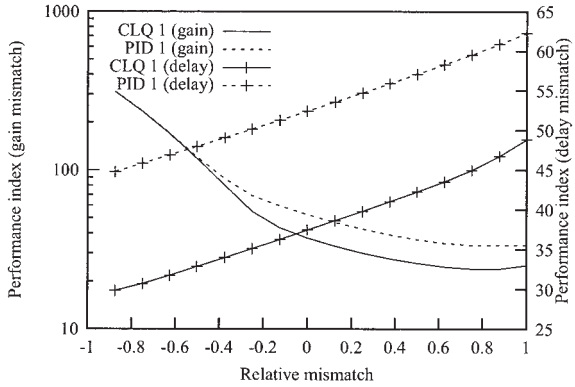

Figure 3. FOPTD system: effect of plant/model mismatch.

Integrating system

The second example is an integrating system

$$G_2(s) = \frac{e^{-2s}}{s}$$

sampled with $T_s = 0.25$. The same input constraints, horizon, setpoint change and disturbances, and estimator parameters as in the first example are considered. CLQ 1 uses a regulator input penalty of $s = 2500$, whereas CLQ 2 uses $s = 10{,}000$. The tuning parameters for PID 1 are chosen according to Luyben's rules (see Luyben and Luyben,[19] p. 97): $K_c = 0.23$, $T_i = 18.7$, $T_d = 0$. The tuning parameters for PID 2 are chosen according to Skogestad's IMC rules[20]: $K_c = 0.23$, $T_i = 17$, $T_d = 0$. Simulation results in the nominal case are reported in Figure 4, whereas Figure 5 shows the simulation results in the presence of random output noise (with variance $\sigma^2 = 0.001$). The performance index Φ vs. the gain and delay relative plant/model mismatch is reported in Figure 6.

Underdamped system

The last example is a second-order underdamped system

$$G_3(s) = \frac{K}{\tau^2 s^2 + 2\tau\xi s + 1}$$

sampled with $T_s = 0.25$, and with nominal parameters of $K = 1$, $\tau = 5$, and $\xi = 0.2$. The same input constraints, horizon, setpoint change and disturbances, and estimator parameters as those in the first example are assumed. CLQ 1 uses a regulator input penalty of $s = 5$, whereas CLQ 2 uses $s = 50$. The tuning parameters for PID 1 are chosen according to Luyben's rules (see Luyben and Luyben,[19] p. 97): $K_c = 7.29$, $T_i = 16.8$, $T_d = 1.21$. The tuning parameters for PID 2 are chosen following the same IMC approach as in Skogestad[20]: $K_c = 0.40$, $T_i = 2$, $T_d = 12.5$. For both PID controllers we filtered the output measurements using an exponential filter with time constant $T_f = 1.1$. If the output is not filtered, the response in the presence of output noise is overly oscillatory because of the derivative action. On the other hand, if no derivative action is used (that is, if a PI controller is chosen), the nominal performance and the stability margins are

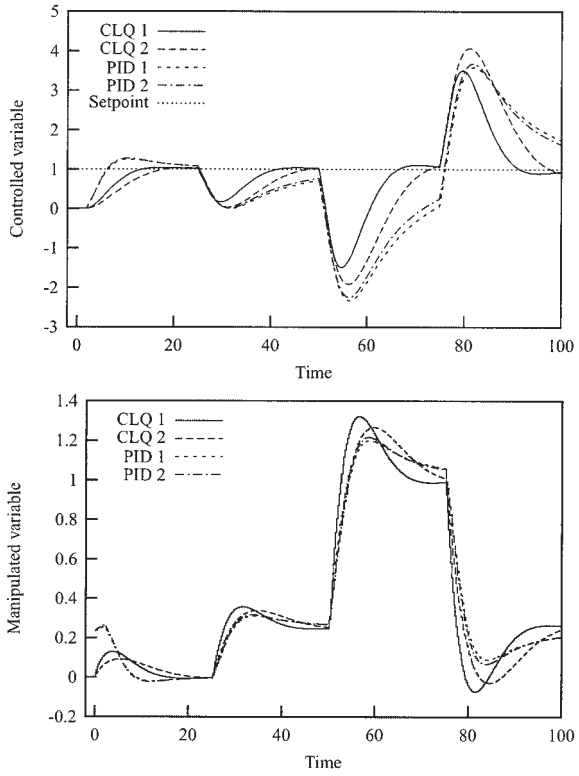

Figure 4. Integrating system: nominal case.

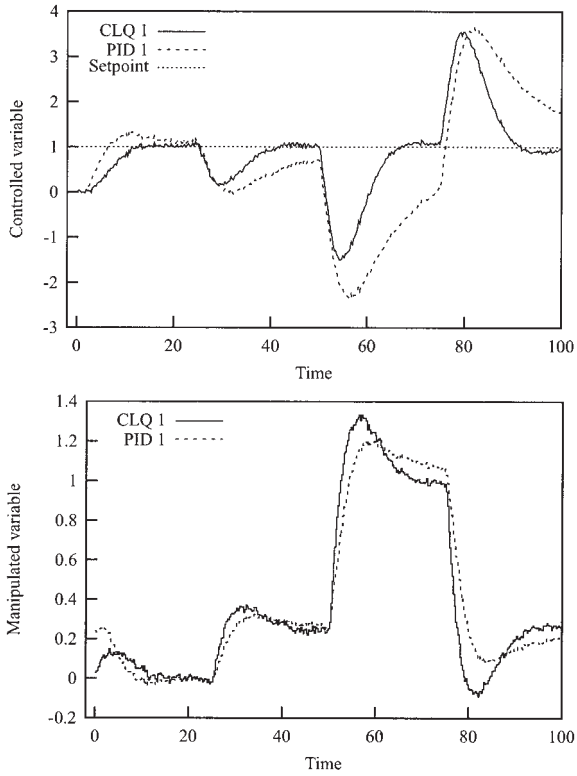

Figure 5. Integrating system: noisy case.

poor. Simulation results for the nominal case are reported in Figure 7, whereas Figure 8 shows the simulation results in the presence of random output noise (with variance $\sigma^2 = 0.001$). The performance index Φ vs. the gain and damping factor relative plant/model mismatch is reported in Figure 9.

Comments and discussion

The results presented in the previous paragraphs clearly show that in all examples (as well as in several others not shown for the sake of space) CLQ outperforms PID both on setpoint changes and disturbance rejection. In the presence of constraints CLQ understands better than PID "when" and "for how long" to saturate the input. Notice that CLQ does not require any anti-windup strategy.

Tuning CLQ is simple: mainly one has to choose only the input penalty s, which quantifies the trade-off between tracking error and input movement. The effect of this tuning parameter is intuitive: the smaller the value of s, the more aggressive the controller. CLQ is robust to plant/model mismatch: one can obtain high-performance closed-loop response in the nominal case and still have robust performance and large stability margins. In Figures 3, 6, and 9 we have shown the performance index for CLQ and PID controllers in a wide relative mismatch range up to 100%. In all examples CLQ controllers outperform PID controllers even within such a wide range of plant/model mismatch.

If one is especially concerned about robust stability, such as in the FOPTD system one can increase s from 5 (shown in Figures 1–3) to 60 and the gain stability range for CLQ is 241% compared to 203% for PID 1, the time-delay stability range for CLQ is 250% (same as for PID 1), and the nominal performance remains 21% better than PID's nominal performance. Similar results can be obtained for the other systems. In other words, it is straightforward to tune a CLQ controller to have simultaneously better nominal performance and larger stability regions than a PID controller tuned with accepted PID tuning rules.

CLQ can be easily tuned to be insensitive to measurement noise by adjusting R_v. If one detects high frequency oscillations in the manipulated variable it is sufficient to increase R_v to suppress this undesirable behavior. In this way, it is not necessary to "slow down " the controller's setpoint response (that is, to increase s) when the measurement is noisy. PID instead suffers from large input oscillations (see Figures 2 and 8), particularly when derivative action is used to improve nominal and robust performance. For this reason, many industrial PID controllers are tuned with $T_d = 0$, and the output measurement is often filtered.

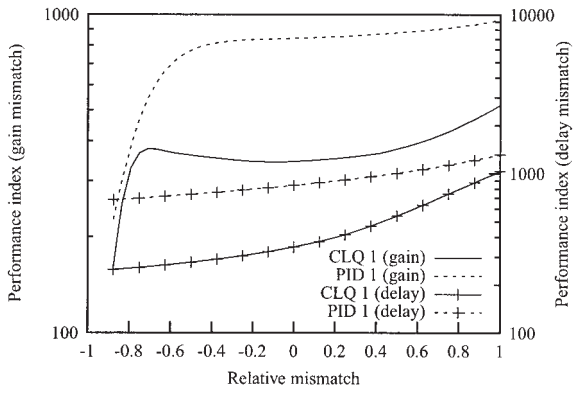

Figure 6. Integrating system: effect of plant/model mismatch.

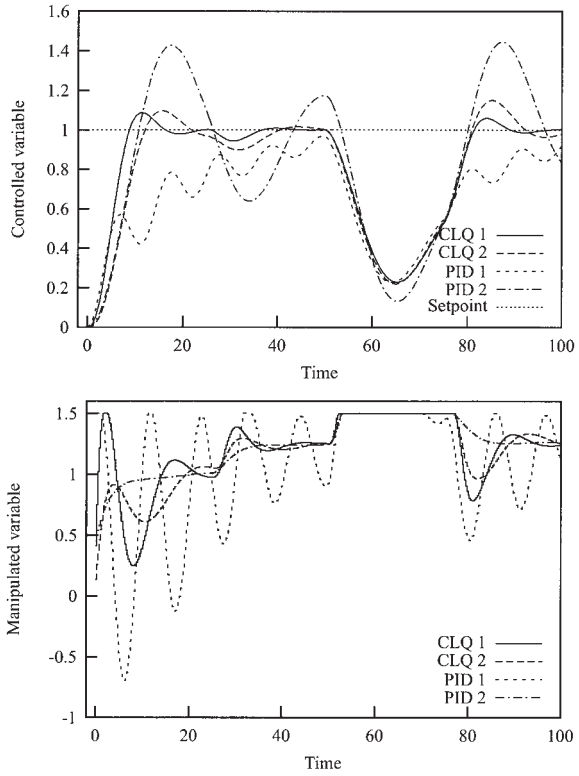

Figure 7. Underdamped system: nominal case.

Finally, it is important to remark that CLQ is efficient and the computational burden is comparable to that of PID. The average CPU time required to compute the control input has been 0.22 ms for CLQ and 0.05 ms for PID (on a 1.7-GHz Athlon PC running Octave[2]). The maximum CPU time has been 0.55 ms for CLQ and 0.10 ms for PID. The computational efficiency comes about because only a small number of simple operations (addition, multiplication, and comparison) are required at each sample time.

Conclusions

In this article, a novel, offset-free, constrained, linear quadratic (CLQ) controller for a SISO system was presented. The purpose of this work was to propose an alternative to digital PID controllers that are commonly available on the distributed control system (DCS).

CLQ has three main modules based on a state-space model of the system: a state and disturbance estimator, a target calculation, and a constrained dynamic optimization. Each module is implemented in an efficient way so that the overall CLQ algorithm has little computational cost and can be applied using simple hardware and software. As shown, the proposed controller outperforms PID controllers in all situations (setpoint changes or disturbance rejections, nominal case, or in the presence of relevant model errors, noise-free or noisy measurements). Moreover, CLQ is simple to tune: there are two main tuning parameters to choose, one in the estimator (R_v) and one in the regulator (s) whose effect on closed-loop performance is intuitive. The proposed controller is "scalable," in the sense that it can be extended to larger multivariable systems in a straightforward fashion. Other possible extensions are:

- the use of feed-forward to reject measured disturbances even more efficiently.
- the coupling of several SISO CLQ controllers by appropriate exchange of information.

Finally, we revisit the six myths, and offer our assessment.

Myth 1. A PID controller is simpler to implement and tune than an LQ controller. The validity of this myth rests largely with the hardware and control software vendors. Certainly it is not difficult to implement a constrained LQ controller if the vendor has programmed the simple active-set table lookup presented in this paper. Regarding tuning, it is not difficult to look up tuning rules for a PID controller. However, it is difficult to find PID tuning parameters that give performance and robustness similar to those of an LQ controller. The LQ controller is not difficult to tune. The effects of its two main tuning parameters (s and R_v) are clear.

Myth 2. A PID controller with model-based tuning is as good as model-based control for simple processes such as SISO, first-order plus time delay. We see no evidence to support this myth. To the contrary, Figure 1 shows the opposite is true. If we restrict the meaning of "simple process" to "first-order process," then this myth has a better chance of holding up.

Myth 3. A well-tuned PID controller is more robust to plant/model mismatch than an LQ controller. Again, we see no evidence to support this myth. Figures 3, 6, and 9 show the opposite is true. We see no superior robustness properties for PID control given any recommended tuning rules.

Myth 3 (Alternate Version). LQ controllers are not very

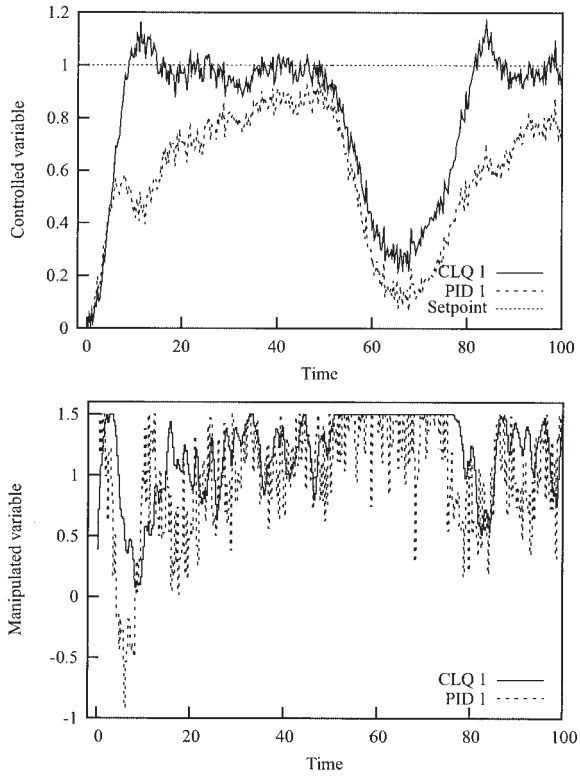

Figure 8. Underdamped system: noisy case.

[2] Octave (http://www.octave.org) is freely distributed under the terms of the GNU General Public License.

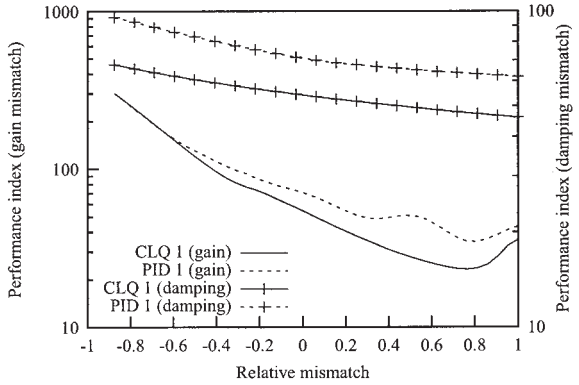

Figure 9. Underdamped system: effect of plant/model mismatch.

robust to plant/model mismatch. As pointed out in previous research,[3] one can construct processes for which the state feedback regulator has good margins but output feedback with the same regulator and a state estimator has arbitrarily poor margins. We have yet to see examples that indicate this issue has industrial significance.

Myth 4. Integrating the tracking error as in PID control is necessary to remove steady-state offset. Applying some anti-windup strategy for this integrator is thus necessary when an input saturates. Integrating the tracking error is not required for offset free control as shown in all of the examples in this paper. Integrating the model error is a sharper idea, which also obviates the need for an anti-windup strategy when the input saturates.

Myth 5. For simple processes (SISO, first-order plus time delay) in the presence of input saturation, a PID controller with a simple anti-windup strategy is as good as model predictive control. The constraint-handling properties of PID are not competitive with MPC. Even in the simple SISO situation, this difference can be noticeable, as shown in Figures 1 and 7.

Myth 6. PID controllers are omnipresent because they work well on most processes. Seeing no evidence that PID controllers work particularly well when compared to an alternative controller, we propose the following explanation rooted more in human behavior than technological advantage. PID controllers are everywhere because vendors programmed them in the DCS when they replaced analog PID controllers.

Acknowledgments

The authors thank Drs. Badgwell, Brambilla, Downs, Mayne, and Ogunnaike for helpful discussion of this paper and also thank John Eaton for his helpful advice regarding the coding of the constraint sets. The authors gratefully acknowledge the financial support of the industrial members of the Texas–Wisconsin Modeling and Control Consortium, National Science Foundation through Grant CTS-0105360 and the University of Pisa.

Literature Cited

1. Chen D, Seborg DE. PID controller design based on direct synthesis and disturbance rejection. Proc of AIChE Annual Meeting, Paper 276i, Reno, NV; 2001.
2. Skogestad S. Probably the best simple PID tuning rules in the world. Proc of AIChE Annual Meeting, Paper 276h, Reno, NV; 2001.
3. Doyle JC. Guaranteed margins for LQG regulators. *IEEE Trans Auto Contr.* 1978;23:756-757.
4. Shinskey FG. *Feedback Controllers for the Process Industries*. New York, NY: McGraw-Hill; 1994.
5. Mukati K, Ogunnaike B. Stability analysis and tuning strategies for a novel next generation regulatory controller. Proc of American Control Conf., Boston, MA; 2004.
6. Soroush M, Muske KR. Analytical model predictive control. In: Allgower F, Zheng A, eds., *Progress in Systems and Control Theory*. Basel, Switzerland: Birkhauser-Verlag; 2000:163-179.
7. Pannocchia G, Laachi N, Rawlings JB. A fast, easily tuned, SISO, model predictive controller. Proc of DYCOPS 7, Boston, MA; 2004.
8. Pannocchia G, Rawlings JB. Disturbance models for offset-free model predictive control. *AIChE J*. 2003;49:426-437.
9. Muske KR, Badgwell TA. Disturbance modeling for offset-free linear model predictive control. *J Process Contr*. 2002;12:617-632.
10. Pannocchia G, Kerrigan EC. Offset-free control of constrained linear discrete-time systems subject to persistent unmeasured disturbances. Proc of 42nd IEEE Conference on Decision and Control, Honolulu, Hawaii; 2003.
11. Pannocchia G. Robust disturbance modeling for model predictive control with application to multivariable. III—Conditioned processes. *J Process Contr*. 2003;13:693-701.
12. Kwakernaak H, Sivan R. *Linear Optimal Control Systems*. New York, NY: Wiley; 1972.
13. Bonné D, Jørgensen JB, Rawlings JB, Jørgensen SB. Robust disturbance modeling for model predictive control with application to multivariable ill-conditioned processes. In preparation, 2005.
14. Blanchini F. Set invariance in control. *Automatica*. 1999;35:1747-1767.
15. Golub GH, Van Loan CF. *Matrix Computations*. Baltimore, MD: The Johns Hopkins Univ. Press; 1996.
16. Bemporad A, Morari M, Dua V, Pistikopoulos EN. The explicit linear quadratic regulator for constrained systems. *Automatica.* 2002;38:3-20.
17. Nocedal J, Wright SJ. *Numerical Optimization*. New York, NY: Springer-Verlag; 1999.
18. Ogunnaike BA, Ray WH. *Process Dynamics, Modeling, and Control*. Oxford, UK: Oxford Univ. Press; 1994.
19. Luyben WL, Luyben ML. *Essentials of Process Control*. New York, NY: McGraw-Hill Int. Editions; 1997.
20. Skogestad S. Simple analytic rules for model reduction and PID controller tuning. *J Process Contr*. 2003;13:291-309.

Manuscript received Dec. 29, 2003, and revision received Aug. 2, 2004.