Conditioning Technique, A General Anti-Windup and Bumpless Transfer Method,

Article *in* Automatica · November 1987

DOI: 10.1016/0005-1098(87)90029-X

3 authors, including:

Raymond Hanus
Université Libre de Bruxelles
98 PUBLICATIONS **1,846** CITATIONS

SEE PROFILE

Michel Kinnaert
Université Libre de Bruxelles
203 PUBLICATIONS **6,726** CITATIONS

SEE PROFILE

Automatica, Vol. 23, No. 6, pp. 729–739, 1987
Printed in Great Britain.

0005–1098/87 $3.00 + 0.00
Pergamon Journals Ltd.

Conditioning Technique, a General Anti-windup and Bumpless Transfer Method*

R. HANUS,† M. KINNAERT† and J.-L. HENROTTE†

A general way to take into account, by an appropriate design of the controller, any discrepancy which can occur between the actual inputs of a process and the desired outputs of its controller yields the so-called conditioned control algorithms and reduces deterioration in control performance.

Key Words—Nonlinear systems; saturation; multivariable systems; cascade control; anti-windup (not in the standard list); bumpless transfer (not in the standard list); initialization (not in the standard list).

Abstract—This paper gives a general way to take into account, by an appropriate design of the controller, any discrepancy which can occur between the actual inputs of a process and the desired outputs of its controller. This yields to the so-called conditioned control algorithms.

The conditioning technique is described for multiple-input–multiple-output nonlinear controllers. Application to complex control structures is explained.

The notion of a self-conditioned controller is defined in order to simplify the implementation of the conditioning technique. Some considerations about the stability of conditioned systems are given. The automatic initialization of any control algorithm is given as an application of the method. It shows that bumpless transfer can be achieved.

1. INTRODUCTION

IN INDUSTRIAL CONDITIONS, it often happens that the actual input of a controlled process is temporarily different from the controller output.

Two main classes of such differences are distinguishable: substitutions and limitations.

Switching from manual to automatic control is an example of substitution. More generally, switching between two controllers in parallel yields a control substitution. Use of such a structure can be found in Hanus (1979) and Debelle (1979).

The mismatch between process input and controller output can also be due to limitation. This is the case when some saturation on the manipulated variable exists. Examples of such limitations are physical limits of the actuators (a fully open or closed valve, for instance), or security requirements on the controller outputs.

When a substitution is performed, bumpless transfer is achieved by a proper positioning of the controller states. In the case of saturation, anti-windup systems attempt to correct the controller states. Actually, the two phenomena can be treated in a similar manner, by using the conditioning principle defined by Hanus (1979).

The mismatch between the control variable u and the manipulated variable u^r (Fig. 1) can deeply deteriorate the performances of the closed-loop. An illustration of such a deterioration can be found in the examples given in Section 7.

The control deterioration can be explained by an inadequacy of the state of the controller, as explained in Hanus (1980a, b), and Åström and Wittenmark (1984). Some solutions to the windup problems have been given in the literature. The first one we found is Fertrik and Ross (1967); followed by Kramer and Jenkins (1971), Vandenbussche (1975) and Khandheria and Luyben (1976). Unfortunately, these different solutions are always designed for a specific application. Their use is restricted to one class of nonlinearities, one class of controllers, and sometimes to one class of process models. Hanus (1979) gives a solution to bumpless transfer and windup problems which is independent of the nature of the controller, of the controlled system and of the nonlinearity acting

* Received 21 December 1984; revised 5 May 1985; revised 31 October 1985; revised 26 June 1986; revised 6 May 1987; revised 11 June 1987. The original version of this paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associate Editor Daniel Tabak under the direction of Editor H. Austin Spang III.

† Laboratoire d'Automatique, Faculté des Sciences appliquées-C.P. 165, Université Libre de Bruxelles, 50 Avenue F. D. Roosevelt, B-1050 Brussels, Belgium.

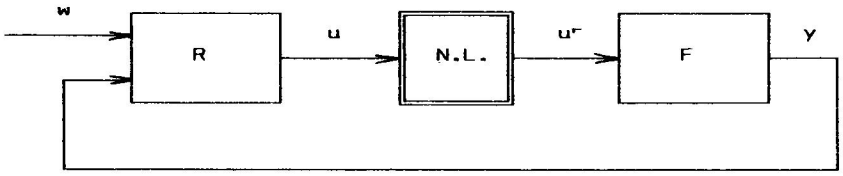

FIG. 1. Classical unconditioned control loop. w: reference variables; u: desired control variables; u^r: actual control variables; y: output variables; R: controller; $N.L.$: nonlinearity; F: process.

on the control variable. This solution used an operator formalism. It was extended to nonlinear systems by using the state space description in Hanus (1980a, b). Åström and Wittenmark (1984) define a general state space model and use an explicit observer in order to take into account some saturations on the control variables (pp. 369–373). With this method, the dynamics of the closed loop system with a constraint on the controller output is usually different from the dynamics in the absence of constraint. Both Åström and Wittenmark (1984) and Hanus (1979, 1980a, b) introduce a measure or an estimation of the actual control signal applied to the controlled system, so that the states of the controller always assume the correct values. Hanus modifies the states further through the use of a "realizable" reference signal. Thanks to this technique, when the difference between the desired control variable and the actual control variable disappears, the closed-loop system reaches steady state with the *same* dynamics as the unconstrained closed-loop system.

In the present paper, the conditioning technique described by Hanus (1979, 1980a, b) is extended to multiple-input–multiple-output (MIMO) nonlinear systems. Moreover, conditioning of complex structures like cascade or parallel block controllers is studied. We also introduce the concept of "self-conditionable" controllers, in order to simplify the implementation of the algorithms. Some considerations about the stability of conditioned systems are given.

The paper is organized as follows. Section 2 presents Hanus' conditioning technique for MIMO systems. Section 3 describes the "self conditionable" structure. Conditioning of complex control structures is discussed in Section 4. Section 5 gives some stability considerations. Section 6 explains the use of conditioning as an initializing technique. Application of the conditioning technique is illustrated in Section 7. A PI controller is used as a simple example. Finally, some concluding remarks are given in Section 8.

2. CONDITIONING TECHNIQUE FOR MIMO SYSTEMS

2.1. *State space description*

Each mismatch between the desired control variables u and the actual ones u^r (see Fig. 1) yields an inadequacy of the controller state variables v. The restoration of the adequacy of these state variables is achieved by the conditioning technique, so called because it "conditions" the controller to come back in to the normal mode as soon as it can. Controllers provided with such tools are said to be "conditioned". The technique is based on a measure or an estimation of the actual control variables u^r. It can be applied to any kind of controller (linear or not, discrete or continuous, time varying or not, single-input–single-output or multiple-input–multiple-output). The cause of the mismatch between the control variables u and u^r need not be known (any kind of substitution or limitation).

We consider a general discrete-time state variable representation for the controller (the method is easily convertible to continuous-time controllers):

$$v(t+1) = f\{v(t), w(t), y(t), t\} \quad (1a)$$

$$u(t) = g\{v(t), w(t), y(t), t\} \quad (1b)$$

where

w = reference variable $\in R^l$

y = output variables $\in R^l$

u = controlling variables $\in R^m$

v = controller state variables $\in R^n$.

When the actual control variables u^r are different from the desired variables u, inadequate values of the state variables v can destroy control performance. In order to restore the state adequacy, auxiliary inputs w^r, called realizable references are used. These realizable references w^r are such that if w^r had been applied to the controller instead of w, the control variables u would have been equal to u^r. We assume that such variables can be calculated. The states, v^r, obtained with these new inputs w^r, are necessarily adequate, because w^r cancels

the effects of the nonlinearities on the control variables. Hence,

$$v^r(t+1) = f\{v^r(t), w^r(t), y(t), t\} \quad (2a)$$

$$u^r(t) = g\{v^r(t), w^r(t), y(t), t\}. \quad (2b)$$

With these new reference variables w^r, the adequacy of the state variables v^r is restored, but the actual reference variables are lost. In order to restore these actual reference variables also, a temporary assumption on the present realizability ($u = u^r$) of the control is made, that is:

$$u(t) = g\{v^r(t), w(t), y(t), t\}. \quad (2c)$$

Equations (2b) and (2c) define implicitly the realizable variables w^r. In some cases, an explicit definition of the realizable variables w^r exists. This is the case when (2b) is linear in $w(t)$, that is

$$u(t) = c\{v(t), y(t), t\} + D\{v(t), y(t), t\}w(t) \quad (3)$$

where D is a functional matrix of appropriate dimensions. Equations (2b) and (2c) become:

$$u^r(t) = c\{v^r(t), y(t), t\} + D\{v^r(t), y(t), t\}w^r(t) \quad (4a)$$

$$u(t) = c\{v^r(t), y(t), t\} + D\{v^r(t), y(t), t\}w(t) \quad (4b)$$

and by the difference,

$$u^r(t) - u(t) = D\{v^r(t), y(t), t\}\{w^r(t) - w(t)\}. \quad (5)$$

The determination of the realizable references $w^r(t)$ implies the resolution of system (5), which is linear in $\{w^r(t) - w(t)\}$. Normally the rank of the matrix D should be full. The opposite would lead to at least one dead time in the control action, and that should be avoided.

Except for particular values of the differences $\{u^r(t) - u(t)\}$, the inversion of system (5) depends on the respective dimensions of $w(t)$ and $u(t)$: l and m. When $l = m$, there is one and only one solution given by:

$$w^r(t) = w(t) + D^{-1}\{v^r(t), y(t), t\}\{u^r(t) - u(t)\}. \quad (6)$$

The realizable references $w^r(t)$ have to be used instead of the actual references $w(t)$, when updating equation (2a), in order to obtain the adequate states $v^r(t)$. The corresponding conditioned controller follows immediately:

$$w^r(t-1) = w(t-1) + D^{-1}\{v^r(t-1), y(t-1), t-1\} \times \{u^r(t-1) - u(t-1)\} \quad (7a)$$

$$v^r(t) = f\{v^r(t-1), w^r(t-1), y(t-1), t-1\} \quad (7b)$$

$$u(t) = c\{v^r(t), y(t), t\} + D\{v^r(t), y(t), t\}w(t). \quad (7c)$$

At time $(t-1)$, the desired variables $u(t-1)$ are known, the actual variables $u^r(t-1)$ are measured or estimated, and the prior values of the reference variables $w^r(t-1)$ can be calculated by equation (7a). These prior values of $w^r(t-1)$ are used to update the adequate states $v^r(t)$, by equation (7b). The adequate states $v^r(t)$, the present measures of the outputs $y(t)$, and the present *actual* values of the references $w(t)$ are used to compute the present desired control variables $u(t)$, by equation (7c). The conditioned algorithm (7) has the same level of complexity as the unconditioned one, increased by equation (7a).

2.2. *Operator description*

The conditioning theory is easily translated to a q operator description when the controllers are of course linear and discrete, and to a z transfer function description when the controllers are moreover time invariant.

For linear controllers, the functions f and g are linear and the system (1) becomes:

$$v(t+1) = A(t)v(t) + B(t)w(t) - E(t)y(t) \quad (8a)$$

$$u(t) = C(t)v(t) + D(t)w(t) - F(t)y(t) \quad (8b)$$

where the matrices $A(t)$, $B(t)$, $C(t)$, $D(t)$, $E(t)$ and $F(t)$ have the appropriate dimensions.

In the linear case, equations (7) become:

$$w^r(t-1) = w(t-1) + D^{-1}(t-1) \times \{u^r(t-1) - u(t-1)\} \quad (9a)$$

$$v^r(t) = A(t-1)v^r(t-1) + B(t-1)w^r(t-1) - E(t-1)y(t-1) \quad (9b)$$

$$u(t) = C(t)v^r(t) + D(t)w(t) - F(t)y(t). \quad (9c)$$

The corresponding unconditioned discrete-time controller fulfills the equation:

$$u(t) = T(q, t)w(t) - S(q, t)y(t) \quad (10)$$

where T and S are the square matrices given by (11a) and (11b) and q is the forward shift operator: $q \cdot x(t) = x(t+1)$:

$$T(q, t) = C(t)\{Iq - A(t)\}^{-1}B(t) + D(t) \quad (11a)$$

$$S(p, t) = C(t)\{Iq - A(t)\}^{-1}E(t) + F(t). \quad (11b)$$

It is easy to see that the matrix $D(t)$ is given by the asymptotical values of $T(q, t)$:

$$D(t) = \lim_{q \to \infty} T(q, t). \quad (12)$$

Relation (12) can be used to evaluate the matrix $D(t)$ when the controller is described by an operator of the form $T(q, t)$. The corresponding conditioned controller fulfills the equation:

$$u(t) = C(t)\{Iq - A(t)\}^{-1}B(t)w^r(t) + D(t)w(t) - S(q, t)y(t) \quad (13)$$

or using (9a), and (11a) in (13)

$$w^r(t-1) = w(t-1) + D^{-1}(t-1) \times \{u^r(t-1) - u(t-1)\} \quad (14a)$$

$$u(t) = \{T(q, t) - D(t)\}w^r(t) - S(q, t)y(t) + D(t)w(t). \quad (14b)$$

It should be pointed out that the operator $\{T(q, t) - D(t)\}$, acting on $w^r(t)$, contains at least one dead time

$$\lim_{q\to\infty} \{T(q, t) - D(t)\} = 0. \quad (15)$$

This condition assures the realizability of the conditioned algorithm. Thus, it is not necessary to know $w^r(t)$ for calculating $u(t)$. The conditioned algorithm is described by equations (14).

These relations are of the same level of complexity as the unconditioned algorithm described by (10), increased by (14a).

3. SELF-CONDITIONABLE STRUCTURE

A general discrete-time state variable representation for a controller (1) leads to the following equation:

$$v(t+1) = h\{v(t), u(t), y(t), t\} \quad (16)$$

where h is a vectorial function of u, whereas f was a vectorial function of w. To obtain (16) it is sufficient to replace, in (1a), $w(t)$ given by the inversion of (3):

$$w(t) = D^{-1}\{v(t), y(t), t\} \times \{u(t) - c\{v(t), y(t), t\}\}. \quad (17)$$

The new representation is now given by equations (16) and (3):

$$v(t) = h\{v(t-1), u(t-1), y(t-1), t-1\} \quad (18a)$$

$$u(t) = c\{v(t), y(t), t\} + D\{v(t), y(t), t\}w(t). \quad (18b)$$

This form is called self-conditionable, because the corresponding conditioned controller is simply obtained by replacing u by u^r in (18a)

$$v^r(t) = h\{v^r(t-1), u^r(t-1), y(t-1), t-1\} \quad (19a)$$

$$u(t) = c\{v^r(t), y(t), t\} + D\{v^r(t), y(t), t\}w(t). \quad (19b)$$

This last form is said to be self-conditioned. It is not necessary to evaluate the realizable references w^r to obtain it.

In the linear case, the function h is such that

$$v^r(t+1) = \{A(t) - B(t)D^{-1}(t)C(t)\}v^r(t) + B(t)D^{-1}(t)u^r(t) + \{B(t)D^{-1}(t)F(t) - E(t)\}y(t) \quad (20a)$$

$$u(t) = C(t)v^r(t) + D(t)w(t) - F(t)y(t). \quad (20b)$$

In the case of operator description, it is also possible to have a self-conditionable form. To this end, it is necessary to be able to evaluate the present values of $u(t)$ from the present values of $w(t)$ and $y(t)$ and the prior values of $y(t)$ and $u(t)$ exclusively [no dependency of the prior values of $w(t)$ can exist].

After some transformations, equation (10) can be rewritten:

$$u(t) = D(t)\{w(t) - T^{-1}(q, t)S(q, t)y(t) + \{D^{-1}(t) - T^{-1}(q, t)\}u(t)\} \quad (21)$$

with

$$\lim_{q\to\infty} \{D^{-1}(t) - T^{-1}(q, t)\} = 0. \quad (22)$$

By inverse transformations, it is easy to prove that (21) is equivalent to (10).

The form (21) is said self-conditionable, because the corresponding self-conditioned form is obtained simply by replacing $u(t)$ by $u^r(t)$ in the right-hand side:

$$u(t) = D(t)\{w(t) - T^{-1}(q, t)S(q, t)y(t) + \{D^{-1}(t) - T^{-1}(q, t)\}u^r(t)\}. \quad (23)$$

Note that the present values of $u(t)$ only depend

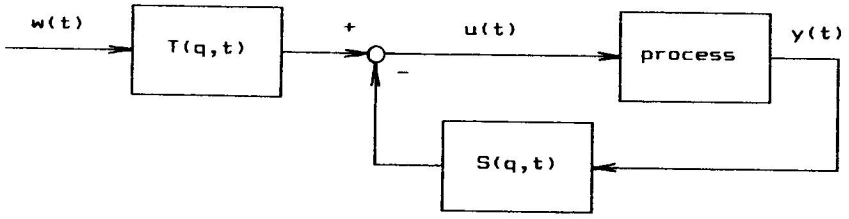

FIG. 2. Non-conditioned structure. w: reference variables; u: control variables; y: output variables; T: feedforward action; S: feedback action.

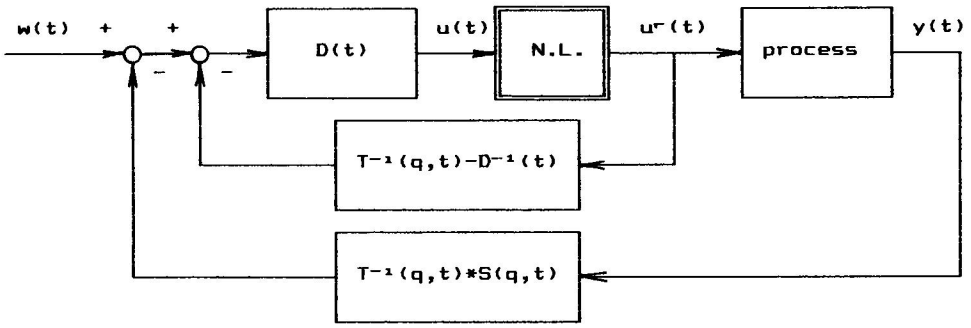

FIG. 3. Self conditioned structure. w: reference variables; u: desired control variables; u^r: actual control variables; y: output variables; T: feedforward action; S: feedback action; D: direct action.

on prior values of $u^r(t)$ due to (22) [see remarks related to (15)].

The non-conditioned structure is given in Fig. 2 and the corresponding self-conditioned one is shown in Fig. 3.

The self-conditioned structure has only a time varying gain matrix in the direct control path [from $w(t)$ to $u(t)$]. The controller states are updated in the feedback control path only.

4. CONDITIONING OF COMPLEX CONTROL STRUCTURES

After this generalization of conditioning techniques and the introduction of self-conditioned structures, we are able to extend their application to more complex control structures.

4.1. *Cascade structure*

The case of multi-loop systems is considered first. A representation of two control loops is given in Fig. 4. The variables belonging to the inner loop are indexed 1, the ones belonging to the outer loop, 2.

The two controllers are described by

$$u_i(t) = T_i(q, t)w_i(t) - S_i(q, t)y_i(t) \qquad i = 1, 2. \quad (24)$$

The cascade is assumed by

$$w_1(t) = u_2^r(t). \quad (25)$$

Conditioning of the inner loop is obtained as before, with

$$w_1^r(t) = w_1(t) + D_1^{-1}(t)\{u_1^r(t) - u_1(t)\} \quad (26)$$

and

$$D_1(t) = \lim_{q\to\infty} T_1(q, t). \quad (27)$$

The actual manipulated variables $u_1^r(t)$ and the realizable references $w_1^r(t)$ are used to update the state variables of $T_1(q, t)$ and $S_1(q, t)$. However, for the outer loop, nonlinearities on both $u_1(t)$ and $u_2(t)$ have to be considered.

According to the general definition, the realizable references $w_2^r(t)$ are variables such that, if w_2^r were applied as inputs of the control loop, the manipulated variables $u_1(t)$ would be equal to $u_1^r(t)$.

In order to compute $w_2^r(t)$, the conditioned state space equations corresponding to both controllers are written:

$$v_i^r(t) = A_i(t-1)v_i^r(t-1) + B_i(t-1) \times w_i^r(t-1) - E_i(t-1)y_i(t-1) \quad (28a)$$

$$u_i(t) = C_i(t)v_i^r(r) + D_i(t)w_i(t) - F_i(t)y_i(t) \quad (28b)$$

$$u_i^r(t) = N.L._i\{u_i(t)\} \quad (28c)$$

where

$$T_i(q, t) = C_i(t)(qI - A_i(t))^{-1}B_i(t) + D_i(t) \quad (29a)$$

$$S_i(q, t) = C_i(t)(qI - A_i(t))^{-1}E_i(t) + F_i(t) \qquad i = 1, 2 \quad (29b)$$

and $w_1^r(t)$ are the realizable references of the

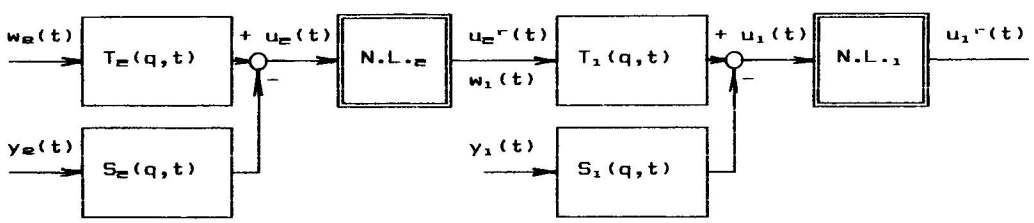

FIG. 4. Multi-loop system with several nonlinearities. w_i: reference variables; u_i: desired control variables; u_i^r: actual control variables; y_i: output variables; $i = 1$: inner loop; $i = 2$: outer loop; T_i: feedforward actions; S_i: feedback actions; $N.L._i$: nonlinearities.

inner loop. They are calculated by (26). The cascade structure implies that these variables are also the realizable controls of the outer loop, which can be different from the limited control variables $u_2^r(t)$.

Taking the above remarks into account, we deduce from (28b), with $i = 1, 2$

$$u_1^r(t) = C_1(t)v_1^r(t) + D_1(t)w_1^r(t) - F_1(t)y_1(t) \tag{30a}$$

$$w_1^r(t) = C_2(t)v_2^r(t) + D_2(t)w_2^r(t) - F_2(t)y_2(t). \tag{30b}$$

By rearrangement of the difference between (28b) with $i = 2$ and (30b):

$$w_2^r(t) = w_2(t) + D_2^{-1}(t)\{w_1^r(t) - u_2(t)\}. \tag{31}$$

Replacing $w_1^r(t)$ by its value given by (26), and using (25),

$$w_2^r(t) = w_2(t) + D_2^{-1}(t)\{\{u_2^r(t) - u_2(t)\} + D_1^{-1}(t)\{u_1^r(t) - u_1(t)\}\}. \tag{32}$$

It is readily seen that extension to n loops requires the following realizable references

$$w_n^r(t) = w_n(t) + \Delta w_n(t) \tag{33a}$$

with

$$\Delta w_n(t) = D_n^{-1}(t)\{u_n^r(t) - u_n(t) + \Delta w_{n-1}(t)\} \tag{33b}$$

and

$$\Delta w_0(t) \triangleq 0. \tag{33c}$$

Or, also,

$$w_n^r(t) = w_n(t) + \sum_{i=1}^{n} K_i(t)\{u_i^r(t) - u_i(t)\} \tag{34a}$$

with

$$K_i(t) = \prod_{j=i}^{n} D_j^{-1}(t) = K_{i+1}(t)D_i^{-1}(t) \tag{34b}$$

and

$$K_{n+1}(t) \triangleq I. \tag{34c}$$

Two remarks have to be made:

(1) each mismatch between $u_i(t)$ and $u_i^r(t)$ has an influence only on the elements located up-stream in the control path;
(2) each mismatch influences the realizable reference signals through a linear combination (34).

4.2. *Parallel structure*

Some other structures can be studied. The parallel structure is given as an example. For the sake of simplicity and because we know how to combine the influence of successive discrepancies between u and u^r, we can study the influence of one of them, e.g. $u_i \neq u_i^r$.

The parallel structure is given in Fig. 5. In this particular case, equation (5) yields the following linear equations:

$$u_i(t) - u_i^r(t) = \{D_{1i}(t) + D_{2i}(t)\} \times \{w_i(t) - w_i^r(t)\} \tag{35}$$

with

$$D_{ji}(t) = \lim_{q \to \infty} T_{ji}(q, t); \quad j = 1, 2. \tag{36}$$

The realizable references are obtained by inversion:

$$w_i^r(t) = w_i(t) + \{D_{1i}(t) + D_{2i}(t)\}^{-1} \times \{u_i^r(t) - u_i(t)\}. \tag{37}$$

The generalization to any number of controllers in parallel is immediate.

5. STABILITY CONSIDERATIONS

Previous work on the stability of single loop control systems with control limitations and antireset-windup circuit is described in Glattefelder and Schaufelberger (1983). Here sufficient conditions for stability of a conditioned loop are described in two particular cases. The

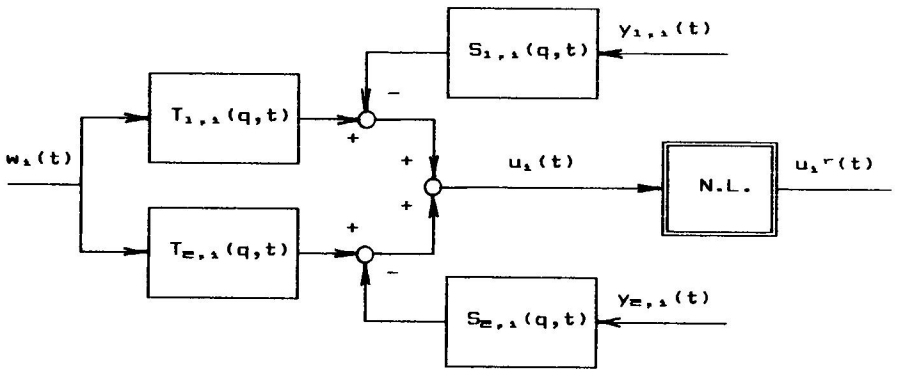

FIG. 5. Structure with blocks in parallel. w_i: reference variables; u_i: desired control variables; u_i^r: actual control variables; y_{ji}: output variables; $j = 1$: upper loop; $j = 2$: down loop; T_{ji}: feedforward actions; S_{ji}: feedback actions; $N.L._i$: nonlinearities.

discrepancy between u and u^r is due to a saturation limitation or a substitution.

Theorem. A constrained control loop operating with a conditioned controller described by (7) or (19) is asymptotically stable for a steady state defined by the asymptotical values of the references $w(t)$ if the following conditions are met:

(1) the unconstrained closed loop is asymptotically stable for the steady state defined by the asymptotical values of the realizable references $w^r(t)$;
(2) (a) for the substitution:
—the substituting loop is asymptotically stable in the steady state defined by the substituting references;
(b) in the particular case of saturation limitations:
—the controlled system is asymptotically stable in the steady state defined by the asymptotical values of the realized control variables $u^r(t)$;
(3) the system described by (19a) is asymptotically stable in the steady state defined by the asymptotical values of the variables $u^r(t)$ and $y(t)$;
(4) the functions described in (19b) tend to asymptotical values for the asymptotical values of the variables $v^r(t)$ and $y(t)$.

Proof. The conditioning principle consists of replacing the actual reference variables by the realizable reference variables w^r so that the prior values of the actual control variables u^r and the prior values of the desired control variables u are equal. Hence the conditioned control loop corresponding to Fig. 1 is equivalent to the unconditioned control loop of Fig. 6, at least with regard to the former time. Condition 1 ensures asymptotical stability of this last system around the asymptotical values of the realizable reference $w^r(t)$. Thus our problem reduces to the proof of the existence of a steady state for w^r. From the definition of $w^r(t)$, (6), we deduce that $w^r(t)$ reach final asymptotical values when $D^{-1}\{v^r(t), y(t), t\}$, $u^r(t)$ and $u(t)$ do so. We first assume that $u^r(t)$ and $y(t)$ reach finite asymptotical values, and we look for conditions ensuring that $u(t)$ reach finite asymptotical values. Since the different forms of the conditioned algorithm are equivalent as far as their stability properties are concerned, we shall use the self-conditioned form only [equation (19)]. Now conditions (3) and (4) ensure that $u(t)$ reach finite asymptotical values. Finally, condition (2) ensures the same property for $u^r(t)$ and $y(t)$. Indeed, when a saturation limitation occurs, $u^r(t)$ obviously have constant values. Hence, by condition (2b), $y(t)$ will reach a finite steady state value. In the case of a substitution, condition (2a) yields the asymptotical convergence of $u^r(t)$ and $y(t)$ towards finite steady state values.

When the controller and/or the controlled system are linear, the asymptotical stabilities can be verified by knowledge of the different system poles.

In this manner, we have established the general asymptotical stability of a conditioned loop when some constraint acts on the control variables. It is evident that as soon as the constraint disappears, the only remaining condition is the first one.

It is also possible to define a BIBO (bounded-input–bounded-output) stability. In order to obtain the conditions of BIBO stability, it is sufficient to replace everywhere above the notions of asymptotical values and constant limits by those of bounded values.

6. CONDITIONING AS INITIALIZATION TECHNIQUE

Each algorithm running in real-time often needs two working modes: aperiodical and periodical. The aperiodical mode is required for initializing the values of the state variables. The initialization of a control algorithm should depend on the prior behaviour of the process. Thanks to the conditioning technique an appropriate initialization can be made. During the initialization of the state variables, the algorithm runs periodically but it is prevented from acting on the process ($u^r \neq u$). However, the measurements are made normally and the realizable inputs w^r, the adequate state variables v^r and the desired outputs u are computed. As

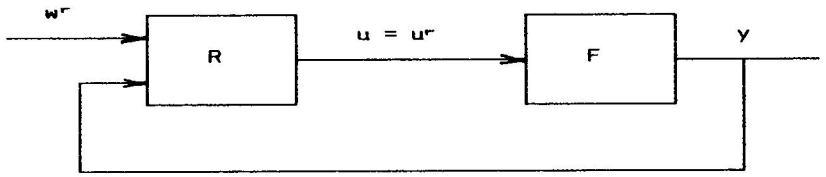

FIG. 6. Equivalent non-constrained control loop. w^r: realizable reference variables; R: controller; u^r: actual control variables; y: output variables; R: controller; F: process.

soon as all the state variables v^r have converged to their proper values, it is possible to switch automatically to work phase ($u^r = u$), without undesirable bump on the outputs u. We talk about a transfer without undesirable bump and not about bumpless transfer. Indeed, a bumpless transfer is only obtained if the desired inputs $w(t)$ are set to the value of the realizable inputs $w^r(t)$. This equality is not always required, but can easily be realized. The convergence of the state variables v^r depends on the initial values of v^r and on the dynamics of the conditioning. When the controller is linear and discrete-time, it is possible to reduce the time of convergence to its minimum value: the dimension of the controller state vector $v(t)$. Indeed, from the operational description (10), it is possible to decompose $T(q, t)$, and $S(q, t)$ in the forms

$$T(q, t) = D(t) + \frac{T'(q^{-1}, t)}{1 + R'(q^{-1}, t)} \quad (38a)$$

$$S(q, t) = F(t) + \frac{S'(q^{-1}, t)}{1 + R'(q^{-1}, t)} \quad (38b)$$

where $T'(q^{-1}, t)$, $S'(q^{-1}, t)$ are polynomial matrices in q^{-1} with no zero degree term, and $R'(q^{-1}, t)$ is a scalar polynomial with the same properties.

The matrices $T'(q^{-1}, t)$ and $S'(q^{-1}, t)$ are given by

$$T'(q^{-1}, t) = q^{-n} C(t) \text{ Adj } \{Iq - A(t)\} B(t) \quad (39a)$$

$$S'(q^{-1}, t) = q^{-n} C(t) \text{ Adj } \{Iq - A(t)\} E(t) \quad (39b)$$

and the scalar $R'(q^{-1}, t)$ is given by

$$R'(q^{-1}, t) = q^{-n} \det \{Iq - A(t)\} - 1. \quad (39c)$$

The unconditioned algorithm as defined by (10) can be rewritten using (38):

$$u(t) = \{D(t)(1 + R'(q^{-1}, t)) + T'(q^{-1}, t)\} w(t) - \{F(t)(1 + R'(q^{-1}, t)) + S'(q^{-1}, t)\} y(t) - R'(q^{-1}, t) u(t). \quad (40)$$

The initializing of the algorithm can be done by putting only $u^r(t)$ in place of $u(t)$ in (40)

$$u(t) = \{D(t)(1 + R'(q^{-1}, t)) + T'(q^{-1}, t)\} w(t) - \{F(t)(1 + R'(q^{-1}, t)) + S'(q^{-1}, t)\} y(t) - R'(q^{-1}, t) u^r(t). \quad (41)$$

This form is said to be conditioned in its poles.

Only the states described by the zeros of the denominator $\{1 + R'(q^{-1}, t)\}$ are conditioned. The zeros of the numerators

$$\{D(t)(1 + R'(q^{-1}, t)) + T'(q^{-1}, t)\}$$

and

$$\{F(t)(1 + R'(q^{-1}, t)) + S'(q^{-1}, t)\}$$

are not conditioned. This artifice reduces the initialization time to its minimum: the time necessary to know the part of the state variables attached to the n poles of the controller. By a nonminimal representation, the splitting of the state variables related to the poles and the zeros of the controller can be more easily achieved. Of course, after this time, the controller can be put into a completely conditioned form; and may be authorized to act directly on the process.

7. A SIMPLE EXAMPLE: THE PI CONTROLLER

A state space model of a direct PI controller can be:

$$v(t + 1) = v(t) + K_I\{w(t) - y(t)\} \quad (42a)$$

$$u(t) = v(t) + K_P\{w(t) - y(t)\} \quad (42b)$$

where K_I is the integral gain and K_P the proportional gain; v the integral term and $(w - y)$ the control error. A realization of the PI controller is given in Fig. 7. Equations (42) are a particular case of equations (8). In the case of the PI controller the general equations of the conditioned controller become, according to equations (9),

$$w^r(t - 1) = w(t - 1) + \frac{u^r(t - 1) - u(t - 1)}{K_P} \quad (43a)$$

$$v^r(t) = v^r(t - 1) + K_I\{w^r(t - 1) - y(t - 1)\} \quad (43b)$$

$$u^r(t) = v^r(t) + K_P\{w(t) - y(t)\}. \quad (43c)$$

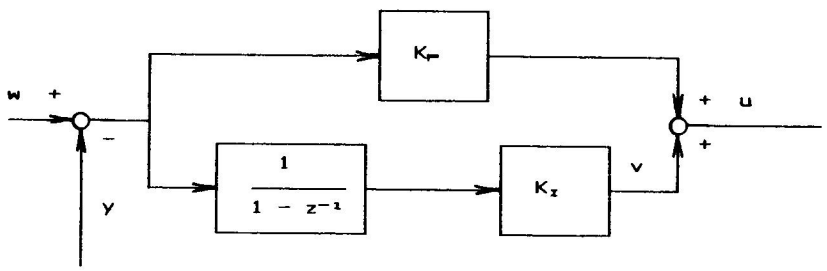

FIG. 7. Unconditioned PI controller. w: reference variable; y: output variable; ε: control error; v: integral term; u: control variable; K_I: integral gain; K_P: proportional gain.

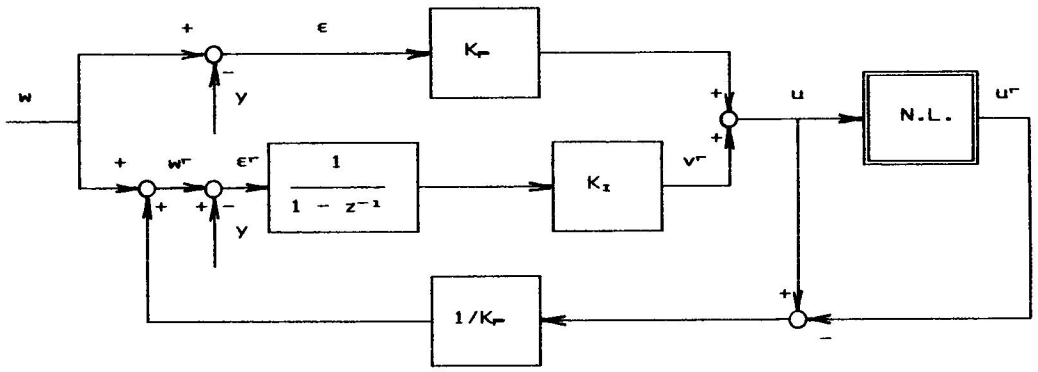

FIG. 8. Conditioned PI controller. w: desired reference variable; y: output variable; ε: control error; w^r: realizable reference variable; ε^r: corrected error; v^r: integral term; u: desired control variable; u^r: actual control variable; K_I: integral gain; K_P: proportional gain; $N.L.$: nonlinearity.

A realization of the conditioned PI is given in Fig. 8. The operator representation of the PI controller is deduced from equations (9), (10) and (11). This yields:

$$u = \left(K_P + \frac{K_I}{z-1}\right)(w - y). \tag{44}$$

Equation (12) gives

$$D = \lim_{z\to\infty}\left(K_P + \frac{K_I}{z-1}\right) = K_P \tag{45}$$

and

$$w^r = w + \frac{u^r - u}{K_P} \tag{46a}$$

$$u = K_P(w - y) + \frac{K_I}{z-1}(w^r - y). \tag{46b}$$

As in the general case, it is possible to express the PI controller in a self-conditioned form. General equations (17), (16) and (19) become, respectively:

$$w(t) = y(t) + \frac{u(t) - v(t)}{K_P} \tag{47}$$

$$v(t) = v(t-1) + \frac{K_I}{K_P}\{u(t-1) - v(t-1)\} \tag{48}$$

$$v^r(t) = v^r(t-1) + \frac{K_I}{K_P}\{u^r(t-1) - v^r(t-1)\} \tag{49a}$$

$$u(t) = v^r(t) + K_P\{w(t) - y(t)\}. \tag{49b}$$

Equation (48) defines the self-conditionable form of the PI, and equations (49) its self-conditioned form. A construction of the self-conditioned PI is shown in Fig. 9. Figure 10 gives the step responses of the closed-loop systems obtained by controlling a first order system with different forms of PI controllers. Curve ① represents the step response of the control loop without constraint. Curve ② is the step response with a conditioned PI constrained by a rate limitation. Curves ③ and ④ correspond respectively to the step responses obtained with the (unconditioned) position and the (unconditioned) velocity algorithms, in the presence of the same rate limitation. The distinction between the last two algorithms is clearly explained in Åström and Wittenmark (1984, p. 181). The response obtained with the conditioned PI is an exact translation of the response obtained without constraint, as soon as it is possible. Notice that the behaviour of a closed-loop system controlled with a conditioned

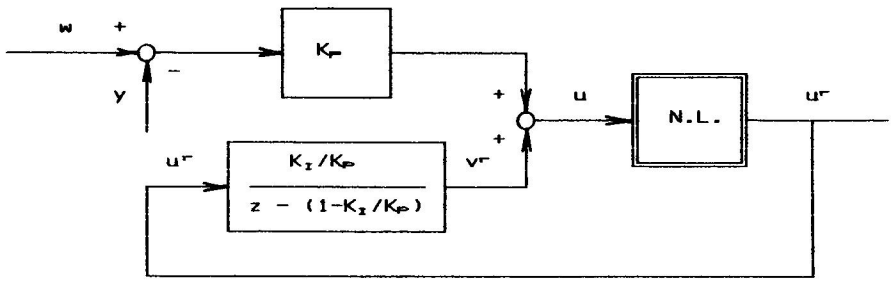

FIG. 9. Self-conditioned PI controller. w: reference variable; y: output variable; ε: control error; v^r: integral term; u: desired control variable; u^r: actual control variable; K_I: integral gain; K_P: proportional gain; $N.L.$: nonlinearity.

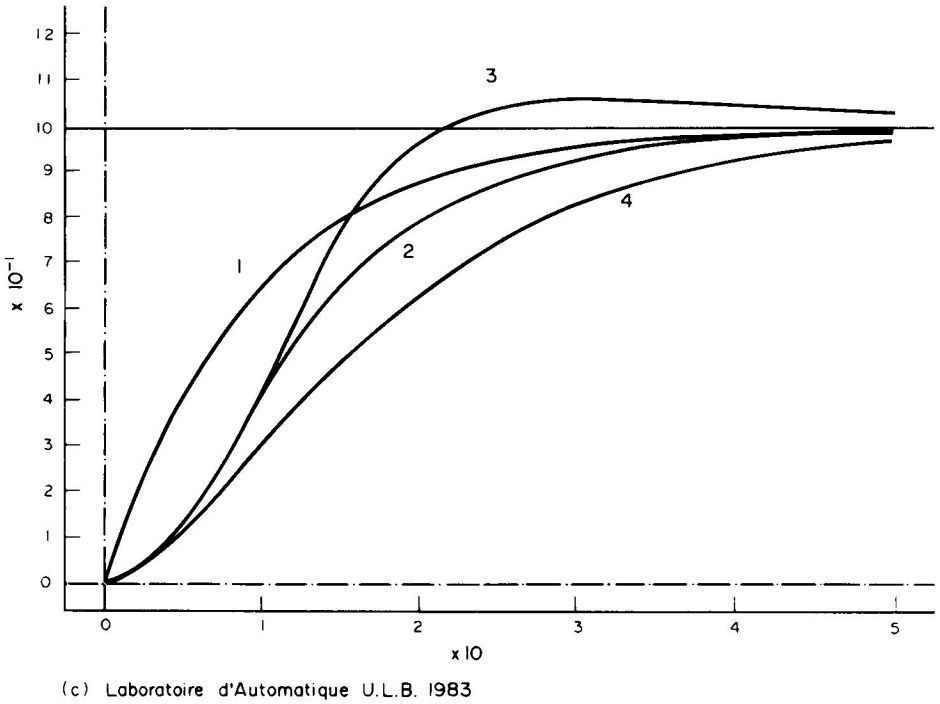

FIG. 10. Step-responses of different PI controllers. Transfer function of the process: $1/1 + 9.49s$. Transfer function of the controller: $1 - 0.9z^{-1}/1 - z^{-1}$. Rate limitation on the control variable: $\Delta u_{max} = 0.1$. ①: unconstrained; ②: conditioned; ③: position algorithm; ④: velocity algorithm.

PI algorithm is independent of the form of the algorithm (position, velocity algorithm, or any other form). The remark is valid for any linear or nonlinear reference signal. The (unconditioned) position algorithm is often destabilizing. The (unconditioned) velocity algorithm often lengthens the response time. However, this algorithm is recommended by Åström and Wittenmark (1984) (pp. 183 and 184). It is usually admitted that, although the velocity

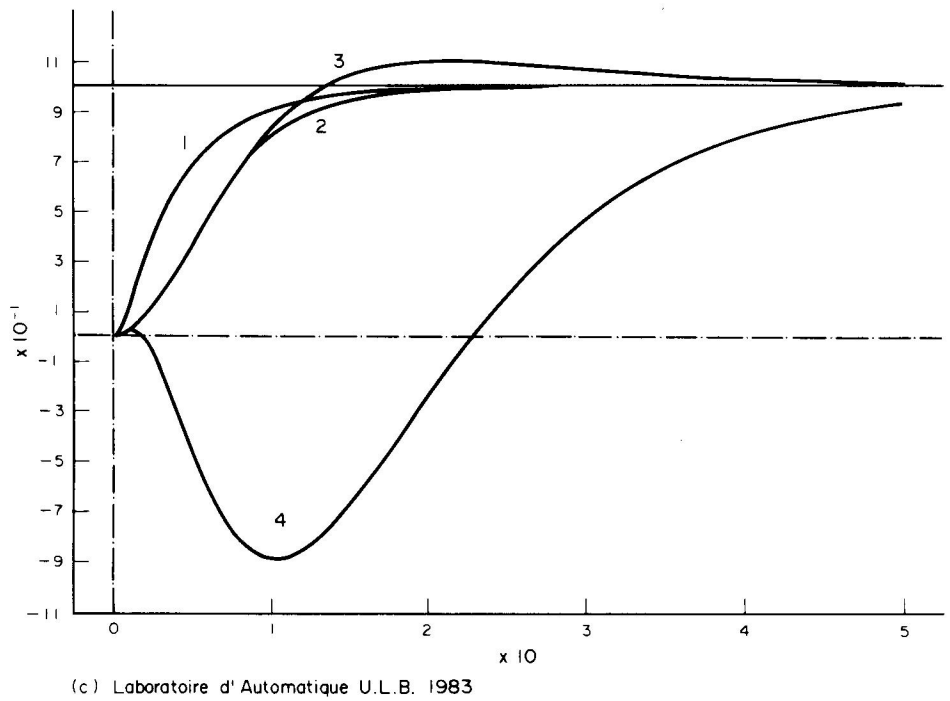

FIG. 11. Step responses of different PID controllers. Transfer function of the process: $1 + 0.0945s/(1 + 4.48s)(1 + 9.49s)$. Transfer function of the controller: $10(1 - 0.9z^{-1})(1 - 0.8z^{-1})/(1 - z^{-1})$. Upper limit on the control variable: $u_{max} = 2$. ① = unconstrained; ② = conditioned; ③ = position algorithm; ④ = velocity algorithm.

algorithm is not perfect, it is however rather safe. A counter-example is given in Fig. 11, where a PID is used as controller.

8. CONCLUSIONS

The conditioning technique was introduced in order to give an answer to the bothersome problem of control windup. It was first applied to single SISO control loop. The present paper generalizes the conditioning concept to complex MIMO control structures. The self-conditionable structure of a controller is introduced in order to make the implementation of the conditioning technique easier. Some basic stability conditions are derived for the closed-loop conditioned system. Finally, the conditioning technique is shown to be very useful for initializing control algorithms. There are certainly many possible extensions to the applications and theory of the conditioning technique. For instance, generalization to systems with m inputs and l outputs ($l \neq m$) and/or to controllers with a singular matrix of direct action D requires further investigations.

Since 1979, the conditioning technique has been systematically used at the Université de Bruxelles by all the research workers and students in the control engineering department, for all kinds of controllers. It has always given full satisfaction. The technique has been used fruitfully in different industries such as power production and distribution, chemical and petrochemical industries, and production of mica sheets and cement tubes. Some Belgian constructors of automatic control materials such as the A.C.E.C. and A.B.S.Y. companies have already incorporated the technique in their products. We are convinced that the conditioning technique still has fine prospects in future use.

Acknowledgements—This work was supported in part by the Belgian National Fund for Scientific Research (F.N.R.S.), by the Belgian Institute for Encouragement of Scientific Research in Industry and Agriculture (I.R.S.I.A.) and the Ministry of the Brussels Region.

REFERENCES

Åström, K.-J. and B. Wittenmark (1984). *Computer Controlled Systems, Theory and Design.* Prentice-Hall, Englewood Cliffs, New Jersey.

Debelle, J. (1979). A control structure based upon process models. *Journal A*, **20,** 71–81.

Fertik, H. A. and C. H. Ross (1967). Direct digital control algorithm with anti-windup feature. Instrument Society of America Preprint, Number 10-1-1CQS-67.

Glattfelder, A. H. and W. Schaufelberger (1983). Stability analysis of single loop control systems with saturation and antireset-windup circuits. *IEEE Trans. Aut. Control,* **AC-28,** 1074–1081.

Hanus, R. (1979). Contribution à la théorie des régulateurs conditionnés. Thèse de doctorat, Faculté des Sciences appliquées, Université Libre de Bruxelles, Belgique.

Hanus, R. (1980a). A new technique for preventing control windup. *Journal A*, **21,** 15–20.

Hanus, R. (1980b). The conditioned control: a new technique for preventing windup nuisances. Proc. IFIP—ASSOPO, Trondheim, 221–224.

Khandheria, J. and W. L. Luyben (1976). Experimental evaluation of digital algorithms of antireset windup. *Ind. Engng Chem. Process. Des Dev.*, **15,** 278–285.

Kramer, L. C. and K. W. Jenkins (1971). A new technique for preventing direct digital control windup. Joint Automatic Control Conference Preprints, Paper no. 6-E4, 571–577.

Vandenbussche, P. (1975). Digital transposition and extension of classical analogical control algorithm. IFAC, 6th Triennal World Congress, Session 5, Part IV–A, 5.6.1–5.6.3.