

Integrator Windup

A controller with integral action combined with an actuator that becomes saturated can give some undesirable effects. If the control error is so large that the integrator saturates the actuator, the feedback path will be broken, because the actuator will remain saturated even if the process output changes. The integrator, being an unstable system, may then integrate up to a very large value. When the error is finally reduced, the integral may be so large that it takes considerable time until the integral assumes a normal value again. This effect is called *integrator windup*. The effect is illustrated in Fig. 8.9.

There are several ways to avoid integrator windup. One possibility is to stop updating the integral when the actuator is saturated. Another method is illustrated by the block diagram in Fig. 8.10(a). In this system an extra feedback path is provided by measuring the actuator output and forming an error signal (e_s) as the difference between the actuator output (u_c) and the controller output (v) and feeding this error back to the integrator through the gain $1/T_i$. The error signal e_s is zero when the actuator is not saturated. When the actuator is saturated the extra feedback path tries to make the error signal e_s equal zero. This means that the integrator is reset, so that the controller output is at the saturation limit. The integrator is thus reset to an appropriate value with the time constant T_i , which is called the tracking-time constant. The advantage with this scheme for *antiwindup* is that it can be applied to any actuator, that is, not only a saturated actuator but also an actuator with arbitrary characteristics, such as a dead zone or an hysteresis, as long as the actuator output is measured. If the actuator output is not measured, the actuator can be modeled and an

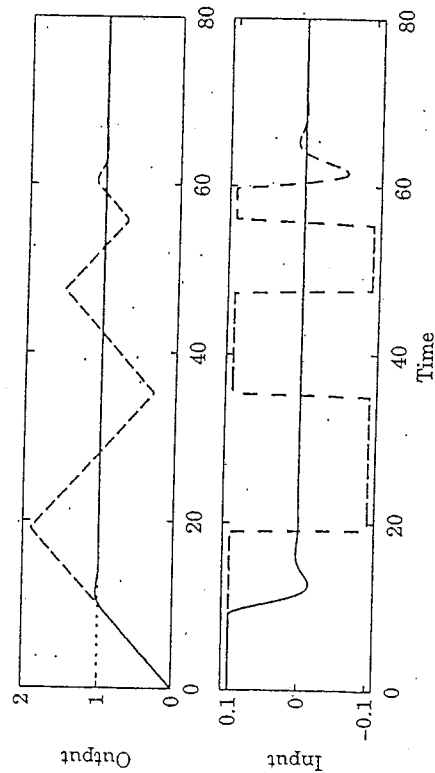


Figure 8.9 Illustration of integrator windup. The dashed lines show the response with an ordinary PID-controller. The solid lines show the improvement with a controller having antiwindup.

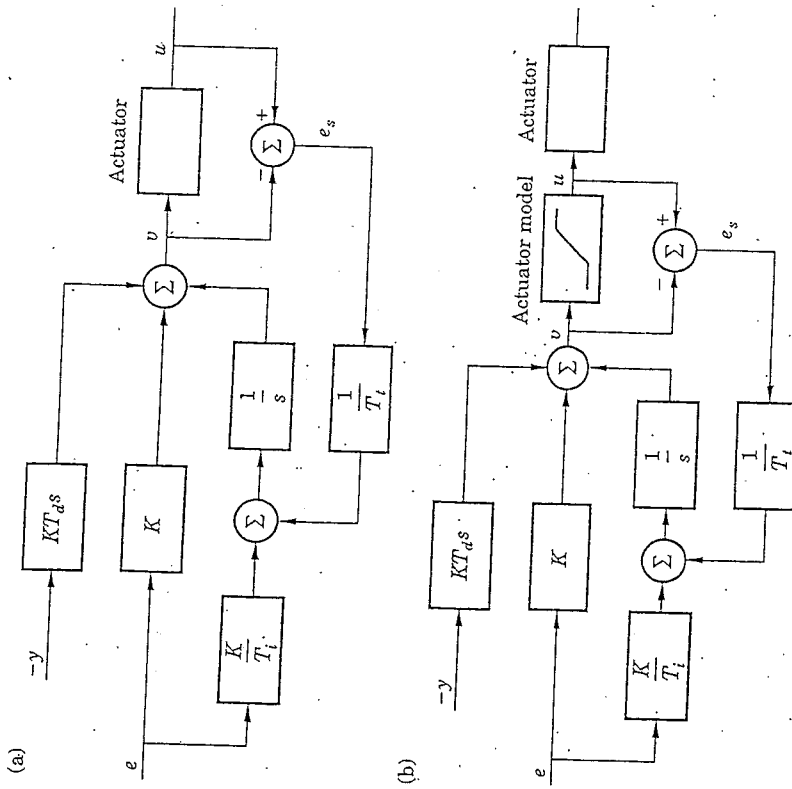


Figure 8.10 Controller with antiwindup. A system in which the actuator output is measured is shown in (a) and a system in which the actuator output is estimated from a mathematical model is shown in (b).

equivalent signal can be generated from the model, as shown in Fig. 8.10(b). Figure 8.9 shows the improved behavior with controllers having an anti-windup scheme. Antiwindup is further discussed in Sec. 9.4.

Operational Aspects

Practically all PID-controllers can run in two modes: manual and automatic. In manual mode the controller output is manipulated directly by the operator, typically by push buttons that increase or decrease the controller output. The controllers may also operate in combination with other controllers, as in a cascade or ratio connection, or with nonlinear elements such as multipliers and selectors. This gives rise to more operational modes. The controllers also have parameters that can be adjusted in operation. When there are changes of

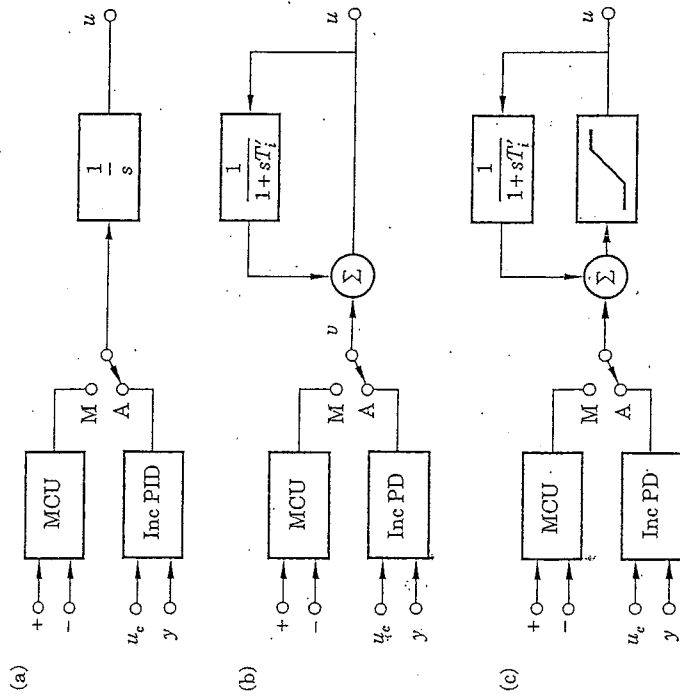


Figure 8.11 Controllers with bumpless transfer from manual to automatic mode. The controller in (a) is incremental. The controllers in (b) and (c) are special forms of position algorithms. The controller in (c) has antiwindup (MCU = Manual Control Unit).

modes and parameters, it is essential to avoid switching transients. The way mode switchings and parameter changes are made depends on the structure chosen for the controller.

Bumpless transfer. Because the controller is a dynamic system it is necessary to make sure that the state of the system is correct when switching the controller between manual and automatic mode. When the system is in manual mode, the controller produces a control signal that may be different from the manually generated control signal. It is necessary to make sure that the value of the integrator is correct at the time of switching. This is called *bumpless transfer*. Bumpless transfer is easy to obtain for a controller in incremental form. This is shown in Fig. 8.11(a). The integrator is provided with a switch so that the signals are either chosen from the manual or the automatic increments. Because the switching only influences the increments, there will not be any large transients. A related scheme for a position algorithm is shown in Fig. 8.11(b). In this case the integral action is realized as positive feedback around a first-order

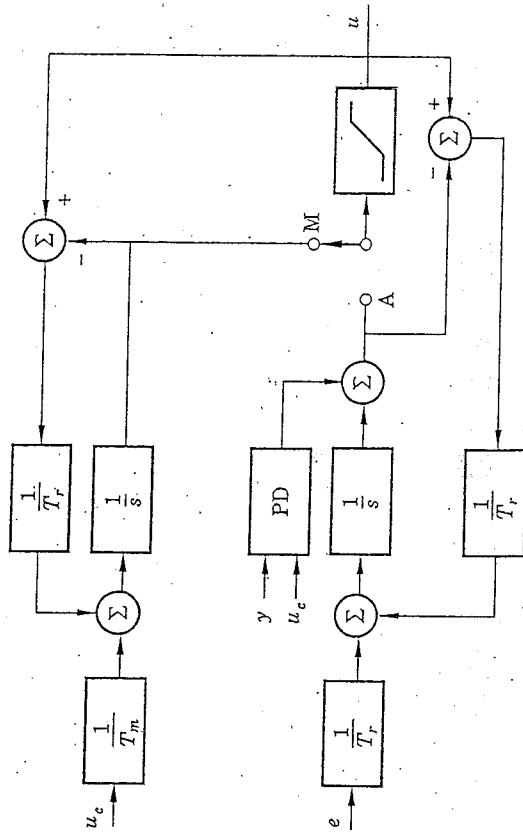


Figure 8.12 PID-controller with bumpless switching between manual and automatic control.

system. The transfer function from v to u is

$$\frac{1}{1 - \frac{1}{1 + sT_r}} = \frac{1 + sT_r}{sT_r}$$

For simplicity the filters are shown in continuous-time form. In a digital system they are of course realized as sampled systems. The system can also be provided with an antiwindup protection, as shown in Fig. 8.11(c). A drawback with this scheme is that the PID-controller must be of the form

$$G(s) = K' \frac{(1 + sT_r')(1 + sT_d')}{sT_i'} \quad (8.27)$$

which is less general than (8.22). Moreover the reset-time constant is equal to T_i' . More elaborate schemes have to be used for general PID-algorithms on position form. Such a controller is built up of a manual control module and a PID-module, each having an integrator. See Fig. 8.12.

Bumpless Parameter Changes

A controller is a dynamic system. A change of the parameters of a dynamic system will naturally result in changes of its output even if the input is kept constant. Changes in the output can in some cases be avoided by a simultaneous

Listing 9.2 Computer code skeleton that implements the control algorithm (9.2). This code has a smaller computational delay than the code in Listing 9.1.

```
Procedure Regulate
begin
  1  Adin y uc
  2  u:=u1+D*y+Dc*uc
  3  Daout u
  4  x:=F*x+G*u+Gc*uc
  5  u1:=C*x
end
```

Outliers and Measurement Malfunctions

The linear filtering theory that will be discussed in Chapter 11 is very useful in reducing the influence of measurement noise. However, there may also be other types of errors, such as instrument malfunction and conversion errors. These are typically characterized by large deviations, which occur with low probabilities. It is very important to try to eliminate such errors so that they do not enter into the control-law calculations. There are many good ways to achieve this when using computer control.

The errors may be detected at the source. In systems with high-reliability requirements, this is done by duplication of the sensors. Two sensors are then combined with a simple logic, which gives an alarm if the difference between the sensor signals is larger than a threshold. A pair of redundant sensors may be regarded as one sensor that gives either a reliable measurement or a signal that it does not work.

Three sensors may be used in more extreme cases. A measurement is then accepted as long as two out of the three sensors agree (two-out-of-three logic). It is also possible to use even more elaborate combinations of sensors and filters.

An observer can also be used for error detection. For example, consider the control algorithm of (9.1) with an explicit observer. Notice that the one-step prediction error

$$\varepsilon(k) = y(k) - \hat{y}(k|k-1) = y(k) - C\hat{x}(k|k-1) \tag{9.4}$$

appears explicitly in the algorithm. This error can be used for diagnosis and to detect if the measurements are reasonable. This will be further discussed in connection with the Kalman filter in Chapter 11.

In computer control there are also many other possibilities for detecting different types of hardware and software errors. A few extra channels in the A-D converter, which are connected to fixed voltages, may be used for testing and calibration. By connecting a D-A channel to an A-D channel, the D-A converter may also be tested and calibrated.

The design methods of Chapters 4, 5, and 8 are all based on the assumption that the process can be described by a linear model. Although linear theory has a wide applicability, there are often some nonlinearities that must be taken into account. For example, it frequently happens that the actuators are nonlinear, as is shown in Fig. 9.3. Valves are commonly used as actuators in process-control systems. This corresponds to a nonlinearity of the saturation type where the limits correspond to a fully open or closed valve. The system shown in Fig. 9.3 can be described linearly when the valve does not saturate. The nonlinearity is thus important when large changes are made. There may be difficulties with the control system during startup and shutdown, as well as during large changes, if the nonlinearities are not considered. A typical example is *integrator windup*. Other typical nonlinearities in practical systems are rate limitations, hysteresis, and backlash.

The rational way to deal with the saturation is to develop a design theory that takes the nonlinearity into account. This can be done using optimal-control theory. However, such a design method is quite complicated. The corresponding control law is also complex. Therefore, it is practical to use simple heuristic methods.

Difficulties occur because the controller is a dynamic system. When the control variable saturates, it is necessary to make sure that the state of the controller behaves properly. Different ways of achieving this are discussed in what follows.

Antiwindup for State-Space Controllers with an Explicit Observer

Consider first the case when the control law is described as an observer combined with a state feedback (9.1). The controller is a dynamic system, whose state is represented by the estimated state $\hat{x}$ in (9.1). In this case it is straightforward to see how the difficulties with the saturation may be avoided.

The estimator of (9.1) gives the correct estimate if the variable u in (9.1) is the actual control variable u_p in Fig. 9.3. If the variable u_p is measured, the estimate given by (9.1) and the state of the controller will be correct even if the control variable saturates. If the actuator output is not measured, it can be estimated—provided that the nonlinear characteristics are known. For the case

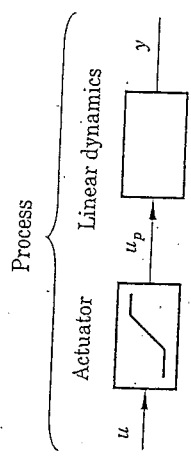


Figure 9.3 Block diagram of a process with a nonlinear actuator having saturation characteristics.

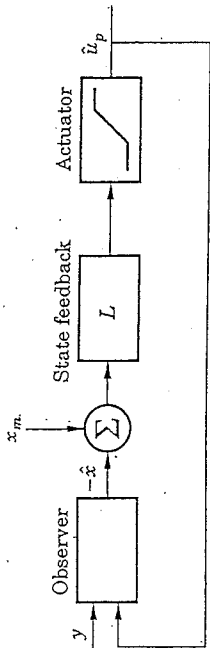


Figure 9.4 Controller based on an observer and state feedback with anti-windup compensation.

of a simple saturation, the control law can be written as

$$\begin{aligned}\hat{x}(k|k) &= \hat{x}(k|k-1) + K(y(k) - C\hat{x}(k|k-1)) \\ &= (I - KC)\Phi\hat{x}(k-1|k-1) + Ky(k) + (I - KC)\Gamma\hat{u}_p(k-1) \\ \hat{u}_p(k) &= \text{sat}(L(x_m(k) - \hat{x}(k|k)) + Du_c(k))\end{aligned}\quad (9.5)$$

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k) + \Gamma\hat{u}_p(k)$$

where the function sat is defined as

$$\text{sat } u = \begin{cases} u_{\text{low}} & u \leq u_{\text{low}} \\ u & u_{\text{low}} < u < u_{\text{high}} \\ u_{\text{high}} & u \geq u_{\text{high}} \end{cases}\quad (9.6)$$

for a scalar and

$$\text{sat } u = \begin{bmatrix} \text{sat } u_1 \\ \text{sat } u_2 \\ \vdots \\ \text{sat } u_n \end{bmatrix}\quad (9.7)$$

for a vector. The values u_{low} and u_{high} are chosen to correspond to the actuator limitations. A block diagram of a controller with a model for the actuator nonlinearity is shown in Fig. 9.4. Observe that even if the transfer function from y to u for (9.1) is unstable, the state of the system in (9.5) will always be bounded if the matrix $(I - KC)\Phi$ is stable. It is also clear that x will be a good estimate of the process state even if the valve saturates, provided that u_{low} and u_{high} are chosen properly.

Antiwindup for the General State-Space Model

The controller may also be specified as a state-space model of the form in (9.2):

$$x(k+1) = Fx(k) + Gy(k)\quad (9.8)$$

$$u(k) = Cx(k) + Dy(k)\quad (9.9)$$

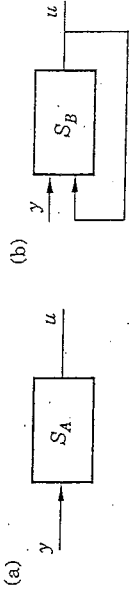


Figure 9.5 Different representations of the control law.

which does not include an explicit observer. The command signals have been neglected for simplicity. If the matrix F has eigenvalues outside the unit disc and the control variable saturates, it is clear that windup may occur. Assume, for example, that the output is at its limit and there is a control error y . The state and the control signal will then continue to grow, although the influence on the process is restricted because of the saturation.

To avoid this difficulty, it is desirable to make sure that the state of (9.8) assumes a proper value when the control variable saturates. In conventional process controllers, this is accomplished by introducing a special *tracking mode*, which makes sure that the state of the controller corresponds to the input-output sequence $\{u_p(k), y(k)\}$. The design of a tracking mode may be formulated as an observer problem. In the case of state feedback with an explicit observer, the tracking is done automatically by providing the observer with the actuator output u_p or its estimate $\hat{u}_p$. In the controller of (9.8) and (9.9), there is no explicit observer. To get a controller that avoids the windup problem, the solution for the controller with an explicit observer will be initiated. The control law is first rewritten as indicated in Fig. 9.5. The systems in (a) and (b) have the same input-output relation. The system S_B is also stable. By introducing a saturation in the feedback loop in (b), the state of the system S_B is always bounded if y and u are bounded. This argument may formally be expressed as follows. Multiply (9.9) by K and add to (9.8). This gives

$$\begin{aligned}x(k+1) &= Fx(k) + Gy(k) + K(u(k) - Cx(k) - Dy(k)) \\ &= (F - KC)x(k) + (G - KD)y(k) + Ku(k) \\ &= F_0x(k) + G_0y(k) + Ku(k)\end{aligned}$$

If the system of (9.8), and (9.9) is observable, the matrix K can always be chosen so that $F_0 = F - KC$ has prescribed eigenvalues inside the unit disc. Notice that this equation is analogous to (9.5). By applying the same arguments as for the controller with an explicit observer, the control law becomes

$$\begin{aligned}x(k+1) &= F_0x(k) + G_0y(k) + Ku(k) \\ u(k) &= \text{sat}(Cx(k) + Dy(k))\end{aligned}\quad (9.10)$$

The saturation function is chosen to correspond to the actual saturation in the actuator. A comparison with the case of an explicit observer shows that (9.10) corresponds to an observer with dynamics given by the matrix F_0 . The system

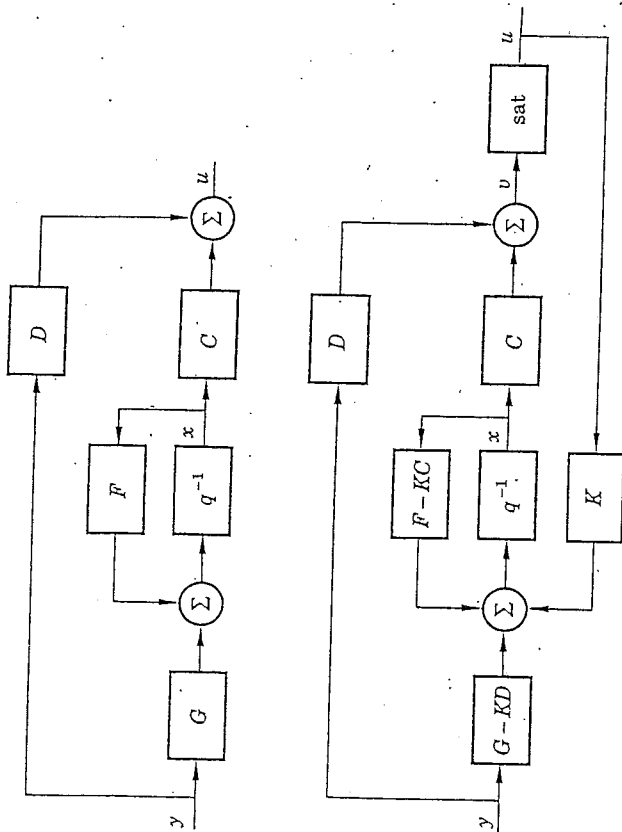


Figure 9.6 Block diagram of the controller (9.2) and the modification in (9.10) that avoids windup.

of (9.10) is also equivalent to (9.2) for small signals. A block diagram of the controller with antireset windup compensation is shown in Fig. 9.6.

Antiwindup for the Input-Output Form

The corresponding construction can also be carried out for controllers characterized by input-output models. Consider a controller described by

$$R(q)u(k) = T(q)u_c(k) - S(q)y(k) \quad (9.11)$$

where R , S , and T are polynomials in the forward-shift operator. The problem is to rewrite the equation so that it looks like a dynamic system with the observer dynamics driven by three inputs, the command signal u_c , the process output y , and the control signal u . This is accomplished as follows.

Let $A_{aw}(q)$ be the desired characteristic polynomial of the antiwindup observer. Adding $A_{aw}(q)u(k)$ to both sides of (9.11) gives

$$A_{aw}u = Tu_c - Sy + (A_{aw} - R)u$$

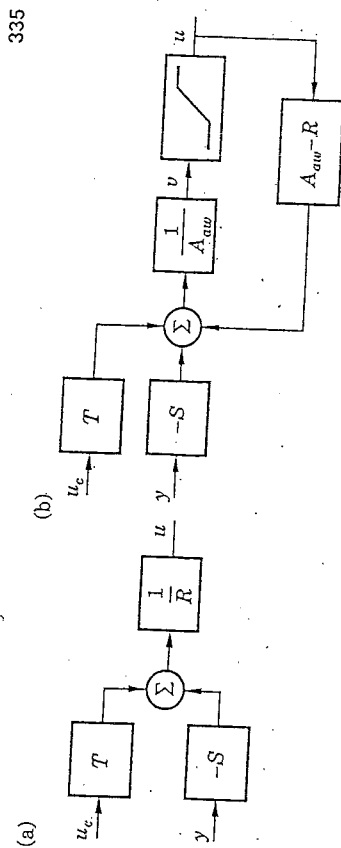


Figure 9.7 Block diagram of the controller of (9.11) and the modification in (9.12) that avoids windup.

A controller with antiwindup compensation is then given by

$$A_{aw}v = Tu_c - Sy + (A_{aw} - R)u. \quad (9.12)$$

$$u = \text{sat } v$$

This controller is equivalent to (9.11) when it does not saturate. When the control variable saturates, it can be interpreted as an observer with dynamics given by polynomial A_{aw} .

A block diagram of the linear controller of (9.11) and the nonlinear modification of (9.12) that avoids windup is shown in Fig. 9.7. A particularly simple case is that of a deadbeat observer, that is, $A_{aw}^* = 1$. The controller can then be written as

$$u(k) = \text{sat} \left(T^*(q^{-1})u_c(k) - S^*(q^{-1})y(k) + (1 - R^*(q^{-1}))u(k) \right) \quad (9.13)$$

An example illustrates the implementation.

Example 9.1 Double integrator with antireset windup

A controller with integral action for the double integrator was designed in Sec. 5.7. In this example we use the same design procedure with parameters $\omega = 0.4$ and $\omega h = 0.2$. The result when using the antireset windup procedure in Fig. 9.7 with $A_{aw} = (q - 0.5)^2$ is shown in Fig. 9.8. The antireset windup gives less overshoot and the control signal is only saturating at the maximum value. The response with the antireset windup is similar but somewhat slower than for the unsaturated case.

A generalization of the antireset windup in Fig. 9.7 is given in Fig. 9.9. An extra degree of freedom is introduced through the polynomial A_n . This polynomial, as well as A_{aw} , should be monic and stable. The case in Fig. 9.7 is obtained for $A_n = 1$. The polynomial A_n can be used to shape the response from errors due to the saturation.

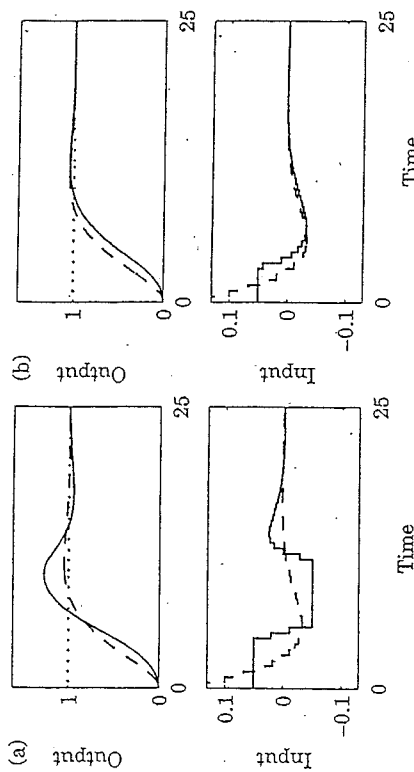


Figure 9.8 Output and input when the double integrator is controlled with a controller (a) without and (b) with antirest windup given by (9.12). The dashed lines show the behavior when there is no saturation.

9.5 Operational Aspects

The interface between the controller and the operator is discussed in this section. This includes an evaluation of the information displayed to the operator and the mechanisms for the operator to change the parameters of the controller. In conventional analog controllers it is customary to display the set point, the measured output, and the control signal. The controller may also be switched from manual to automatic control. The operator may change the gain (or proportional band), the integration time, and the derivative time. This organization was motivated by properties of early analog hardware. When computers are used to implement the controllers, there are many other possibilities. So far the

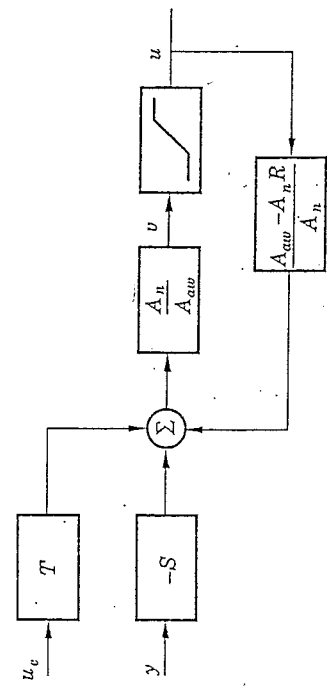


Figure 9.9 A generalization of the antiwindup scheme in Fig.9.7.

potentials of the computer have been used only to a very modest degree.

To discuss the operator interface, it is necessary to consider how the system will be used operationally. This is mentioned in Sec. 6.2 and a few additional comments are given here. First, it is important to realize the wide variety of applications of control systems. There is no way to give a comprehensive treatment, so a few examples are given. For instance, the demands are very different for an autopilot, a process-control room, or a pilot plant.

Example 9.2 Importance of operational aspects

To illustrate that the operational aspects and security are important we take two examples from practical implementations. The first example is a control system for a steel rolling mill. In this application the control, signal conditioning and logic took about 30% of the code and the rest was related to operator interface and security measures. The second example is the implementation of an autotuner based on relay feedback. A straightforward implementation of the tuning algorithm could be done in 1.5 pages of C code. The commercial algorithm with all bells and whistles needed for operator communication and security required 15 pages of code.

Operating Modes

It is often desirable to have the possibility of running a system under manual control. A simple way to do this is to have the arrangement shown in Fig. 9.10, where the control variable may be adjusted manually. Manual control is often done with push buttons for increasing or decreasing the control variable.

Because the controller is a dynamic system, the state of the controller must have the correct value when the mode is switched from manual to automatic. If this is not the case, there will be a switching transient. A smooth transition is called *bumpless transfer*, or *bumpless transition*.

In conventional analog controllers, it is customary to handle bumpless transition by introducing a *tracking mode*, which adjusts the controller state so that it is compatible with the given inputs and outputs of the controller. A tracking mode may be viewed as an implementation of an observer.

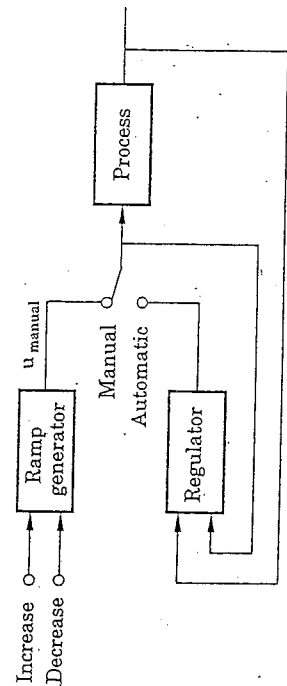


Figure 9.10 Control system with manual and automatic control modes.