# 13 EL2520 Lecture notes 13: Dealing with Hard Constraints

The focus of this course is on control of systems that can be described by linear time-invariant (LTI) models. This is the case for many, if not most, applications. The reason is that the aim of a control system typically is to keep a system close to some desired steady-state, and then the behavior can usually be well described by an LTI model[1]. However, there is one non-linearity present in essentially any control system that can not be described by a linear model, and that is hard constraints on the control inputs. Almost any control input, be it e.g., a valve or the power of an engine, has hard lower and upper limits. A valve can for instance at most be fully open or fully closed. In the previous lecture we saw how such constraints can be included in the control optimization in MPC, providing optimal control under input (and possibly output) constraints. In this lecture we will consider an approach based on adding a modification to a controller that has been designed based on an LTI model, e.g., an LQG-controller, an $\mathcal{H}_\infty$-optimal controller or a simple PID controller. This approach is usually termed *Anti Reset Windup*, referring to the fact that the integral (reset) part of a controller "winds up", i.e., just grows and grows, when there is a non-zero control error and the input is at a constraint.

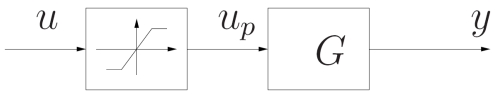


Figure 13.1: Hard constraint on input.

Consider the open-loop system in Figure 13.1. Here $u$ is the input computed by the controller, i.e., the controller output, while $u_p$ is the input actually implemented on the system. A hard constraint implies that $u_p = u_{min}$ when $u < u_{min}$, $u_p = u$ when $u_{min} < u < u_{max}$ and $u_p = u_{max}$ when $u > u_{max}$. Thus, when the computed input is outside the hard constraints, the implemented input is constant implying that the feedback loop is broken. See also Figure 13.2. With the feedback loop broken it implies that the poles of the system equals the open-loop poles, i.e., the union of the poles of $F_y(s)$ and $G(s)$. Thus, if either $F_y(s)$ or $G(s)$ is unstable the system becomes unstable as the input $u_p$ saturates (is at a constraint). If $G(s)$ is unstable, it is not much we can do other than ensure we do not hit a constraint in the input. However, if the controller $F_y(s)$ is unstable then we can in principle modify the controller so that it becomes stable when the input is saturated. Note that most controllers are input-output unstable since they typically contain integral action, i.e., poles at $s = 0$.

We shall develop a modification of the controller as discussed above, starting with a

[1] In fact, feedback control usually tends to make a system much more linear than it is without control.

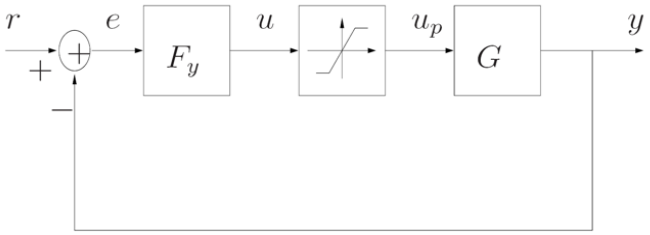


Figure 13.2: Feedback loop with constraint on input.

controller consisting of an observer with feedback of observed states

$$\begin{aligned} \dot{\hat{x}} &= A\hat{x}(t) + Bu(t) + K(y(t) - C\hat{x}(t)) \quad (1) \\ u(t) &= -L\hat{x}(t) \quad (2) \end{aligned}$$

On input-output form, the controller transfer-function is

$$U(s) = -F_y(s)Y(s) = -L(sI - A + BL + KC)^{-1}KY(s) \quad (3)$$

Note that we use the computed input $u$ in the observer. A simple modification, provided we know the input constraints or we measure the true input $u_p$, is to replace $u$ with the true input $u_p$ in the observer. In the former case we effectively get a non-linear observer since we include a model of the saturation in the observer. The observer can now be written

$$\dot{\hat{x}} = A\hat{x}(t) + Bu_p(t) + K(y(t) - C\hat{x}(t)) = (A - KC)\hat{x}(t) + Bu_p(t) + Ky(t) \quad (4)$$

and on input-output form the controller becomes

$$U(s) = -L(sI - A + KC)^{-1}Y(s) + L(sI - A + KC)^{-1}BU_p(s) \quad (5)$$

When the input is not saturated, $u_p = u$ and the controller is the same as in (3) with poles given by the eigenvalues of $A - BL - KC$. However, when the input is saturated, $u_p$ is a constant and the controller poles are given by the eigenvalues of $A - KC$ as can be seen from (5). Note that the controller in the latter case has poles equal to the poles of the observer which always should be stable. Thus, when the input saturates, the controller becomes stable. The modification correponds to what is known as anti reset windup.

The modification of the observer, as discussed above, is only applicable to controllers which can be written on the form of an observer with state feedback. However, we shall see that the resulting modification of the controller can be seen as input tracking, i.e., attempting to force $u$ to be equal to $u_p$, and this idea can then be used also for other type of controllers. To this end, write (4) as

$$\dot{\hat{x}} = (A - KC)\hat{x}(t) + B(u_p(t) + u(t) - u(t)) + Ky(t) \quad (6)$$

With $u = -L\hat{x}$ we get

$$\dot{\hat{x}} = (A - KC - BL)\hat{x} + B(u_p(t) - u(t)) + Ky(t) \quad (7)$$

The controller can then be written

$$\begin{aligned} U(s) &= -L(sI - A + BL + KC)^{-1}KY(s) - L(sI - A + BL + KC)^{-1}B(U_p(s) - U(s)) \\ &= -F_y(s)Y(s) + W(s)(U_p(s) - U(s)) \end{aligned}$$

Thus, the controller is partly trying to keep $Y$ small (or, more generally, equal to the setpoint) and partly trying to keep the computed control input $u(t)$ equal to the implemented control input $u_p$. The control structure is illustrated in Figure 13.3. A typical choice for $W(s)$ is

$$W(s) = \frac{1}{T_t s}$$

that is, a pure integrator. The smaller the integral time $T_t$ is chosen, the more emphasis is put on keeping $u(t) = u_p(t)$ and hence avoiding integral windup in the feedback controller $F_y(s)$ when the input $u_p(t)$ is at a constraint. Such input tracking is known as *anti reset windup*.

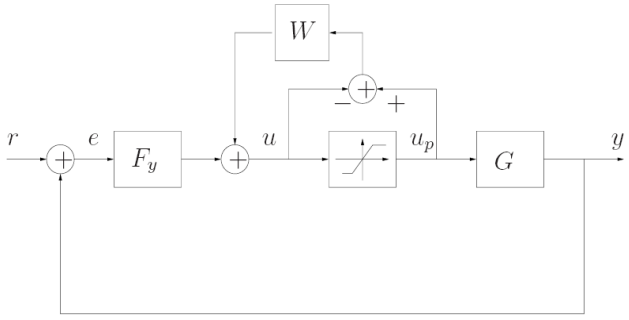


Figure 13.3: Feedback loop with constraint on input and input tracking.