第5章 抗饱和(Anti-Windup)设计

——2019年春季学期

授课教师： 马 杰 （控制与仿真中心）

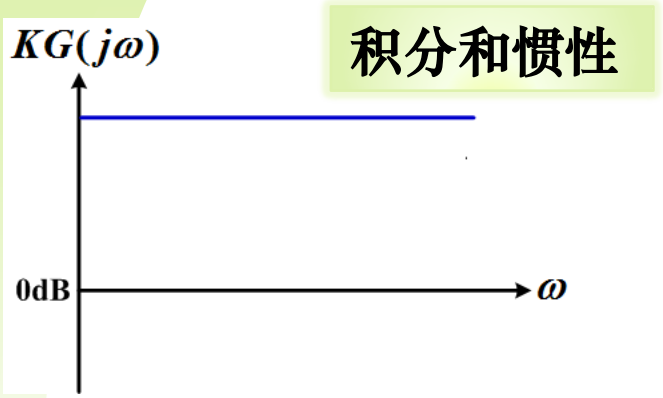
罗 晶 （控制科学与工程系）

马克茂（控制与仿真中心）

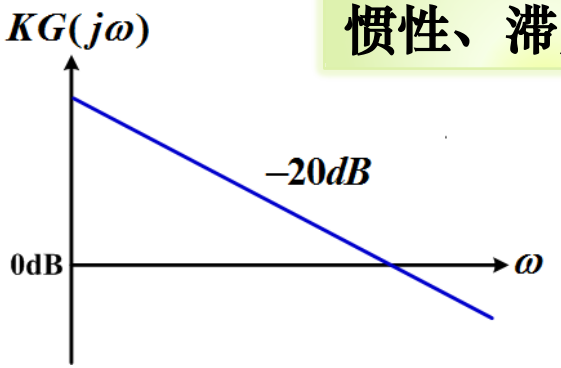
陈松林（控制与仿真中心）

上一节课内容回顾

带宽设计的第一种情况（自身带宽较宽，需要压低带宽）



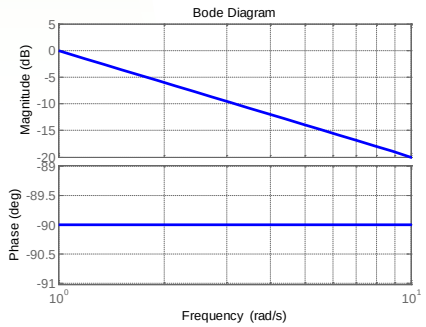
积分和惯性



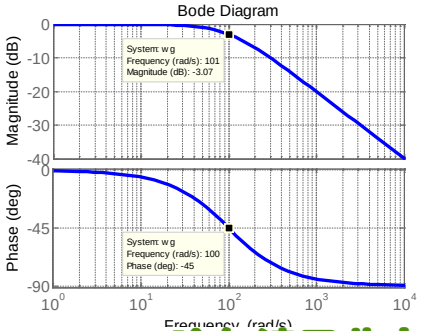
惯性、滞后、PI

压低带宽的主要方法（积分，惯性，滞后，PI）

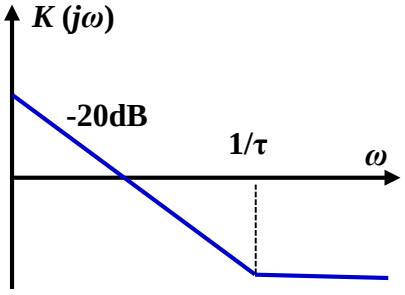
$$G_c(s)=\frac{1}{s}$$



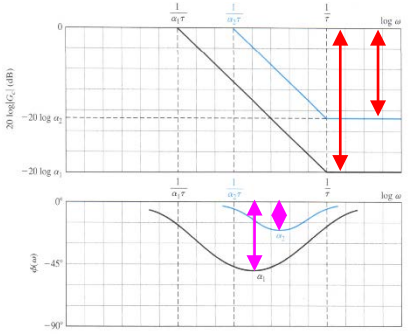
$$G_c(s)=\frac{1}{\tau s+1}$$



$$G_c(s)=\frac{K_1(\tau s+1)}{s}$$

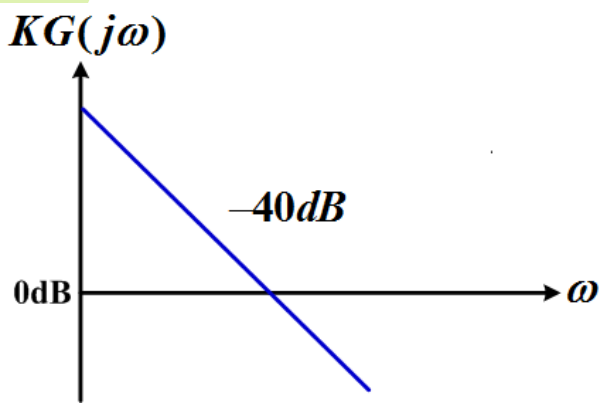


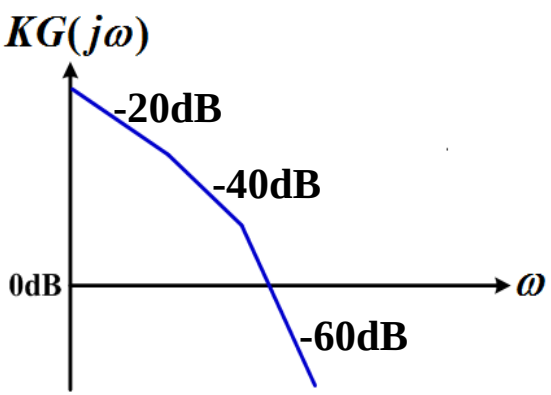
$$G_c(s)=\frac{\tau s+1}{\alpha\tau s+1}$$



上一节课内容回顾

带宽设计的第二种情况（自身带宽较窄，需要拓展带宽）





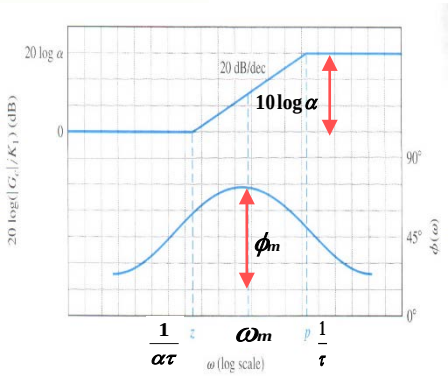
拓展带宽的主要方法（超前、近似微分/PD/PID）

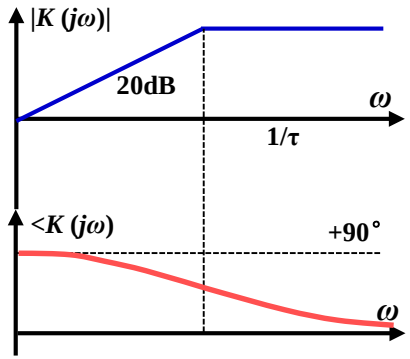
$$G_c(s)=\frac{\alpha\tau s+1}{\tau s+1}$$

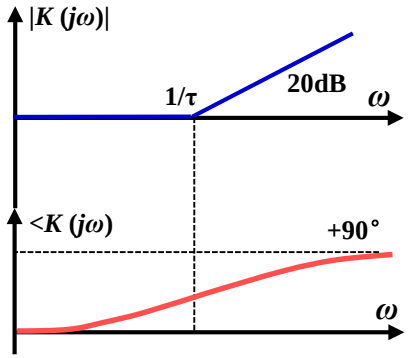
$$K(s)=\frac{Ks}{1+\tau s}$$

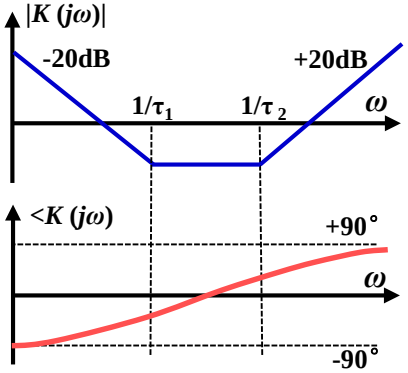
$$K_{PD}(s)=K_1(\tau s+1)$$

$$K_{PID}(s)=\frac{K_1(\tau_1 s+1)(\tau_2 s+1)}{s}$$



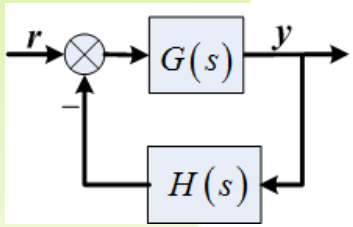


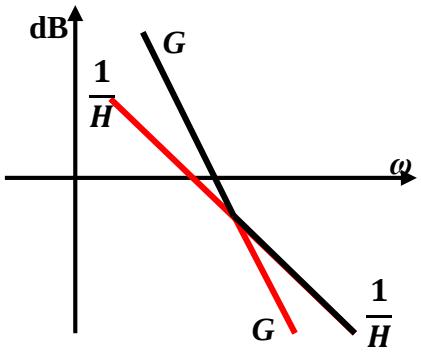




上一节课内容回顾

反馈校正可以直接改变闭环系统的带宽

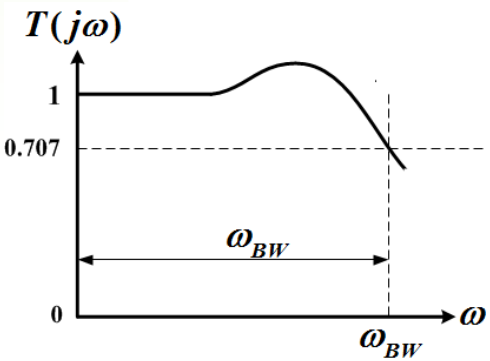


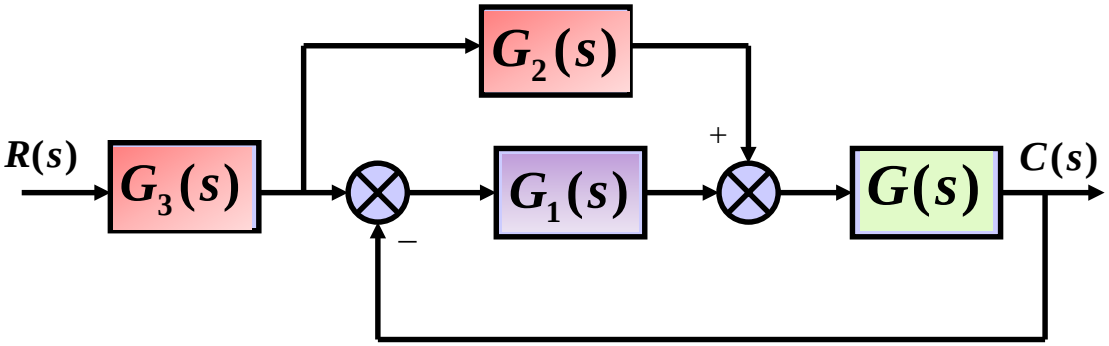


$$T(s)=\frac{G}{1+GH}=\begin{cases}\dfrac{1}{H}, & GH\gg 1,\ \text{即}G\gg\dfrac{1}{H}\\ G, & GH\ll 1,\ \text{即}G\ll\dfrac{1}{H}\end{cases}$$

当G比$\frac{1}{H}$阶次高时，低频$\frac{1}{H}$为主导，高频由G主导

拓展闭环系统带宽的方法

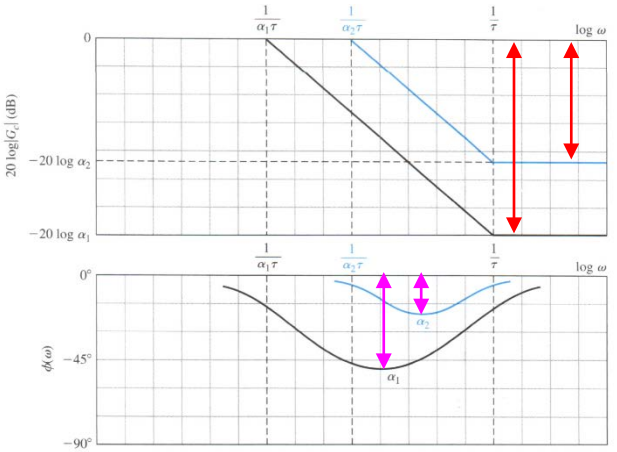




上一节课内容回顾

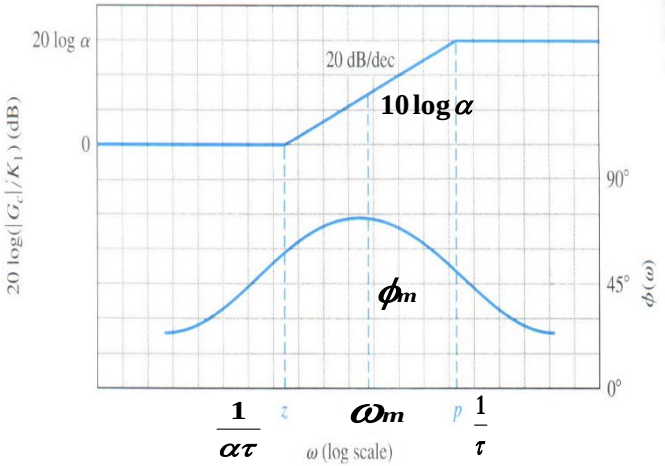
关于滞后环节（改增益）

- 滞后可以用在低频，抬高低频增益，减小误差，但要考虑相位损失，使用时，可以用**多个中心频率错开**的滞后环节，**避免相角的集中损失，从而导致条件稳定。**
- 也可以用在高频，压低高频增益，降低穿越频率，但要注意其在剪切频率处带来的相位滞后，保证系统有足够的稳定裕度。



关于超前环节（补相角）

- 超前环节一般用在高频，也就是在剪切频率处，一个超前能够补偿的相角是有限的，通常需要多个超前环节一起使用，应用时每个超前环节的**中心频率都要和期望的剪切频率一致**，这样可以**集中补偿相角**，减小对高频增益的提升和对低频增益的衰减。

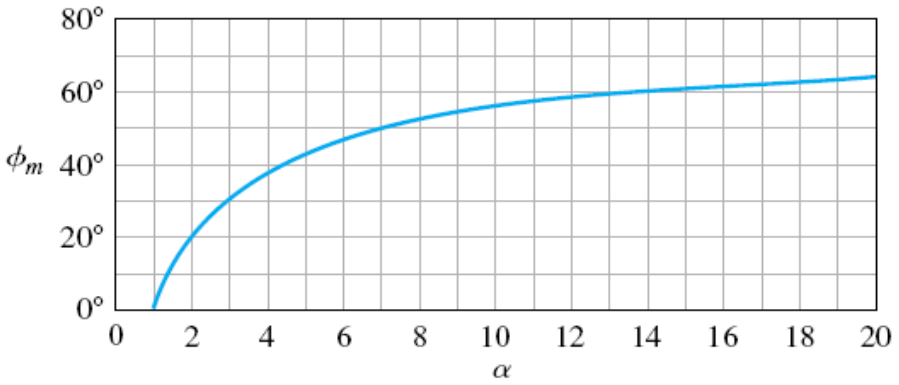


上一节课内容回顾

如何减小环节的副作用（在达到目的前提下）

- 一阶超前校正环节可提供的最大超前角最大不超过70°；
- 为了避免过多抬高高频增益（压低低频增益），尽量使用小相角的超前环节来补偿相角。

超前校正环节最大超前角与α的关系



- 每种校正环节都有其适用范围，不能随意使用；
- 多数情况下，校正环节在改变系统特性的同时会带来副作用；
- 这些环节的合理选取和使用（优化），可以减小副作用；
- 考虑到控制系统自身的约束和校正环节的副作用，带宽的拓展是有上限的，不能任意拓展；
- 控制设计时要懂得折中和取舍，要学会灵活应用。

上一节课内容回顾

频域与时域指标的关系

- 开环频域指标γ和时域指标σ和 t_s 的关系

（1）越大，σ%越小；γ越小，σ%越大。

一般希望$30° \le \gamma \le 70°$；

（2）ω_c 越大，t_s 越小；

- 闭环频域指标M_r和ω_b与时域指标σ和 t_s的关系

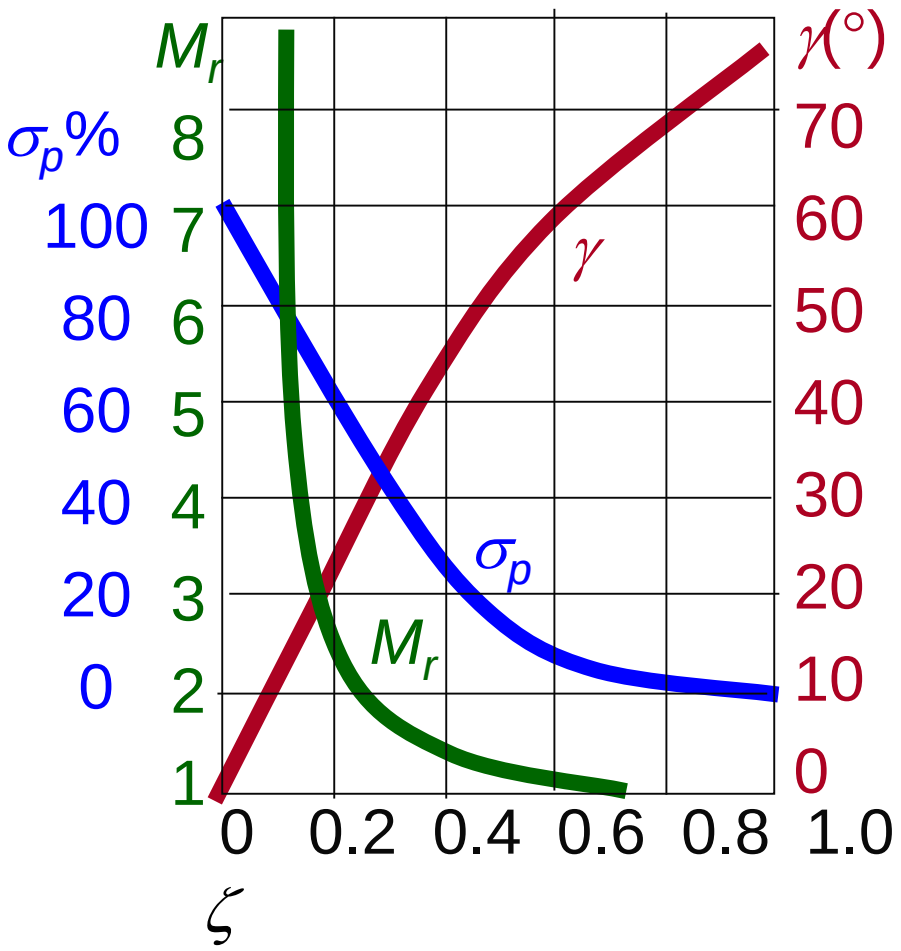
（1）M_r 越大，σ%越大；

（2）ω_b 越大，t_s 越小；

- 开环频域指标σ和 t_s 和闭环频域指标M_r和ω_b 的关系

（1）γ越大，M_r越小（$M_r \approx 1/\sin\gamma$）；

（2）ω_b越大，ω_c越大（$\omega_c < \omega_b < 2\omega_c$）；



学习目标

本节课需要掌握的内容

- 了解饱和产生的原因，可能带来的问题；
- 学会饱和和转换速率限制的数学描述；
- 掌握针对积分饱和的处理方法；
- 掌握基于反馈思想的处理饱和的方法。

本章主要内容

A1 执行器约束问题

A2 Anti-Windup设计

A3 Anti-Windup控制器的一般形式

5.1 执行器的约束问题

5.1.1 执行器约束问题的提出

5.1.2 执行器约束的描述方法

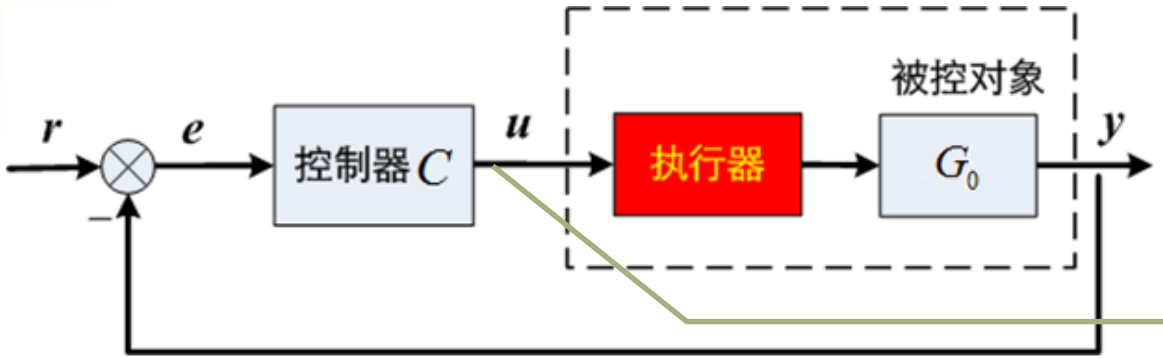
5.1.3 积分器的Windup问题

5.1.1 执行器约束问题的提出

控制变量增大的原因分析

控制系统的存在很多非线性因素，有些可以忽略，有些在设计必须要考虑。**执行器存在一些典型的物理限制：**如电机的最大转速限制、峰值力矩限制；阀门的开合不能超过全开或全闭等。

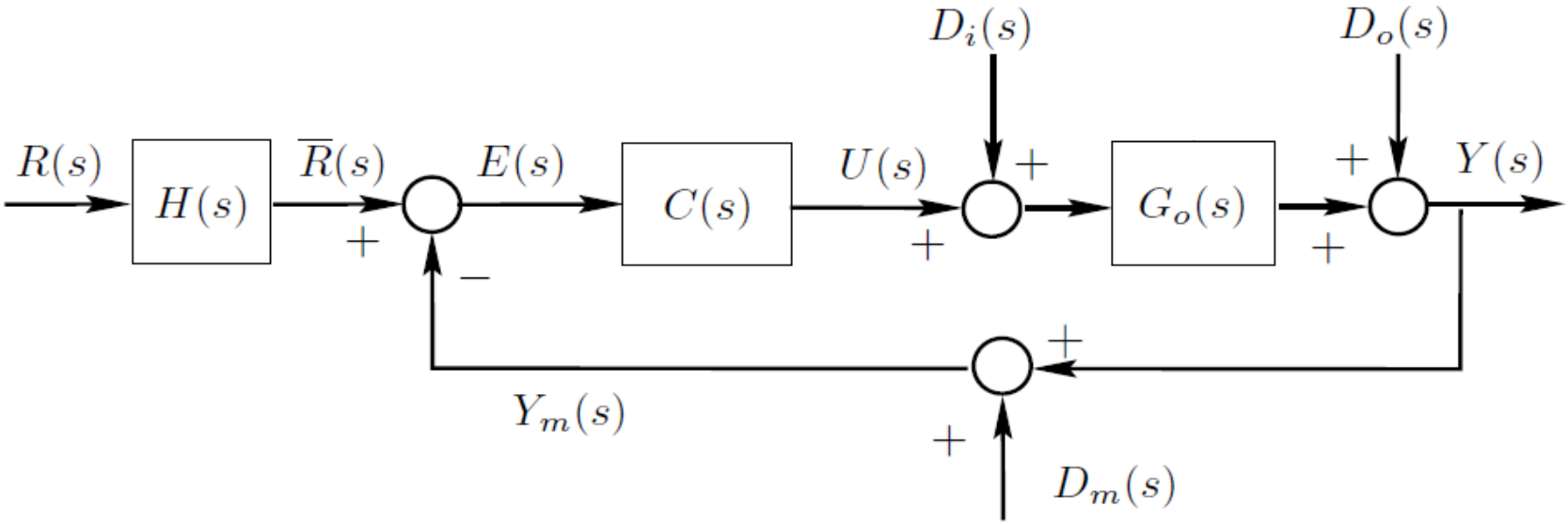
对于在比较宽范围内运行的控制系统，**控制量有可能达到执行器的极限（幅值极限或变化速率极限）**。当这种情况发生时，**反馈回路失效**，系统将运行于**开环状态**，即只要执行器处于饱和，即执行器停留在其极限状态，系统的输出相当于开环响应。



导致控制变量增大的原因是什么?

5.1.1 执行器约束问题的提出

控制变量增大的原因分析



双自由度闭环回路

$$T_o \triangleq \frac{G_0 C}{1+G_0 C}$$

$$S_o \triangleq \frac{1}{1+G_0 C}$$

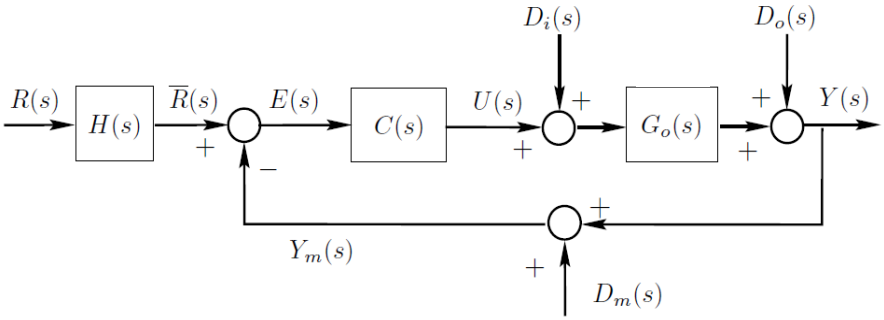
$$S_{uo} \triangleq \frac{C}{1+G_0 C}$$

$$S_{io} \triangleq \frac{G_0}{1+G_0 C}$$

5.1.1 执行器约束问题的提出

控制变量增大的原因分析

输入与输出表达式关系如下：



$$Y(s) = T_o(s)\big(H(s)R(s) - D_m(s)\big) + S_o(s)D_o(s) + S_{io}(s)D_i(s)$$

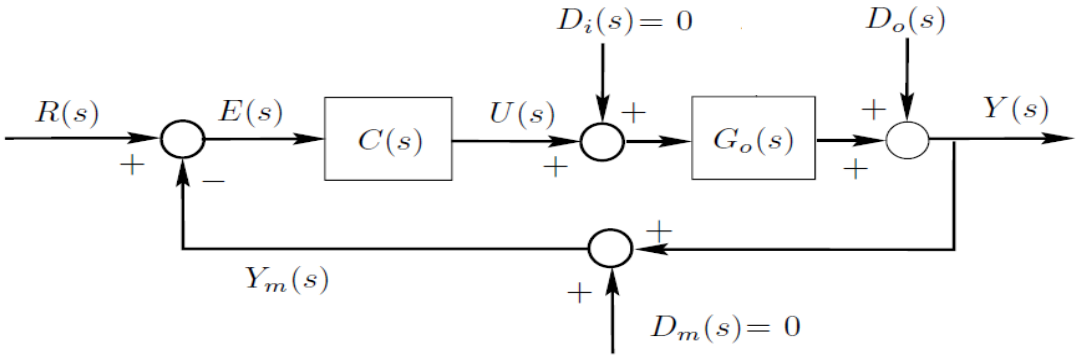
$$U(s) = S_{uo}(s)\left(H(s)R(s) - D_m(s) - D_o(s) - G_o(s)D_i(s)\right)$$

$$T_o \triangleq \frac{G_0C}{1+G_0C} \qquad S_o \triangleq \frac{1}{1+G_0C} \qquad S_{uo} \triangleq \frac{C}{1+G_0C} \qquad S_{io} \triangleq \frac{G_0}{1+G_0C}$$

5.1.1 执行器约束问题的提出

控制变量增大的原因分析

参考信号$R(s)$或者输出扰动信号$D_o(s)$存在大的快速变化时，通常会引起控制器输出信号较大的峰值和变化速率，而输入扰动的高频变化通常已经由对象特性滤掉。



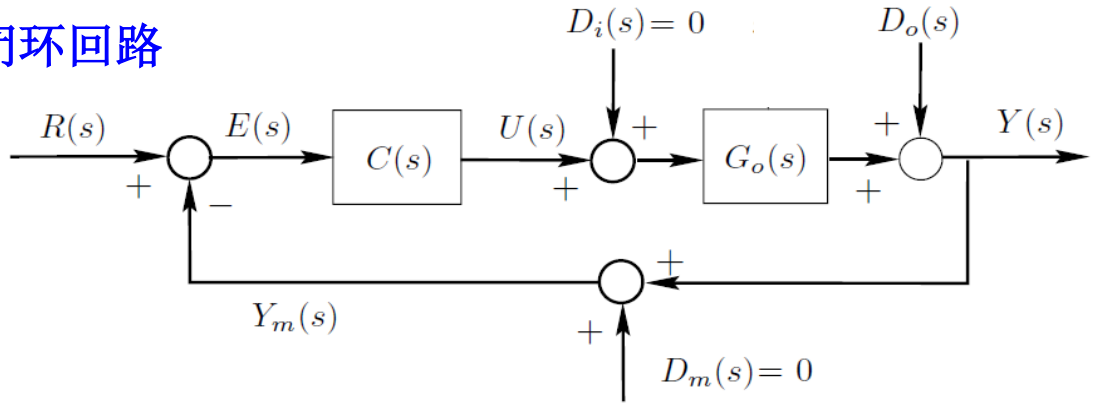
单自由度闭环回路

$$U(s) = S_{uo}(s)\left(R(s) - D_m(s) - D_o(s) - G_o(s)D_i(s)\right)$$

5.1.1 执行器约束问题的提出

控制变量增大的原因分析

单自由度闭环回路



$$U(s) = S_{uo}(s)\left(R(s) - D_m(s) - D_o(s) - G_o(s)D_i(s)\right)$$

$$U(s) = S_{uo}(s)(R(s) - D_o(s))$$

$$S_{uo}(s) \triangleq \frac{T_o(s)}{G_o(s)}$$

$$sU(s) = S_{uo}(s)[sR(s) - sD_o(s)] = \frac{T_o(s)}{G_o(s)}[sR(s) - sD_o(s)]$$

5.1.1 执行器约束问题的提出

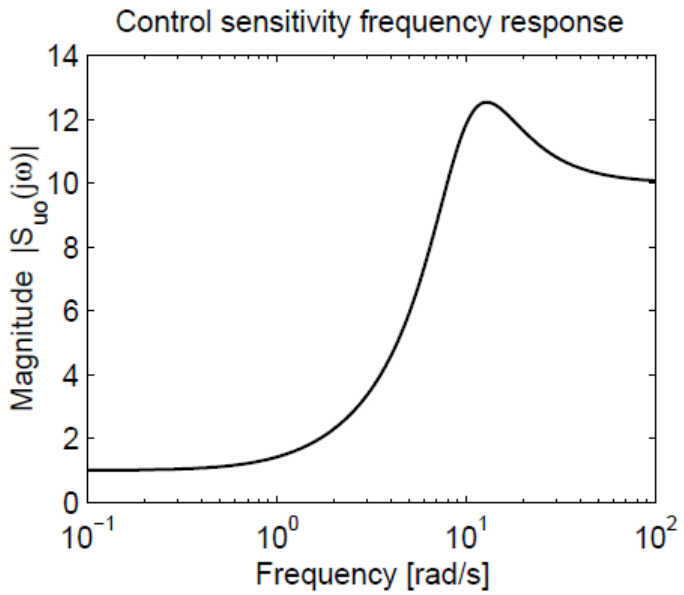
控制变量增大的原因分析

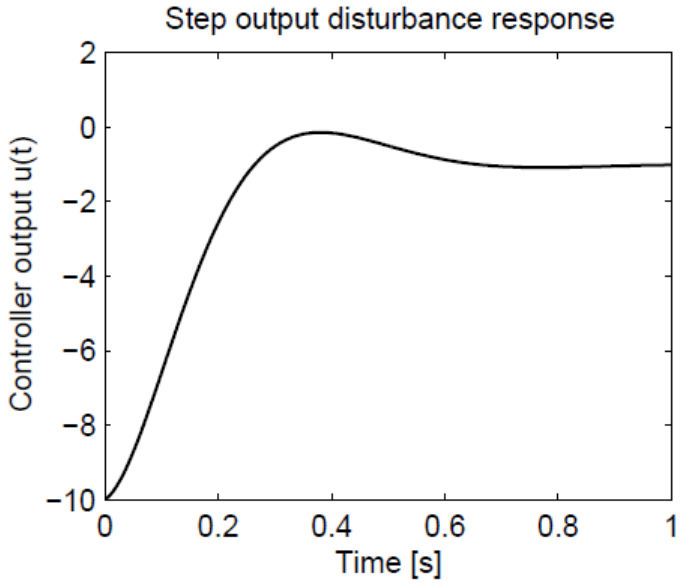
◆例1

$$G_o(s)=\frac{10}{(s+10)(s+1)}$$

$$T_o(s)=\frac{100}{s^2+12s+100}$$

闭环带宽近似10倍于对象带宽





如果闭环带宽远大于开环模型的带宽，则S_{uo}将显著地放大参考信号和输出扰动中的高频成分。

$$U(s)=S_{uo}(s)(R(s)-D_o(s))$$

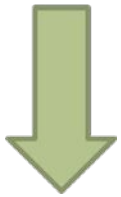
$$sU(s)=S_{uo}(s)[sR(s)-sD_o(s)]$$

$$S_{uo}(s)\triangleq\frac{T_o(s)}{G_o(s)}$$

5.1.1 执行器约束问题的提出

问题的提出——执行器约束带来的设计约束

对于实际的执行器，其能力都是有限的，存在幅值和变化速率的约束（分别称为**饱和**与**转换速率**限制），忽视这些约束，可能带来严重的性能下降，甚至使系统失稳。



为了避免执行器饱和或转换速率限制引起的问题，通常有必要对**闭环带宽**设置一个上界！

5.1.1 执行器约束问题的提出

问题的提出——执行器约束带来的设计约束

在设计流程的方案设计阶段，需要根据指标要求、信号分析等确定执行器的能力并完成选型。

对于有些控制问题（比如时间最优控制问题），在问题描述时就需要明确执行器的约束，并在设计时加以考虑；否则，可能会导致问题本身失去意义。

$$\dot{x} = f(t, x, u), \qquad x(t_0) = x_0 \qquad u \in U \subset \mathbb{R}^m$$

$$J(u) := \int_{t_0}^{t_f} L(t, x(t), u(t))dt + K(t_f, x_f)$$

5.1 执行器的约束问题

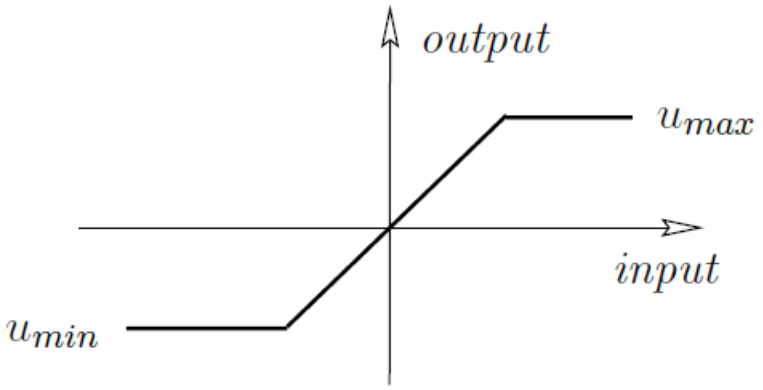
5.1.1 执行器约束问题的提出

5.1.2 执行器约束的描述方法

5.1.3 积分器的Windup问题

5.1.2 执行器约束的描述方法

执行器饱和模型



$$u(t) = Sat\langle \hat{u}(t) \rangle \triangleq \begin{cases} u_{max} & \text{if } \hat{u}(t) > u_{max}, \\ \hat{u}(t) & \text{if } u_{min} \le \hat{u}(t) \le u_{max}, \\ u_{min} & \text{if } \hat{u}(t) < u_{min}. \end{cases}$$

u_{max}——执行器输出的最大幅值；

u_{min}——执行器输出的最小幅值；

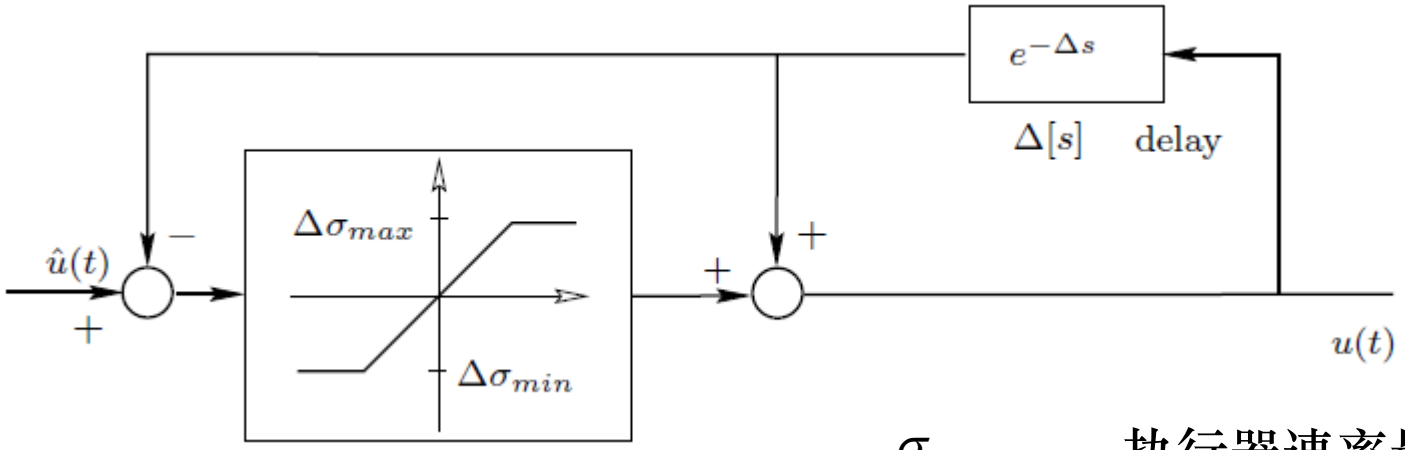
$u(t)$——执行器实际输出；

$\hat{u}(t)$——执行器期望输出；

$Sat\langle \bullet \rangle$——饱和函数。

5.1.2 执行器约束的描述方法

执行器转换速率限制模型



$$\dot{u}(t) = Sat\langle\dot{\hat{u}}(t)\rangle \triangleq \begin{cases} \sigma_{max} & \text{if } \dot{\hat{u}}(t) > \sigma_{max}, \\ \dot{\hat{u}}(t) & \text{if } \sigma_{min} \le \dot{\hat{u}}(t) \le \sigma_{max}, \\ \sigma_{min} & \text{if } \dot{\hat{u}}(t) < \sigma_{min}. \end{cases}$$

$\sigma_{\max}$ ——执行器速率最大值；

$\sigma_{\min}$ ——执行器速率最小值；

Δ ——时延因子；

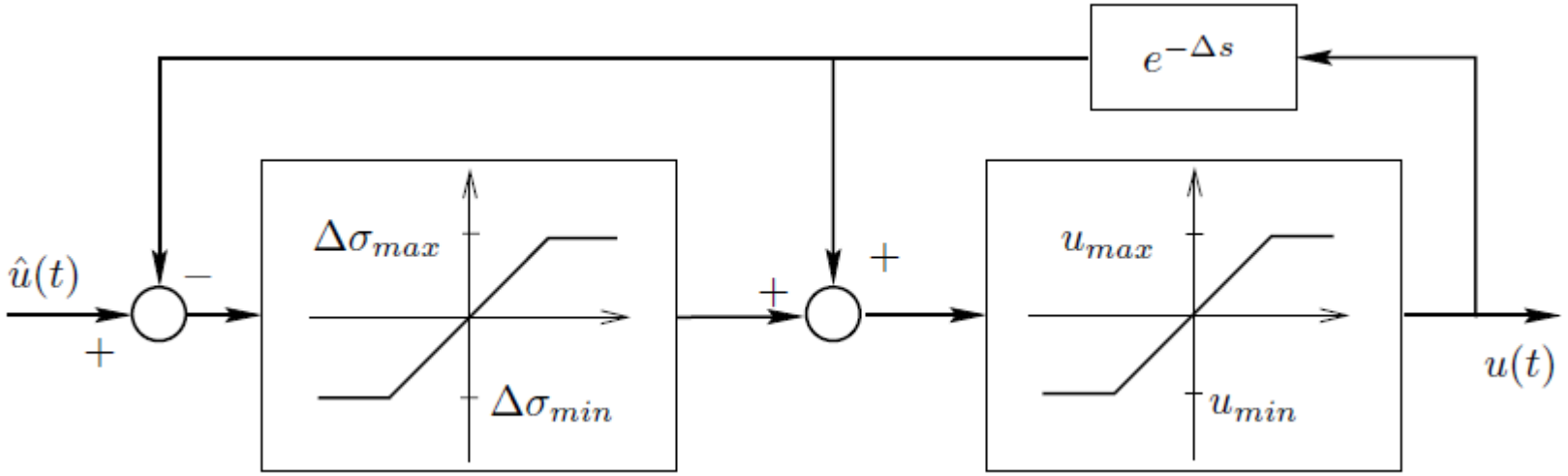
$\dot{u}(t)$ ——执行器实际输出速率；

$\dot{\hat{u}}(t)$ ——执行器期望输出速率；

$Sat\langle\bullet\rangle$ ——饱和函数。

5.1.2 执行器约束的描述方法

执行器同时具有饱和与转换速率限制的模型



u_{max}——执行器输出的最大幅值；

u_{min}——执行器输出的最小幅值；

$u(t)$——执行器实际输出；

$\hat{u}(t)$——执行器期望输出；

$Sat\langle\bullet\rangle$——饱和函数。

$\sigma_{\max}$——执行器速率最大值；

$\sigma_{\min}$——执行器速率最小值；

Δ——时延因子；

$\dot{u}(t)$——执行器实际输出速率；

$\dot{\hat{u}}(t)$——执行器期望输出速率；

$Sat\langle\bullet\rangle$——饱和函数。

5.1 执行器的约束问题

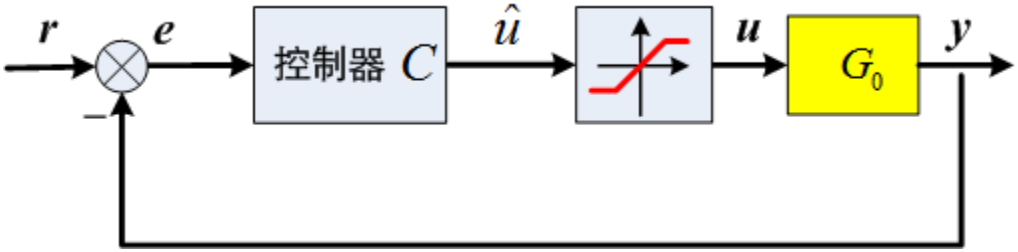
5.1.1 执行器约束问题的提出

5.1.2 执行器约束的描述方法

5.1.3 积分器的Windup问题

5.1.3 积分器的Windup问题

控制中经常使用积分器来消除系统的稳态误差，其前提是回路工作在线性范围内。当执行器达到其约束边界进入饱和后，在误差的作用下，积分器不断累积，控制器的输出 $\hat{u}$ 可能会累积到很大的值，但执行器由于处于饱和状态而无响应，即u仍受到饱和约束的限制。控制器的输出 $\hat{u}$ 回到饱和边界之内（线性范围）后，这一积分值作为初始条件，会导致很大的暂态响应。这一现象称为**Windup**，会严重影响系统的性能，甚至使系统失稳。



5.1.3 积分器的Windup问题

◆例2——积分器的Windup问题

被控对象

$$G_o(s)=\frac{2}{(s+1)(s+2)}$$

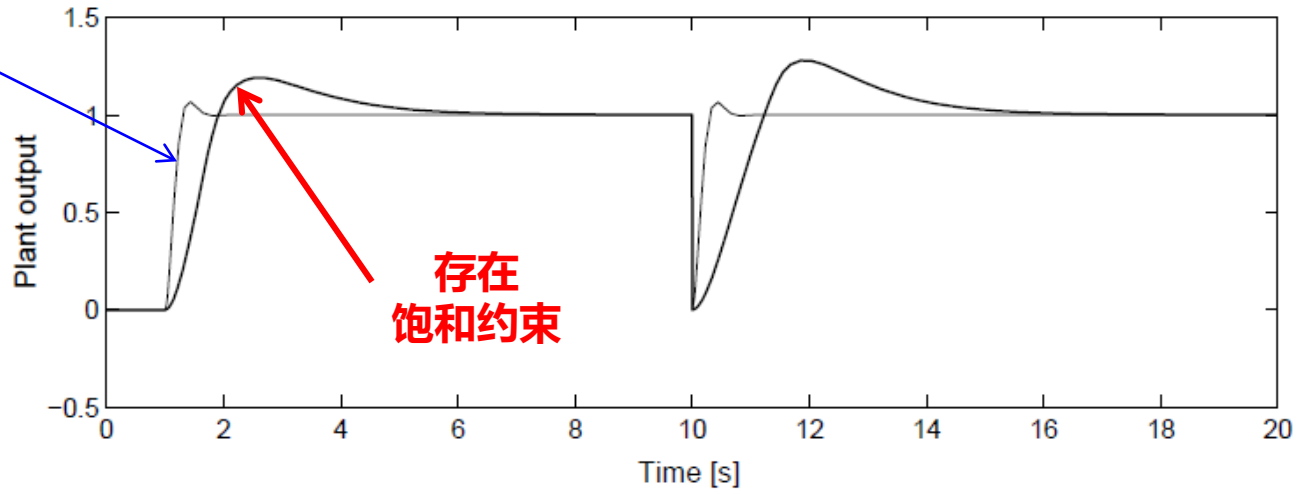
控制器

$$C(s)=\frac{50(s+1)(s+2)}{s(s+13)}$$

闭环传函

$$T_o(s)=\frac{100}{s^2+13s+100}$$

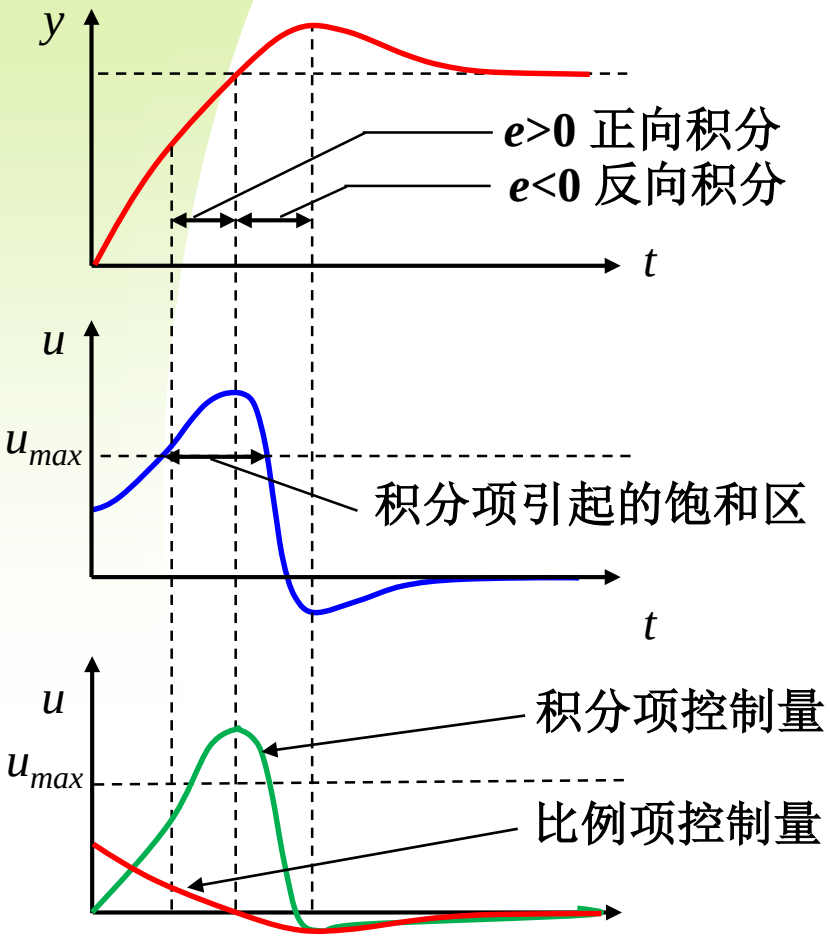
不存在
饱和约束



在1秒时施加单位阶跃输入信号，在10秒存在一个负向单位阶跃输出扰动信号。执行器的线性工作范围是[-3, 3] 。

5.1.3 积分器的Windup问题

积分器引起饱和的原因



- 当$e(t)>0$时，即积分器输入符号不变时，积分器的输出会持续增大，时间越长输出越大；
- 当$e(t)<0$时，即积分器误差符号改变时，由于积分器的初值较大，需要较长时间才能改变符号；
- 由于积分器的上述特性，积分器的存在很容易使系统进入饱和状态，使系统进入开环状态，失去对误差的调节能力

5.1.3 积分器的Windup问题

执行器约束的解决办法

（1）降低性能要求，保证执行器不会超出范围。

——对执行器要求过高，或导致性能不必要的下降。

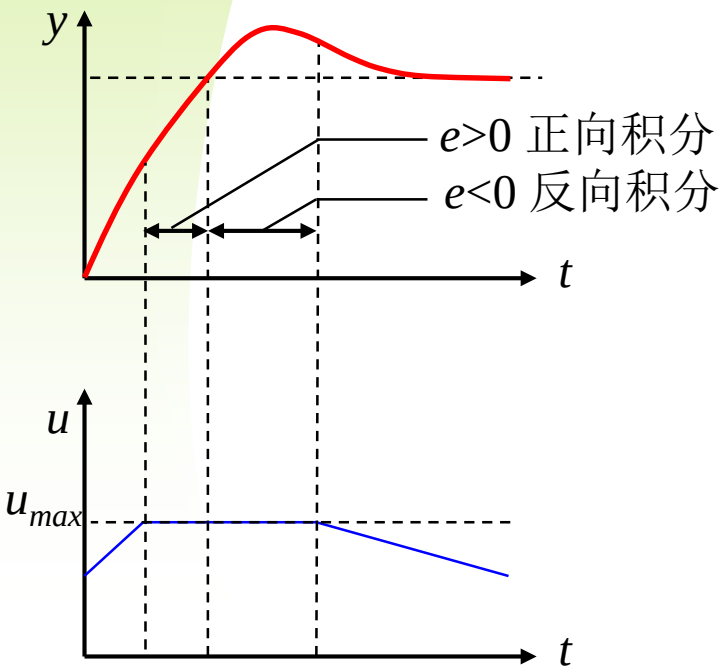
（2）修改设计，应对约束。

——对于一定程度（如100%）的超出约束，可获得满意的效果。对于存在更严重的超出约束的情况，可能是由于针对指定的性能指标执行器选型不当。

当执行器达到约束的边界时切断积分作用！

5.1.3 积分器的Windup问题

解决积分器饱和的方法



- 积分分离法
  - 方法：e>设定限值时，改用纯P调节
  - 作用：既不会积分饱和又能在小偏差时利用积分作用消除偏差
- 遇限削弱积分法
  - 方法：u_{PI}>设定限值时，只累加负偏差，反之亦然
  - 作用：可避免控制量长时间停留在饱和区

Thank You !

授课教师： 马 杰 （控制与仿真中心）

罗 晶 （控制科学与工程系）

马克茂（控制与仿真中心）

陈松林（控制与仿真中心）

第5章 抗饱和(Anti-Windup)设计

——2019年春季学期

授课教师： 马 杰 （控制与仿真中心）

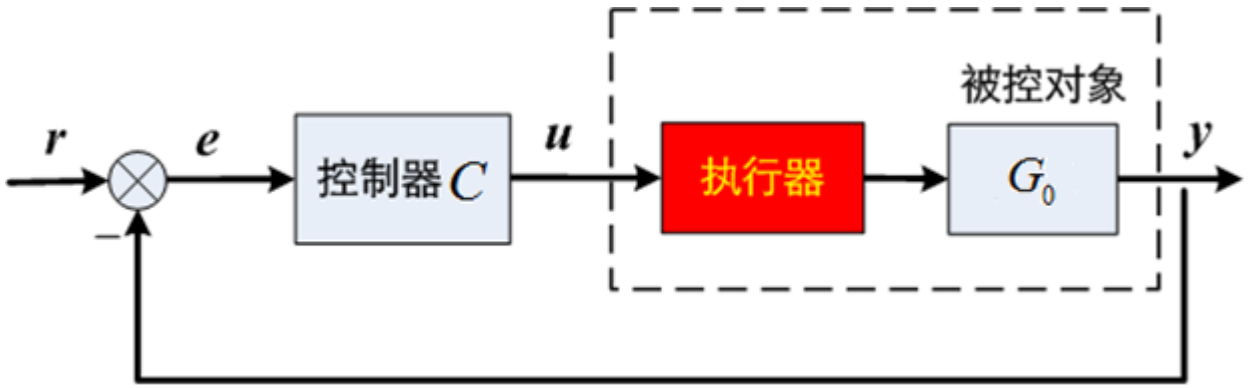
罗 晶 （控制科学与工程系）

马克茂（控制与仿真中心）

陈松林（控制与仿真中心）

执行器约束问题

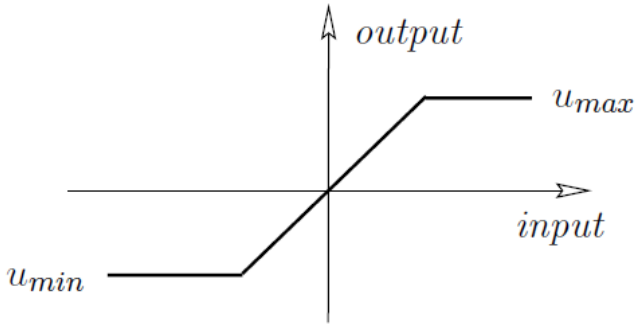
执行器存在一些典型的限制：电机最大转速限制、峰值力矩限制；阀门的开合不能超过全开或全闭等。



当控制量达到执行器的极限（幅值极限或变化速率极限）时，反馈回路失效，系统将运行于开环状态。

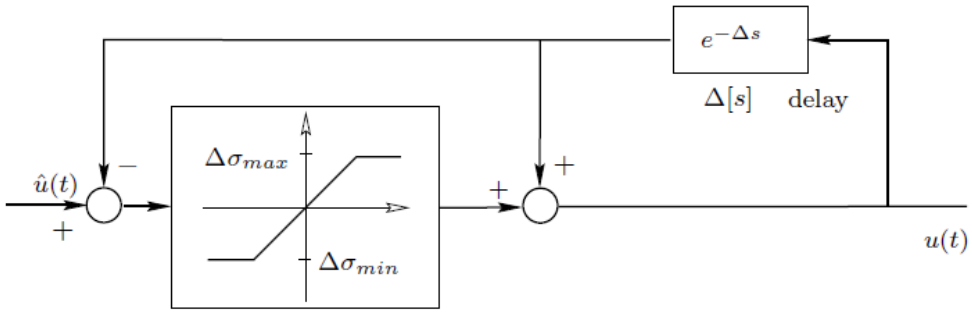
执行器约束模型

- 执行器饱和模型



$$u(t) = Sat\langle \hat{u}(t) \rangle \triangleq \begin{cases} u_{max} & \text{if } \hat{u}(t) > u_{max}, \\ \hat{u}(t) & \text{if } u_{min} \le \hat{u}(t) \le u_{max}, \\ u_{min} & \text{if } \hat{u}(t) < u_{min}. \end{cases}$$

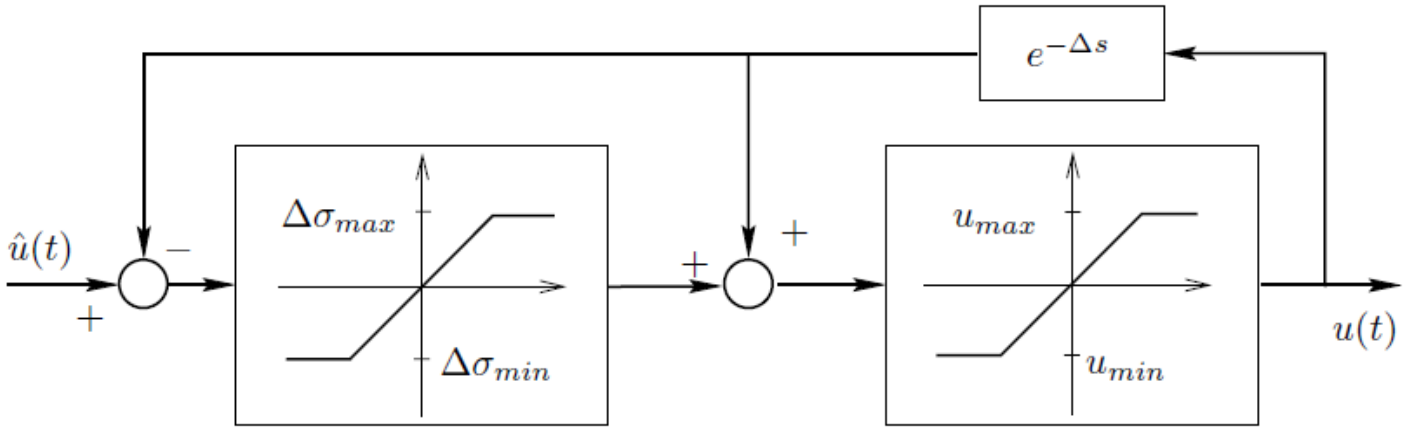
- 执行器转换速率限制模型



$$\dot{u}(t) = Sat\langle \dot{\hat{u}}(t) \rangle \triangleq \begin{cases} \sigma_{max} & \text{if } \dot{\hat{u}}(t) > \sigma_{max}, \\ \dot{\hat{u}}(t) & \text{if } \sigma_{min} \le \dot{\hat{u}}(t) \le \sigma_{max}, \\ \sigma_{min} & \text{if } \dot{\hat{u}}(t) < \sigma_{min}. \end{cases}$$

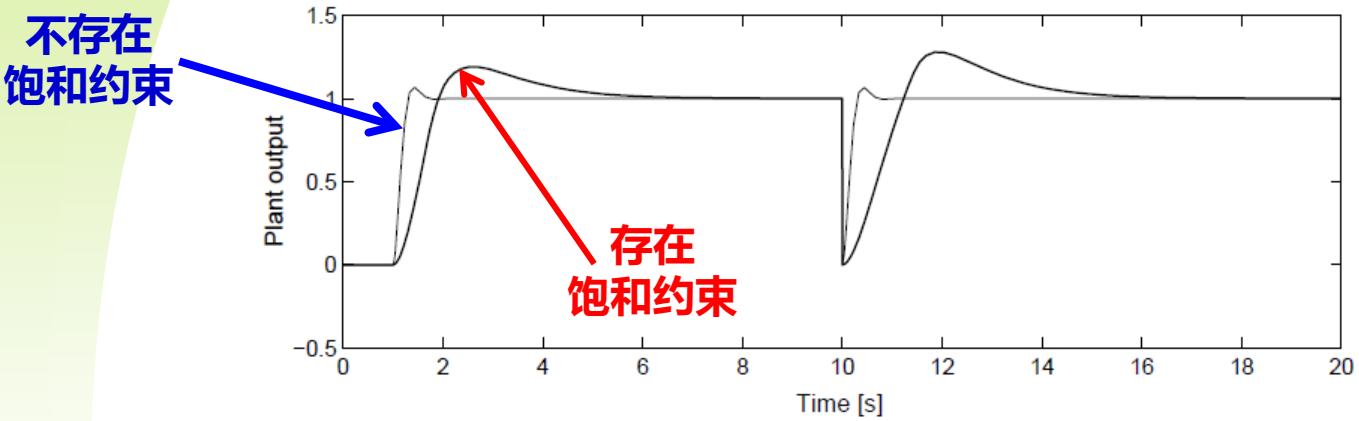
执行器约束模型

- 执行器同时具有饱和与转换速率限制的模型



$$u(t) = Sat\langle\hat{u}(t)\rangle \triangleq \begin{cases} u_{max} & \text{if } \hat{u}(t) > u_{max}, \\ \hat{u}(t) & \text{if } u_{min} \le \hat{u}(t) \le u_{max}, \\ u_{min} & \text{if } \hat{u}(t) < u_{min}. \end{cases} \qquad \dot{u}(t) = Sat\langle\dot{\hat{u}}(t)\rangle \triangleq \begin{cases} \sigma_{max} & \text{if } \dot{\hat{u}}(t) > \sigma_{max}, \\ \dot{\hat{u}}(t) & \text{if } \sigma_{min} \le \dot{\hat{u}}(t) \le \sigma_{max}, \\ \sigma_{min} & \text{if } \dot{\hat{u}}(t) < \sigma_{min}. \end{cases}$$

积分器的Windup问题



- **执行器约束的解决办法**

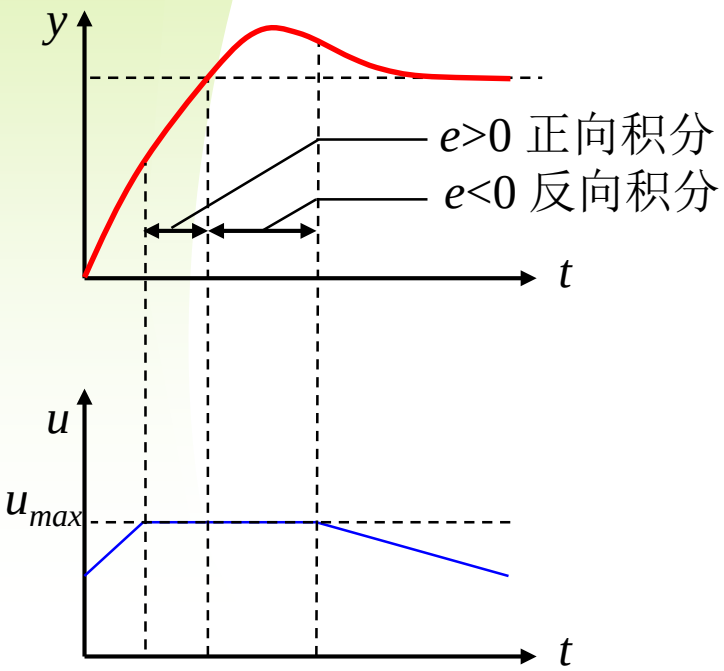
（1）保证执行器不会超出范围（做好选型）

（2）降低性能指标要求（降低增益，带宽，避免使用Ⅰ）

（3）修改设计，应对约束（避免系统进入深度饱和）

5.1.3 积分器的Windup问题

解决积分器饱和的方法



- 积分分离法
  - 方法：e>设定限值时，改用纯P调节
  - 作用：既不会积分饱和又能在小偏差时利用积分作用消除偏差
- 遇限削弱积分法
  - 方法：u_{PI}>设定限值时，只累加负偏差，反之亦然
  - 作用：可避免控制量长时间停留在饱和区

学习目标

本节课需要掌握的内容

- 掌握针对PI控制器的Anti-Windup控制设计方法
- 基本Anti-Windup控制设计的原理和方法;
- 熟悉其他形式Anti-Windup控制设计方法
- 理解上述方法中的基本思想

本章主要内容

A1 执行器约束问题

A2 Anti-Windup设计

A3 Anti-Windup控制器的一般形式

5.2 Anti-Windup设计

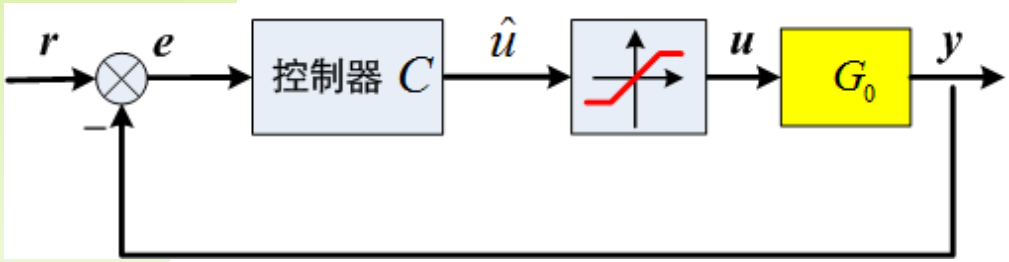
5.2.1 Anti-Windup设计策略

5.2.2 Anti-Windup设计原理分析

5.2.3 Anti-Windup的多种实现形式

5.2.1 Anti-Windup的设计策略

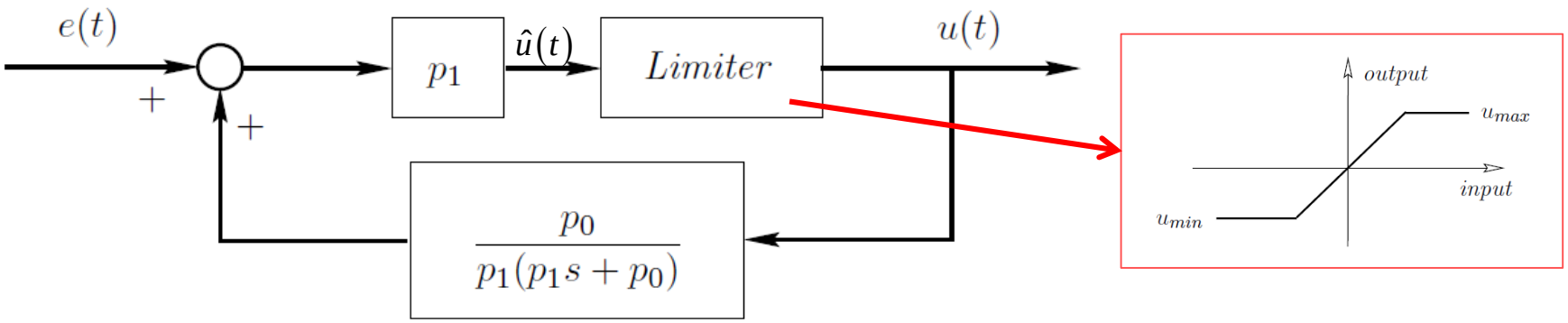
带有Anti-Windup的PI控制器



线性工作范围内的传递函数

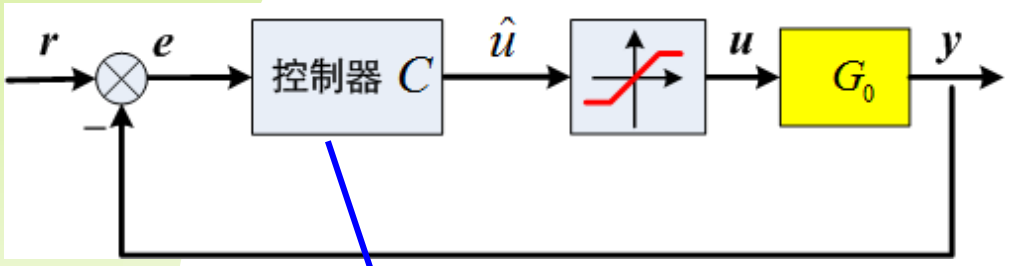
$$\frac{U(s)}{E(s)} = \frac{p_1 s + p_0}{s}$$

在线性工作范围内，实现比例+积分的作用，
而达到约束边界时，切除了积分作用。



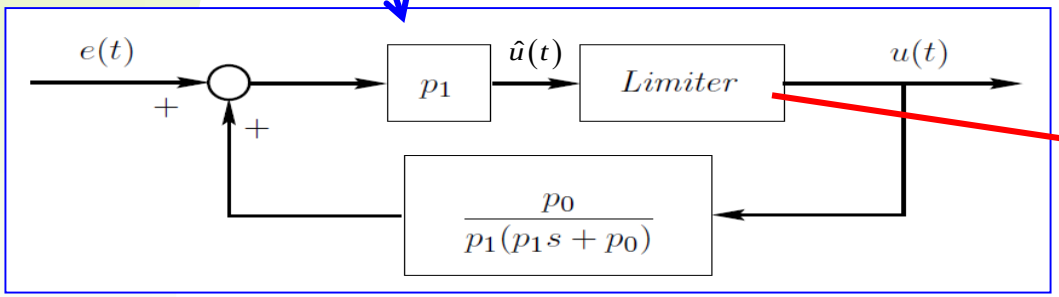
5.2.1 Anti-Windup的设计策略

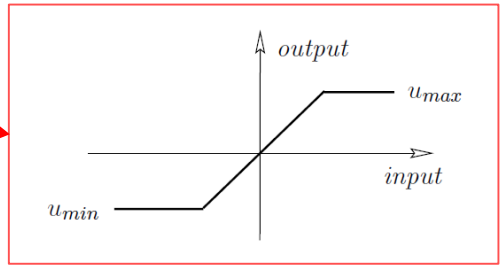
带有Anti-Windup的PI控制器



线性工作范围内的传递函数

$$\frac{U(s)}{E(s)} = \frac{p_1 s + p_0}{s}$$





重新推导*E*到*U*的传递函数

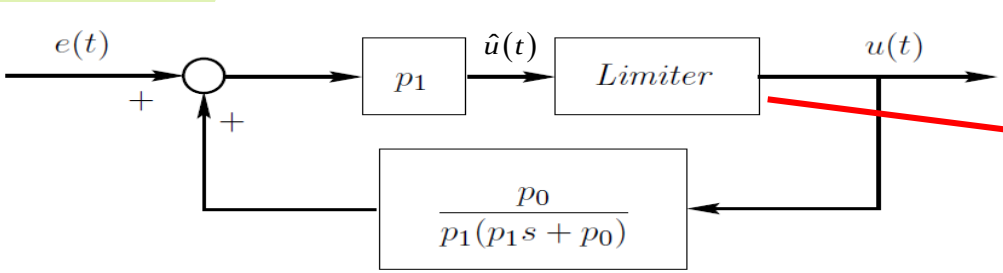
$$\left[e(t) + \frac{p_0}{p_1(p_1 s + p_0)} u(t) \right] p_1 = \hat{u}(t)$$

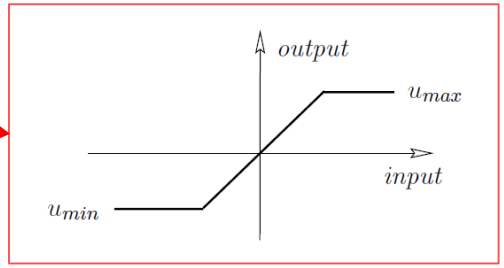
$$p_1 e(t) = \hat{u}(t) - \frac{p_1 p_0}{p_1(p_1 s + p_0)} u(t)$$

5.2.1 Anti-Windup的设计策略

带有Anti-Windup的PI控制器

$$\frac{U(s)}{E(s)}=\frac{p_1 s+p_0}{s}$$





$$p_1 e(t)=\hat{u}(t)-\frac{p_1 p_0}{p_1(p_1 s+p_0)}u(t)$$

线性区 $u(t)=\hat{u}(t)$ if $u_{\min}\le\hat{u}(t)\le u_{\max}$

$$u(t)=Sat\langle\hat{u}(t)\rangle\triangleq\begin{cases}u_{max} & \text{if } \hat{u}(t)>u_{max},\\ \hat{u}(t) & \text{if } u_{min}\le\hat{u}(t)\le u_{max},\\ u_{min} & \text{if } \hat{u}(t)<u_{min}.\end{cases}$$

$$p_1 e(t)=\frac{p_1 p_1 s}{p_1(p_1 s+p_0)}u(t)$$

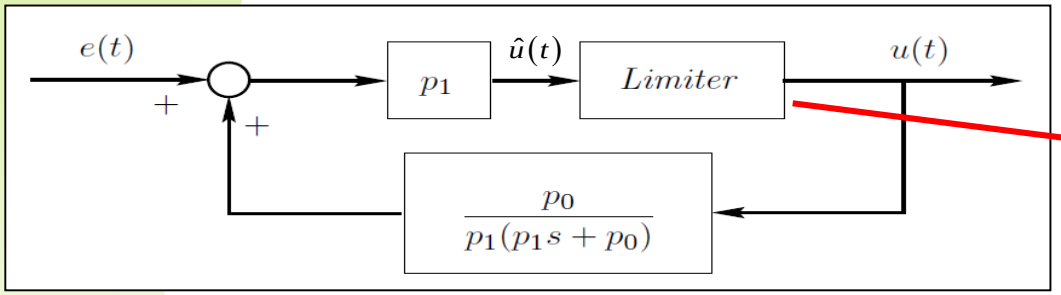
$$\frac{U(s)}{E(s)}=\frac{p_1 s+p_0}{s}$$

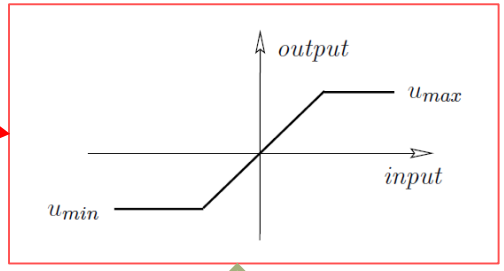
仍然是PI控制器

5.2.1 Anti-Windup的设计策略

带有Anti-Windup的PI控制器

$$\frac{U(s)}{E(s)}=\frac{p_1 s+p_0}{s}$$





$$p_1 e(t)=\hat{u}(t)-\frac{p_1 p_0}{p_1(p_1 s+p_0)}u(t)$$

非线性区 $u(t)=u_{\max}$ if $\hat{u}(t)\geq u_{\max}$

$$p_1 e(t)+\frac{p_1 p_0}{p_1(p_1 s+p_0)}u_{\max}=\hat{u}(t)$$

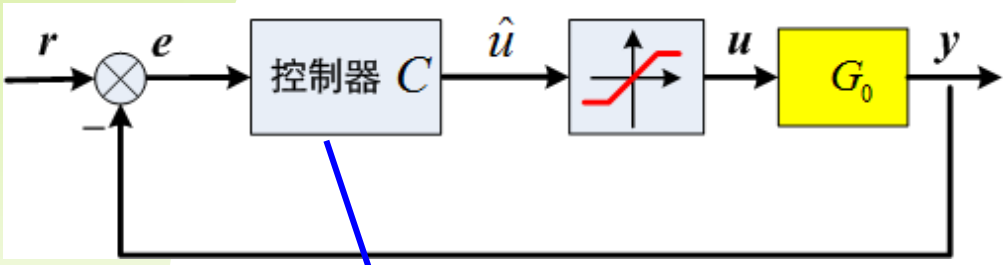
$$\hat{u}(t)=p_1 e(t)+u_{\max}$$

$$u(t)=Sat\langle\hat{u}(t)\rangle\triangleq\begin{cases}u_{max} & \text{if } \hat{u}(t)>u_{max},\\ \hat{u}(t) & \text{if } u_{min}\leq\hat{u}(t)\leq u_{max},\\ u_{min} & \text{if } \hat{u}(t)<u_{min}.\end{cases}$$

控制量为误差的比例项加一个常值。此时，积分器停止工作，不会出现深度饱和现象

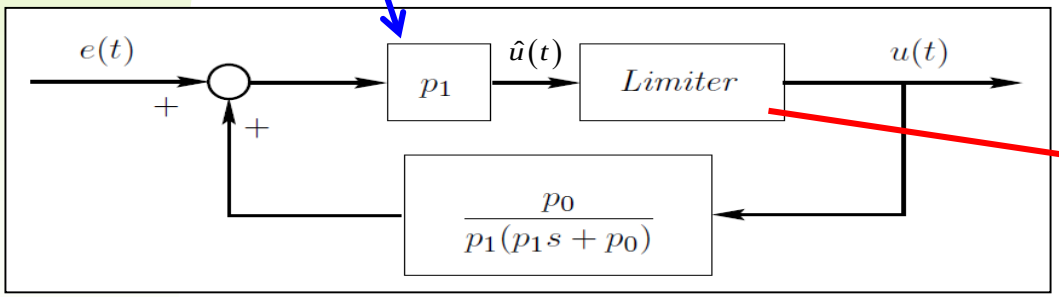
5.2.1 Anti-Windup的设计策略

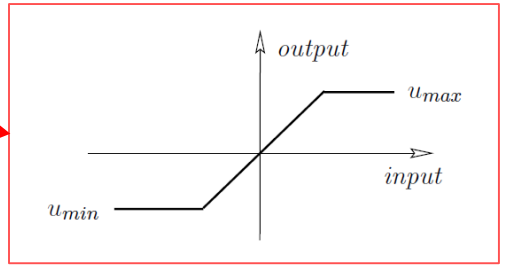
带有Anti-Windup的PI控制器



线性工作范围内的传递函数

$$\frac{U(s)}{E(s)} = \frac{p_1 s + p_0}{s}$$





饱和并不一定都是积分引起的，而且很多控制器中并不一定存在积分项，对于一般的控制器，如何进行Anti-Windup设计？

5.2.1 Anti-Windup的设计策略

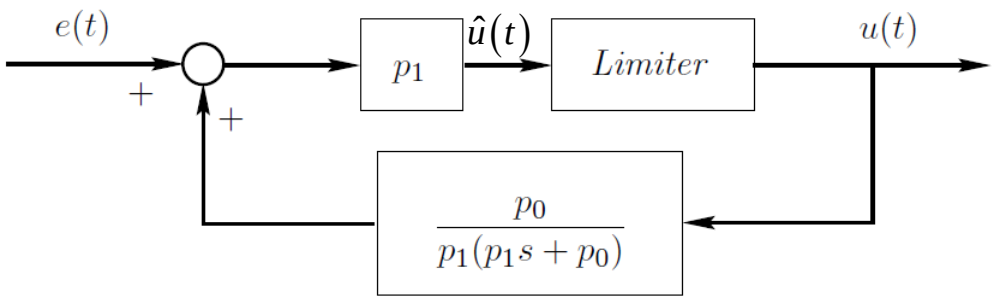
带有Anti-Windup的PI控制器

——总结Anti-Windup设计策略

$$\frac{U(s)}{E(s)} = \frac{p_1 s + p_0}{s}$$

（1）控制器的动态由对象的实际输入信号（执行器的实际输出）来驱动；

（2）由对象的实际输入信号驱动时，控制器的动态是稳定的。



执行器达到约束边界时，控制器切除了积分作用！

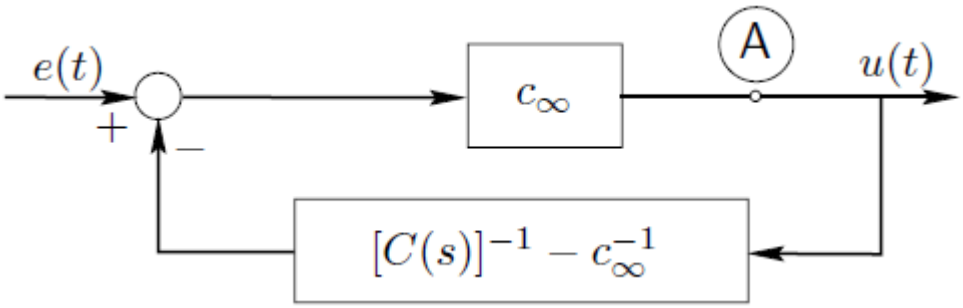
5.2.1 Anti-Windup的设计策略

Anti-Windup的基本结构

假设控制器是双正则的最小相位系统，将其分解为比例项和严格正则项：

$$C(s) = c_\infty + \bar{C}(s)$$

$$\begin{aligned}\frac{U(s)}{E(s)} &= \frac{c_\infty}{1 + ([C(s)]^{-1} - c_\infty^{-1})c_\infty} \\ &= \frac{c_\infty}{[C(s)]^{-1}c_\infty} \\ &= C(s)\end{aligned}$$



严格正则的最小相位控制器可通过适当添加远离虚轴的最小相位零点变为双正则的形式。

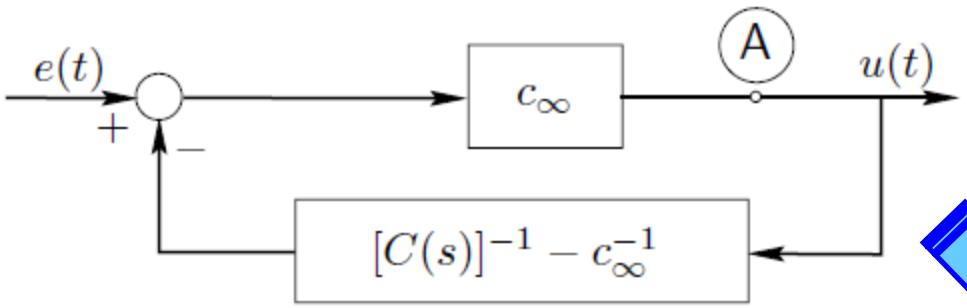
设计策略：

（1）控制器的动态由对象的实际输入信号来驱动；

（2）由对象的实际输入信号驱动时，控制器的动态是稳定的。

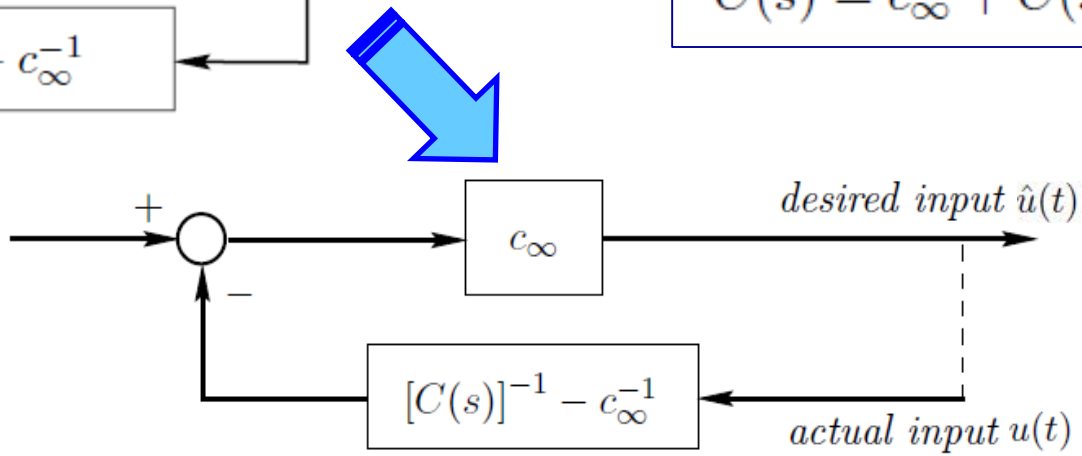
5.2.1 Anti-Windup的设计策略

Anti-Windup的基本结构



双正则控制器：

$$C(s) = c_\infty + \bar{C}(s)$$



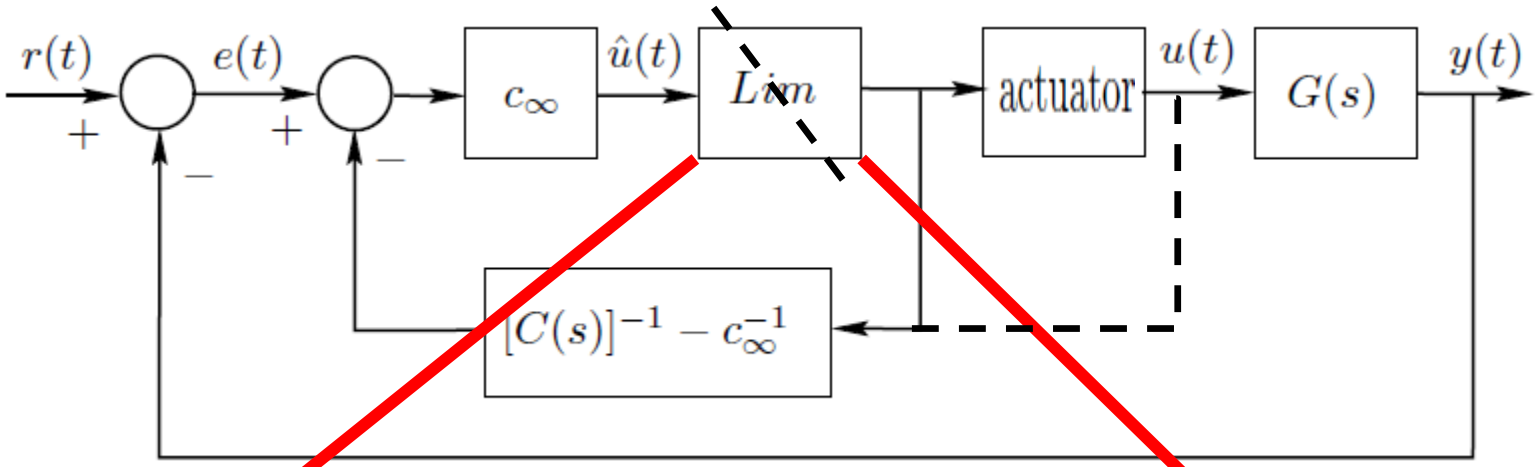
设计策略：

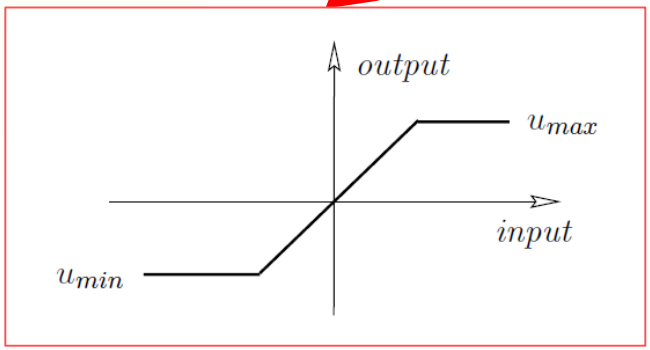
（1）控制器的动态由对象的**实际输入信号**来驱动；

（2）由对象的实际输入信号驱动时，控制器的**动态是稳定的**。

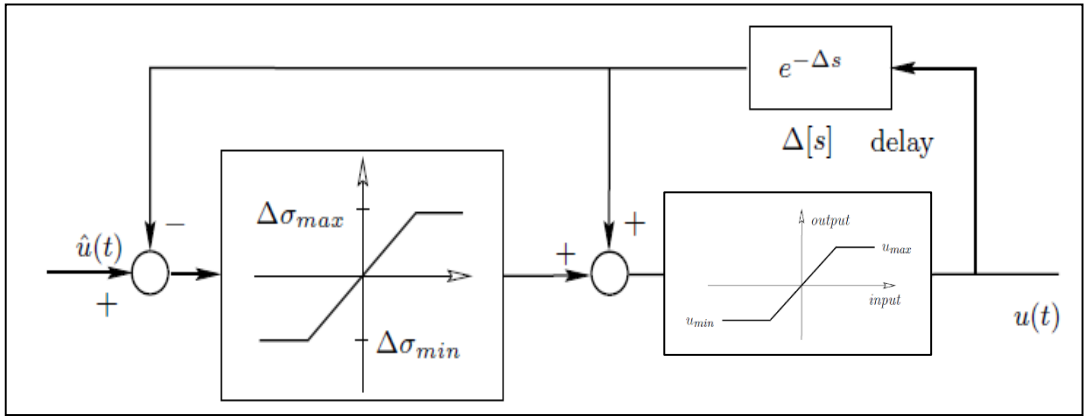
5.2.1 Anti-Windup的设计策略

Anti-Windup控制回路——含有执行器约束





or



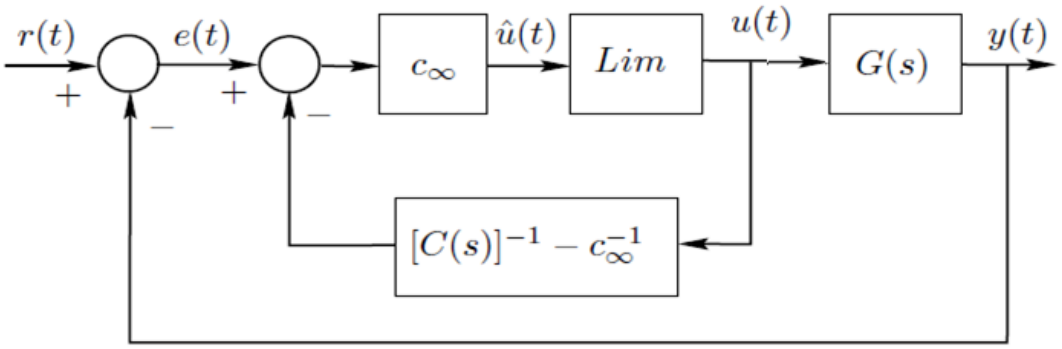
5.2.1 Anti-Windup的设计策略

◆ 例2 Anti-Windup设计（含有执行器幅值约束）

$$G_o(s) = \frac{2}{(s+1)(s+2)}$$

$$C(s) = \frac{50(s+1)(s+2)}{s(s+13)}$$

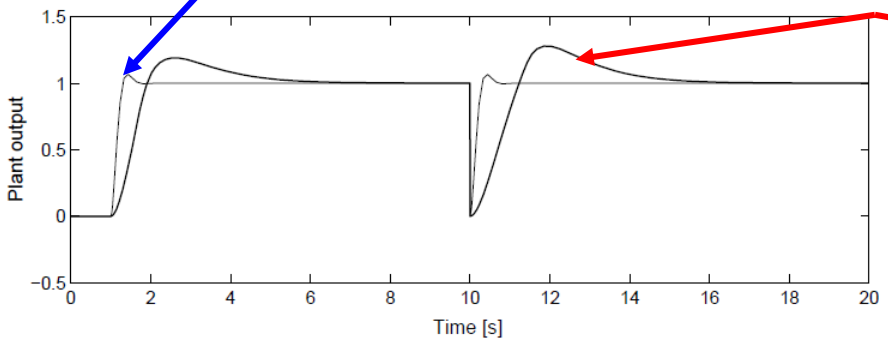
$$T_o(s) = \frac{100}{s^2+13s+100}$$

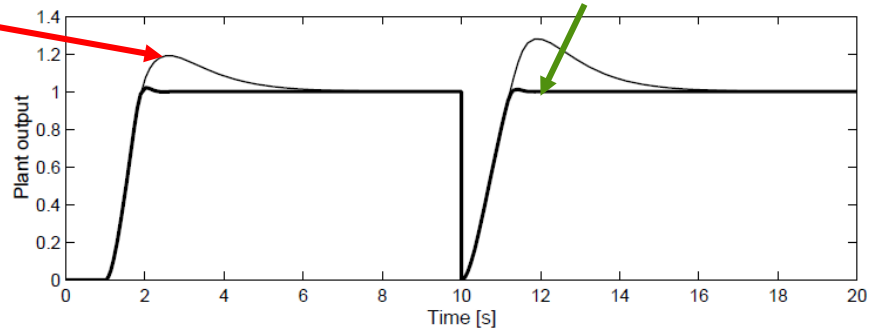


不存在饱和约束

存在饱和约束
非Anti-Windup设计

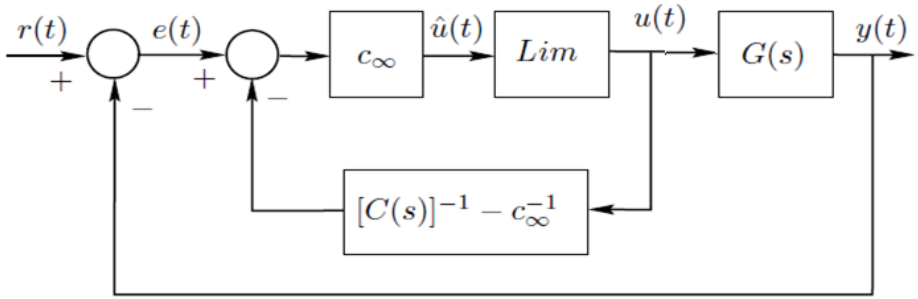
存在饱和约束
Anti-Windup设计

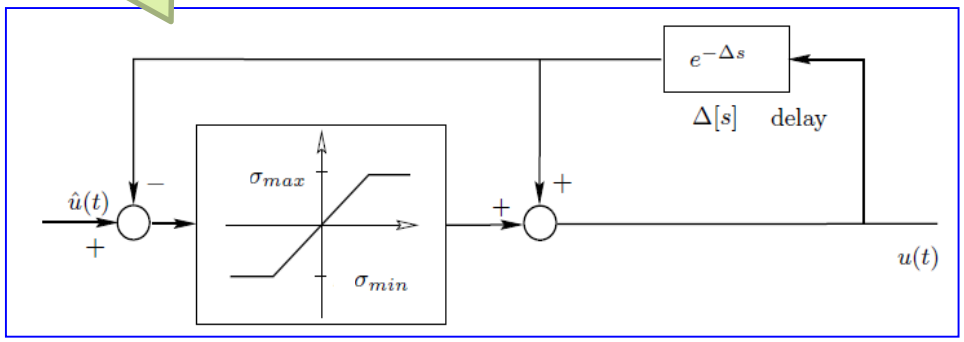




5.2.1 Anti-Windup的设计策略

◆ 例3 Anti-Windup设计(含有执行器转换速率约束)





对象模型：

$$Y(s) = e^{-s}\left(\frac{1}{(s+1)^2}U(s) + D_g(s)\right)$$

$$\begin{cases} C_{PI}(s) = K_p\left(1 + \dfrac{1}{T_r s}\right) \\ K_P = 0.5 \\ T_r = 1.5[s] \end{cases}$$

假设执行器的转换速率小于等于$0.2s^{-1}$。

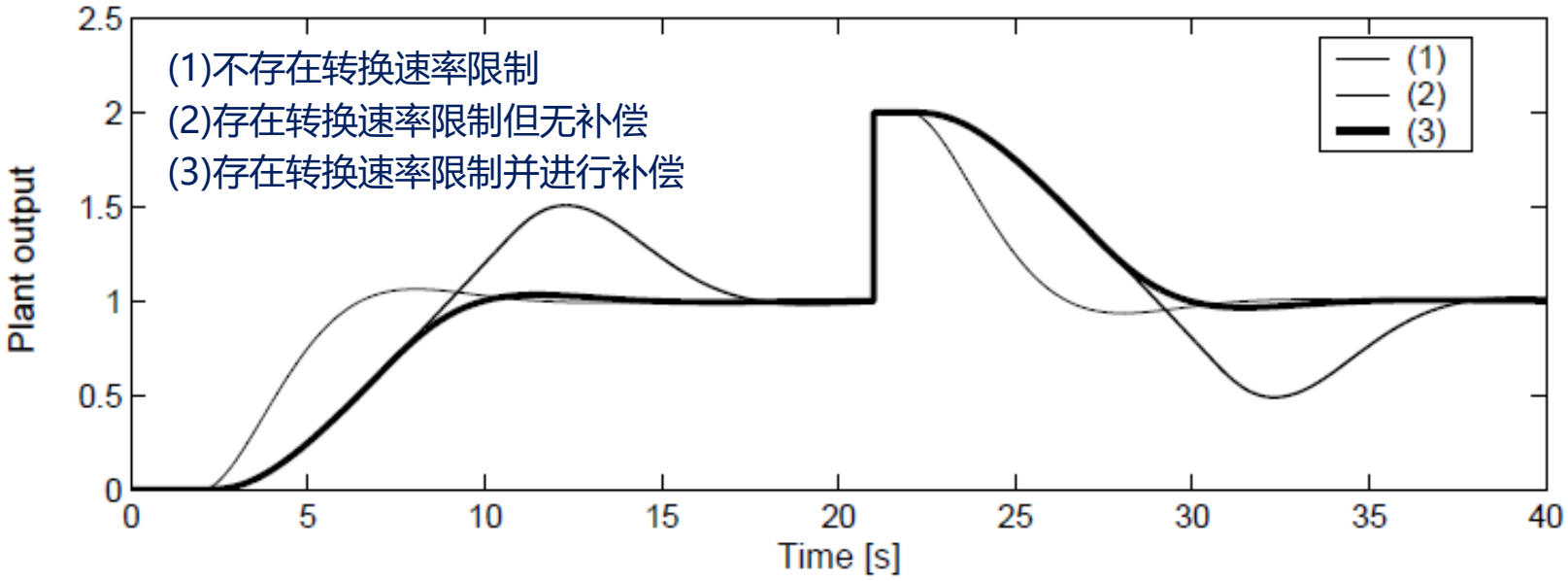
$$\begin{cases} c_\infty = K_p = 0.5 \\ [C(s)]^{-1} - c_\infty^{-1} = -\dfrac{1}{K_p(T_r s+1)} = -\dfrac{2}{(1.5s+1)} \end{cases}$$

5.2.1 Anti-Windup的设计策略

◆ 例3 Anti-Windup设计(含有执行器转换速率约束)

$$Y(s) = e^{-s}\left(\frac{1}{(s+1)^2}U(s) + D_g(s)\right) \qquad c_\infty = K_p = 0.5 \qquad [C(s)]^{-1} - c_\infty^{-1} = -\frac{2}{(1.5s+1)}$$

在1s时输入单位阶跃参考信号，在20s时加入输出端单位阶跃扰动输入。



5.2 Anti-Windup设计

5.2.1 Anti-Windup设计策略

5.2.2 Anti-Windup设计原理分析

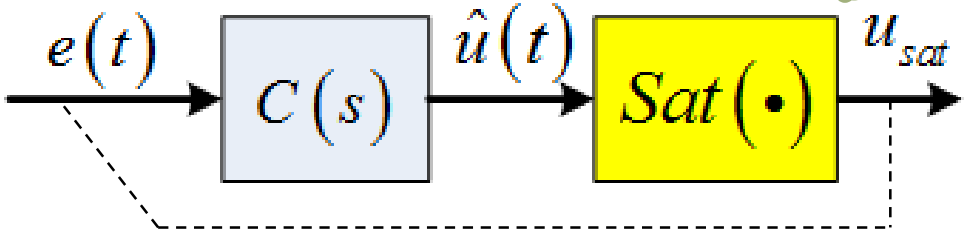
5.2.3 Anti-Windup的多种实现形式

5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路

如何构造出不进入饱和的期望控制 $\hat{u}(t)$?

考虑执行器饱和的控制器如下：



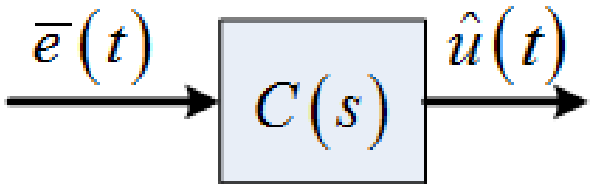
控制器的输入信号$e(t)$过大会导致执行器饱和，Ainti-Windup的思想是基于控制器输入信号$e(t)$，找到一个期望的控制器输入信号$\bar{e}(t)$，使得期望的控制量$\hat{u}(t)$不会超出实际控制量的限幅u_{sat}。

Anti-Windup设计思路

假设控制器是最小相位、双正则的，将其分解为比例项和严格正则项： $C(s)=C_\infty+\bar{C}(s)$

期望的控制量（执行器的期望输入）表示为

$$\hat{u}(t)=C\bar{e}=\bar{C}\bar{e}+C_\infty\bar{e}$$



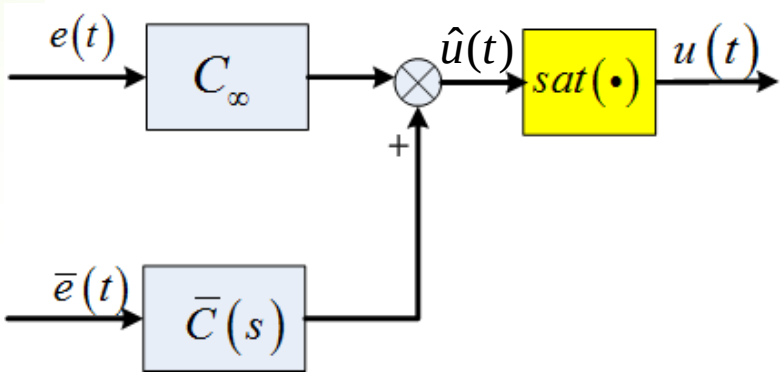
5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路

对实际控制量 $u(t)$ （执行器实际输入）做如下修改：

$$u(t)=\bar{C}\,\bar{e}+C_{\infty}\,e$$

当执行器饱和时，由 $\bar{e}(t)$ 形成期望的误差转化。



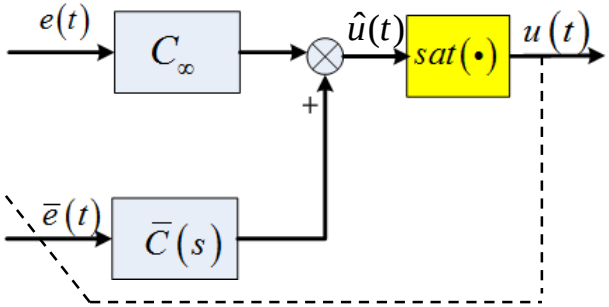
$$u(t)=sat\left(\bar{C}\,\bar{e}+C_{\infty}\,e\right)$$

5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路

假设在达到饱和的瞬间，通过及时修改误差信号可以避免饱和，且保证之后任何时刻均有$\hat{u}(t)=u(t)$，则应保证：

$$\begin{cases}\hat{u}(t)=\bar{C}\bar{e}+C_\infty\bar{e}\\ u(t)=sat\left(\bar{C}\bar{e}+C_\infty e\right)\end{cases}$$



$$\bar{C}\bar{e}+C_\infty\bar{e}=sat\left(\bar{C}\bar{e}+C_\infty e\right)$$

执行器线性工作范围内：

$$\bar{e}(t)=e(t)$$

$$\bar{e}=C_\infty^{-1}\left[Sat\left(\bar{C}\bar{e}+C_\infty e\right)-\bar{C}\bar{e}\right]$$

问题：$e(t)\to\bar{e}(t)$

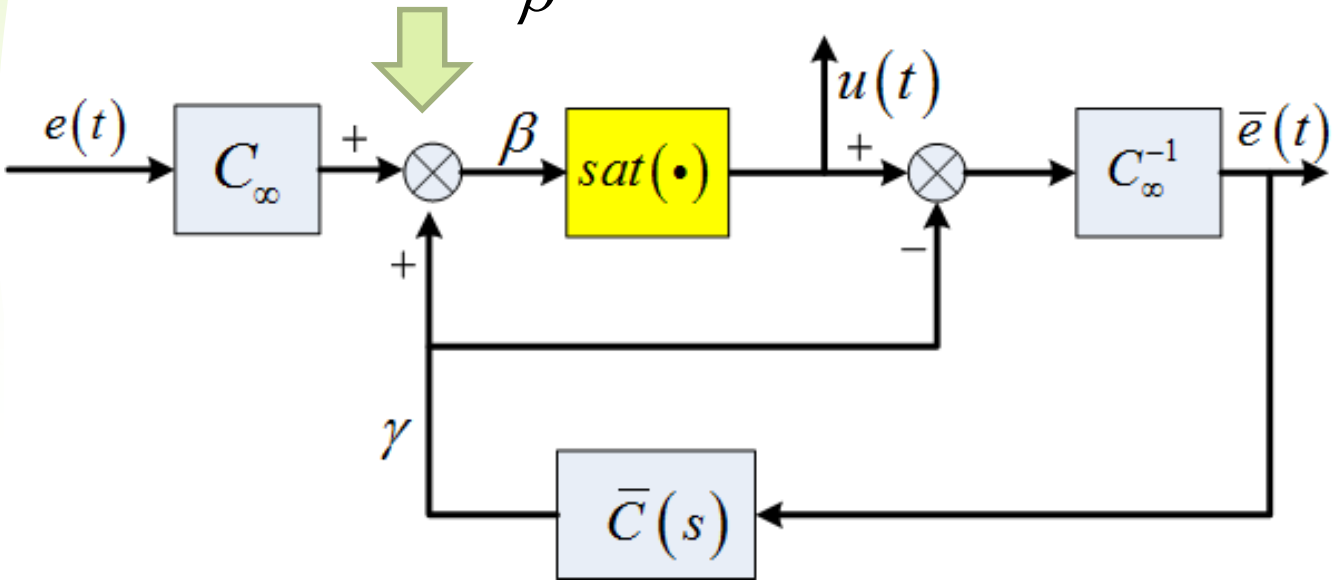
需要满足什么条件？

5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路

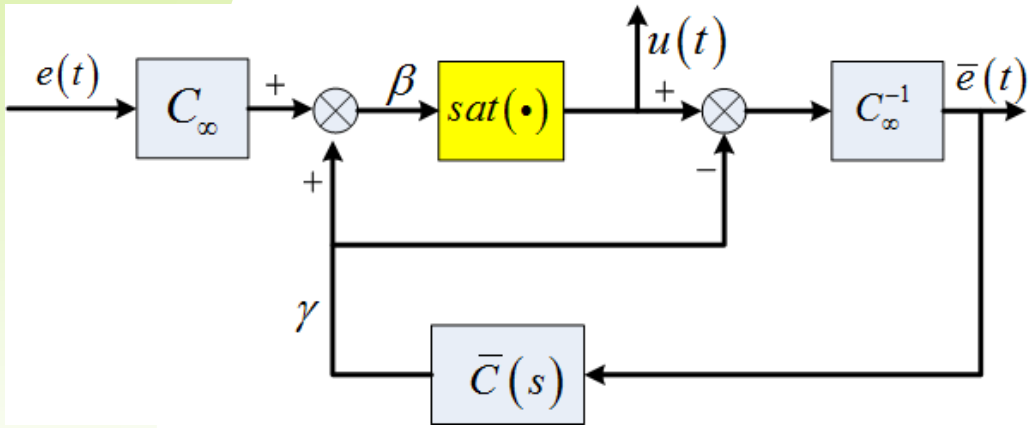
$$\bar{e} = C_\infty^{-1}\left[Sat\left(\underbrace{\bar{C}\bar{e} + C_\infty e}_{\beta}\right) - \bar{C}\bar{e}\right]$$

问题：$e(t) \to \bar{e}(t)$ 需要满足什么条件?



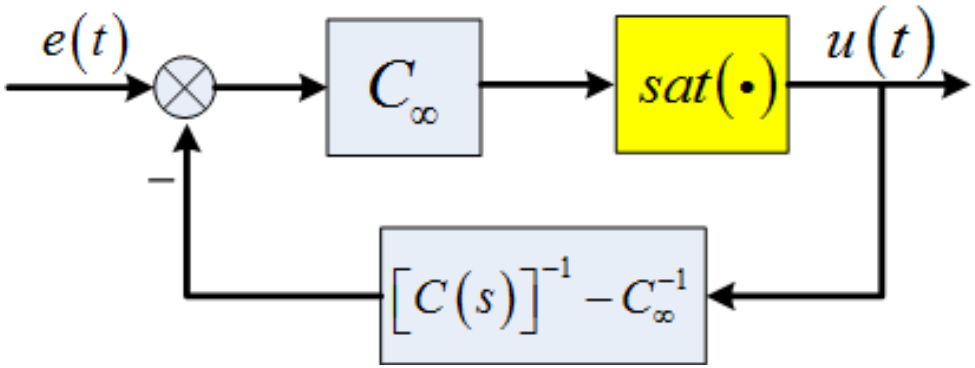
5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路



等价吗?

$$\bar{e} = C_\infty^{-1}\left[Sat\left(\bar{C}\bar{e} + C_\infty e\right) - \bar{C}\bar{e}\right]$$



推导 u 跟 e 之间的关系，来判断两种结构间的等价关系

5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路

$$\gamma(t)=C_\infty^{-1}\bar{C}\left[u(t)-\gamma(t)\right]$$

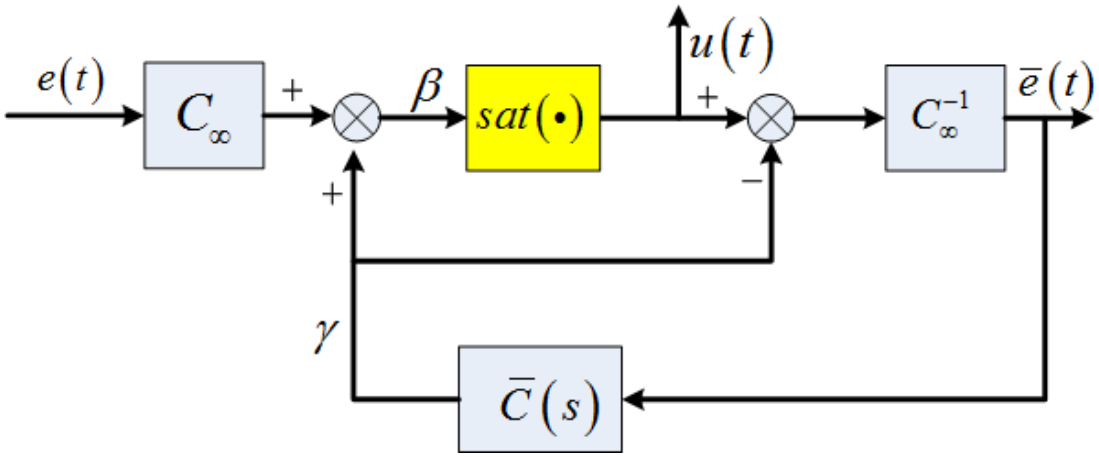
$$C_\infty\gamma(t)=\bar{C}u(t)-\bar{C}\gamma(t)$$

$$\left(C_\infty+\bar{C}\right)\gamma=\bar{C}u(t)$$

$$C\gamma(t)=\left(C-C_\infty\right)u(t)$$

$$\gamma(t)=-C^{-1}C_\infty u(t)+C^{-1}Cu(t)$$

$$\gamma(t)=-C_\infty\left[C^{-1}-C_\infty^{-1}\right]u(t)$$



$$\beta(t)=C_\infty e(t)+\gamma(t)$$

$$=C_\infty e(t)-C_\infty\left[C^{-1}-C_\infty^{-1}\right]u(t)$$

$$\beta(t)=C_\infty\left[e(t)-\left(C^{-1}-C_\infty^{-1}\right)u(t)\right]$$

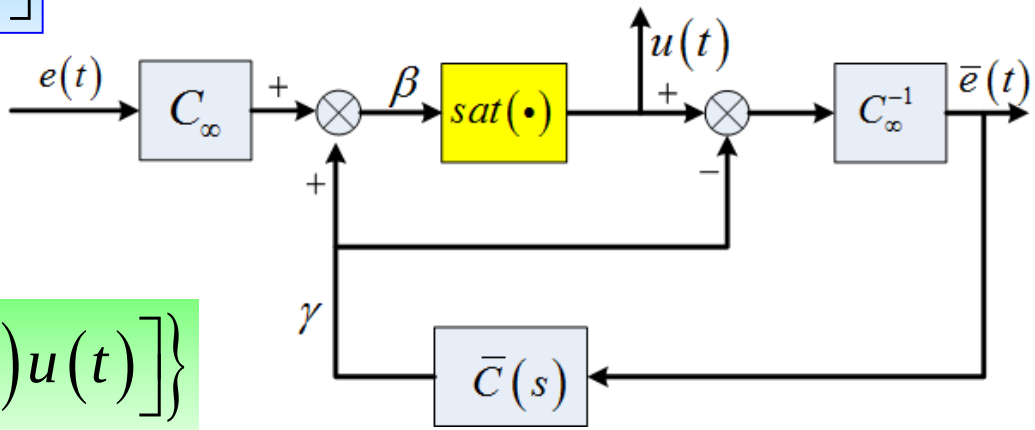
5.2.2 Anti-Windup设计原理分析

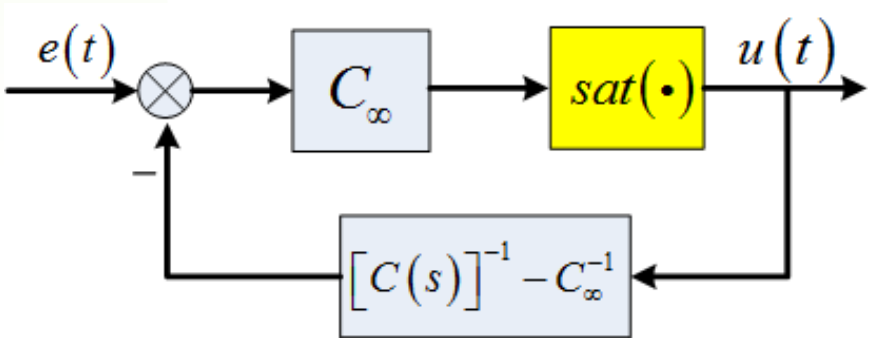
Anti-Windup设计思路

$$\beta(t)=C_\infty\left[e(t)-\left(C^{-1}-C_\infty^{-1}\right)u(t)\right]$$

$$u(t)=sat(\beta)$$

$$u(t)=sat\left\{C_\infty\left[e(t)-\left(C^{-1}-C_\infty^{-1}\right)u(t)\right]\right\}$$

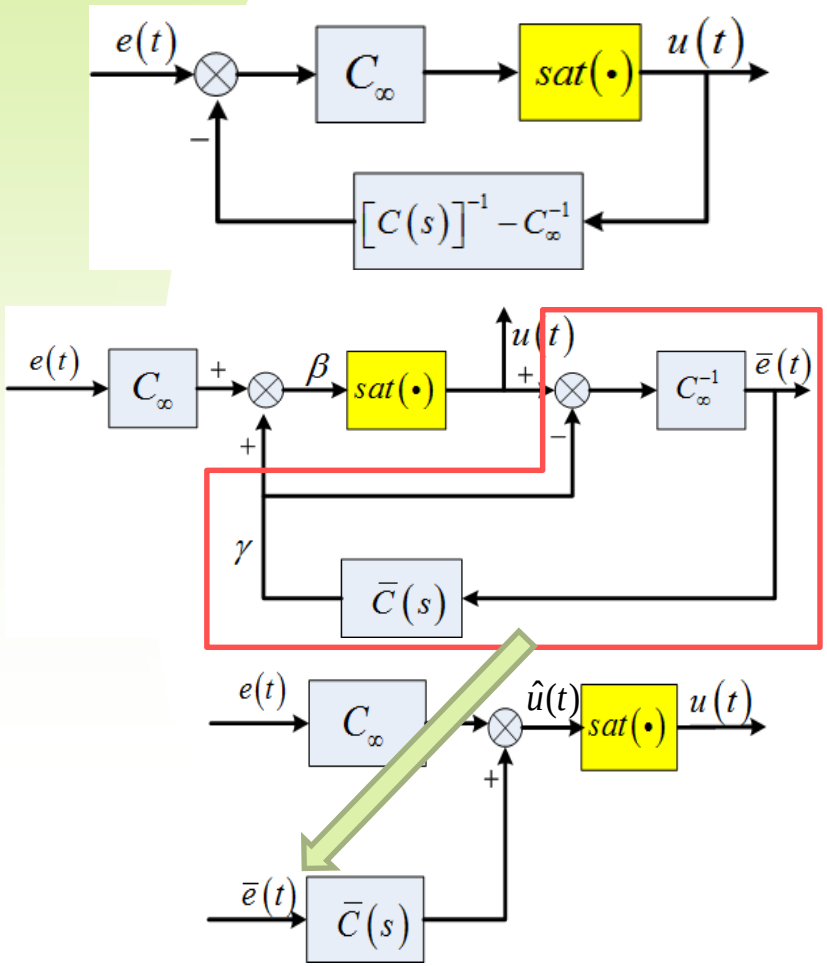




等价!

5.2.2 Anti-Windup设计原理分析

Anti-Windup设计思路



Anti-Windup 设计，从原理上可以理解为，**用实际的控制量来调节控制器的输入**，也就是误差$e(t)$，使其能够始终保证$\hat{u}(t)=u(t)$，这样能够有效避免系统进入深度饱和，而且在误差变号后，快速退出饱和。

5.2 Anti-Windup设计

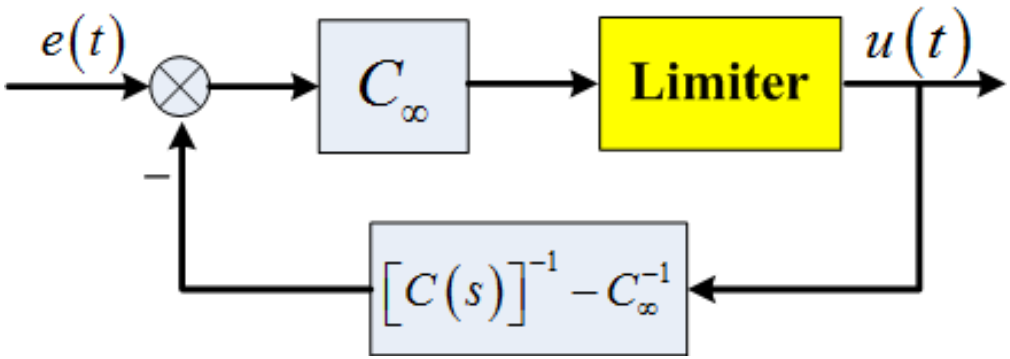
5.2.1 Anti-Windup设计策略

5.2.2 Anti-Windup的设计原理分析

5.2.3 Anti-Windup的多种实现形式

5.2.3 Anti-Windup的多种实现形式

Anti-Windup设计得到了广泛研究，已有多种关于抗饱和的设计方法，这里只列举两种简单的其他形式的Anti-Windup控制器。

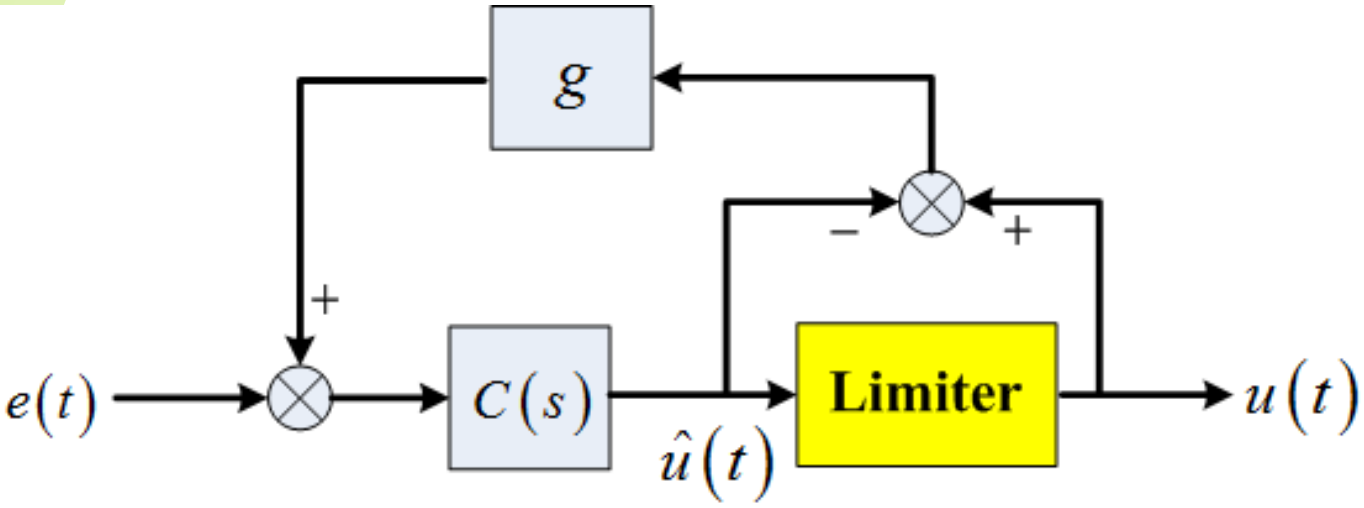


Anti-Windup 控制器基本结构

适用条件：控制器双正则，最小相位

5.2.3 Anti-Windup的多种实现形式

Anti-Windup的其他形式——第一种



Anti-Windup 控制器控制器的一般形式——第一种

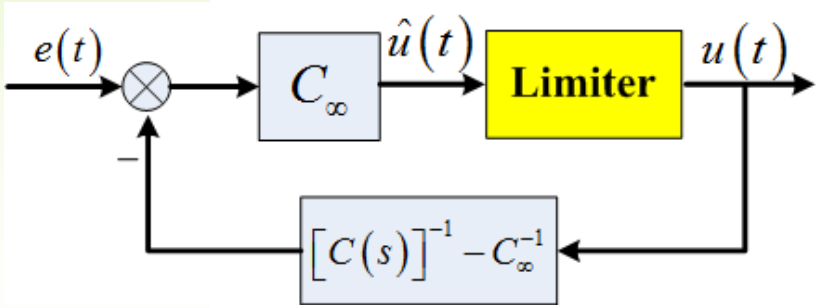
g：静态增益

控制器无双正则、最小相位需求

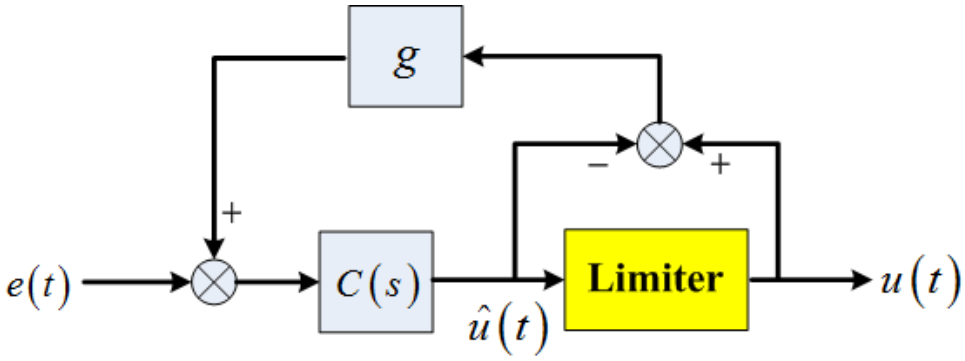
5.2.3 Anti-Windup的多种实现形式

◆ 例4——两种Anti-Windup控制器对比分析

考虑积分型控制器 $C(s)=\dfrac{k}{s}$



(a) Anti-Windup基本结构



(b) Anti-Windup一般形式-第一种

5.2.3 Anti-Windup的多种实现形式

◆ 例4——两种Anti-Windup控制器对比分析

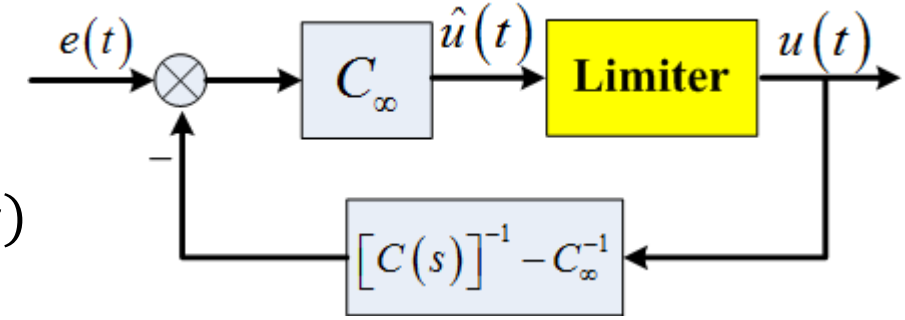
考虑积分型控制器 $C(s)=\dfrac{k}{s}$

通过添加一个远离虚轴、稳定的零点$\varepsilon s+1$，可以变为近似等价的双正则最小相位的形式： $C(s)=\dfrac{k(\varepsilon s+1)}{s}$

$$\hat{u}(t)=C_\infty\left[e(t)-\left(C(s)^{-1}-C_\infty^{-1}\right)u(t)\right]$$

$$=u(t)+C_\infty e(t)-C_\infty C(s)^{-1}u(t)$$

$$=u(t)+\left[k\varepsilon e(t)-\frac{\varepsilon s}{\varepsilon s+1}u(t)\right]$$



(a) Anti-Windup基本结构

5.2.3 Anti-Windup的多种实现形式

◆ 例4——两种Anti-Windup控制器对比分析

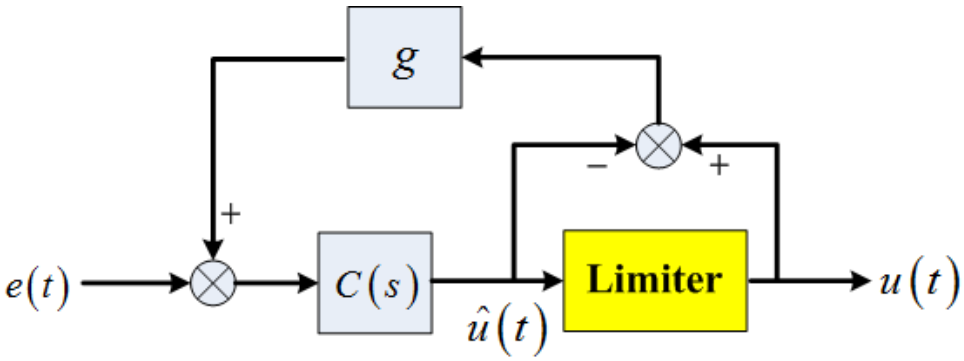
考虑积分型控制器 $C(s)=\dfrac{k}{s}$

$$\hat{u}(t)=C(s)\left\{e(t)+g\left[u(t)-\hat{u}(t)\right]\right\}=C(s)e(t)+C(s)g\,u(t)-C(s)g\,\hat{u}(t)$$

$$\hat{u}(t)+\frac{gk}{s}\hat{u}(t)=\frac{k}{s}e(t)+\frac{gk}{s}u(t)$$

$$\hat{u}(t)=\frac{gk}{s+gk}u(t)+\frac{k}{s+gk}e(t)$$

$$\hat{u}(t)=u(t)+\left[\frac{k}{s+gk}e(t)-\frac{s}{s+gk}u(t)\right]$$

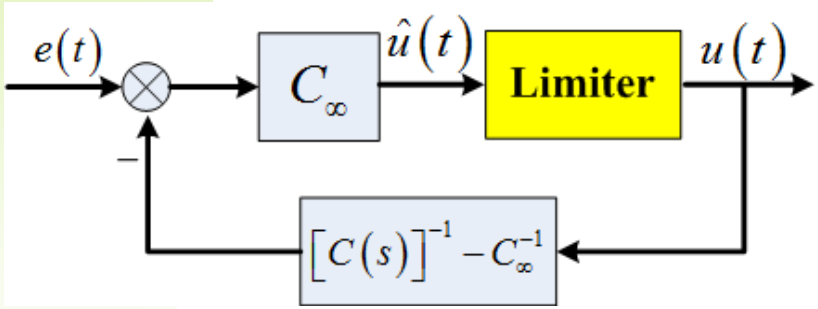


(b) Anti-Windup一般形式—第一种

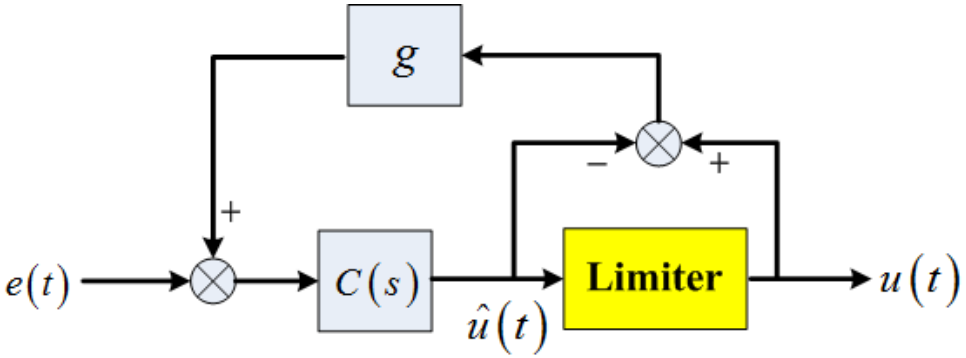
5.2.3 Anti-Windup的多种实现形式

◆ 例4——两种Anti-Windup控制器对比分析

考虑积分型控制器 $C(s)=\dfrac{k}{s}$



(a) Anti-Windup基本结构



(b) Anti-Windup一般形式—第一种

$$\hat{u}(t)=u(t)+\left[k\varepsilon e(t)-\frac{\varepsilon s}{\varepsilon s+1}u(t)\right] \Longleftrightarrow \hat{u}(t)=u(t)+\left[\frac{k}{s+kg}e(t)-\frac{s}{s+gk}u(t)\right]$$

$$\varepsilon=\frac{1}{gk}$$

近似等价

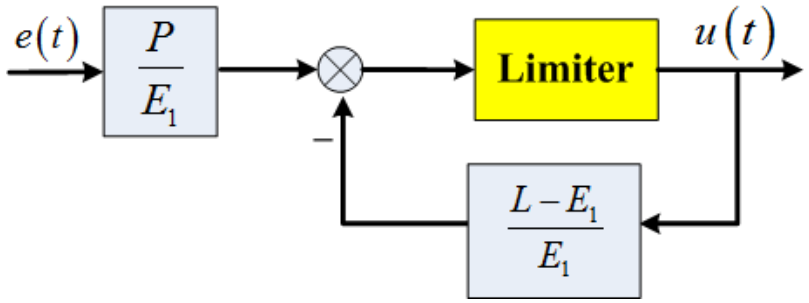
5.2.3 Anti-Windup的多种实现形式

Anti-Windup的一般形式——第二种

控制器 $C(s)=\dfrac{P(s)}{L(s)}$，其中 $L(s)=s^n+l_{n-1}s^{k-1}+\cdots+l_0$

$P(s)=p_n s^n+p_{n-1}s^{n-1}+\cdots+p_0$

闭环极点 $s=-\alpha_i \quad i=1\cdots N>n$



Anti-Windup一般形式——第二种

$E_1(s)$为n阶（与$L(s)$的阶次相同）的首一的Hurwitz多项式：

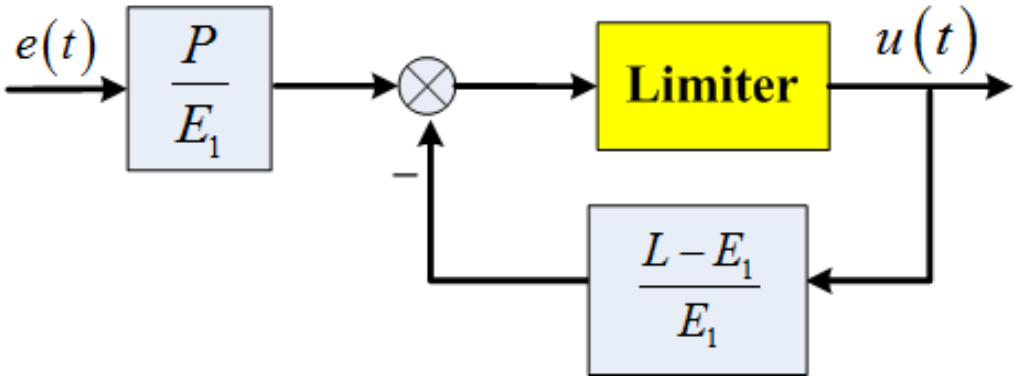
$$E_1=(s+\alpha_{m_1})(s+\alpha_{m_2})+\cdots+(s+\alpha_{m_n})$$

无限制条件，适用于：
非最小相位控制器、
不稳定控制器、
⋮

5.2.3 Anti-Windup的多种实现形式

Anti-Windup的一般形式——第二种

从原理上讲，$E_1(s)$可选择为任意的n阶首一Hurwitz多项式，但不同的选择会导致控制性能的差异。一种经验的选择方法是从闭环极点中选择(n)个最快的极点作为$E_1(s)$的根，大多数情况下可获得满意的性能。



Anti-Windup一般形式——第二种

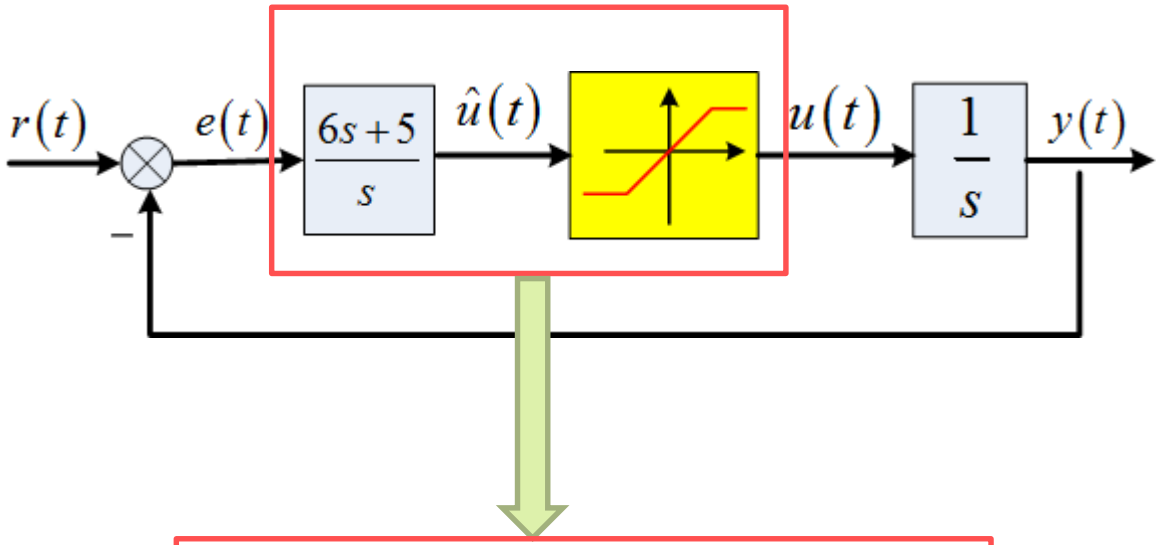
$$E_1=\left(s+\alpha_{m_1}\right)\left(s+\alpha_{m_2}\right)+\cdots+\left(s+\alpha_{m_n}\right)$$

闭环极点：$s=-\alpha_i \quad i=1\cdots N>n$

5.2.3 Anti-Windup的多种实现形式

◆ 例5——Anti-Windup一般形式控制器举例

$$U(s)=\frac{P(s)}{L(s)}E(s)$$



控制器 $L(s)=s$

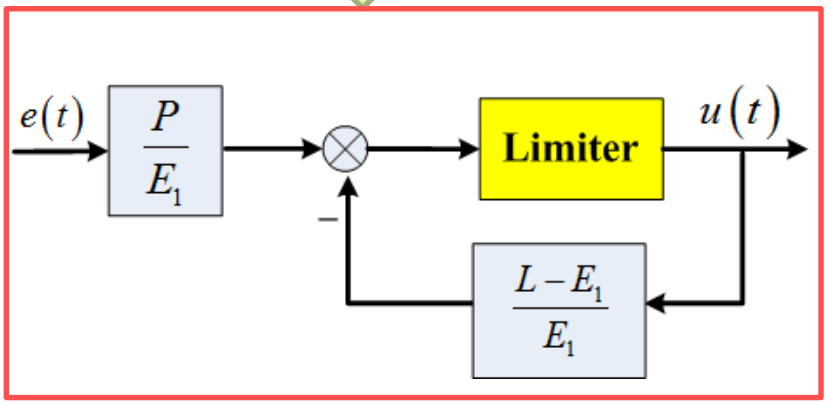
$P(s)=6s+5$

执行器限幅

$u_{\max}=1 \quad u_{\min}=-1$

闭环极点

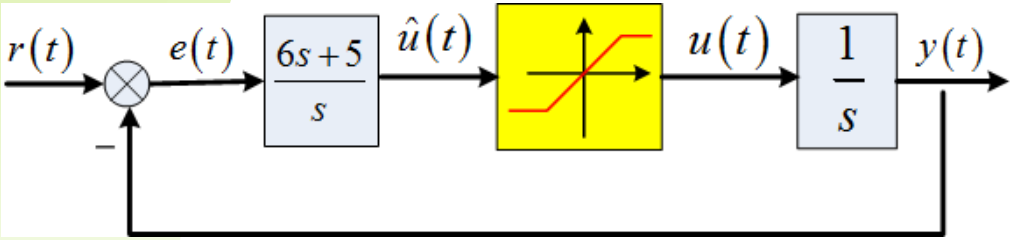
$s_1=-1, \quad s_2=-5$

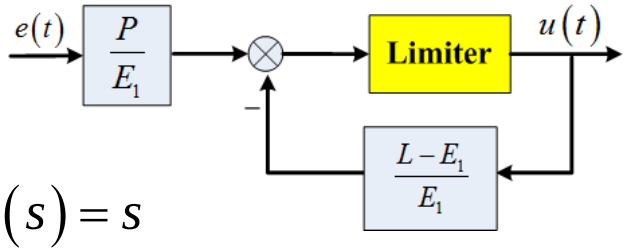


5.2.3 Anti-Windup的多种实现形式

◆ 例5——Anti-Windup一般形式控制器举例

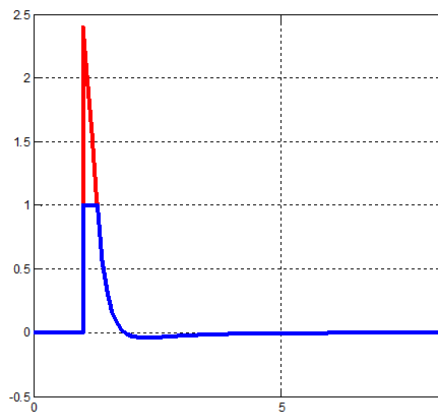
——非Anti-Windup 设计仿真结果



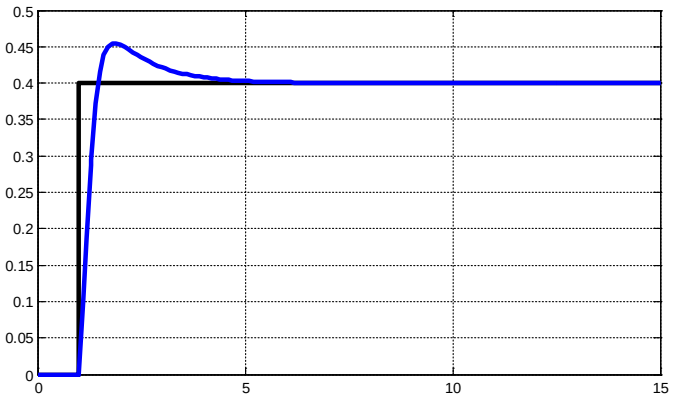


$$L(s)=s$$

$$P(s)=6s+5 \qquad E_1=(s_2+5)$$



u & $\hat{u}$

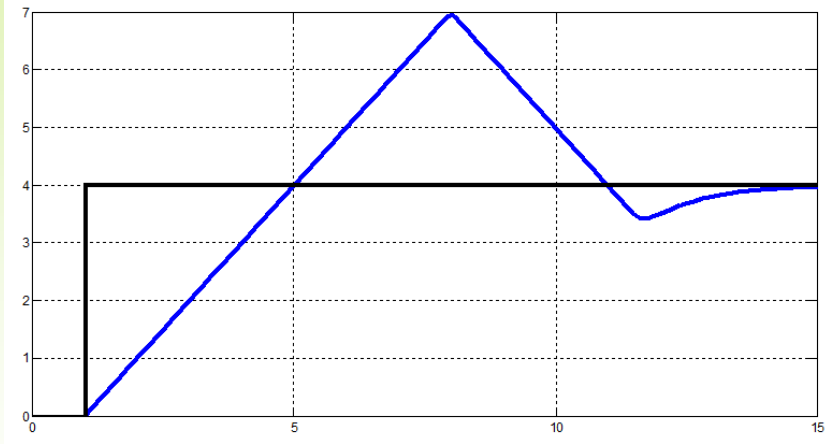


阶跃响应（r=0.4）

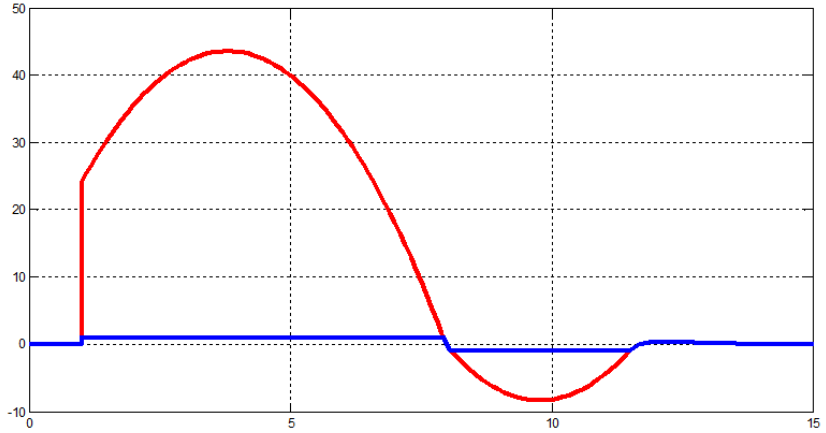
5.2.3 Anti-Windup的多种实现形式

◆ 例5——Anti-Windup一般形式控制器举例

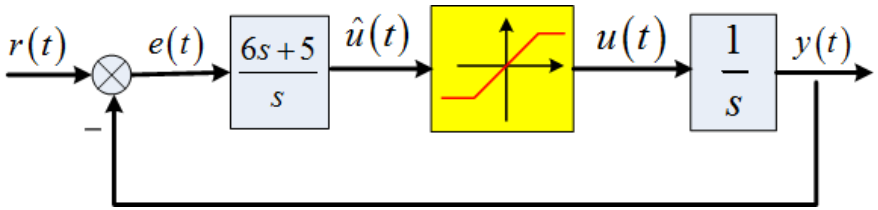
——非Anti-Windup 设计仿真结果



阶跃响应（r=4）



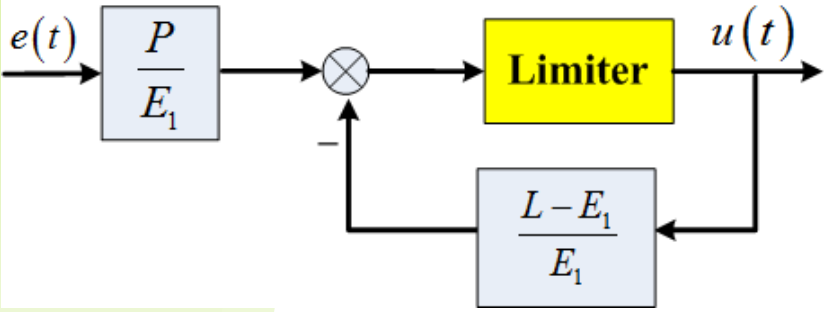
u & $\hat{u}$



控制系统框图
（非Anti-Windup控制器）

5.2.3 Anti-Windup的多种实现形式

◆ 例5——Anti-Windup一般形式控制器举例

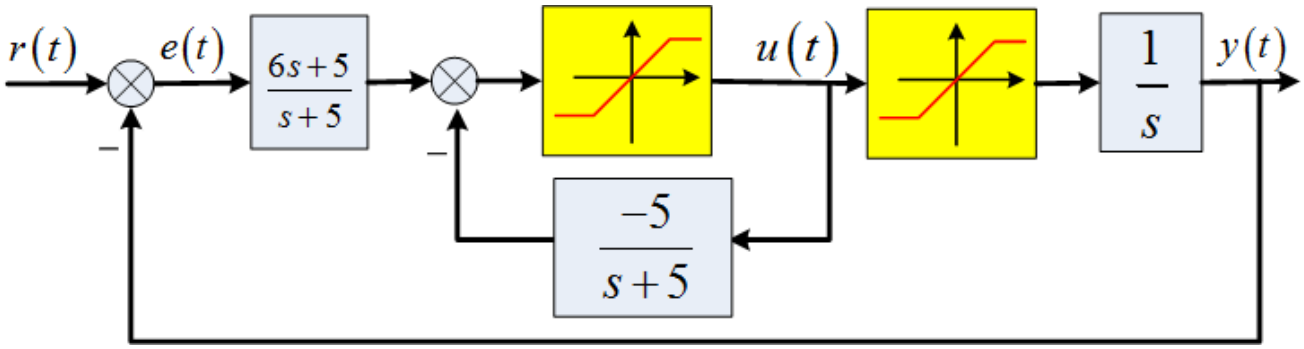


——非Anti-Windup 设计仿真结果

$$E_1(s) = s+5$$

$$\frac{P(s)}{E_1(s)} = \frac{6s+5}{s+5}$$

$$\frac{L(s)-E_1(s)}{E_1(s)} = \frac{-5}{s+5}$$

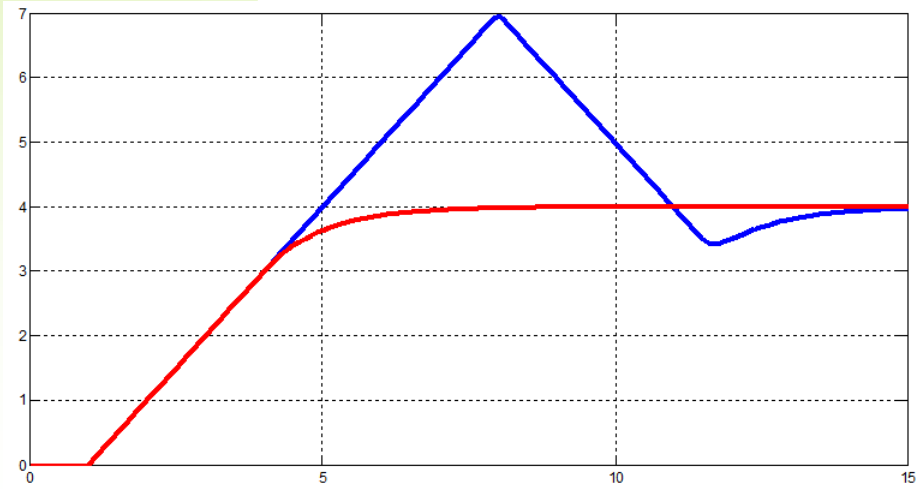


控制系统框图
(Anti-Windup控制器)

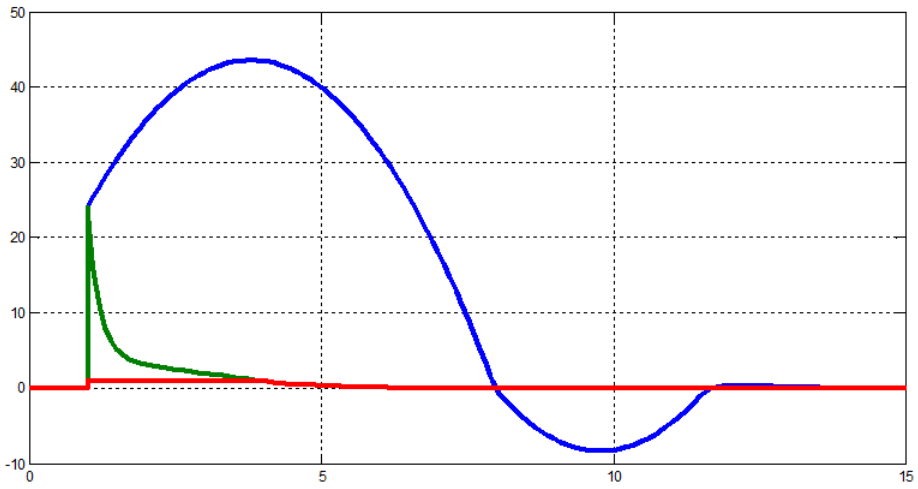
5.2.3 Anti-Windup的多种实现形式

◆ 例5——Anti-Windup一般形式控制器举例

——非Anti-Windup 设计仿真结果



阶跃响应对比（r=4）



控制量对比（r=4）

5.2.3 Anti-Windup的多种实现形式

◆ 例6——Anti-Windup一般形式控制器举例

不稳定对象 $G(s)=\dfrac{1}{s-1}$

控制器 $L(s)=s$

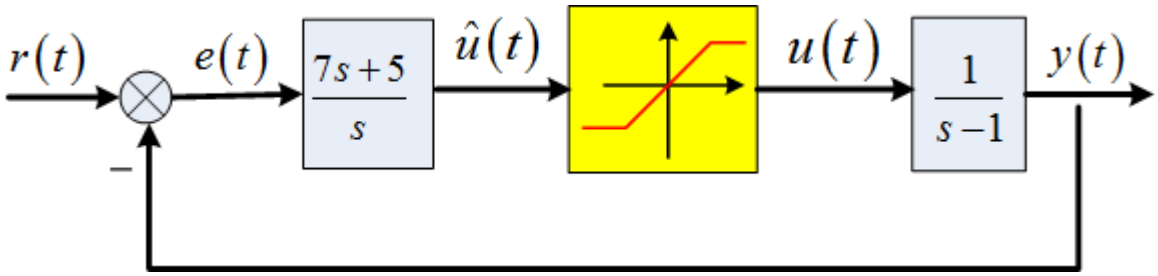
$P(s)=7s+5$

执行器限幅

$u_{\max}=1 \quad u_{\min}=-1$

闭环极点

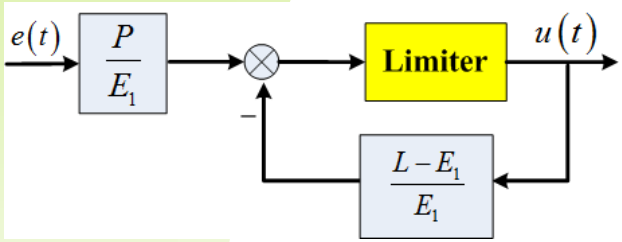
$s_1=-1, \quad s_2=-5$



控制系统框图
（非Anti-Windup控制器）

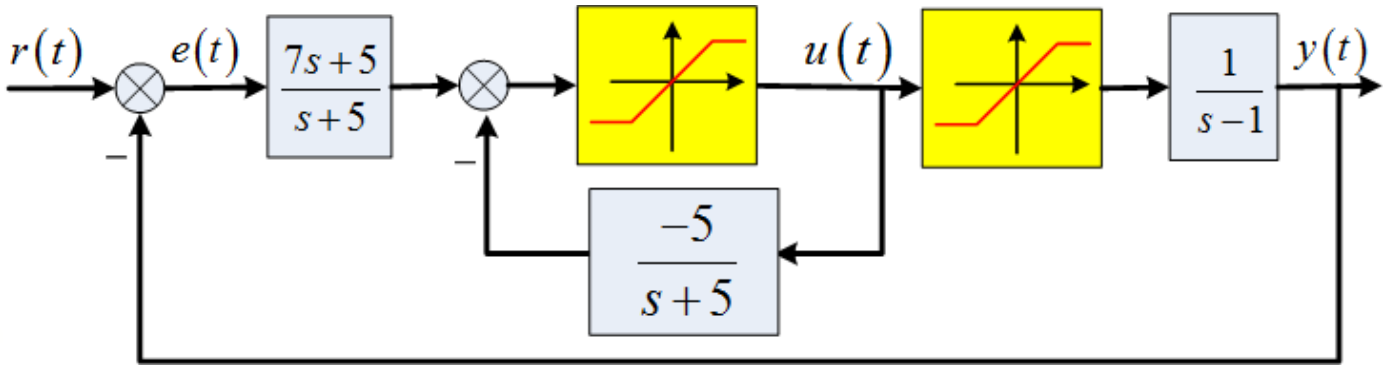
5.2.3 Anti-Windup的多种实现形式

◆ 例6——Anti-Windup一般形式控制器举例



$$E_1(s) = s+5$$

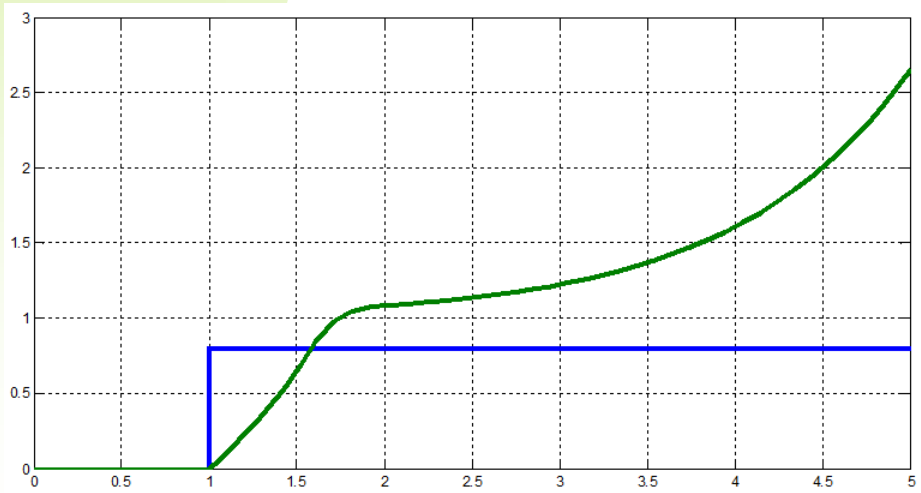
$$\frac{P(s)}{E_1(s)} = \frac{7s+5}{s+5} \qquad \frac{L(s)-E_1(s)}{E_1(s)} = \frac{-5}{s+5}$$



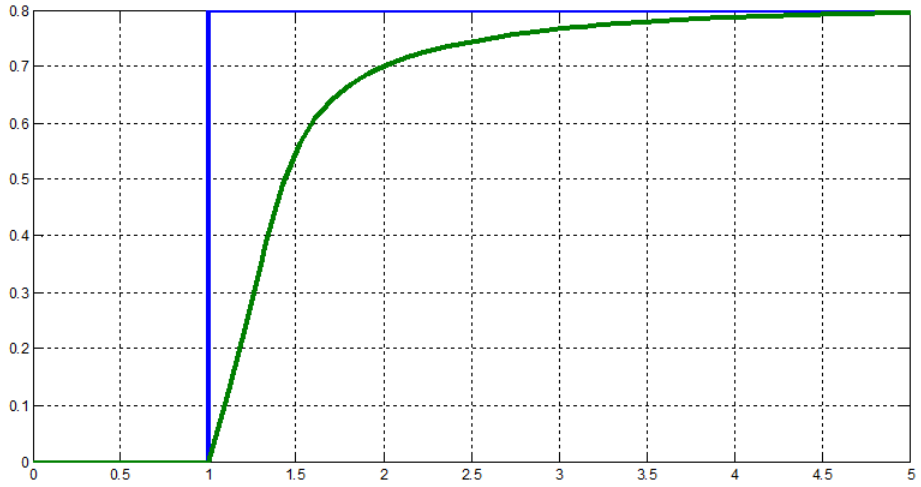
控制系统框图
(Anti-Windup控制器)

5.2.3 Anti-Windup的多种实现形式

◆ 例6——Anti-Windup一般形式控制器举例



阶跃响应对比（r=0.8）
非Anti-Windup控制

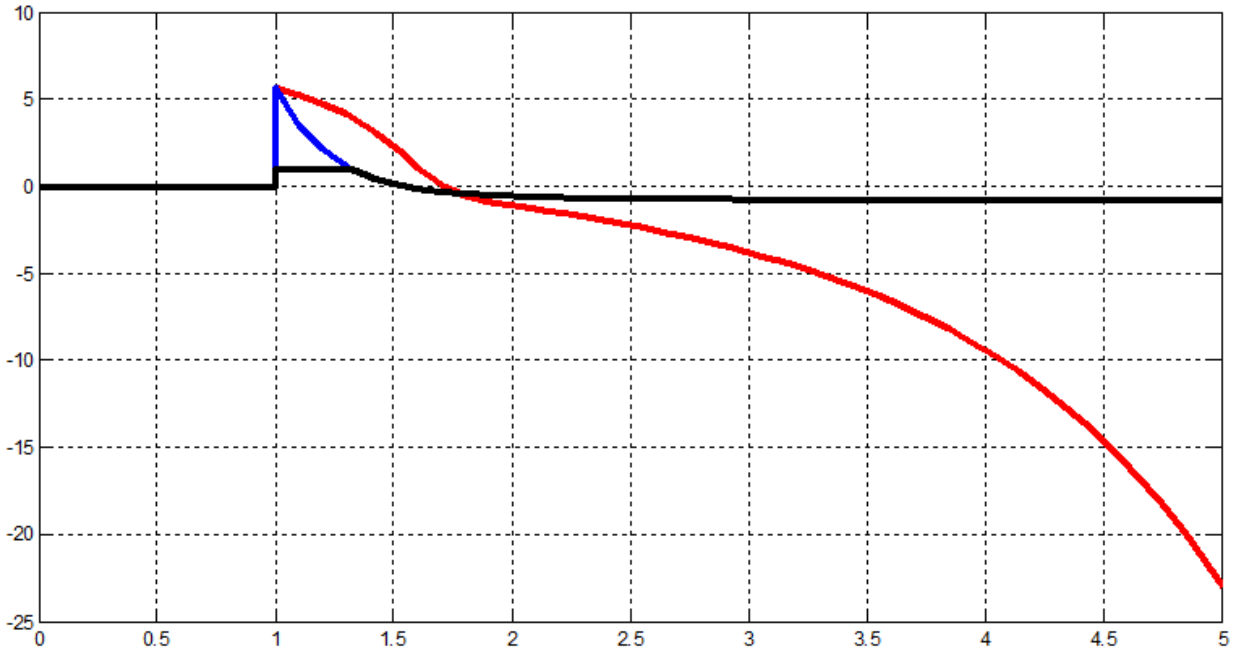


阶跃响应对比（r=0.8）
Anti-Windup控制

5.2.3 Anti-Windup的多种实现形式

◆ 例6——Anti-Windup一般形式控制器举例

——Anti-Windup 设计仿真结果



阶跃响应（r=0.8）下的控制量对比

非Anti-Windup控制　Anti-Windup期望控制　Anti-Windup实际控制

小　结

1. 执行器的饱和或转换速率限制并不总是能够完全避免，但可以通过多种措施减弱其对控制性能的影响；

2. 执行器进入饱和后，如不能及时退出饱和（如积分存在Windup时），则反馈作用被切断，系统处于开环状态，对于不稳定的被控对象，这是非常危险的；

3. Anti-Windup设计的主要思想是控制器的动态由对象的实际输入信号来驱动的，且此时控制器的动态是稳定的。

Thank You !

授课教师： 马 杰 （控制与仿真中心）

罗 晶 （控制科学与工程系）

马克茂（控制与仿真中心）

陈松林（控制与仿真中心）

Anti-Windup设计

作业

Anti-Windup控制的仿真：

针对例3所给的系统，验证如下框图中Anti-Windup控制策略各模块中的控制模型，然后利用Simulink分别搭建带有和不带有Anti-Windup策略的仿真模型，在不同时间点施加单位阶跃输入信号和负的单位阶跃输出扰动信号，对比两个仿真模型响应效果。