Christopher Lum
lum@uw.edu

Lecture 08a
Practical Implementation Issues with a PID Controller

Outline

-Practical Implementation of PID Controllers
    -Derivative Issues
    -Integral Issues

Practical Implementation Issues with a PID Controller

Recall that the control law of a PID controller has the form

$$u(t) = u_P(t) + u_I(t) + u_D(t) \qquad \textbf{(Eq.1)}$$

where $u_P(t) = K_P\, e(t)$

$u_I(t) = K_I \int_0^t e(\tau)\, d\tau$

$u_D(t) = K_D \frac{d\, e(t)}{d\, t}$

Mathematically, this is a fine representation of a control law, but there are several issues to consider when implementing PID controllers on real, digital hardware (particularly with the derivative and integral components).

Derivative Issues

Let us first investigate the derivative control component.

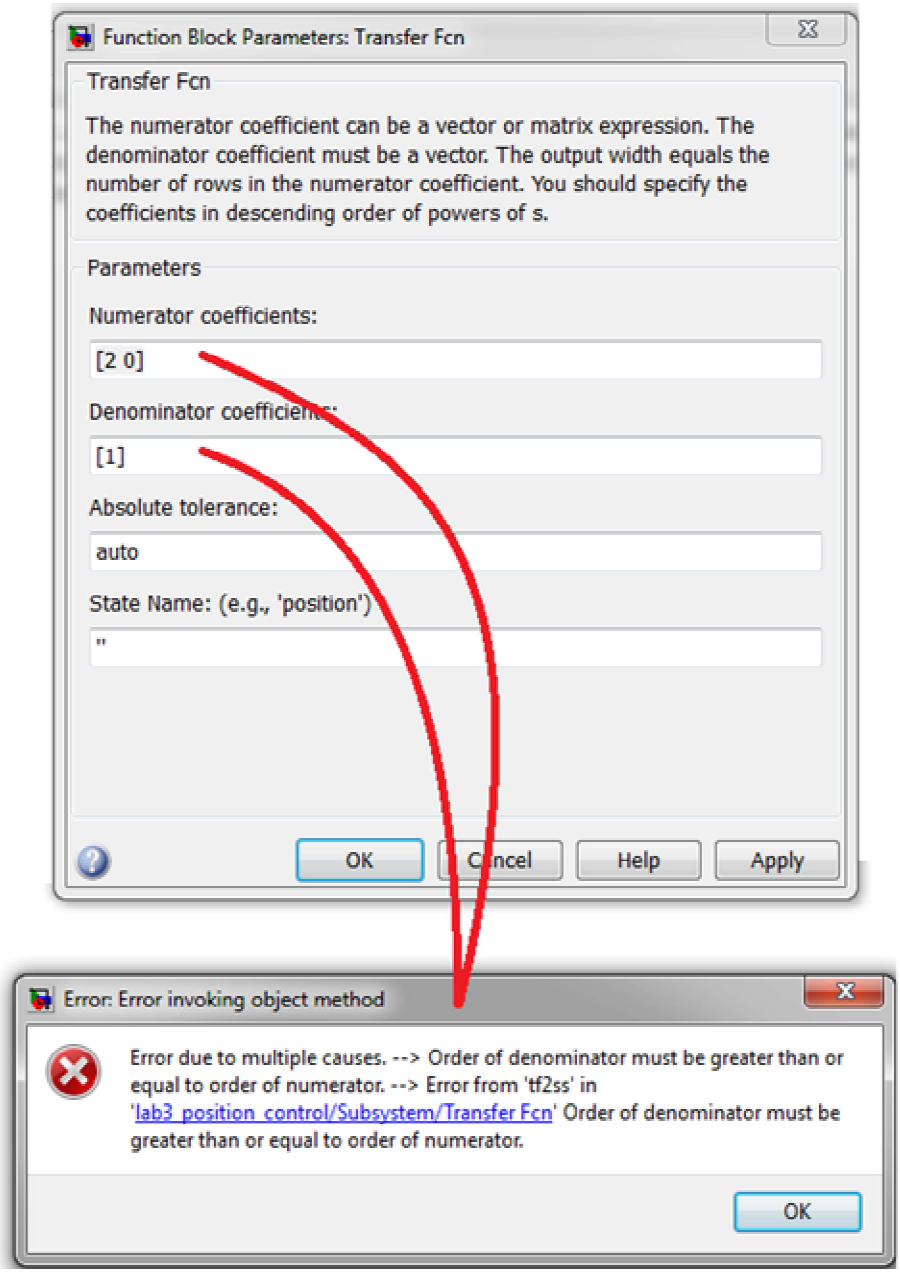
$$u_D(t) = K_D \frac{d\, e(t)}{d\, t} \qquad \textbf{(Eq.2)}$$

In the Laplace domain, we have

$$u_D(s) = K_D\, s\, E(s)$$

$$\frac{u_D(s)}{E(s)} = K_D\, s \qquad \textbf{(Eq.3)}$$

It may be tempting to attempt to implement this subsystem with a transfer function block as shown below for a $K_D = 2$ value (but we see this produces an error).



We see that it is non-proper (numerator order is higher than the denominator). One reason behind this error is that Simulink must convert this continuous time model to a discrete system in order to numerically solve. In order to do this, it can use various conversion methods which are beyond the scope of this class, but one resulting difference equation for a sampling time T is given as

$$u_D(k) = K_D\left(\frac{e(k+1)-e(k)}{T}\right) \qquad \textbf{(Eq.4)}$$

This appears to be a non-causal system (requires information at time $k+1$ in order to compute the control at time k. This is a characteristic of non-proper systems.

In all likelihood, we do not have a direct measurement of $d\,e(t)/d\,t$, instead, the controller has access only to $e(t)$ and therefore must numerically calculate the time derivative of the signal. To get around the non-causal problem, we can approximate the derivative of the error as

$$\dot{\tilde{e}}(k) \approx \frac{e(k)-e(k-1)}{T} \qquad \textbf{(Eq.5)}$$

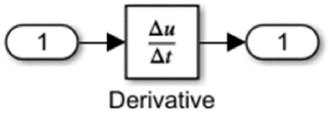
The problem with this simple numerical derivative is if the error signal has some noise (as shown below), there may be large spikes in the $\dot{e}$ signal.

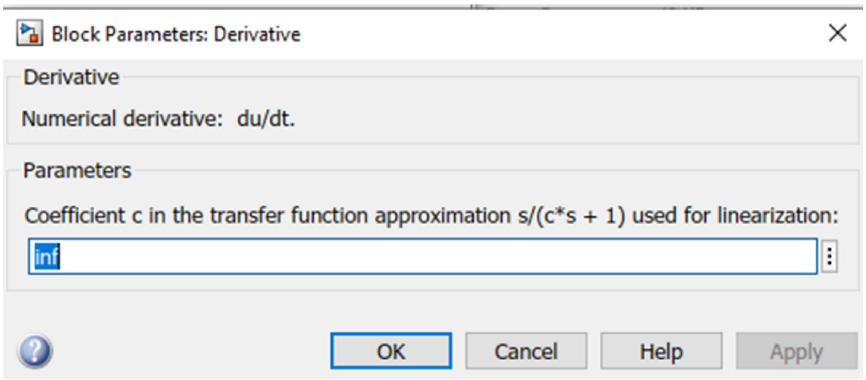
$$\dot{\tilde{e}}(k) \approx \frac{e(k)-e(k-1)+n(k)}{T} \quad \text{where } n(k) = \text{noise at time } k$$

$$= \frac{e(k)-e(k-1)}{T} + \frac{n(k)}{T}$$

We see that we would like to measure the term $\frac{e(k)-e(k-1)}{T}$ but if the time sample is small, then the term $\frac{n(k)}{T}$ will likely be large and wash out the signal.

It may be tempting to us the simple numerical derivative of Eq.5 to compute $\dot{e}(k)$ as shown below





The problem is if the error signal has some noise to it and we sample at a fast rate, the noise will be amplified and the resulting signal for $\dot{e}(k)$ will be very noisy as shown in the first subplot of a following plot.

To combat this problem, we propose to use a pseudo derivative of the form

$$\frac{\dot{E}(s)}{E(s)} = \frac{a\,s}{s+a} \qquad \textbf{(Eq.6)}$$

Note that this is the equivalent to the transfer function approximation suggested in the block parameters

$$\frac{a\,s}{s+a} = \frac{a\,s}{a\left(\frac{1}{a}s+1\right)}$$

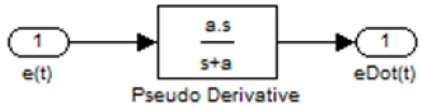
$$= \frac{s}{\frac{1}{a}s+1} \qquad \text{let } c = 1/a$$

$$\frac{a\,s}{s+a} = \frac{s}{c\,s+1}$$

where $c = 1/a$

So $c = \infty$ corresponds to $a = 0$

A block diagram is shown below

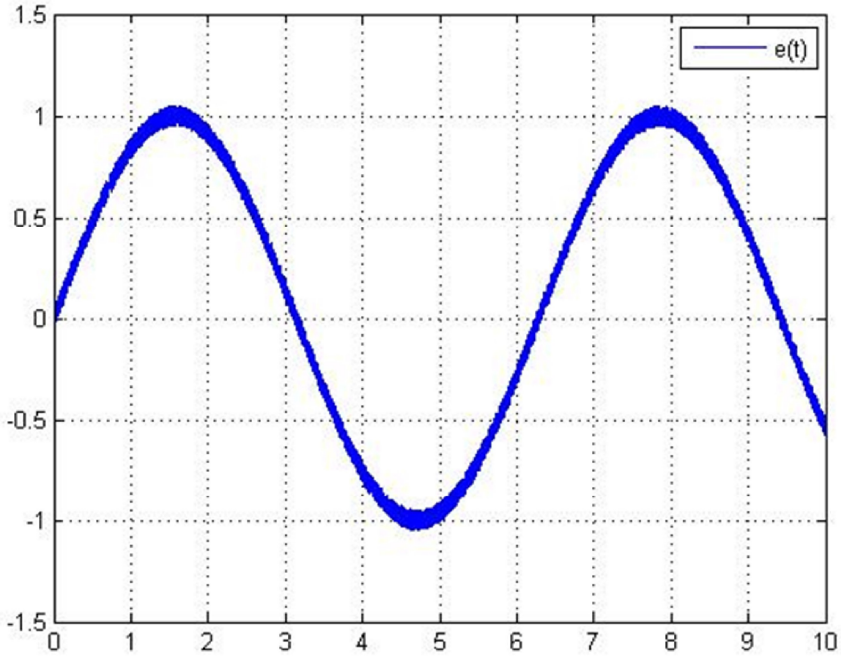


We see that this is simply a derivative combined with a 1st order low pass filter. If the cutoff frequency $\omega_n = a$ is chosen sensibly, then the noise can be somewhat filtered out. An additional benefit is that this is a proper transfer function which can be converted into a causal difference equation for the purposes of digital/discrete/numerical implementation. We see from the second subplot of the following plot that this improves the derivative somewhat.

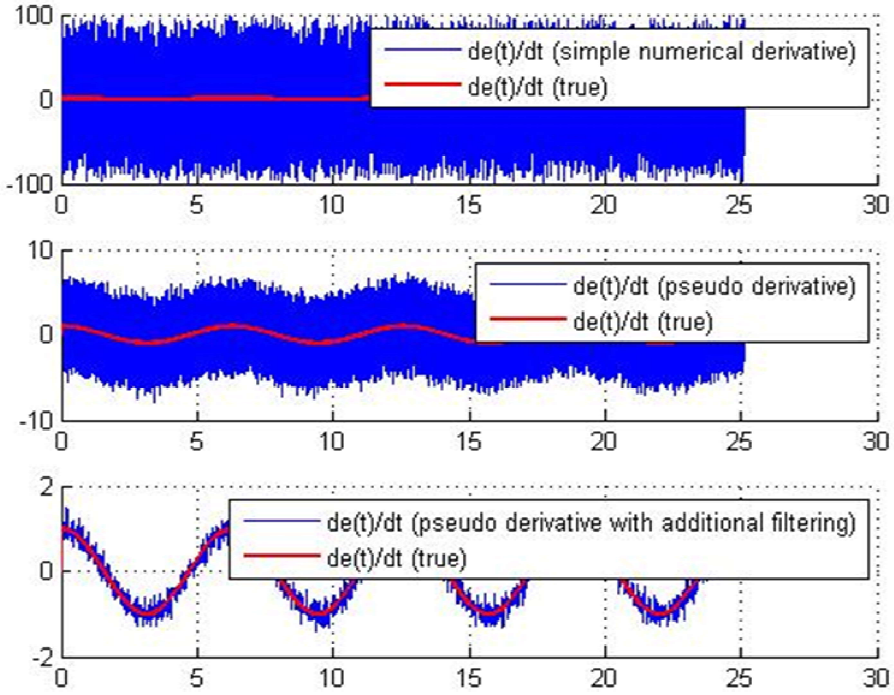
We can extend this idea and further filter the error signal to combat the noise issue

$$\frac{\dot{E}(s)}{E(s)} = \left(\frac{a}{s+a}\right)^n \left(\frac{a\,s}{s+a}\right) \qquad \textbf{(Eq.7)}$$

In this case, we simply use 3 low pass filters in series to cut out high frequency noise (at the cost of inducing more phase lag to the system). We see from the third subplot of the following plot that this yields a good approximation of the derivative.

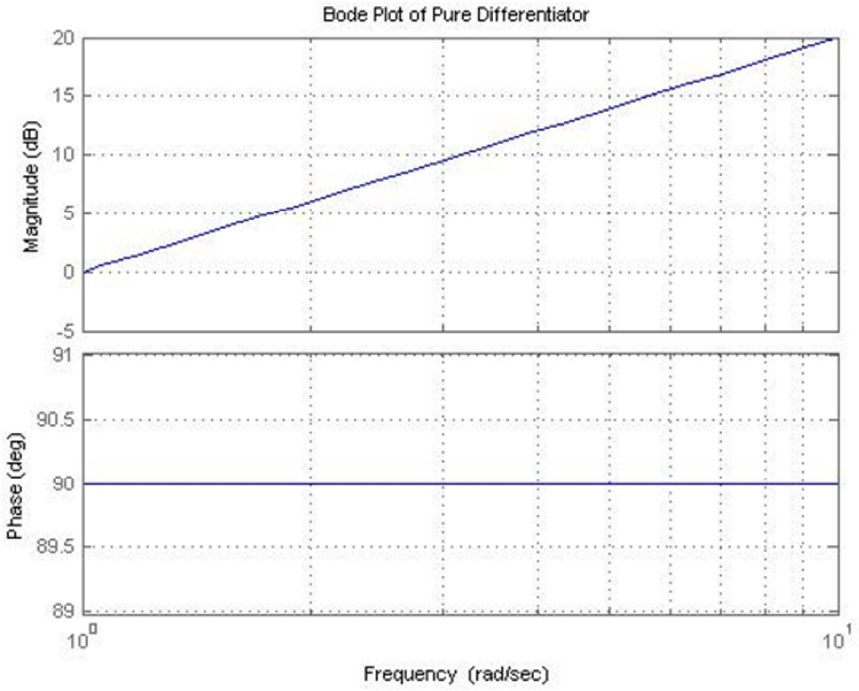


If we compute the derivative using the various method discussed previously, we obtain (for an input frequency of 1 rad/s)

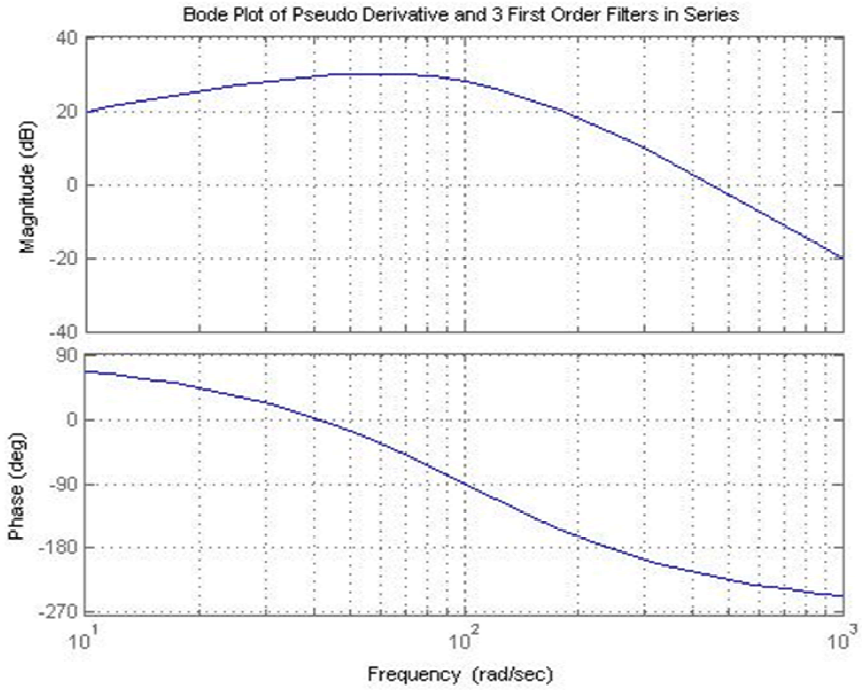


We can gain insight on the cause of this noise amplification by looking at the bode plot of a pure differ-

entiator, $\frac{\dot{E}(s)}{E(s)} = s$

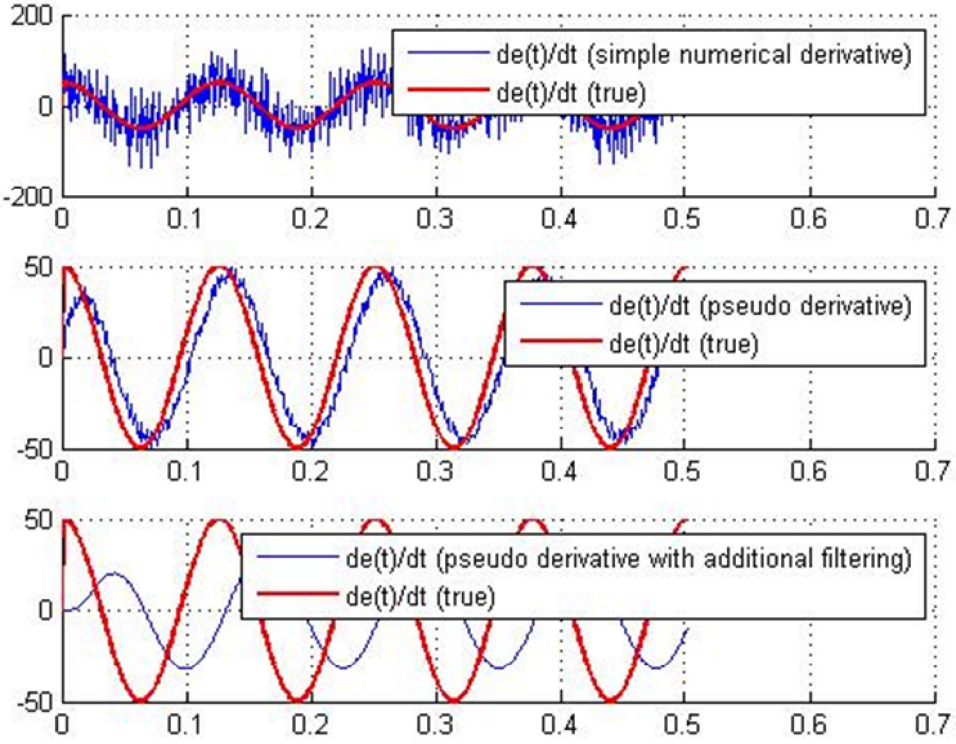


As can be seen, this amplifies high frequency noise. As the frequency increases, so does the amplification of high frequency noise. We can investigate the bode plot of the filtered, pseudo-derivative filter, $\frac{\dot{E}(s)}{E(s)} = \left(\frac{a}{s+a}\right)^3 \left(\frac{a\,s}{s+a}\right)$ with $a = 100$.



As we see here, the pseudo-derivative followed by 3 low pass filters yields a system which filters out noise above 100 rad/s but also ceases to act like a differentiator at higher frequencies (it fails to compute the derivative and also incurs phase lag at higher frequencies).

In summary, at higher frequencies, this system's approximation to a derivative begins to break down as can be seen below for input frequency of approximately 50 rad/s)



The pseudo-derivative is also beneficial in situations where there are discontinuities in the error signal. For example, if there is a step change in setpoint, there will be a discontinuity in the error signal which will lead to theoretically infinite $\dot{e}(t)$. This problem is present even if the signal has no noise. A pseudo-derivative will "smooth out" the $\dot{e}(t)$ signal in these situations.

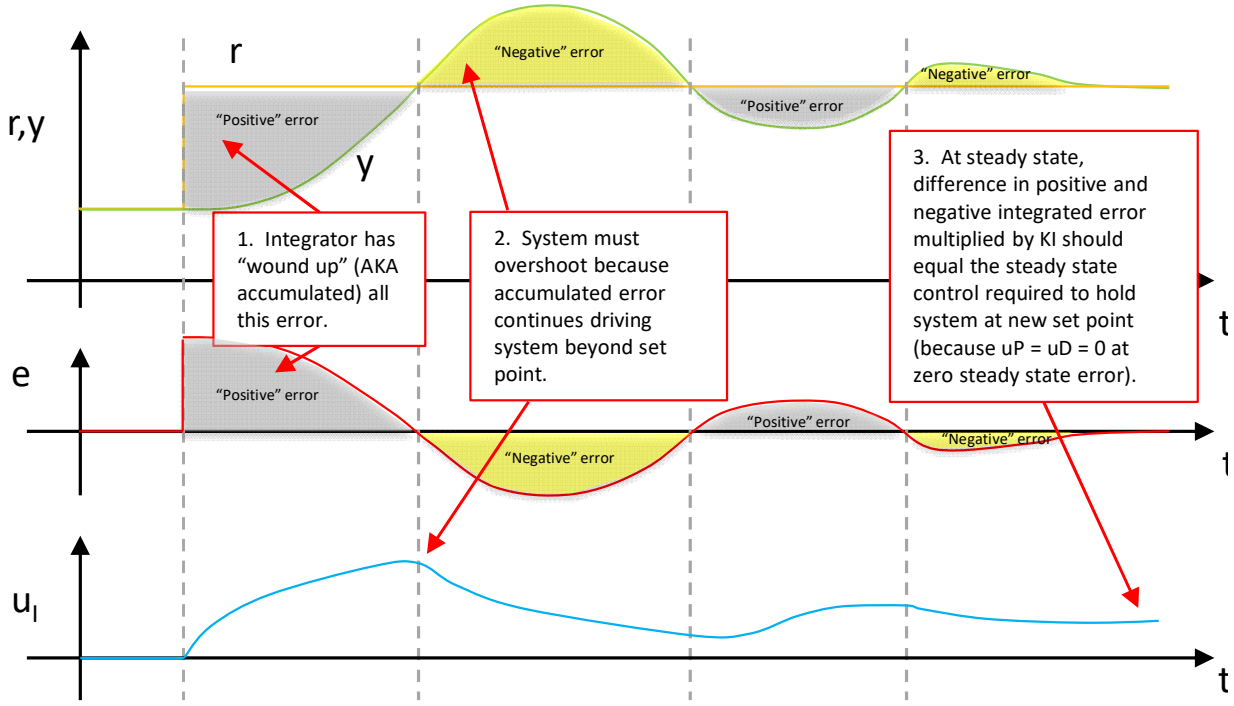
Integral Issues

We now turn our attention to the integral control component.

$$u_I(t) = K_I \int_0^t e(\tau)\, d\tau \qquad \textbf{(Eq.8)}$$

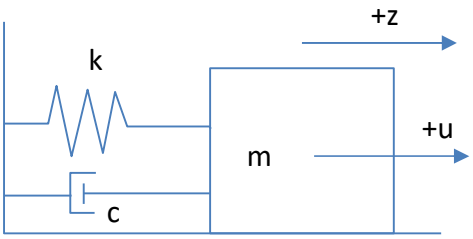
One of the major issues with integral control is a phenomena known as integrator windup. This is easiest to understand by examining an example.

Consider a simple integral controller with $K_I = 1$. Assume that the system is at steady state and we introduce a step in desired setpoint.



Example: Spring Mass Damper

Consider the simple spring mass damper system



EOMs are

$$-k\,z(t) - c\,\dot{z}(t) + u(t) = m\,\ddot{z}(t)$$

$$-\frac{k}{m}\,z(t) - \frac{c}{m}\,\dot{z}(t) + \frac{1}{m}\,u(t) = \ddot{z}(t)$$

$$\overline{x}(t) = \begin{pmatrix} z(t) \\ \dot{z}(t) \end{pmatrix}$$

$$\dot{\overline{x}}(t) = \begin{pmatrix} \dot{z}(t) \\ \ddot{z}(t) \end{pmatrix}$$

$$= \begin{pmatrix} \dot{z}(t) \\ -\frac{k}{m} z(t) - \frac{c}{m} \dot{z}(t) + \frac{1}{m} u(t) \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 1 \\ -\frac{k}{m} & \frac{-c}{m} \end{pmatrix} \begin{pmatrix} z(t) \\ \dot{z}(t) \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{m} \end{pmatrix} u(t)$$

We can simulate controlling this system with a PID controller.

-Show case with oscillations

-Show case with no oscillations

This can have potentially disastrous results for your system if you are not careful.

Example: Aerosonde *Piper* Crash Incident

History

-1991 Dr. Tad McGeer founds Insitu

-August 1998, transatlantic crossing with the Aerosonde, 2000 miles, 26 hours, 29 pound aircraft on 1.5 US gallons of fuel

3 attempts

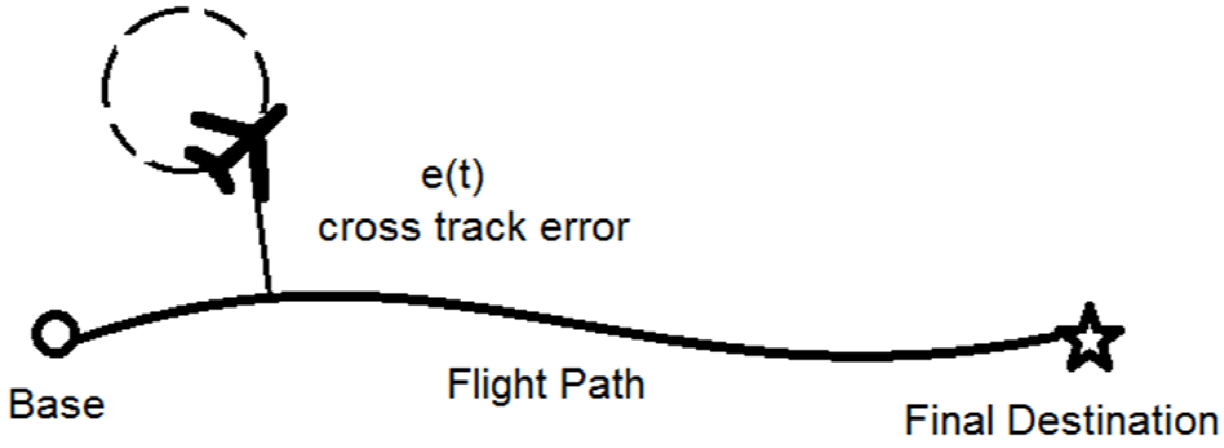
1. *Trumper* - lost at sea
2. *Piper* - crash at launch due to integrator wind up
3. *Laima* - successful

-Laima is now hanging in the Museum of Flight in Seattle

-2008 Insitu acquired by Boeing

This may have caused a crash of the *Piper*. Some of this is based on stories from Juris Vagners, UW professor named the 3rd successful aircraft Laima after the Latvian diety of good fortune.

Consider an aircraft on a mission as shown below. In this illustration, the aircraft controller controls the bank angle of the aircraft as a function of the cross track error.



Consider the case when the aircraft is in a holding pattern near the base before starting the mission. If the integral controller is turned on while the aircraft is in manual mode, the integrator will begin wind-

ing up error. In the case of the Aerosonde, when the system is transitioned from manual to autopilot control via a ground station command, the integrator state was reset properly. However, in the case of *Piper*, during the preflight launch, there was a temporary communication drop and autopilot automatically switched on. It turns out that in this edge case scenario where the autopilot transition was triggered via communication failure rather than explicit ground station command, the integrator windup was not reset and taken into account and this crashed.

References

-https://www.washington.edu/news/1998/08/21/aerosonde-robotic-airplane-completes-historic-trans-atlantic-flight/

-https : // aerovel.com / wp – content / uploads / 2015 / 03 / Laima_the –

first – Atlantic – crossing – by – unmanned – aircraft1.pdf

Show simulation of this

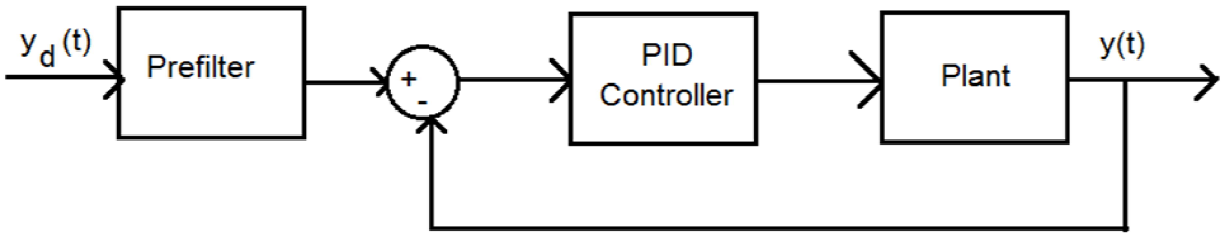
Addressing Integrator Windup

The problem is that if there is a large setpoint change or if the system ever has a chance to accumulate too much error, it may require significant overshoot to compensate for. Therefore, several solutions may include:

Solution 1. Increasing the setpoint in a gradual fashion to avoid large errors accumulating
Solution 2. Disable the integral function until the system reaches some region of the setpoint
Solution 3. Limiting the time period over which the integral error is calculated.
Solution 4. Limiting the max/min state of the integrator (AKA the amount of accumulated error)

Solution 1

If the setpoint is changed in a gradual fashion, this will not allow the error to be so large. This can be accomplished using a prefilter type architecture as shown below.



Some important design characteristics of the prefilter.

-DC gain of unity (see video on DC gain https://youtu.be/sgTt7v4LYfE)

-Bandwidth which does not handicap the system (see video on 'Bandwidth of a Dynamic System' https://youtu.be/evVi_D7C6mA)

Add this to mass/spring/damper example and show how it helps alleviate integrator wind-up

Solution 2

Notice that at steady state, if the error is zero and the system is steady, the proportional and derivative

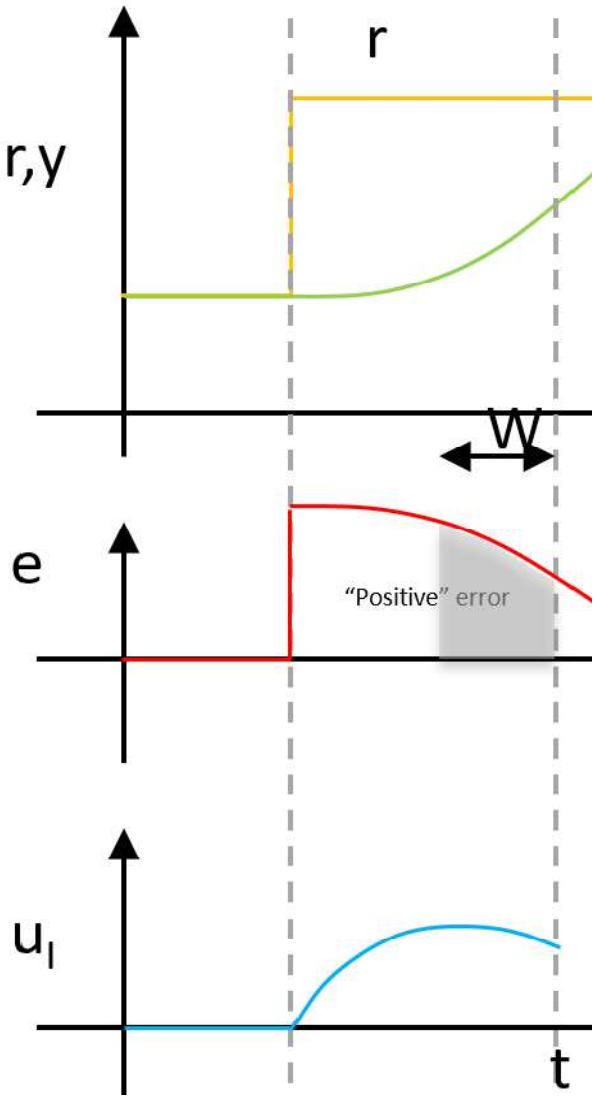
components of the PID controller should do nothing and the integral should be generating the required control signal to hold the system at the setpoint. Therefore, the integral term serves to "cleanup" the small steady state error and can simply be turned on once the proportional and derivative components have moved the system to the region of the setpoint.

Solution 3

Approach 3 and 4 are sometimes referred to as “integrator anti-wind-up” schemes.

Instead of integrating all the error from time 0 up to the current time, we can only integrate error over a finite lookback window of length W

$$u_I(t) = K_I \int_0^t e(\tau)\, d\tau \quad \Rightarrow \quad u_I(t) = K_I \int_{t-W}^t e(\tau)\, d\tau$$

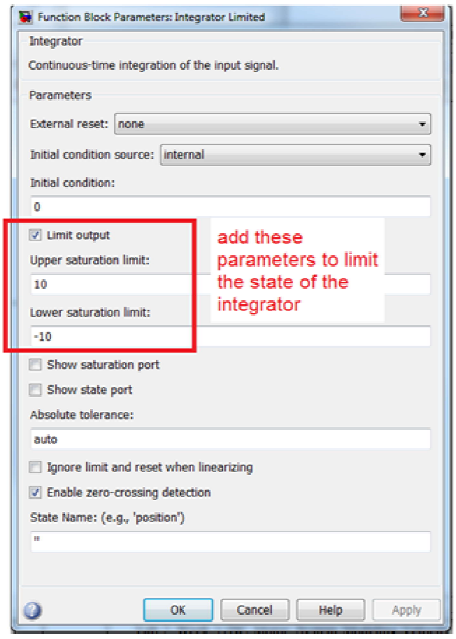


The effect of this is to limit the amount of error that the integrator can accumulate.

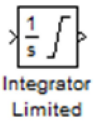
Solution 4

Addressing this problem involves limiting the state of the integrator.

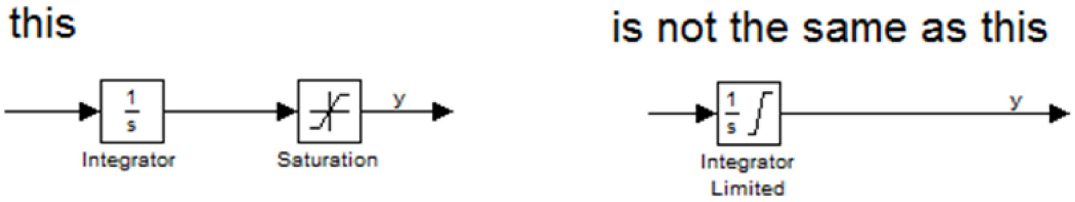
You can modify a standard “Integrator” block to have this functionality by doing as shown below

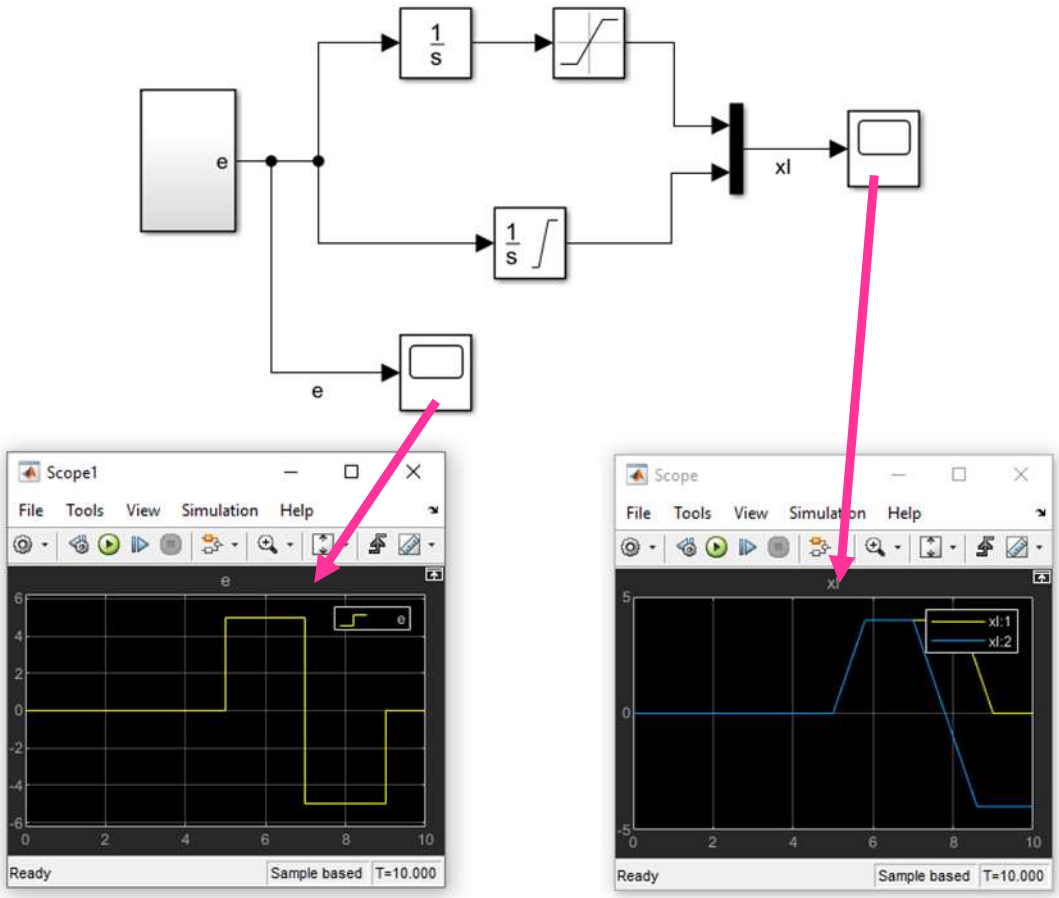


The block now appears as



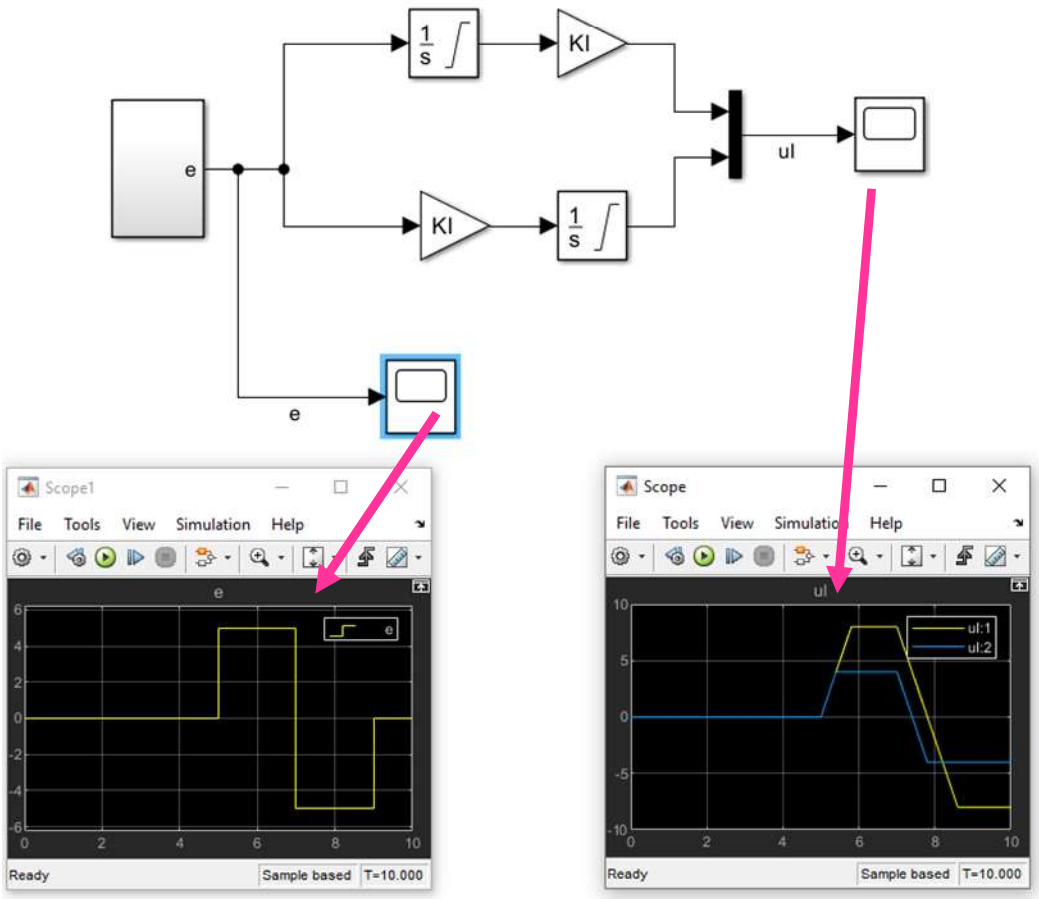
Note that the wording of “Limit output” is misleading. This is not the same as simply putting a saturation block on the output of the integrator. In other words





Optional: see homework assignment which investigates looks at how to implement integrator anti-wind up in code

Furthermore, it now matters if the gain K_I comes before or after the limited integrator.



Note that I personally prefer to have the saturated integrator after the K_I gain so the saturation limit is the actual control authority of the integrator.

Show how this fixes both the mass spring damper system and the aerosonde problem

Note that items 2, 3, and 4 are nonlinear operation and therefore make the entire system nonlinear. This means we cannot no longer make guarantees about performance, stability, etc.

Closing Thoughts

Note that if you include a pseudo-derivative and a prefilter, the system is still linear.

If you add any type of non-linear integrator anti-wind up, the system becomes nonlinear.