Microcontroller Application Series
PID Control 2/3

Credit : Shanying Zhu

Department of Automation, SJTU

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Today's Lecture- PID/2

1 Introduction

2 Analog PID Controller

3 Digital PID Controller

4 PID Controller Tuning

5 Summary

3 Digital PID Controller

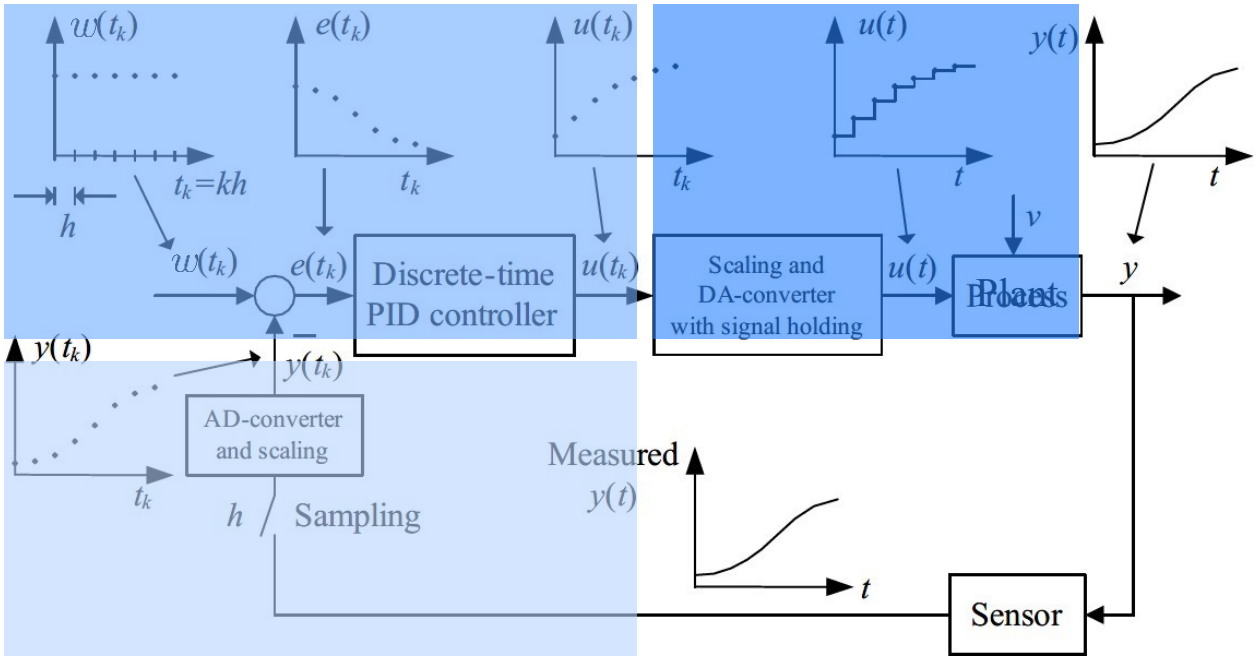
- Introduction
- Discretization of an analog PID
- Practical issues and remedies
- Summary

Introduction

- **Control loop implemented with a computer**

The control signal is scaled and sent to the DA converter, where it is held constant during the present time step, followed by the control action of the plant.

The resulting digital signal, y(tk), is used in the control function, calculating the value of the control signal u(tk).



The computer registers the process measurement signal viaan AD converter. The AD converter produces a numerical value which represents the measurement.

Discretizing a PID Controller

- **Digital PID controller**

$$u(t) = K_{\mathrm{p}}\left(e(t) + \frac{1}{T_{\mathrm{i}}}\int_0^t e(s)\,\mathrm{d}s + T_{\mathrm{d}}\frac{\mathrm{d}e(t)}{\mathrm{d}t}\right)$$

  - Numerical approximation for P, I, D terms

integral $\int$	summation, e.g., rectangle method $\sum$
derivative $\frac{\mathrm{d}}{\mathrm{d}t}$	backward/forward difference $\frac{(t)-(t-T)}{T}; \frac{(t+T)-(t)}{T}$

- Two prevailing forms
  - **Position/absolute algorithm**: the output of the controller is the absolute value of the control signal, e.g., a valve position.
  - **Incremental/velocity algorithm**: the output of the controller represents the increments of the control signal, e.g., a stepping motor.

Position/Absolute Algorithm

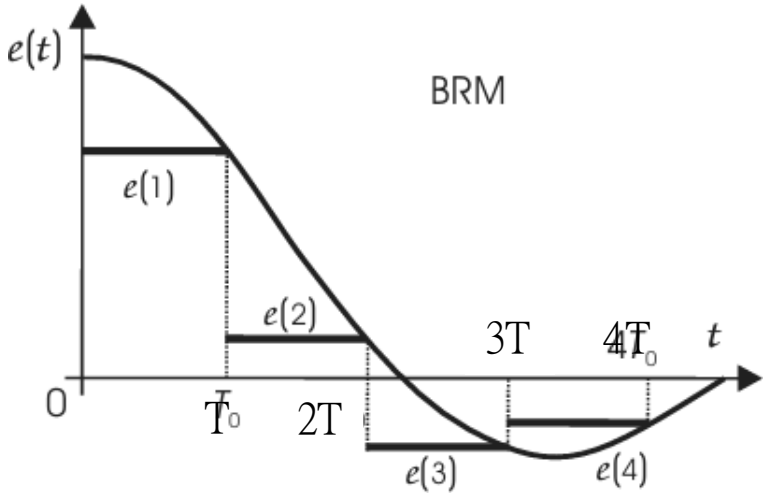
- Sampling period T
- The error value at the k-th moment of sampling, i.e., $t = kT$, is

$$e(t) \approx e_k \triangleq e(kT)$$

- then

$$\int_0^t e(t)\mathrm{d}t \approx T\sum_{j=1}^{k} e_j$$

backward rectangle method(BRM)



$$\frac{\mathrm{d}e(t)}{\mathrm{d}t} \approx \frac{e_k - e_{k-1}}{T}$$

backward difference

- **Basic form**: need to store all previous errors $e_1, e_2, .., e_{k-1}$

$$u_k = K_\mathrm{p}\left(e_k + \frac{T}{T_\mathrm{i}}\sum_{j=1}^{k} e_j + \frac{T_\mathrm{d}}{T}(e_k - e_{k-1})\right)$$

- **Component form:** Only the last value of the integral component $P_\mathrm{i}(k-1)$ and e_{k-1} are retained in the memory.

$$u_k = P_\mathrm{p}(k) + P_\mathrm{i}(k) + P_\mathrm{d}(k)$$

- where

$$P_\mathrm{p}(k) = K_\mathrm{p} e_k$$

$$P_\mathrm{i}(k) = K_\mathrm{p}\frac{T}{T_\mathrm{i}}\sum_{j=1}^{k} e_j = P_\mathrm{i}(k-1) + K_\mathrm{p}\frac{T}{T_\mathrm{i}} e_k$$

$$P_\mathrm{d}(k) = K_\mathrm{p}\frac{T_\mathrm{d}}{T}(e_k - e_{k-1})$$

Incremental/Velocity Algorithm

- Define the incremental control value

$$\Delta u_k \triangleq u_k - u_{k-1}$$

- **Incremental PID algorithm**:

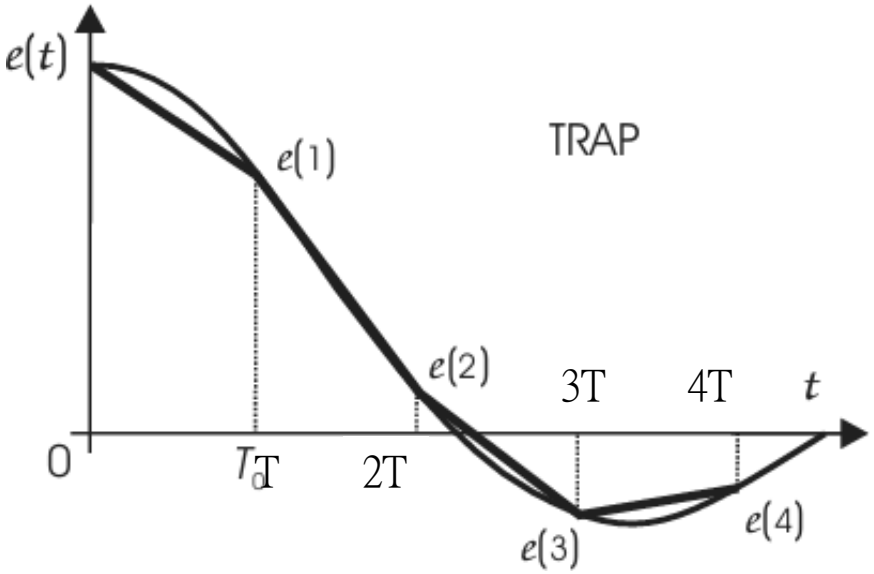
$$\begin{aligned}\Delta u_k &= K_{\mathrm{p}}\left(e_k - e_{k-1} + \frac{T}{T_{\mathrm{i}}}e_k + \frac{T_{\mathrm{d}}}{T}(e_k - 2e_{k-1} + e_{k-2})\right)\\ &= d_0 e_k + d_1 e_{k-1} + d_2 e_{k-2}\end{aligned}$$

- where

$$d_0 = K_{\mathrm{p}}\left(1 + \frac{T}{T_{\mathrm{i}}} + \frac{T_{\mathrm{d}}}{T}\right),\ d_1 = -K_{\mathrm{p}}\left(1 + 2\frac{T_{\mathrm{d}}}{T}\right),\ d_2 = K_{\mathrm{p}}\frac{T_{\mathrm{d}}}{T}$$

- The incremental algorithm is particularly useful if the actuator, e.g., stepping motor, is controlled by an incremental signal. It is only Δu_k that is sent to the stepping motor.

- **Exercise:** Use the trapezoidal method and backward difference method to approximate the integral and derivative, respectively,
  - Position algorithm?
  - Incremental algorithm?

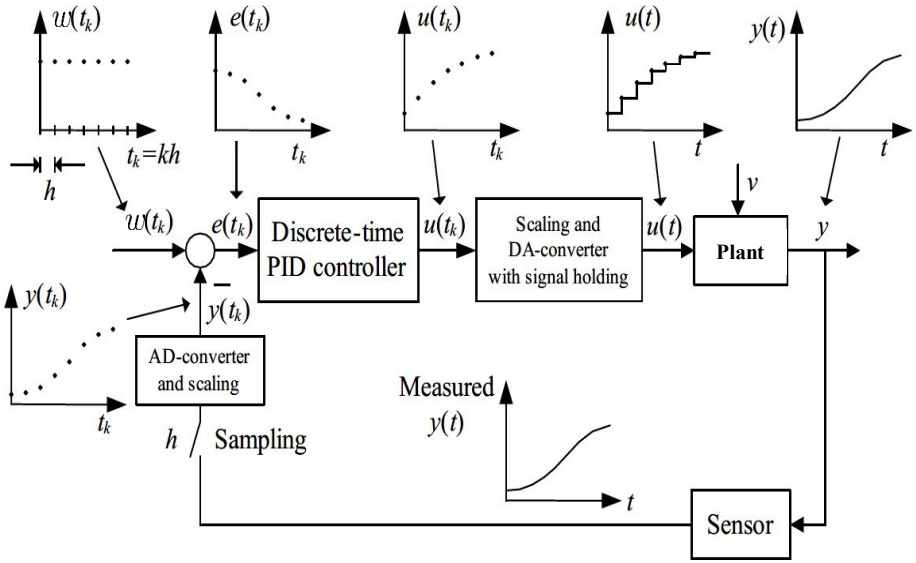


Practical Issues Related to Implementation-1/4

- **Selecting the sampling period:** Approximation of a analog PID controller is only suitable where the sampling period T is small enough compared with the system dynamics.
- Shannon's sampling theorem:

$$\frac{1}{T} \geq 2f_{\max}$$

- Variance of quantization noise due to ADC rounding increases with T
- ZOH in DAC introduces additional delay in the control loop$\approx T/2$



- Decreasing T requires more expensive hardware and aggravate problems caused by the finite-word representation of parameters.

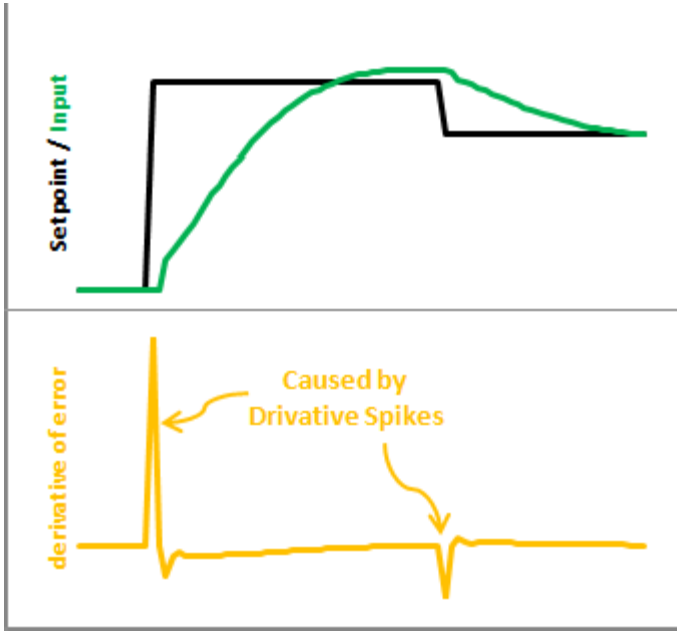
$$\frac{1}{T} \geq M f_{\max}, \quad M = 5 \sim 10$$

Practical Issues Related to Implementation-2/4

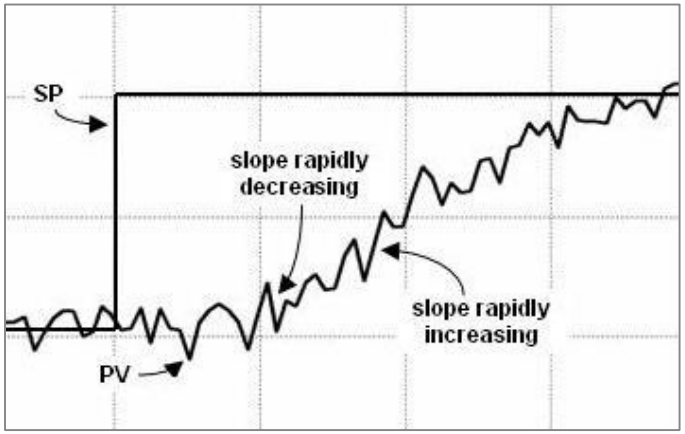
- **Derivative kick:** the derivative term momentarily become very large

$$\frac{\mathrm{d}e(t)}{\mathrm{d}t} = \frac{\mathrm{d\ setpoint}}{\mathrm{d}t} - \frac{\mathrm{d}y(t)}{\mathrm{d}t}$$

- On one hand, any change in setpoint causes an instantaneous change in error. The derivative of this change is infinity, resulting in a spike.



- On the other hand, the derivate of a noisy measurement signal will change wildly, if it contains high-frequency, random fluctuations, and D action will amplify the noise.



Eliminating Derivative Kick

- **1. Filtering the D action:** Limits the gain of D control

$$U_{\mathrm{d}}(s) = sK_{\mathrm{d}}E(s) = sK_{\mathrm{p}}T_{\mathrm{d}}E(s)$$

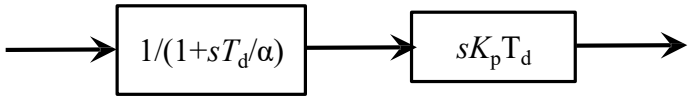
$$sK_{\mathrm{p}}T_{\mathrm{d}} \longrightarrow \frac{sK_{\mathrm{p}}T_{\mathrm{d}}}{1+sT_{\mathrm{d}}/\alpha}, \quad 3 \le \alpha \le 20$$

- Modified D control:

$$U_{\mathrm{d}}(s) = \frac{sK_{\mathrm{p}}T_{\mathrm{d}}}{1+sT_{\mathrm{d}}/\alpha}E(s) \Leftrightarrow \frac{T_{\mathrm{d}}}{\alpha}\frac{\mathrm{d}u_{\mathrm{d}}(t)}{\mathrm{d}t} + u_{\mathrm{d}}(t) = K_{\mathrm{p}}T_{\mathrm{d}}\frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

- Interpreted as the ideal derivative filtered by a first-order system (single capacity filter) with the time constant T_{d}/α
- Discretize it using backward difference:

$$u_{\mathrm{d}}(k) = \frac{T_{\mathrm{d}}u_{\mathrm{d}}(k-1) + K_{\mathrm{p}}T_{\mathrm{d}}\alpha[e_k - e_{k-1}]}{T_{\mathrm{d}} + \alpha T}$$



Low-pass filter Ideal D control

- **2. Suppressing large changes resulting from setpoint changes:** Use the output y_k instead of e_k in the D component

$$e_k \Longrightarrow -y_k$$

- A significant decrease in the controller output at the moment of a set point change.

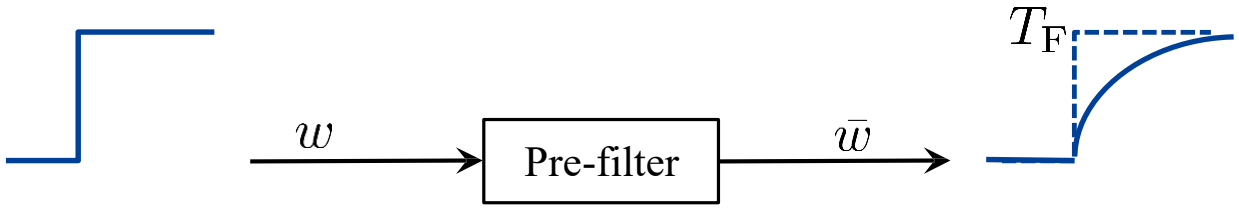
- Modified positon algorithm

$$u_k = P_{\mathrm{p}}(k) + P_{\mathrm{i}}(k) + K_{\mathrm{p}}\frac{T_{\mathrm{d}}}{T}(-y_k + y_{k-1})$$

- Modified incremental algorithm

$$\Delta u_k = K_{\mathrm{p}}\left(e_k - e_{k-1} + \frac{T}{T_{\mathrm{i}}}e_k + \frac{T_{\mathrm{d}}}{T}(-y_k + 2y_{k-1} - y_{k-2})\right)$$

- **Smoothing the setpoint changes**



$$\bar{w}_i = \sigma \bar{w}_{i-1} + (1 - \sigma) w_i$$

- where $\sigma = \exp(-T/T_F)$, T_F is the time constant of the pre-filter.
- This strategy changes the set point gradually by ramping it to the new value, thus limits the rate of change of the set point and reduces the derivative kick.

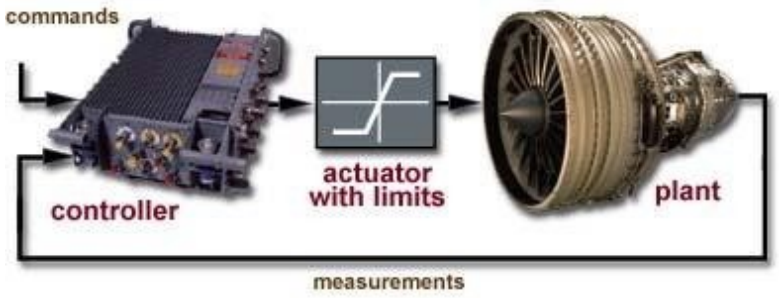
- Mix all the ingredients together
  - **(F)** Filtration of D control;
  - **(S)** Suppression of large change due to setpoint change;
  - **(SW)** Setpoint weighting for P control (accounting for proportional kick)
- Modified digital PID controller

$$u_k = P_{\mathrm{p}}^m(k) + P_{\mathrm{i}}(k) + P_{\mathrm{d}}^m(k)$$

$P_{\mathrm{p}}^m(k) = K_{\mathrm{p}}(\beta w_k - y_k),\ 0 \le \beta \le 1$	**(SW)** for P control to reduce the maximum overshoot of the process output
$P_{\mathrm{i}}(k) = K_{\mathrm{p}}\frac{T}{T_{\mathrm{i}}}\sum_{j=1}^{k} e_j = P_{\mathrm{i}}(k-1) + K_{\mathrm{p}}\frac{T}{T_{\mathrm{i}}}e_k$	No change
$P_{\mathrm{d}}^m(k) = \frac{T_{\mathrm{d}}}{T_{\mathrm{d}}+\alpha T}P_{\mathrm{d}}^m(k-1) + \frac{K_{\mathrm{p}}T_{\mathrm{d}}\alpha}{T_{\mathrm{d}}+\alpha T}[-y_k + y_{k-1}]$	**(F)** to limit the amplification of high-frequency noise and **(S)** to damp the response to the setpoint change

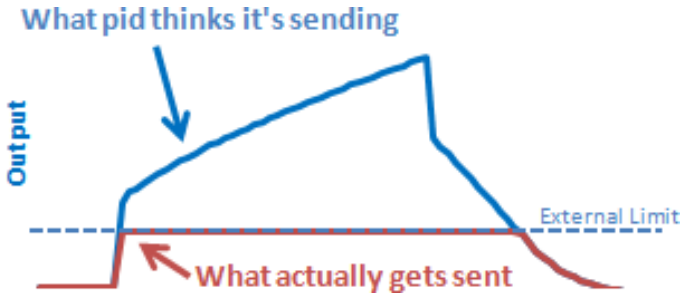
Practical Issues Related to Implementation-3/4

- **Control saturation:** the control signal exceeds actuator's physical limits, e.g., a motor has limited speed, a valve cannot be more than fully opened or fully closed



- Saturation function

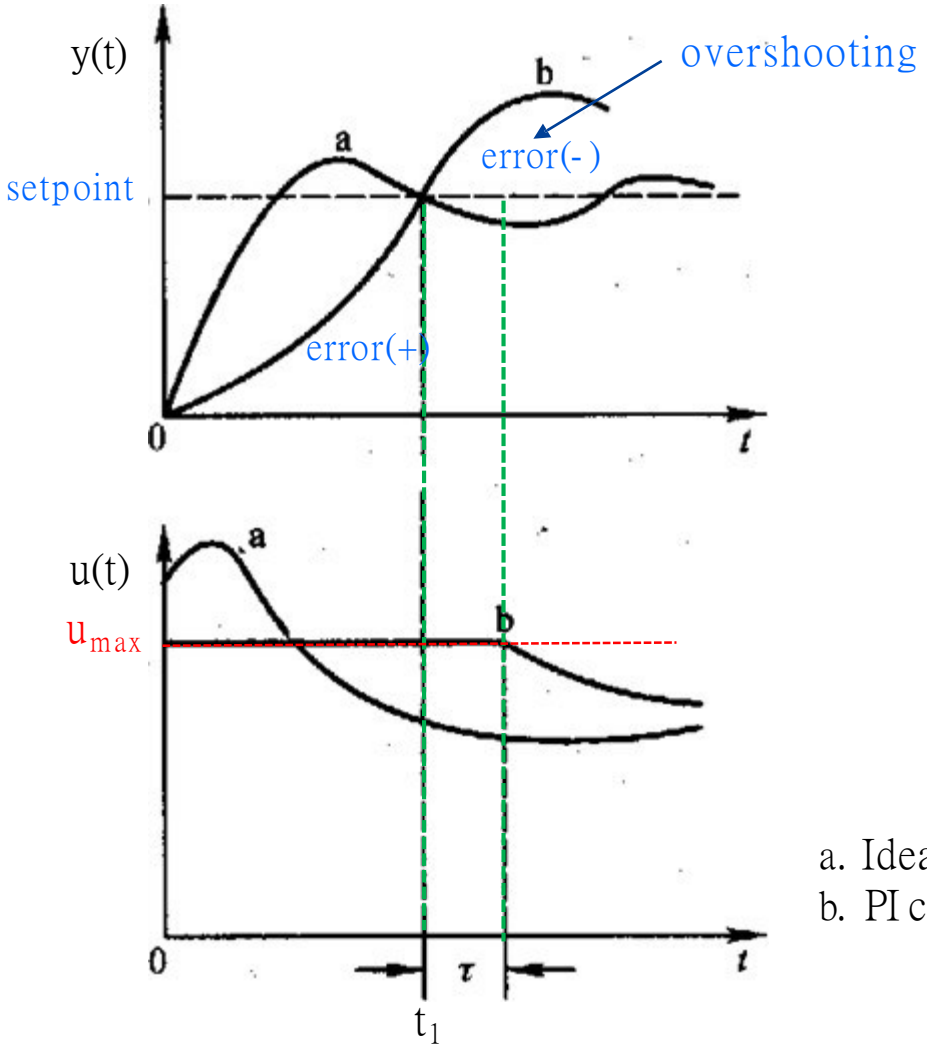
$$u_{sat}(t) = \begin{cases} u_{\max}, & \text{if } u > u_{\max} \\ u(t), & \text{if } u_{\min} \le u \le u_{\max} \\ u_{\min}, & \text{if } u < u_{\min} \end{cases}$$



- **Integral windup (reset windup):**

- For a control system with a wide range of operating conditions, it may happen that the control variable u_k reaches the actuator limits, i.e., the actuator saturates. In this case, the control variable u_k is independent of the feedback and the system runs as an open loop. If a controller with integrating action is used, the error will continue to be integrated. This means that the integral term may become very large or "winds up".
- The consequence is that any controller with integral action may give large transients when the actuator saturates.
- Windup is associated with the I component only.

- Illustration of integral windup



One specific problem of integral windup is
- ✓ the excess overshooting

a. Ideal PI controller
b. PI controller with saturation

Anti-windup Techniques

- All anti-windup algorithms must be based on the modification of the I component.
- Applies only to position form

$$u_k = K_\mathrm{p} e_k + K_\mathrm{i} \sum_{j=0}^{k} e_j + K_\mathrm{d}(e_k - e_{k-1})$$

- For the incremental form, no summation appears (no error accumulation), and thus the reset windup problem is avoided.

$$\Delta u_k = K_\mathrm{p}\left(e_k - e_{k-1} + \frac{T}{T_\mathrm{i}} e_k + \frac{T_\mathrm{d}}{T}(e_k - 2e_{k-1} + e_{k-2})\right)$$

- **Main idea of Conditional Integration method**: Switch off the integration at certain conditions.

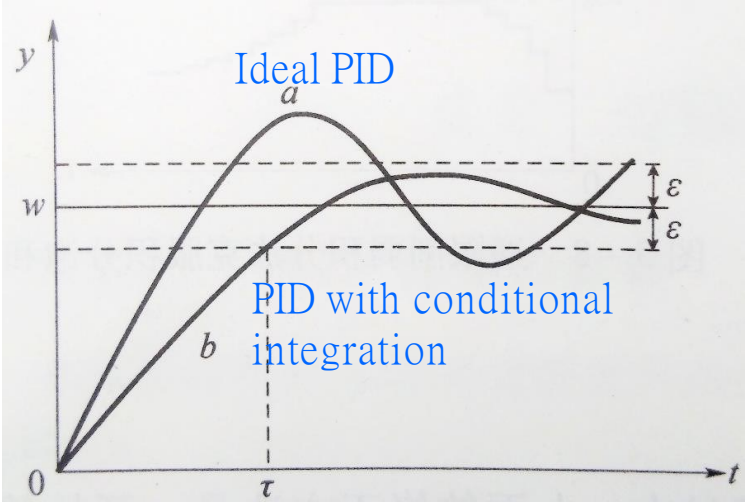
- **Conditional integration method 1:**
  - Stop summation when the control error is large, i.e., $|e_k| > \epsilon$

$$u_k = P_{\mathrm{p}}(k) + \chi_{e_k} P_{\mathrm{i}}(k) + P_{\mathrm{d}}(k)$$

  - where

$$\chi_{e_k} = \begin{cases} 0, & \text{if } |e_k| > \epsilon \\ 1, & \text{if } |e_k| \leq \epsilon \end{cases}$$

  - It has to be avoided that the controller gets stuck at the steady-state value (may be away from setpoint) such that the control error is still greater than the threshold ε.
  - ε has to be carefully selected in order to properly handle the trade-off between the need of avoiding integral windup and the need of assuring a zero steady-state error.



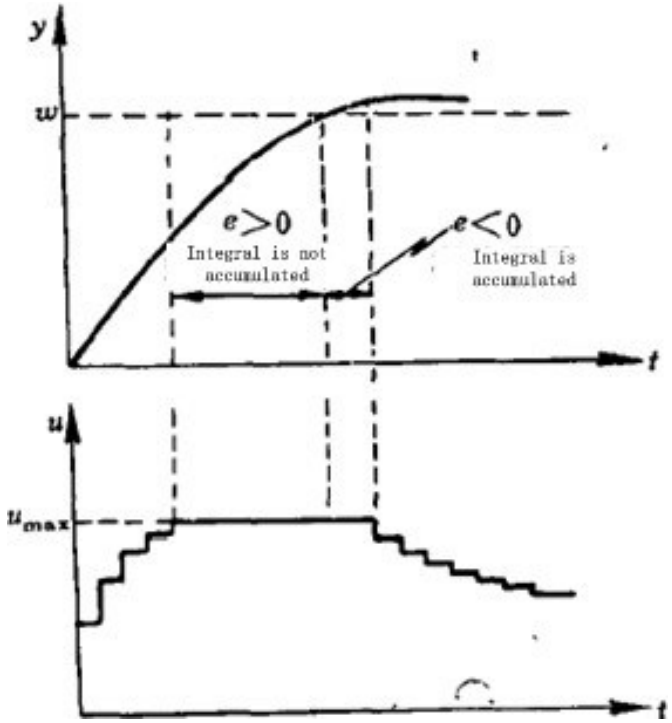
- **Conditional integration method 2:**
  - Stop summation when the control input saturates, and the control error and the control variable have the same sign.

$$u_k = P_{\mathrm{p}}(k) + \chi_{e_k} P_{\mathrm{i}}(k) + P_{\mathrm{d}}(k)$$

  - where

$$\chi_{e_k} = \begin{cases} 0, & \text{if } u_{k-1} \neq u_{sat}(k-1) \& u_{k-1} e_k > 0 \\ 1, & \text{otherwise} \end{cases}$$

  - The condition $u_{k-1} e_k > 0$ implies that the control increases rather than corrects the error due to windup.
  - If the actuator saturates, only add those errors that decrease the integral action.



- **Effective error method:** Re-calculate the value of e_k that just causes the actuator to saturate. When saturation occurs, use this value as the error term e_{k-1} in the next controller calculation.
- If the actuator saturates at k, i.e.,

$$u_k > u_{\max} \quad \text{or} \quad u_k < u_{\min}$$

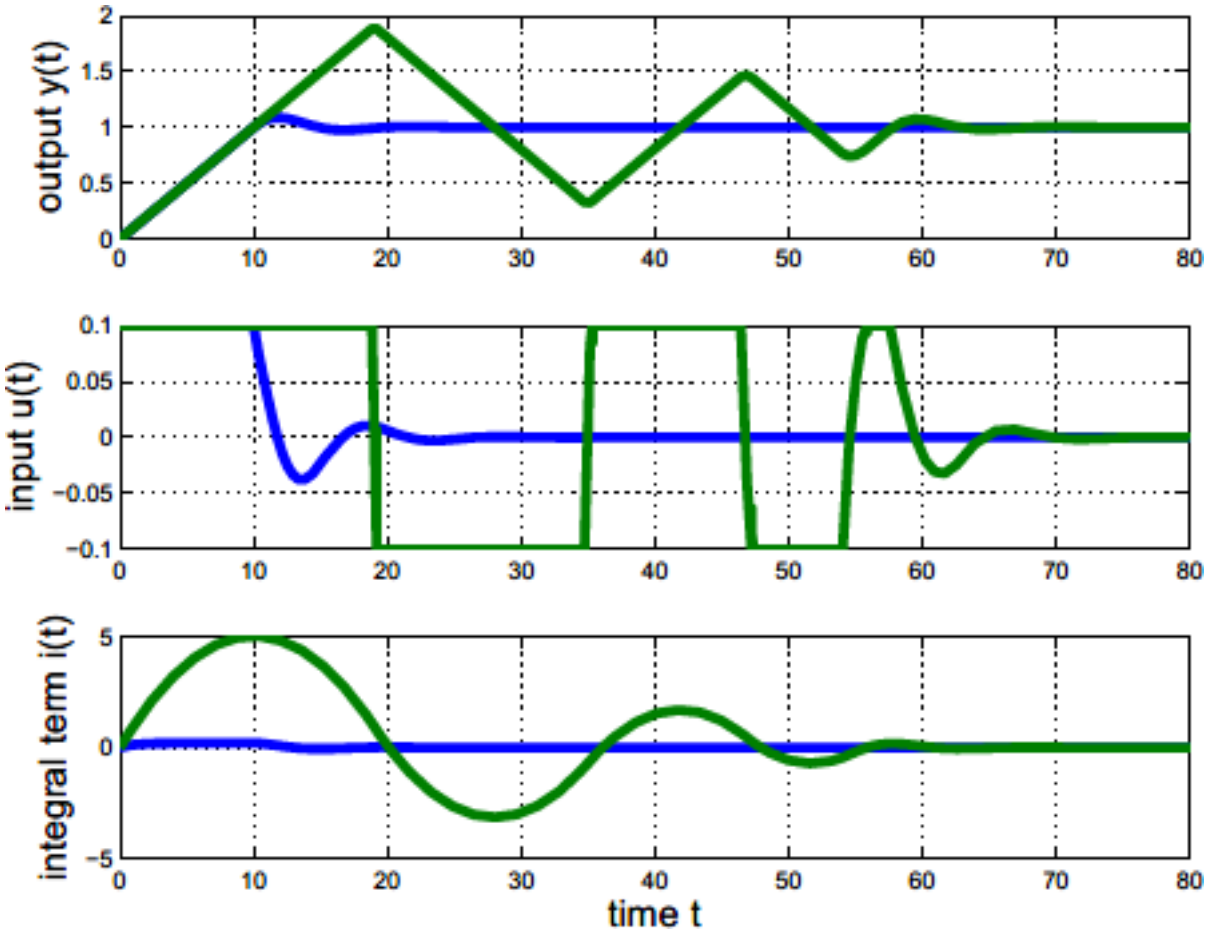
- then calculate the effective error

$$e_k \leftarrow \frac{\frac{1}{K_\mathrm{p}}u^* - \frac{T}{T_\mathrm{i}}\sum_{j=1}^{k-1} e_j + \frac{T_\mathrm{d}}{T}e_{k-1}}{1 + \frac{T}{T_\mathrm{i}} + \frac{T_\mathrm{d}}{T}}$$

- where $u* = u_{\max}$ or $u_{\min}$.
- Update of the control action

$$u_{k+1} = K_\mathrm{p}\left(e_{k+1} + \frac{T}{T_\mathrm{i}}\sum_{j=1}^{k+1} e_j + \frac{T_\mathrm{d}}{T}(e_{k+1} - e_k)\right)$$

- Difference between having and not having an anti-windup scheme



Note that in case of windup we have

- ✓ strong output oscillations
- ✓ a longer time to reach the steady-state
- ✓ the peaks of control signal

Comparison between position PID and incremental PID

- Position PID algorithm
- Large accumulation error due to the summation of all historical signal error value

$$\sum_{j=1}^{k} e_j$$

- Great misoperation effect, if the computer fails.
- Integral windup phenomenon occurs when the controller output saturates.

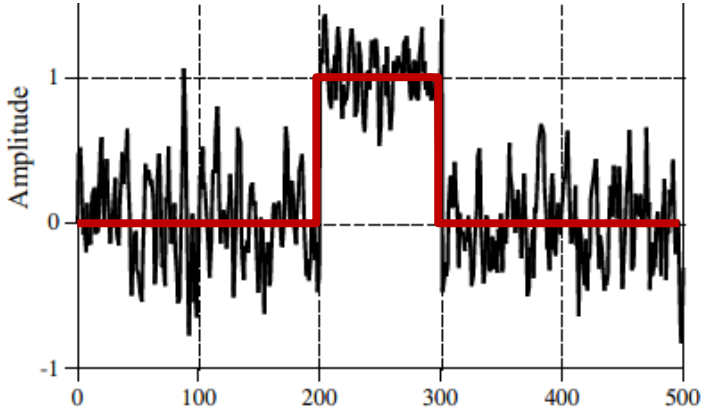
- Incremental PID algorithm
- Useful if the actuator is controlled by an incremental signal, e.g., stepping motor.
- No accumulation error.
- It inherently contains anti-windup, since no summation of errors.
- Transferring the controller from manual to automatic model does not require any initialization of the output.
- A minor disadvantage is that the Imode must be included.

$$u_k = K_{\mathrm{p}}\left(e_k + \frac{T}{T_{\mathrm{i}}}\sum_{j=1}^{k} e_j + \frac{T_{\mathrm{d}}}{T}(e_k - e_{k-1})\right)$$

$$\Delta u_k = K_{\mathrm{p}}\left(e_k - e_{k-1} + \frac{T}{T_{\mathrm{i}}}e_k + \frac{T_{\mathrm{d}}}{T}(e_k - 2e_{k-1} + e_{k-2})\right)$$

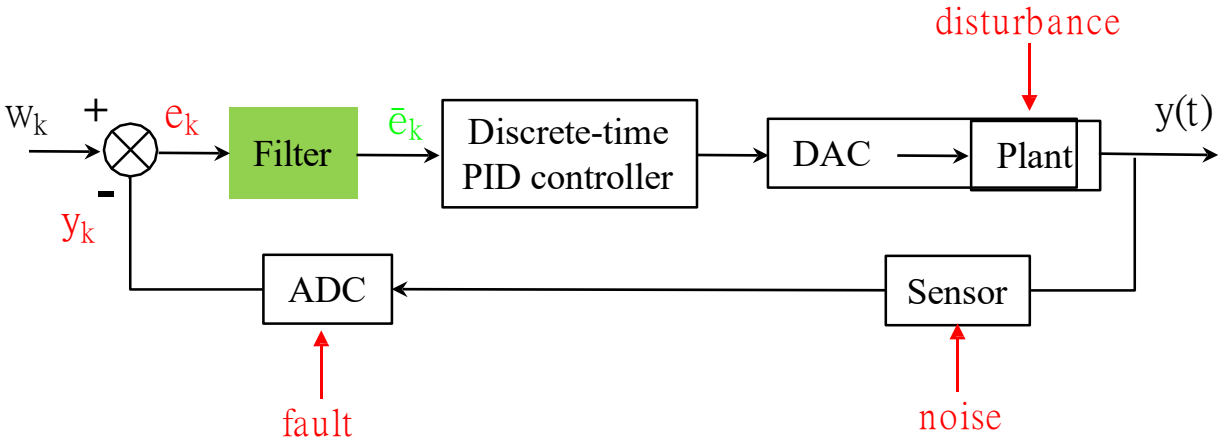
Practical Issues Related to Implementation-4/4

- **Noise:** May arise from the measurement device, e.g., sensor, or electronic equipment, or the process itself.



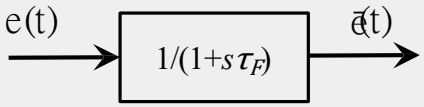
Rectangular signal disrupted by noise

- The error signal must be filtered before calculating the P-I-D components.



Digital Filters

- **Exponential filter:** Discrete-time version of the low-pass filter introduced earlier.



Low-pass filter

Analog filter:

$$\bar{e}(t) + \tau_F \frac{\mathrm{d}\bar{e}(t)}{\mathrm{d}t} = e(t)$$

where τ_F is the time constant of the filter.

- Discrete-time version:

$$\bar{e}_k = \frac{\tau_F}{T + \tau_F}\bar{e}_{k-1} + \frac{T}{T + \tau_F}e_k$$

- The filtered measurement $\bar{e}_k$ is a weighted sum of the filtered value at the previous sampling instant $\bar{e}_{k-1}$ and the current measurement e_k.

- **Moving average filter:** Average a specified number of past data points, giving equal weight to each data point.

$$\bar{e}_k = \frac{1}{N} \sum_{j=k-N+1}^{k} e_j$$

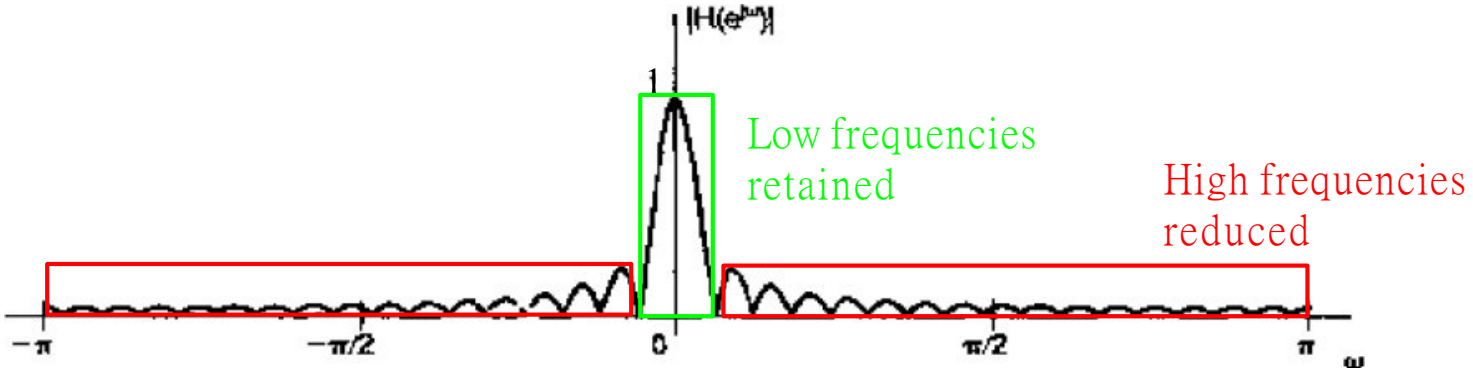
- Recursive form: Do not need to store all previous data

$$\bar{e}_k = \bar{e}_{k-1} + \frac{1}{N}[e_k - e_{k-N}]$$

- A moving average is a type of convolution and so it can be viewed as an example of a low-pass filter used in signal processing (?), which thus eliminates high-frequency noise.

- **Moving average filter is a low-pass filter!** (see Oppenheim et al. 1997)
- The filter's frequency response

$$H(e^{j\omega}) = \frac{1}{N}\sum_{k=0}^{N-1} e^{-j\omega k}$$
$$= \frac{1}{N} e^{-j\omega(N-1)/2} \frac{\sin(\omega N/2)}{\sin(\omega/2)}$$



- **Introducing the forgetting factor** $0 < \lambda \leq 1$**:** Place more weight to the most recentdata, e.g., exponentially downweights older observations.

$$\bar{e}_k = \frac{1}{sum_\lambda}(\lambda^{N-1}e_{k-N+1} + \cdots + \lambda e_k)$$

$$= \frac{1}{sum_\lambda}\sum_{j=k-N+1}^{k}\lambda^{k-j}e_j$$

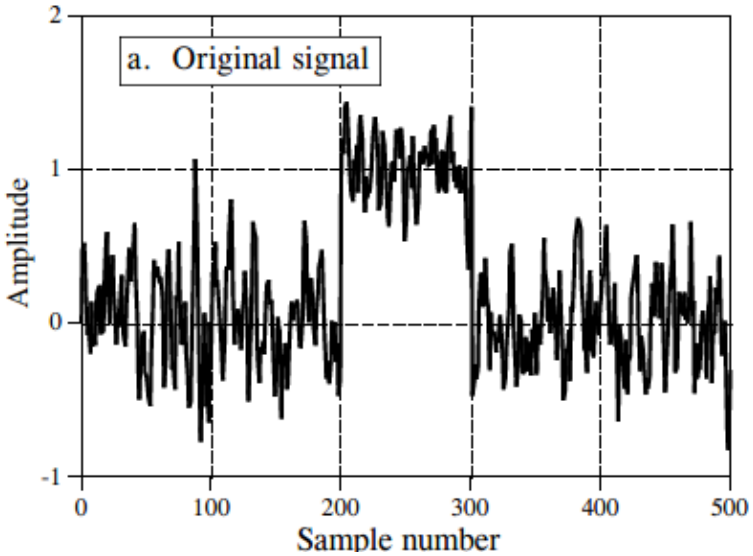
- where

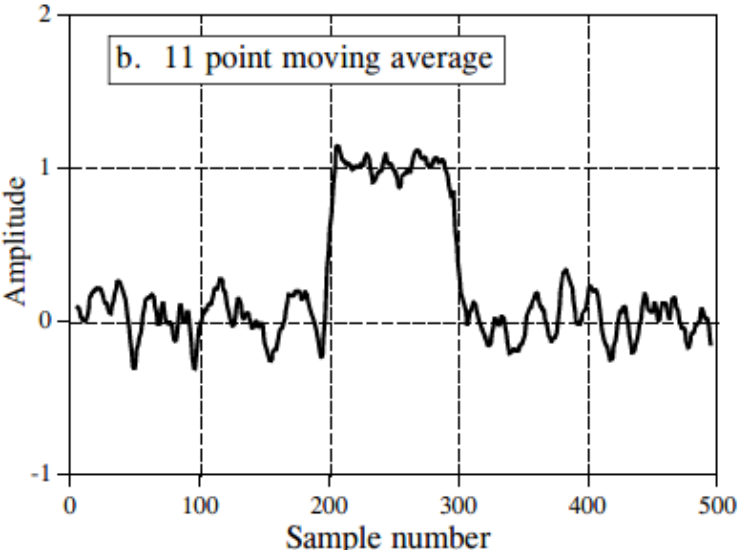
$$sum_\lambda = \sum_{j=k-N+1}^{k}\lambda^{k-j} = \sum_{j=0}^{N-1}\lambda^j$$

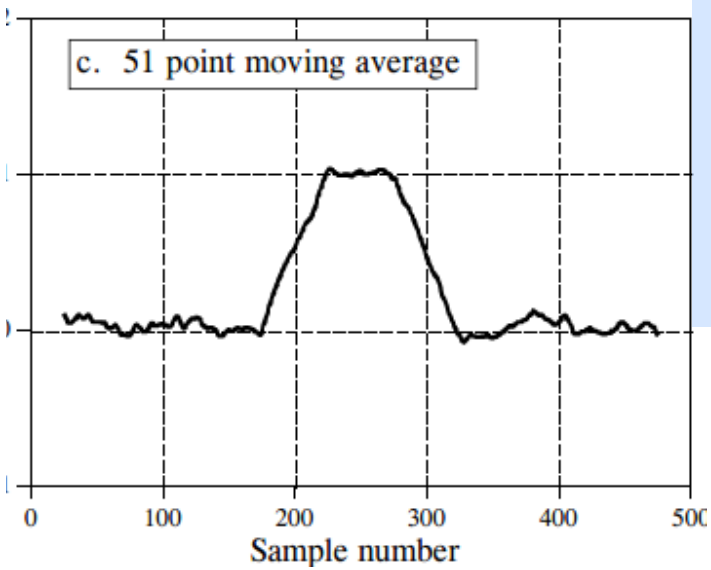
- It is a low-pass filter! More effective ($\lambda < 1$) than the moving average filter ($\lambda = 1$).
- Recursive form:

$$\bar{e}_k = \frac{1}{\lambda}\bar{e}_{k-1} + \frac{1}{sum_\lambda}(\lambda^{N-1}e_{k-N+1} - e_{k-1})$$

Rectangular signal disrupted by noise

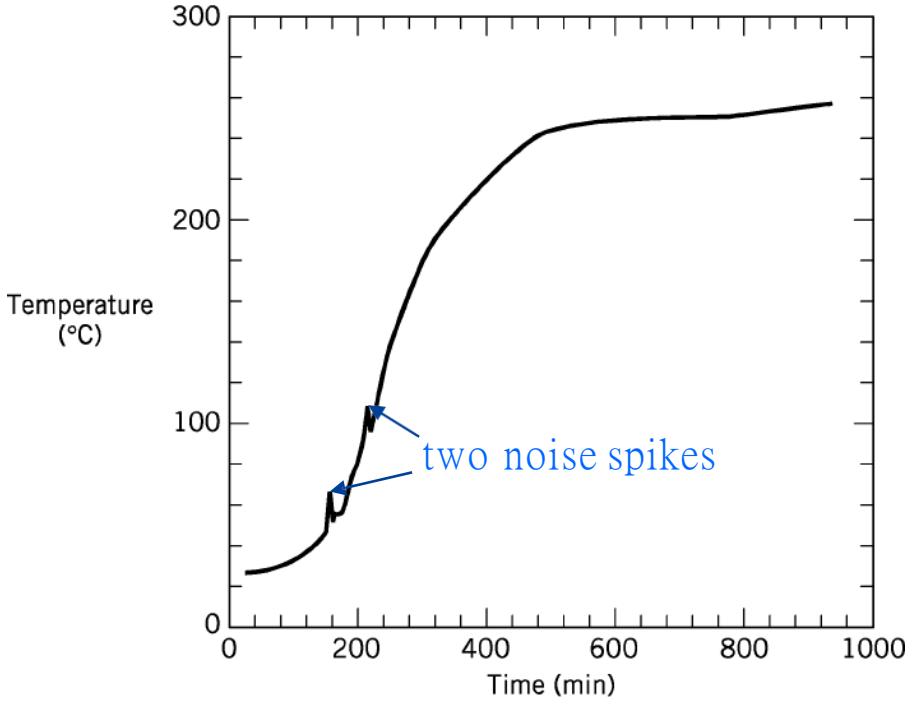






- ✓ As the number of points in the filter increases, the noise becomes lower; however, the edges becoming less sharp.
- ✓ Use small number of past data points results in faster response but larger variation.

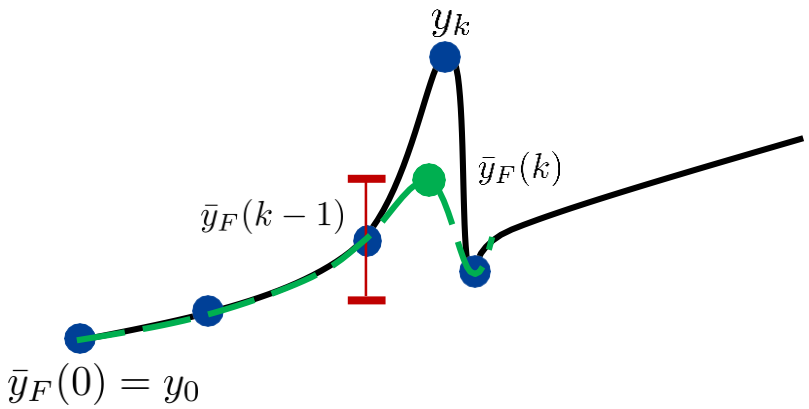
- **Noise spike:** If a noisy measurement changes suddenly by a large amount and then returns to the original value (or close to it) at the next sampling instant, a noise spike is said to occur.



- If noise spikes are not removed by filtering before the noisy measurement is sent to the controller, the controller will produce large, sudden changes in the manipulated variable.

- **Noise-spike filter(rate-of-change filter):** If a large change in the measurement occurs, the filter replaces the measurement by the previous filter output plus (or minus) the maximum allowable change Δ.

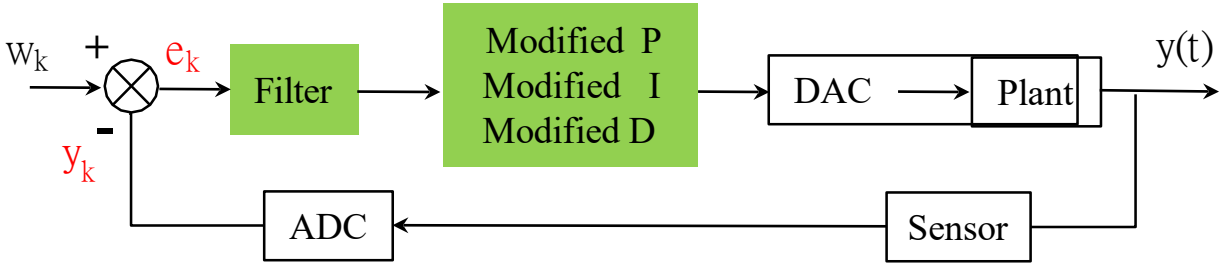
$$\bar{y}_F(k) = \begin{cases} y_k, & \text{if } |y_k - \bar{y}_F(k-1)| \le \Delta \\ \bar{y}_F(k-1) - \Delta, & \text{if } y_k - \bar{y}_F(k-1) < -\Delta \\ \bar{y}_F(k-1) + \Delta, & \text{if } y_k - \bar{y}_F(k-1) > \Delta \end{cases}$$



- This filter can also be used to detect instrument malfunctions such as a power failure, a break in a thermocouple or instrument line, or an ADC "glitch."

Summary

- Digital PID controllers:
  - Position form
  - Incremental form
- Practical issues in implementing PID controllers and remedies:
  - Proportional kick: Setpoint weighting
  - Derivative kick: Filtering the D action; suppressing changes resulting from setpoint changes; setpoint weighting
  - Integral windup: Conditional integration; back-calculation
  - Noise: Exponential filter; moving average filter; noise-spike filter



References

- V. Bobál, J. Böhm, J. Fessl, and J. Macháček, **Digital Self-tuning Controllers: Algorithms, Implementaion and Applications**. London: Springer-Verlag, 2005.
- Genke Yang, Jianying Xie, **Micro-Computer Control Technology**, 4th ed. Changsha: National Defense Industry Press, 2016 (in Chinese).
- D. E. Seborg, T. F. Edgar, D. A. Mellichamp, and F. J. Doyle Ⅲ, **Process Dynamics and Control**, 3rd ed. Hoboken, NJ: John Wiley & Sons, 2011.
- K.J. Åström, and T. Hägglund, **PID Controllers: Theory, Design, and Tuning**, 2nd ed. Research Triangle Park, NC: Instrument Society of America, 1995.
- A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, **Signals and Systems**, 2nd ed. Prentice-Hall, 1997.

Thanks for your attention!

ShyZhu（善迎）

Shanghai Minhang

E-mail: shyzhu@sjtu.edu.cn

Wechat ID: S14528707