New PID designs for sampling control and batch process optimization

Tao Liu

Institute of Advanced Control Technology
Dalian University of Technology
Outline

1. Introduction
2. PID design for sampled control systems
3. Learning PID design for batch optimization
4. Conclusions & Outlook
Introduction

Some facts of PID controllers

Advantages:
- Simple structure and low cost;
- Easily understood and commanded by users;
- Most widely used and commercialized in industrial applications.

Disadvantages [1, 2]:
- Generally not optimal in control performance;
- (often used as an inferior to demonstrate other advanced control designs)
- Difficult to analyze the robust stability against system uncertainties.
- (lowly valued for theoretical contribution in top control-relevant journals)

PID Controller: >1000 papers in 2017

PI Controller: >400 papers in 2017

More applications for various systems

Steadily increasing over the past 30 years!

Motivation of this talk

The internal model control (IMC) structure

Process model:
\[ \hat{G} = \frac{B(z)}{A(z)} z^{-d} \]

Frequency domain:
\[ \hat{G} = \frac{B(s)}{A(s)} e^{-\theta s} \]

IMC: applicable for stable processes

Equivalent relationship
\[ K = \frac{C}{1 - \hat{G}C} \]

PID: applicable for stable, integrating, and unstable processes
Review of the IMC design in frequency domain

Step 1. Decompose the model into the minimum-phase (MP), non-MP, and all-pass parts.

For example: 

\[
\hat{G} = \frac{k_p(-\tau_0s + 1)}{s^2(\tau_1s + 1)(\tau_2s + 1)} e^{-\theta s}
\]

where \( \tau_0 > 0, \tau_1 > 0, \tau_2 > 0 \).

\[
\hat{G} = \hat{G}_{mp} \hat{G}_{nmp} \hat{G}_{ap}
\]

\[
\hat{G}_{ap} = e^{-\theta s}
\]

\[
\hat{G}_{mp} = \frac{k_p}{s^2(\tau_1s + 1)(\tau_2s + 1)}
\]

\[
\hat{G}_{nmp} = -\tau_0s + 1
\]

Step 2. Specify the desired closed-loop transfer function for set-point tracking.

\[
T = F_1 F_2 \hat{G}_{nmp} \hat{G}_{ap}
\]

where \( F_1 \) and \( F_2 \) are two low-pass filters.

For a stable process described as above,

\[
F_1 = \frac{1}{(\lambda s + 1)^3}
\]

\[
F_2 = \frac{1}{\tau_0s + 1}
\]

\[
T = \frac{1}{(\lambda s + 1)^3} \frac{-\tau_0s + 1}{\tau_0s + 1} e^{-\theta s}
\]

Desired closed-loop transfer function

single tuning parameter \( \lambda \)

No overshoot

Minimal ISE
Discrete-time domain IMC design

Step 1. Decompose the model into the minimum-phase (MP), non-MP, and all-pass parts.

For example:

\[ \hat{G} = \frac{k_p(z-b_1)(z-b_2)}{(z-a_1)(z-a_2)} z^{-d} \]

where \( |a_1| < 1, |a_2| < 1, |b_1| < 1, |b_2| > 1 \).

\[ \hat{G} = \hat{G}_{mp} \hat{G}_{nmp} \hat{G}_{ap} \]

\[ \hat{G}_{mp} = \frac{k_p(z-b_1)}{(z-a_1)(z-a_2)} \]

\[ \hat{G}_{nmp} = z - b_2 \]

Step 2. Specify the desired closed-loop transfer function for set-point tracking.

\[ T = F_1 F_2 \hat{G}_{nmp} \hat{G}_{ap} \]

where \( F_1 \) and \( F_2 \) are two low-pass filters.

For a stable process described as above,

\[ F_1 = \frac{(1 - \lambda_c)^2}{(z - \lambda_c)^2} \]

ensure \( F_1 \hat{G}_{mp} \) strictly proper

\[ F_2 = \frac{1 - b_2^{-1}}{(1 - b_2)(z - b_2^{-1})} \]

ensure \( F_2 \hat{G}_{nmp} \) all-pass, i.e.

\[ (1 - b_2^{-1})(z - b_2) \]

\[ (1 - b_2)(z - b_2^{-1}) \]

Desired closed-loop transfer function

\[ T = \frac{(1 - \lambda_c)^2}{(z - \lambda_c)^2} \frac{(1 - b_2^{-1})(z - b_2)}{(1 - b_2)(z - b_2^{-1})} z^{-d} \]
Discrete-time domain IMC design

Step 3. Determine the IMC controller.

\[ C_{\text{IMC}}(z) = \frac{T(z)}{\hat{G}(z)} \]

For a stable process described as above,

\[
C_{\text{IMC}}(z) = \frac{(z - a_1)(z - a_2)(1 - \lambda_c)^2}{k_p (z - b_1)(z - \lambda_c)^2} \frac{(1 - b_2^{-1})}{(1 - b_2)(z - b_2^{-1})}
\]

single tuning parameter \( \lambda_c \)

In case \(-1 < b_1 < 0\) , it could provoke \textbf{inter-sample ringing} in the control signal!
Discrete-time domain IMC design

Solution: Introduce another filter to remove such a zero for implementation

\[ F_3(z) = z^{-1} \frac{z - b_1}{1 - b_1} \quad \Rightarrow \quad C_{\text{RIMC}}(z) = F_3(z)C_{\text{IMC}}(z) \]

Control performance assessment

For a stable process described by

\[ G_1(z) = \frac{K_p}{z - z_p} z^{-d} \]

\[ R(z) = \frac{z}{z - 1} \]

\[ T_d(z) = \frac{1 - \lambda_c}{z - \lambda_c} \]

\[ Y(z) = \frac{1 - \lambda_c}{z - \lambda_c} z^{-d} R(z) \]

Step response in time domain

\[ y(kT_s) = \begin{cases} 0, & k \leq d, \\ 1 - \lambda_c^{k-d}, & k > d. \end{cases} \]

Set-point tracking error

\[ IAE_r = \sum_{k=d+1}^{\infty} [1 - y(kT_s)] = \sum_{k=d+1}^{\infty} \lambda_c^{k-d} = \lim_{n \to \infty} \frac{\lambda_c (1 - \lambda_c^n)}{1 - \lambda_c} = \frac{\lambda_c}{1 - \lambda_c} \]

Disturbance rejection error to a step change

\[ IAE_d = \sum_{k=d+1}^{\infty} y(kT_s) = \frac{K_p}{(1 - \lambda_c)(1 - z_p)} \]

Quantitative tuning
IMC-based PID design in frequency domain

Step 4. Determine the equivalent controller in a unity feedback control structure.

\[
K = \frac{C}{1 - \hat{G}C}
\]

For \( \hat{G} = \frac{k_p(-\tau_0s + 1)}{(\tau_1s + 1)(\tau_2s + 1)}e^{-\theta s} \)

\[
K = \frac{(\tau_1s + 1)(\tau_2s + 1)}{k_p[(\lambda s + 1)^3(\tau_0s + 1) - (-\tau_0s + 1)e^{-\theta s} ]}
\]

Step 5. Determine the PID controller by the Taylor approximation [1].

Since \( \lim_{s \to 0} K(s) = \infty \), let \( K(s) = \frac{M}{s} \)

\[
K(s) = 1 \left[ M(0) + M'(0)s + \frac{M''(0)}{2!}s^2 + \cdots \right]
\]

\[
K(s) = k_C + \frac{1}{\tau_I s} + \frac{\tau_D s}{\tau_F s + 1}
\]

\[
\begin{align*}
\tau_F &= (0.01 - 0.1)\tau_D \\
k_C &= M'(0) \\
\tau_I &= 1/M(0) \\
\tau_D &= M''(0)/2
\end{align*}
\]

IMC-based PID design in frequency domain

Alternatively, the delay term in the process model or the IMC controller may be rationally approximated to derive a PID, e.g.,

The first-order Taylor approximation [2]:

\[ e^{-\theta s} \approx 1 - \theta s \]

Pade approximation [3]:

\[ e^{-\theta s} \approx \left(1 - \frac{\theta}{2} s\right) / \left(1 + \frac{\theta}{2} s\right) \]

Model reduction [4, 5]:

\[
\frac{1}{(\tau_1 s + 1)(\tau_2 s + 1)} \approx \frac{1}{(\tau_1 + \frac{\tau_2}{2})s + 1} e^{-\frac{\tau_2}{2}s} \quad \text{‘Half rule’}
\]

Frequency response fitting [6]:

\[
\min \sum_{i=1}^{m} |K_{\text{PID}}(j \omega_i) - K(j \omega_i)| \quad \omega_i \in (0.1, 1)\omega_{cb}
\]

---


IMC-based PID design in discrete-time domain

For sampling control implementation, the first-order differentiation is generally used to discretize the above frequency domain design.

For direct PID design in discrete-time domain, let the equivalent controller in a unity feedback control structure be

\[
K(z) = \frac{M(z)}{z-1}
\]

owing to \( \lim_{z \to 1} K(z) = \infty \).

A PID controller is determined by the Taylor approximation \([1, 2]\).

\[
K(z) = \frac{1}{z-1} \left[ M(1) + M'(1)(z-1) + \frac{M''(1)}{2!}(z-1)^2 + \ldots \right]
\]

\[
K_{\text{PID}}(z) = k_c \left[ 1 + \frac{1}{\tau_1(z-1)} + (1-\alpha\tau_D) \frac{\tau_D(z-1)}{z-\alpha\tau_D} \right]
\]

\[
\begin{align*}
k_c &= M'(1) \\
\tau_1 &= M'(1) / M(1) \\
\tau_D &= M''(1) / 2M'(1)
\end{align*}
\]

Single tuning parameter \( \lambda \)

\[\alpha \in (0.01, \ 0.1)\]

IMC-based PID design in discrete-time domain

Closed-loop system robust stability analysis

The $T - \Delta$ structure

Small gain theorem

$$\|T(z)\Delta_m(z)\|_\infty < 1$$

$$T(z) = \frac{G_p(z)C_{PID}(z)}{1 + G_p(z)C_{PID}(z)}$$

$$\Delta_m(z) = \left[ G(z) - G_m(z) \right] / G_m(z)$$

Magnitude/Phase margin, Maximal sensitivity index $M_s$

For a stable process described by

$$G_1(z) = \frac{k_p}{z - \tau}$$

No analytical solution!

For

$$\frac{k_p}{z - \tau} z^{-d} K \left[ 1 + \frac{1}{T_i(z - 1)} + (1 - \alpha T_d) \frac{T_d(z - 1)}{z - \alpha T_d} \right] \left[ 1 + \frac{1}{T_i(z - 1)} + (1 - \alpha T_d) \frac{T_d(z - 1)}{z - \alpha T_d} \right]$$
**IMC-based PID design in discrete-time domain**

**Disturbance rejection performance**

\[ G(z) = \frac{0.0065904(z + 0.9222)}{(z - 0.8669)(z - 0.9048)} z^{-d} \]

**Disturbance response peak**

**Tuning guideline:** The single adjustable parameter \( \lambda_c \) can be monotonically increased or decreased in a range of \( \lambda_c \in (0.8, 1) \) to meet a good trade-off between the closed-loop control performance and its robust stability.
IMC-based PID design in discrete-time domain

An illustrative example:

\[ G(s) = \frac{1}{s + 1} e^{-0.25s} \]

Sampling period: \( T_s = 0.02(s) \)

\[ G(z) = \frac{0.0003947(z + 0.9868)}{(z - 1)(z - 0.9608)} z^{-25} \]

\[ C_{IMC}(z) = \frac{(z - 0.99)(1 - \lambda_c)}{0.00995(z - \lambda_c)} \]

\[ C_{PID}(z) = 1.7049 \left[ 1 + \frac{1}{105.2388(z - 1)} + \frac{2.4956(z - 1)}{z - 0.4791} \right] \]

Avoid differential kick!

Improved IMC-based PID design for disturbance rejection

For a slow process described by

\[ G = \frac{k_p e^{-s}}{\tau_p s + 1} \]

with a large time constant \( \tau_p \),

Discrete-time domain model: \( G(z) = \frac{k_p}{z - z_p} z^{-d} \) with a pole \( |z_p| < 1 \) close to the unit circle

Load disturbance transfer function

\[ \frac{y}{d_i} = G(1 - GC) \]
\[ \frac{y}{d_o} = G_d (1 - GC) \]

The slow pole of \( G \) or \( G_d \) affects the disturbance rejection performance!

Solution: Introduce an asymptotic constraint to remove the effect of slow dynamics.

Modify the desired transfer function:

\[ T_{RIMC} = \frac{(\alpha s + 1)e^{-\theta s}}{(\lambda \tau s + 1)^2} \]

\[ T_d(z) = \frac{(1 - \lambda_c)^{n_d} (\beta_0 + \beta_1 z)}{(z - \lambda_c)^{n_d}} \]

\[ \alpha = \tau_p [1 - \left( \frac{\lambda \tau}{\tau_p} - 1 \right)^2 e^{-\theta \tau_p}] \]

\[ \lim_{s \to -1/\tau_{RIMC}} (1 - T_{RIMC}) = 0 \]

\[ \lim_{s \to -1/\tau_{RIMC}} (1 - T_d) = 0 \]

Discrete-time domain:

\[ \lim_{z \to 1} (1 - T_d) = 0 \]
\[ \lim_{z \to -1/z_p} (1 - T_d) = 0 \]

\[ \begin{align*}
\beta_1 &= \frac{(z_p - \lambda_c)^{n_d} - (1 - \lambda_c)^{n_d}}{(z_p - 1)(1 - \lambda_c)^{n_d}} \\
\beta_0 &= 1 - \beta_1
\end{align*} \]
Improved IMC-based PID design for disturbance rejection

An illustrative example:

\[ G = \frac{e^{-s}}{(20s + 1)(2s + 1)} \]

Slow pole: \[ s = -\frac{1}{20} \]

Comparison with the standard IMC and SIMC by Skogestad (JPC, 2003)

Discrete-time PID design for integrating and unstable processes

Problem: The classical PID control structure could not suppress large overshoot for set-point tracking! There exists severe water-bed effect!

Solution: A two-degree-of-freedom (2DOF) control structure

Advantage: the set-point tracking is decoupled from load disturbance rejection.

Set-point tracking: open-loop control

IMC design: $T_r = GC_s$

Disturbance rejection: closed-loop control using a PID controller, $C_f$

The desired transfer function for set-point tracking is the same as the standard IMC design

The desired transfer function for disturbance rejection:

$$T_d(z) = (\beta_0 + \beta_1 z + \cdots + \beta_m z^m) \frac{(1 - \lambda_f)^{m+1}}{(z - \lambda_f)^{m+1}} z^{-d}$$
Discrete-time PID design for integrating and unstable processes

Integrating process:

\[ \hat{G}_I(z) = \frac{k_p(z - z_0)}{(z - 1)(z - z_p)} z^{-d} \]

Unstable process:

\[ \hat{G}_U(z) = \frac{k_p(z - z_0)}{(z - z_u)(z - z_p)} z^{-d} \]

The following asymptotic tracking constraints must be satisfied,

\[ \lim_{z \to 1} (1 - T_d) = 0 \]

\[ \lim_{z \to \eta} (1 - T_d) = 0 \quad \eta = z_u \text{ or } \eta = z_p \quad \text{(close to the unit circle)} \]

\[ \lim_{z \to 1} \frac{d}{dz} (1 - T_d) = 0 \]

\[ T_d = \frac{\hat{G}C_f}{1 + \hat{G}C_f} \]

Derive the closed-loop controller:

\[ C_f = \frac{T_d}{1 - T_d} \cdot \frac{1}{G} \quad \text{Taylor approximation} \quad \text{PID} \]
Discrete-time PID design for integrating and unstable processes

Integrating process:

\[ G(s) = \frac{0.1e^{-5s}}{s(5s+1)} \]

Frequency domain design:

\[ C_s(s) = \frac{s(5s+1)}{0.1(\lambda_c s + 1)^2} \]
\[ T_r(s) = \frac{e^{-5s}}{(\lambda_c s + 1)^2} \]
\[ C_f(s) = \frac{1}{G(s)} \cdot \frac{T_d(s)}{1 - T_d(s)} \]
\[ T_d(s) = \frac{\eta_2 s^2 + \eta_1 s + 1}{(\lambda_f s + 1)^4} e^{-5s} \]
\[ \eta_1 = 4\lambda_f + 5 \]
\[ \eta_2 = 5\eta_1 + 25(0.2\lambda_f - 1)^4 e^{-1} - 25 \]

Discrete-time PID design for integrating and unstable processes

Sampling period: \( T_s = 0.2(s) \)

\[
G(z) = \frac{0.0003947(z + 0.9868)}{(z - 1)(z - 0.9608)} z^{-25}
\]

\[
C_s(z) = \frac{(z - 1)(z - 0.9608)(1 - \lambda_c)^2}{0.0007842(z - \lambda_c)^2}
\]

\[
T_d(z) = \frac{(\beta_0 + \beta_1 z + \beta_2 z^2)(1 - \lambda_f)^4}{(z - \lambda_f)^4} z^{-25}
\]

\[
\beta_1 = d + \frac{4}{1 - \lambda_f} - 2\beta_2
\]

\[
\beta_2 = \frac{(\tau_p - \lambda_f)^4 \tau_p^d + 4(1 - \lambda_f)^3(1 - \tau_p) + (d - d\tau_p - 1)(1 - \lambda_f)^4}{(1 - \lambda_f)^4(\tau_p - 1)^2}
\]

\[
C_t(z) = \frac{(1 - \lambda_f)^4(\beta_0 + \beta_1 z + \beta_2 z^2)(z - 1)(z - \tau_p)}{k_p z(1 - \lambda_f)^4(1 - \lambda_f)^4 z^{-25}}
\]

Perturbation: the process gain and time constant are actually 20% larger

Discrete-time PID design for long time delay processes

Predictor based control structure


Discrete-time PID design for long time delay processes

The nominal system transfer function

\[ y(z) = K_f(z) \frac{K(z)G_p(z)}{1 + K(z)\hat{G}(z)} z^{-d_p} r(z) + \frac{G_p(z)}{1 + K(z)\hat{G}(z)} \left[1 + K(z)F_1(z)\right] z^{-d_p} w(z) \]

Closed-loop transfer function for disturbance rejection

\[ \frac{u(z)}{w(z)} = T_d(z) = \frac{K(z)G(z)}{1 + K(z)G(z)} \]

The desired closed-loop transfer function (free of time delay)

\[ T_d(z) = \frac{(1 - \lambda_c)^{n_d}}{(z - \lambda_c)^{n_d}} \sum_{i=0}^{l} \beta_i z^i \]

\[ \sum_{i=0}^{l} \beta_i = 1 \]

Taylor approximation

\[ K(z) = \frac{T_d(z)}{1 - T_d(z)} \cdot \frac{1}{G_p(z)} \]

PID
Discrete-time PID design for long time delay processes

Example: \( G(s) = \frac{e^{-27.5s}}{52.5s + 1} \)

Sampling period: \( T_s = 0.5(s) \)

\( G(z) = \frac{0.009479}{z - 0.9905}z^{-55} \)

PI controller: \( K(z) = \frac{(1 - \lambda_c)^2(\beta_1 z + \beta_0)}{0.009479(z - 1)} \)

\( \beta_2 = 1 - \beta_1 \)
\( \beta_1 = (z_p - \lambda_c)^2 - (1 - \lambda_c)^2 / (z_p - 1)(1 - \lambda_c)^2 \)

\( K_f(z) = \frac{(z - \lambda_c)^2(1 - \lambda_f)^2z}{(1 - \lambda_c)^2(\beta_0 + \beta_1 z)(z - \lambda_f)^2} \)


Discrete-time PID design for long time delay processes

A temperature control system of a 4-liter jacketed reactor for pharmaceutical crystallization

Sampling period: \( T_s = 3 \text{s} \)

Step response identification:

Integrating type process model:

\[
G(s) = \frac{0.0004529}{s(760.40s + 1)} e^{-100.25s}
\]

\[
G(z) = \frac{2.6765 \times 10^{-6}(z + 0.9989)}{(z - 1)(z - 0.9961)} z^{-34}
\]

Closed-loop controller:

\[
K(z) = \frac{(1 - \lambda_f)^4(\beta_0 + \beta_1 z + \beta_2 z^2)}{4.9557 \times 10^{-6} z(z - 1)(z - 0.9899)}
\]

\[
\beta_0 = 1 - \beta_1 - \beta_2 \quad \beta_1 = 4/(1 - \lambda_f) - \beta_2
\]

\[
\beta_2 = \frac{(0.9961 - \lambda_f)^4}{(0.9961 - 1)^2(1 - \lambda_f)^4} - \frac{4}{(1 - 0.9961)(1 - \lambda_f)} - \frac{1}{(0.9961 - 1)^2}.
\]

Set-point filter:

\[
K_f(z) = \frac{z(z - \lambda_f)^4(1 - \lambda_s)^3}{(1 - \lambda_f)^4(\beta_0 + \beta_1 z + \beta_2 z^2)(z - \lambda_s)^3}
\]
Discrete-time PID design for long time delay processes

Heat up the aqueous solution from the room temperature (25°C) to 45°C. Load disturbance arises from feeding 200(ml) solvent of distilled water.

**Input constraint:** \( 0 \leq u \leq 100 \) regulate the heating power

Compared with the filtered PID method [21] based on delayed output feedback, more than 30 minutes are saved for recovering the solution temperature to the operation zone of \((45 \pm 0.1)\degree C\) against the load disturbance.

---


Our recent publications on discrete-time domain PID design


PID design for batch process optimization

Industrial batch processes, e.g., injection molding machines, chemical reactors

Features:
1. Repetitive operation for production;
2. Historical cycle information for progressively improving system performance;
3. Time or batch varying uncertainties.
PID design for batch process optimization

IMC-based iterative learning control (ILC) for perfect tracking using historical cycle information

Advantage: batch control optimization

---

PID design for batch process optimization

Type I: PID-based ILC system
(modify the input error for batch operation)

Type II: ILC-based PID control system
(controller retuning for batch operation)

PID design for batch process optimization

Indirect-type ILC based on the PI control structure

Advantage: No need to modify the closed-loop PI controller for ILC design, i.e., The PI controller and ILC updating law can be separately designed.

PID design for batch process optimization

Indirect-type ILC design based on the PI control structure and current output error

PID design for batch process optimization

Indirect-type ILC based on the PID control structure plus feedforward control

\[ P_\Delta : \begin{align*}
    x(t+1,k) &= [A_m + \Delta \hat{A}(t,k)] x(t,k) + [B_m + \Delta \hat{B}(t,k)] u(t,k) + \omega(t,k) \\
    y(t,k) &= C x(t,k), \quad 0 \leq t \leq T_p; \\
    x(0,k) &= x(0), \quad k=1,2,\ldots.
\end{align*} \]

Time index \( t \), Batch index \( k \), Batch period \( T_p \)

Robust PID tuning for indirect-type ILC

A PID control law in discrete-time domain

\[ u_{\text{PID}}(t) = k_p e(t) + k_i \sum e(t) + k_d [e(t + 1) - e(t)] \]

\[ e(t, k) = Y_r(t) - y(t, k) \]

approximate

\[ e(t) - e(t - 1) \approx \frac{[e(t) + e(t - 2) - 2e(t - 1)]}{2} \]

State-space closed-loop PID system description

\[
\begin{align*}
\begin{bmatrix} x(t + 1) \\ \sum e(t) \end{bmatrix} &= \begin{bmatrix} \tilde{A} + \tilde{B}[\hat{k}_d(CI - A_m) - \hat{k}_p C] & \tilde{B}\hat{k}_i \\ -C & I \end{bmatrix} \begin{bmatrix} x(t) \\ \sum e(t - 1) \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} \omega(t) \\
y(t) &= \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \sum e(t - 1) \end{bmatrix}
\end{align*}
\]

where

\[ \hat{A} = A_m + \Delta \tilde{A}(t) \]

\[ \hat{k}_p = (I + k_d CB_m)^{-1}(k_p + k_i) \]

\[ \hat{B} = B_m + \Delta \tilde{B}(t) \]

\[ \hat{k}_i = (I + k_d CB_m)^{-1}k_i \]

\[ \hat{k}_d = (I + k_d CB_m)^{-1}k_d \]
Robust PID tuning for indirect-type ILC

The H infinity control objective for closed-loop system robust stability

\[ \|e(t)\|_2 < \gamma_{\text{PID}} \|\omega(t)\|_2 \]

where \( \gamma_{\text{PID}} \) denotes the robust performance level.

Theorem 1: The PID control system is guaranteed robustly stable if there exist \( P_{11} > 0 \), \( P_{22} > 0 \) matrices \( P_{12}, R_1, R_2 \), and positive scalars \( \varepsilon_1, \varepsilon_2 \), such that

\[
\begin{bmatrix}
-P + \varepsilon_1 \Phi_{A1}^T \Phi_{A1} + \varepsilon_2 \Phi_{B1}^T \Phi_{B1} & \Gamma & D_g & 0 & 0 & 0 \\
* & -P & 0 & PH^T C^T & P\Phi_{A2}^T & P\Phi_{B2}^T \\
* & * & -\gamma_{\text{PID}} I & 0 & 0 & 0 \\
* & * & * & -\gamma_{\text{PID}} I & 0 & 0 \\
* & * & * & * & -\varepsilon_1 I & 0 \\
* & * & * & * & * & -\varepsilon_2 I \\
\end{bmatrix} < 0
\]

where \( D_g = [I \ 0]^T \), \( H = [I \ 0] \), \( \Phi_{A1} = [\Delta \bar{A}_1^T, 0]^T \), \( \Phi_{A2} = [\Delta \bar{A}_2 P_{11}, \Delta \bar{A}_2 P_{12}] \), \( \Phi_{B1} = [\Delta \bar{B}_1^T, 0]^T \), \( \Phi_{B2} = [\Delta \bar{B}_2 R_1, \Delta \bar{B}_2 R_2] \).
Robust PID tuning for indirect-type ILC

Correspondingly, the PID controller is determined by

\[
\begin{bmatrix}
\hat{k}_D (CI - A_m) - \hat{k}_p C \\
\hat{k}_1
\end{bmatrix} = [R_1 \quad R_2] P^{-1}
\]

\[k_1 = \hat{k}_1 (I + k_D CB_m)\]

\[k_p = \hat{k}_p (I + k_D CB_m) - k_1\]

where \(k_D\) is user specified for implementation. If \(k_D = 0\), it is a PI controller.

An optimal program for tuning PID to accommodate for the uncertainty bounds,

\[
\min_{\Delta \bar{A}(t), \Delta \bar{B}(t)} \gamma_{\text{PID}}
\]

**Guideline:** A smaller value of \(\gamma_{\text{PID}}\) leads to faster output response with a more aggressive control action, and vice versa.

---

Robust PI tuning for indirect-type ILC

Another robust tuning of PI controller by assigning the closed-loop system poles to a prescribed circular region, \( D(\alpha, r) \) centered at \( (\alpha, 0) \) with radius \( r \) and \( |\alpha| + r < 1 \), i.e.,

\[
\lambda(\tilde{A}) \subset D(\alpha, r)
\]

while the closed-loop transfer function \( H(z) = \tilde{C}(zI - \tilde{A})^{-1}\tilde{D} \) satisfies

\[
\| H(z) \|_\infty < \gamma_{PI}
\]

**Theorem 2:** The PI control system is guaranteed robustly D-stable if there exist matrices \( P_1 > 0 \), \( P_3 > 0 \), \( P_2 = P_2^T \), \( R_1 \), \( R_2 \), and positive scalar \( \varepsilon \), such that

\[
\begin{bmatrix}
\Lambda_1 & 0 & \Lambda_2 & P\hat{C}^T & 0 & P\hat{F}^T \\
* & -\beta_1^{-1}\gamma_{PI}^2 I & \hat{D}^T & 0 & \hat{D}^T & 0 \\
* & * & \Lambda_3 & 0 & 0 & 0 \\
* & * & * & -\beta_1^{-1}I & 0 & 0 \\
* & * & * & * & -\beta_2 P & 0 \\
* & * & * & * & * & -\varepsilon I \\
\end{bmatrix} < 0
\]

where \( \beta_1 = 1 - |\alpha| \), \( \beta_2 = (\beta_1^{-1} - 1)^{-1} \)

\[
P = \begin{bmatrix} P_1 & P_2 \\ P_2 & P_3 \end{bmatrix} \quad P\hat{F}^T = \begin{bmatrix} P_1 F_A^T - R_1^T F_B^T \\ P_2^T F_A^T - R_2^T F_B^T \end{bmatrix} \quad \hat{A} = \begin{bmatrix} A & \beta_1 P & P \end{bmatrix} \quad \Lambda_1 = -\alpha \hat{P}^T - \alpha \hat{A} \hat{P} + (\alpha^2 - r^2) P + \varepsilon \alpha^2 \hat{E} \hat{E}^T \\
\Lambda_2 = P \hat{A}^T - \varepsilon \alpha \hat{E} \hat{E}^T \\
\Lambda_3 = -P + \varepsilon \hat{E} \hat{E}^T
\]
Correspondingly, the PI controller is determined by

\[
\begin{bmatrix}
(K_p + K_I)C & -K_I
\end{bmatrix} = \begin{bmatrix} R_1 & R_2 \end{bmatrix} P^{-1}
\]

\[
(R_1 \quad R_2) P^{-1} = (\hat{K}_p \quad \hat{K}_I)
\]

\[
K_I = -\hat{K}_I
\]

\[
K_p = \hat{K}_p C^T (CC^T)^{-1} + \hat{K}_I
\]

To optimize the robust H infinity control performance, the PI controller can be determined by solving the following optimization program,

\[
\text{Min}_{\Delta \hat{A}(t), \Delta \hat{B}(t)} \gamma_{\text{PI}}
\]

Guideline: A smaller value of \( \gamma_{\text{PI}} \) leads to faster output response with a more aggressive control action, and vice versa.
PI based set-point learning design for indirect-type ILC

Based on the PI or PID closed-loop, a learning set-point command is designed as

\[ y_s(t, k) = y_s(t, k - 1) + L_1 e(t + 1, k - 1) + L_2 \delta e_s(t - 1, k) + L_3 \delta \sum e_s(t - 1, k) \]

where \( y_s(t, k) \) is the set-point command in the previous cycle,

\[ \delta e_s(t - 1, k) = e_s(t - 1, k) - e_s(t - 1, k - 1) \]

\[ \delta \sum e_s(t, k) = \delta \sum e_s(t - 1, k) + \delta e_s(t, k) \]
Feedforward controllers are used to adjust the process input

\[ u(t, k) = u_{\text{PID}}(t, k) + F_1 e_s(t, k) + F_2 e_s(t - 1, k) + F_3 \sum e_s(t - 1, k) \]

Note: The setpoint tracking errors at the current moment, one-step ahead moment, and the error integral in the current cycle are used to construct the feedforward control.
PI based set-point learning design for indirect-type ILC

Two-dimensional (2D) system description of the indirect ILC scheme

\[
\begin{bmatrix}
\delta x(t+1,k) \\
\delta e_s(t,k) \\
\delta \sum e_s(t,k) \\
e(t+1,k)
\end{bmatrix}
= \tilde{\Psi}
\begin{bmatrix}
\delta x(t,k) \\
\delta e_s(t-1,k) \\
\delta \sum e_s(t-1,k) \\
e(t+1,k-1)
\end{bmatrix}
+ D_w \sigma(t)
\]

\[
\varsigma(t,k) = G
\begin{bmatrix}
\delta x(t,k) \\
\delta e_s(t-1,k) \\
\delta \sum e_s(t-1,k) \\
e(t+1,k-1)
\end{bmatrix}
\]

where

\[
\varsigma(t,k) = e(t+1,k-1)
\]

\[
G = \begin{bmatrix} 0 & 0 & 0 & I \end{bmatrix}
\]

\[
D_w = \begin{bmatrix} I & 0 & 0 & -C^T \end{bmatrix}^T
\]

\[
\tilde{\Psi} =
\begin{bmatrix}
\tilde{A} - \tilde{B}(k_p + k_i + k_d + F_1)C & \tilde{B}[(k_p + k_i + k_d + F_1)L_2 + F_2 - k_d] \\
-C & -C \\
-C\tilde{A} + C\tilde{B}(k_p + k_i + k_d + F_1)C & -C\tilde{B}[(k_p + k_i + k_d + F_1)L_2 + F_2 - k_d] \\
\tilde{B}[(k_p + k_i + k_d + F_1)L_3 + F_3 + k_i] & \tilde{B}(k_p + k_i + k_d + F_1)L_1 \\
L_2 & L_2 \\
L_3 & L_3 \\
I + L_3 & I + L_3 \\
-C\tilde{B}[(k_p + k_i + k_d + F_1)L_3 + F_3 + k_i] & I - C\tilde{B}(k_p + k_i + k_d + F_1)L_1
\end{bmatrix}
\]
PI based set-point learning design for indirect-type ILC

The control objectives for robust tracking from batch to batch

\[
J_{BP} = \sum_{i=0}^{N_1=T_p} \sum_{k=0}^{N_2=\infty} (\gamma_{ILC}^{-1} \|\zeta(t, k+1)\|_2^2 - \gamma_{ILC} \|\sigma(t, k+1)\|_2^2) < 0
\]

2D Roesser’s system stability [1]:

\[
\begin{align*}
\begin{bmatrix}
    x^h(i+1, j) \\
    x^v(i, j+1)
\end{bmatrix} &= \begin{bmatrix}
    A_{11} + \Delta A_{11} & A_{12} + \Delta A_{12} \\
    A_{21} + \Delta A_{21} & A_{22} + \Delta A_{22}
\end{bmatrix} \begin{bmatrix}
    x^h(i, j) \\
    x^v(i, j)
\end{bmatrix} + \omega(i, j) \\
y(i, j) &= \begin{bmatrix}
    C_1 & C_2
\end{bmatrix} \begin{bmatrix}
    x^h(i, j) \\
    x^v(i, j)
\end{bmatrix}
\end{align*}
\]

\[
i, j=0,1,2,\ldots.
\]

Robust stability condition [1]:

\[
\tilde{A}^T P \tilde{A} - P < 0
\]

\[
\tilde{A} = \begin{bmatrix}
    A_{11} + \Delta A_{11} & A_{12} + \Delta A_{12} \\
    A_{21} + \Delta A_{21} & A_{22} + \Delta A_{22}
\end{bmatrix}
\]

\[
P = \text{diag}\{ P_1, P_2 \}
\]

PI based set-point learning design for indirect-type ILC

Define

\[
x^h(t,k) = \begin{bmatrix}
\delta x(t,k) \\
\delta e_s(t-1,k) \\
\delta \sum e_s(t-1,k)
\end{bmatrix}
\]

\[
x^v(t,k) = e(t+1,k)
\]

Lyapunov-Krasovskii function used for analyzing 2D asymptotic stability

\[
\Delta V = V_Q \begin{bmatrix}
x^h(t+1,k) \\
x^v(t,k)
\end{bmatrix} - V_Q \begin{bmatrix}
x^h(t,k) \\
x^v(t,k-1)
\end{bmatrix}
\]

The objective function of robust batch operation for minimization

\[
J_{BP} = \sum_{t=0}^{N_1=T_p} \sum_{k=0}^{N_2 \to \infty} (\gamma_{ILC}^{-1} \| \varsigma(t,k+1) \|^2 - \gamma_{ILC} \| \varpi(t,k+1) \|^2 + \Delta V) - \sum_{t=0}^{N_1=T_p} \sum_{k=0}^{N_2 \to \infty} \Delta V < 0
\]

\[
\begin{cases}
\delta x(0,0) = \delta x(0,1) = \delta x(1,0) = 0; \\
\delta e_s(0,0) = \delta e_s(0,1) = \delta e_s(1,0) = 0; \\
\delta \sum e_s(0,0) = \delta \sum e_s(0,1) = \delta \sum e_s(1,0) = 0; \\
e(0,0) = e(0,1) = e(1,0) = 0.
\end{cases}
\]

\[
\sum_{t=0}^{N_1=T_p} \sum_{k=0}^{N_2 \to \infty} \Delta V > 0
\]
PI based set-point learning design for indirect-type ILC

Theorem 3: The 2D control system is guaranteed robustly stable with a H infinity control performance level, $\gamma_{ILC}$, if there exist $Q_1 > 0$, $Q_2 > 0$, $Q_3 > 0$, $Q_4 > 0$, matrices $\hat{F}_2$, $\hat{F}_3$, $\hat{L}_1$, $\hat{L}_2$, $\hat{L}_3$, and positive scalars $\varepsilon_1$, $\varepsilon_2$, such that

$$
\begin{bmatrix}
-Q + \varepsilon_1 \Omega_{A_1} \Omega_{A_1}^T + \varepsilon_2 \Omega_{B_1} \Omega_{B_1}^T & \Pi & D_w & 0 & 0 & 0 \\
* & -Q & 0 & QG^T & P \Omega_{A_2}^T & P \Omega_{B_2}^T \\
* & * & -\gamma_{ILC} I & 0 & 0 & 0 \\
* & * & * & -\gamma_{ILC} I & 0 & 0 \\
* & * & * & * & -\varepsilon_1 I & 0 \\
* & * & * & * & * & -\varepsilon_2 I
\end{bmatrix} < 0
$$

where $Q = \text{diag}\{Q_1, Q_2, Q_3, Q_4\}$, $D_g = [I \ 0]^T$, $H = [I \ 0]$

$$
\begin{align*}
\Omega_{A_1} &= [\Delta \bar{A}_1^T, \ 0, \ 0, \ -\Delta \bar{A}_1^T C^T]^T \\
\Omega_{A_2} &= [\Delta \bar{A}_2, \ 0, \ 0, \ 0] \\
\Omega_{B_1} &= [\Delta \bar{B}_1^T, \ 0, \ 0, \ -\Delta \bar{B}_1^T C^T]^T 
\end{align*}
$$

$$
\Omega_{B_2} = \begin{bmatrix}
-\Delta \bar{B}_2 (k_p + k_i + k_d + F_1) C, \\
\Delta \bar{B}_2 [(k_p + k_i + k_d + F_1) \hat{L}_2 + \hat{F}_2 - k_d], \\
\Delta \bar{B}_2 [(k_p + k_i + k_d + F_1) \hat{L}_3 + \hat{F}_3 + k_i] \\
\Delta \bar{B}_2 (k_p + k_i + k_d + F_1) \hat{L}_1
\end{bmatrix}
$$

$$
\Pi = \begin{bmatrix}
A_m Q_1 - B_m (k_p + k_i + k_d + F_1) C Q_1 & B_m (k_p + k_i + k_d + F_1) \hat{L}_2 + B_m \hat{F}_2 - B_m k_d Q_2 \\
-CQ_1 & -CQ_1 \\
-CA_m Q_1 + CB_m (k_p + k_i + k_d + F_1) C Q_1 & -CB_m (k_p + k_i + k_d + F_1) \hat{L}_2 - CB_m \hat{F}_2 + CB_m k_d Q_2 \\
B_m (k_p + k_i + k_d + F_1) \hat{L}_3 + B_m \hat{F}_3 + B_m k_d Q_3 & B_m (k_p + k_i + k_d + F_1) \hat{L}_1 \\
B_m \hat{L}_3 & \hat{L}_1 \\
Q_3 + \hat{L}_i & \hat{L}_i \\
-CB_m (k_p + k_i + k_d + F_1) \hat{L}_3 - CB_m \hat{F}_3 - CB_m k_d Q_3 & Q_4 - CB_m (k_p + k_i + k_d + F_1) \hat{L}_1
\end{bmatrix}
$$
PI based set-point learning design for indirect-type ILC

Correspondingly, the PI type ILC controller is determined by

\[
\begin{align*}
L_1 &= \hat{L}_1 Q_4^{-1} \\
L_2 &= \hat{L}_2 Q_2^{-1} \\
L_3 &= \hat{L}_3 Q_3^{-1}
\end{align*}
\]

The feedforward controller is determined by

\[
\begin{align*}
F_2 &= \hat{F}_2 Q_2^{-1} \\
F_3 &= \hat{F}_3 Q_3^{-1}
\end{align*}
\]

To optimize the set-point tracking performance, the PI type ILC controller can be determined by solving the following optimization program,

\[
\text{Min } \gamma_{\text{ILC}}
\]

Guideline: A smaller value of \( \gamma_{\text{ILC}} \) leads to faster output response with a more aggressive control action, and vice versa.

PI based indirect-type ILC for batch injection molding

The nozzle pressure response to the hydraulic valve input was modeled [1] by

\[
y(t, k + 1) = \frac{1.239(\pm 5\%) z^{-1} - 0.9282(\pm 5\%) z^{-2}}{1 - 1.607(\pm 5\%) z^{-1} + 0.6086(\pm 5\%) z^{-2}} u(t, k + 1) + \omega(t, k + 1)
\]

PI based indirect-type ILC for batch process optimization

Equivalently, the process model is rewritten in a state-space form

\[
\begin{align*}
x(t+1,k+1) &= \left( \begin{array}{cc} 1.607 & 1 \\ -0.6086 & 0 \\ \end{array} \right)x(t,k+1) + \Delta \tilde{A})x(t,k+1) + \left( \begin{array}{cc} 1.239 & 1 \\ -0.9282 & 0 \\ \end{array} \right)u(t,k+1) + \left( \begin{array}{c} 1 \\ 0 \\ \end{array} \right)\omega(t,k+1) \\
y(t,k+1) &= \left[ \begin{array}{c} 1, 0 \end{array} \right]x(t,k+1)
\end{align*}
\]

Time-varying uncertainties

\[
\Delta \tilde{A}(t) = \left[ \begin{array}{cc} 0.0804 \delta(t) & 0 \\ -0.0304 \delta(t) & 0 \\ \end{array} \right] = \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \\ \end{array} \right]\delta(t) - \left[ \begin{array}{cc} 0 & 0 \\ 0 & 1 \\ \end{array} \right]\delta(t)\left[ \begin{array}{cc} 0.0804 & 0 \\ -0.0304 & 0 \\ \end{array} \right]
\]

\[
\Delta \tilde{B}(t) = \left[ \begin{array}{c} 0.062 \delta(t) \\ -0.0464 \delta(t) \end{array} \right] = \left[ \begin{array}{cc} 1 & 0 \\ 0 & 1 \\ \end{array} \right]\delta(t)\left[ \begin{array}{c} 0.062 \\ -0.0464 \end{array} \right]
\]

Robustly tuned PI controllers: \( k_p = 1.2889 \quad k_i = 0.0336 \)

ILC controllers: \( F_2 = 0 \quad F_3 = -0.0097 \)

\( L_1 = 0.1776 \quad L_2 = 0 \quad L_3 = -0.029 \)
PI based indirect-type ILC for batch process optimization

Case 1: time-invariant uncertainties

Case 2: repetitive disturbance

Desired output profile

\[
Y_r = \begin{cases} 
200, & 0 \leq t \leq 100; \\
200+5(t-100), & 100 < t \leq 120; \\
300, & 120 < t \leq T_p = 200.
\end{cases}
\]
Case 3: Time-varying uncertainties

$$|\delta(t)| \leq 0.1$$

$$\omega(t, k + 1) = \sin(t + \theta(k))$$

$$\theta(k) \in [0, 2\pi]$$

$$\text{ATE}(k) = \sum_{t=1}^{T_p} |e(t, k)| / T_p$$

Plot of the output error for batch operation


Main Results:

- Analytical PID design in discrete-time domain for sampled control systems
- 2DOF control structure based PID design for improving disturbance rejection
- Predictor-based PID design for long time delay systems
- Robust PID tuning methods with respect to the system uncertainty bounds
- PI based indirect type ILC design for batch process optimization

Outlook:

- Data-driven PID tuning for sampled control systems
- Fractional-order PID design & PID scheduling for nonlinear systems
- PID +Memory for learning/intelligent control of industrial batch processes, repetitive systems, and robots etc.
Acknowledgement

Thanks to my collaborators and students for PID design

Pedro Albertos  Furong Gao  Pedro García  Wojciech Paszke  Youqing Wang

My group photo in 2017
Research interests:

➢ Advanced control system design & system optimization;
➢ On-line monitoring, control design and control optimization of chemical production batch process;
➢ Real-time model predictive control and optimization of crystallization & drying processes;
➢ Design of in-situ measurement and control devices;
➢ PBM and CFD modeling of crystallization processes.
Thanks for your attention & comments!

Lab website: http://act.dlut.edu.cn/