SICE関西支部主催のオンライン勉強会2020

Pythonで学ぶ制御工学

Part2: 伝達関数モデルを用いた制御系設計

大阪大学
OSAKA UNIVERSITY

南 裕樹

勉強会の概要

流れや雰囲気を体感してください
独学のハードルを下げるのが目的です！本を読んでください！

Part 1

制御工学の基礎
- 制御とは，フィードバック制御，制御系設計

Pythonプログラミングの基礎
- Jupyter Notebook の使い方
- Pythonプログラミング　★ Python演習 1

制御系設計のためのモデル
- 伝達関数モデル，ブロック線図，時間応答，周波数応答　★ Python演習 2

Part 2

制御系設計のためのモデル（復習）
- 伝達関数モデル，時間応答，周波数応答

伝達関数モデルを用いた制御系設計
- 閉ループ系の設計仕様，PID制御，モデルマッチング　★ Python演習 3

Part 3

ループ整形による制御系設計
- 開ループ系の設計仕様，位相進み・遅れ補償　★ Python演習 4

状態空間モデルを用いた制御系設計
- 状態空間モデル，状態フィードバック　★ Python演習 5

今日の目標

Level☆☆★

・閉ループ系の設計仕様の用語を説明できる
・閉ループ系の安定性をPythonでチェックできる
・PID制御系の応答特性（時間・周波数）のグラフをPythonで描ける

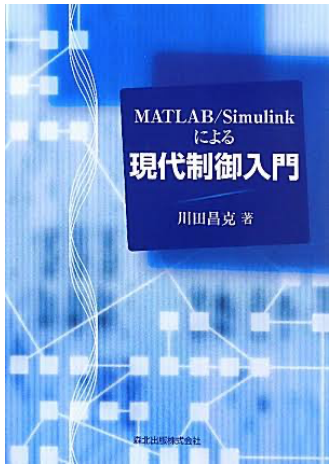
Level☆★★

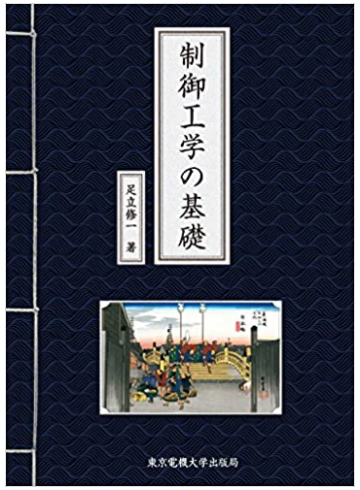
・限界感度法，ステップ応答法でPIDゲインを設計できる
・モデルマッチング法でPIDゲインを設計できる

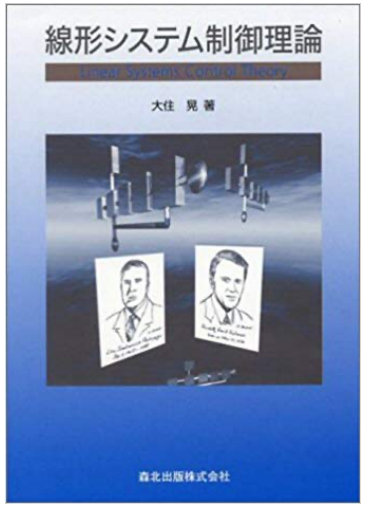
Level★★★

・２次遅れ系の行き過ぎ量やバンド幅，ピークゲインを手計算できる
・不完全微分が必要な理由を説明でき，それを用いた解析・設計ができる

参考図書







導入にもってこい

押さえておけば安心

数学的にしっかり勉強

補助資料の紹介

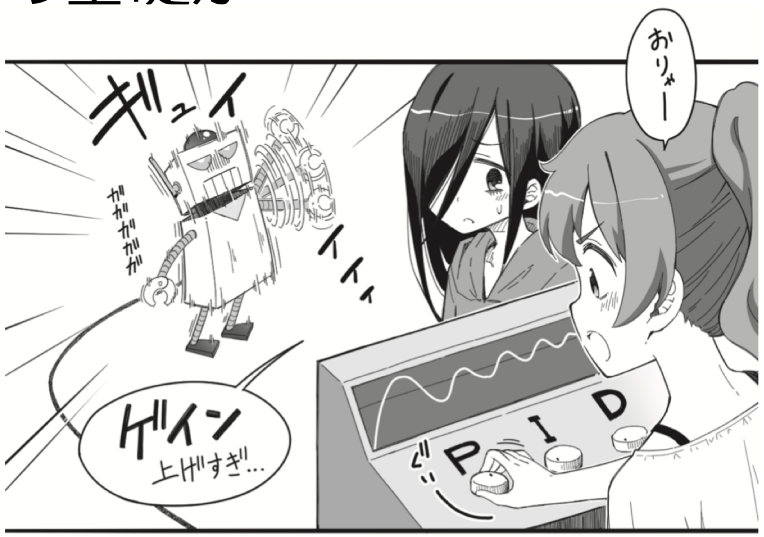
「Python × 制御工学」の日本初の本

古典制御，現代制御，ロバスト制御の基礎がギュッとつまっている

数学的な説明は少なめ

対話形式の説明やイラストを配置

サポートページには，
サンプルコードが公開されている
→ Python & MATLABコード



- 2,600 円＋税
- A5 272頁
- 2019/05/22 発行　Kindle版もあります

勉強会の概要

Part 1

制御工学の基礎
- 制御とは，フィードバック制御，制御系設計

Pythonプログラミングの基礎
- Jupyter Notebook の使い方
- Pythonプログラミング　★ Python演習 1

制御系設計のためのモデル
- 伝達関数モデル，ブロック線図，時間応答，周波数応答　★ Python演習 2

制御系設計のためのモデル（復習）
- 伝達関数モデル，時間応答，周波数応答

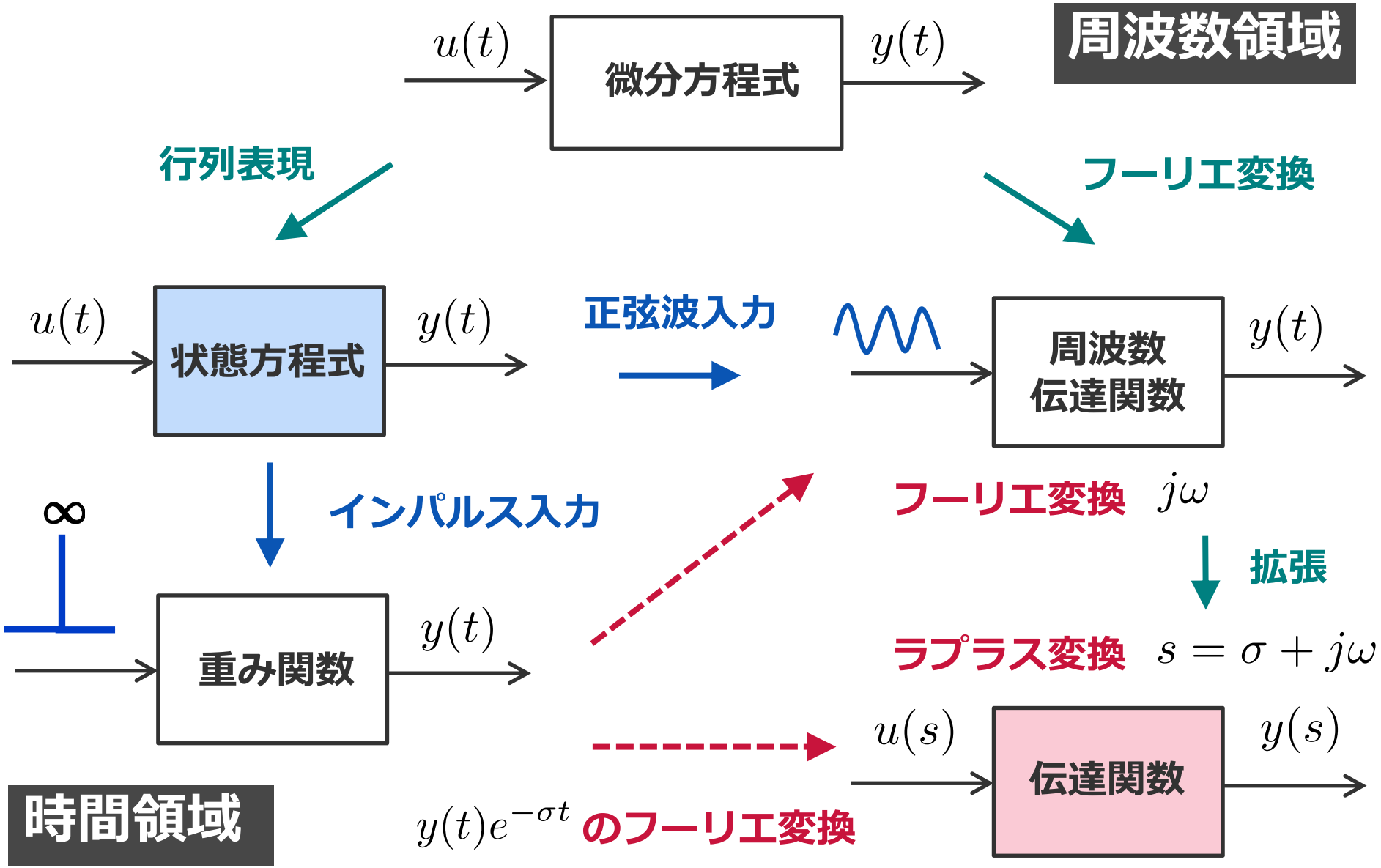
伝達関数モデルを用いた制御系設計
- 閉ループ系の設計仕様，PID制御，モデルマッチング　★ Python演習 3

Part 3

ループ整形による制御系設計
- 開ループ系の設計仕様，位相進み・遅れ補償　★ Python演習 4

状態空間モデルを用いた制御系設計
- 状態空間モデル，状態フィードバック　★ Python演習 5

設計モデルの関係


$u(t)$
微分方程式
$y(t)$
周波数領域
行列表現
フーリエ変換
$u(t)$
状態方程式
$y(t)$
正弦波入力
周波数
伝達関数
$y(t)$
インパルス入力
フーリエ変換 $j\omega$
拡張
∞
重み関数
$y(t)$
ラプラス変換 $s = \sigma + j\omega$
$u(s)$
伝達関数
$y(s)$
時間領域
$y(t)e^{-\sigma t}$ のフーリエ変換

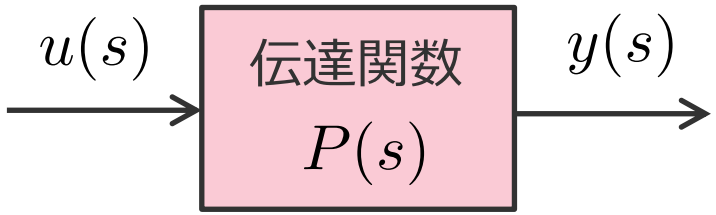
制御のためのモデル（設計モデル）

動的システム

$$\frac{\mathrm{d}^n}{\mathrm{d}t^n}y(t) + a_{n-1}\frac{\mathrm{d}^{n-1}}{\mathrm{d}t^{n-1}}y(t) + \cdots + a_1\frac{\mathrm{d}}{\mathrm{d}t}y(t) + a_0 y(t)$$
$$= b_m\frac{\mathrm{d}^m}{\mathrm{d}t^m}u(t) + b_{m-1}\frac{\mathrm{d}^{m-1}}{\mathrm{d}t^{m-1}}u(t) + \cdots + b_1\frac{\mathrm{d}}{\mathrm{d}t}u(t) + b_0 u(t)$$

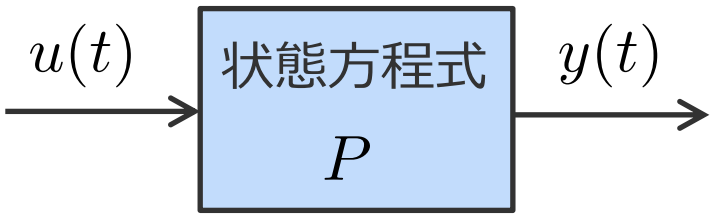
変換

伝達関数モデル　ラプラス変換を用いて代数方程式で表す



$$P(s) = \frac{b_m s^m + b_{m-1}s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}$$

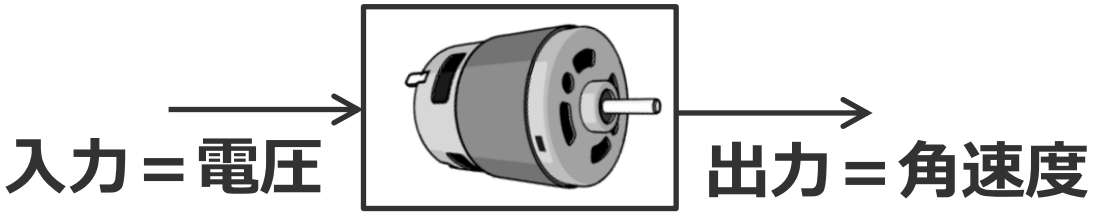
状態空間モデル　行列表現を用いて 1 階の微分方程式で表す



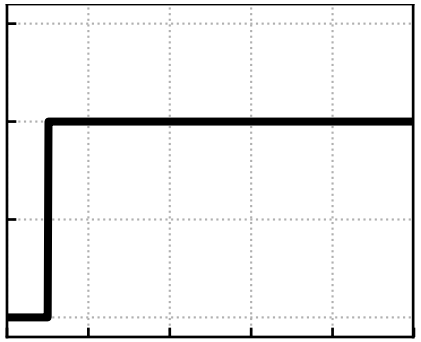
$$P : \begin{cases} \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}u(t) \\ y(t) &= \boldsymbol{C}\boldsymbol{x}(t) \end{cases}$$

モデルの特徴を知る

システム＝モータ



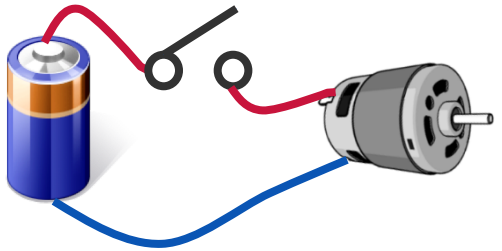
入力を加えて出力を観測する
→ 特徴を調べる

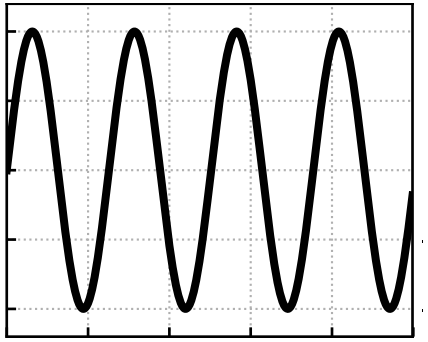


ステップ入力

電池をつないだときの出力の振る舞いは？

→ システムのステップ応答（過渡・定常特性）

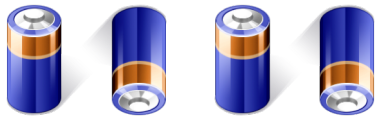




正弦波入力

電池の＋-を交互に切り替えるとどんな振る舞いになるか？

→ システムの周波数応答（ゲイン，位相）



Python演習２

伝達関数モデルのステップ応答

以下のコードを実行してみましょう

$$P(s) = \frac{1}{0.5s+1} = \frac{0s+1}{0.5s+1}$$

```
from control.matlab import *
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots(figsize=(3, 2.3))

P = tf([0, 1], [0.5, 1])

y, t = step(P, np.arange(0, 5, 0.01))
ax.plot(t, y, color='k')

ax.set_xticks(np.linspace(0, 5, 6))
ax.grid(ls=':’)
stepinfo(P)
```

Python ：伝達関数モデルの定義

```
tf( [分子の係数], [分母の係数] )
```

Python ：ステップ応答

```
y, t = step(モデル, 時間)
```

出力 時間

Python ：ステップ応答の情報

```
Info = stepinfo(モデル)
```

Python演習２

伝達関数モデルの周波数応答

以下のコードを実行してみましょう

$$P(s) = \frac{1}{0.5s+1} = \frac{0s+1}{0.5s+1}$$

Python : 周波数応答

gain, phase, w = bode(モデル, 周波数域)

ゲイン(倍) 位相（rad） 周波数（rad/s）

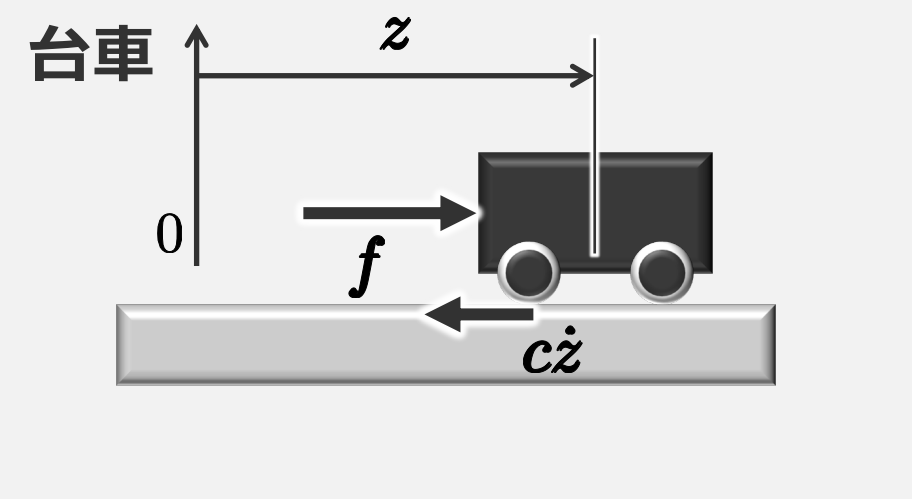
```
K = 1
T = 0.5

fig, ax = plt.subplots(2,1,figsize=(4,3.5))

P = tf([0, K],[T, 1])
gain, phase, w = bode(P, logspace(-2,2), Plot=False)
ax[0].semilogx(w, 20*np.log10(gain))
ax[1].semilogx(w, phase*180/np.pi)

ax[0].grid(which="both", ls=':')
ax[0].set_ylabel('Gain [dB]')
ax[1].grid(which="both", ls=':')
ax[1].set_xlabel('$¥omega$ [rad/s]')
ax[1].set_ylabel('Phase [deg]')
```

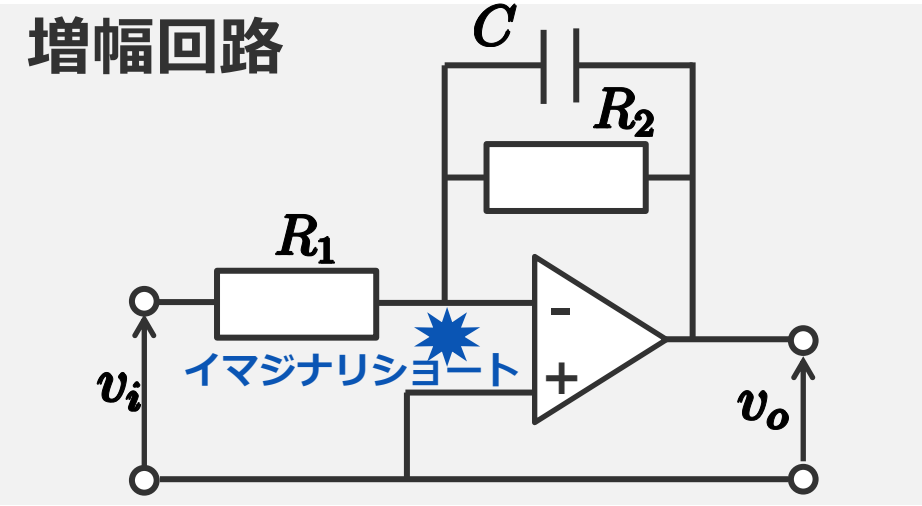
0.01 ~ 100 → logspace(-2,2)

dBへの変換：20*np.log10(gain)

degへの変換：phase*180/np.pi

1次遅れ系





$$\frac{1}{ms+c}$$

$$-\frac{R_2}{R_1R_2Cs+R_1}$$

1次遅れ系：$G(s)=\dfrac{K}{1+Ts}$

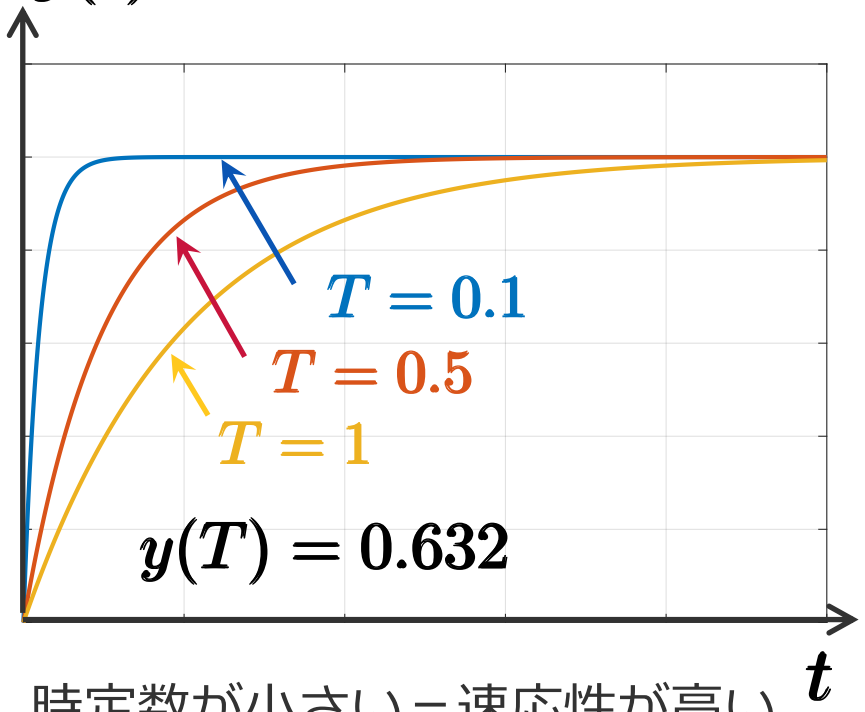
T ：時定数

K ：ゲイン

1次遅れ系

$$G(s) = \frac{1}{1+Ts}$$

$$y(t) = 1 - e^{-\frac{t}{T}}$$

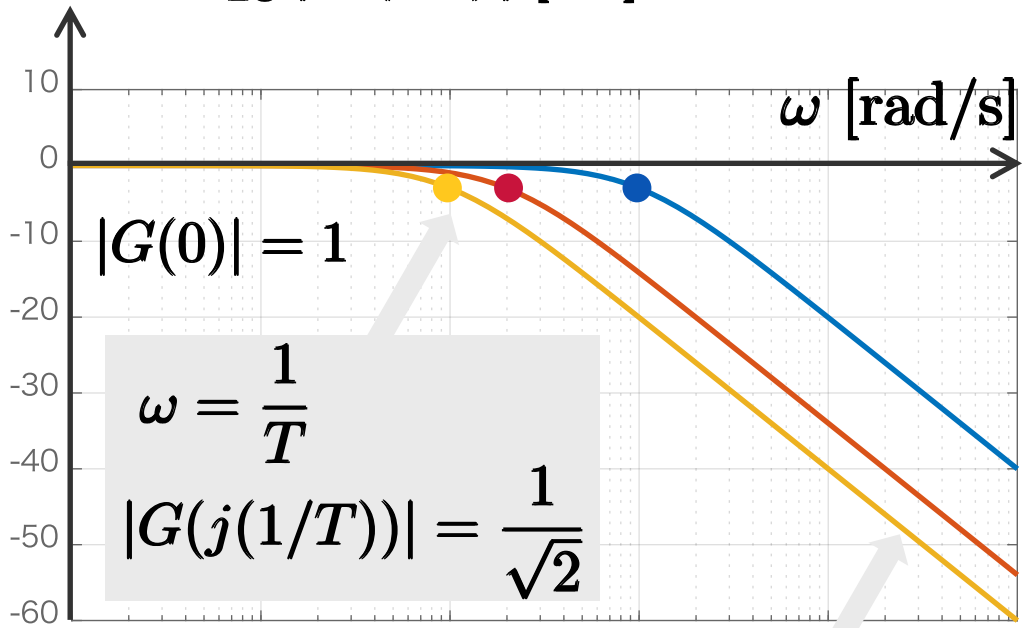


時定数が小さい＝速応性が高い

振動しない→オーバーシュートなし

$$G(j\omega) = \frac{1}{1+j\omega T}$$

$20\log_{10}|G(j\omega)|$ [dB]



$$|G(j\omega)| \simeq \frac{1}{\omega T} \rightarrow \text{-20[dB/dec]}$$

1 次遅れ系

- 1 次遅れ系のステップ応答は，振動せずに収束する
- 時定数は速応性に関するパラメータ

$$y(T) = 0.632$$

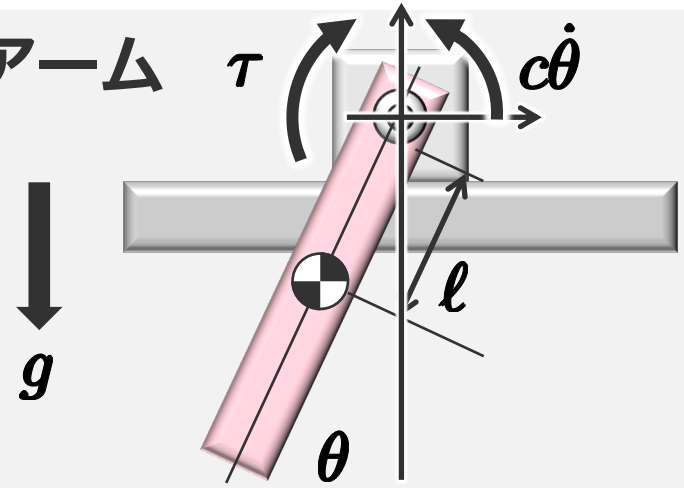
→ 定常値（=1）の63.2% に至る時間

$$\omega = \frac{1}{T} \quad |G(j(1/T))| = \frac{1}{\sqrt{2}}$$

→ 時定数の逆数は，周波数特性のカットオフ周波数（制御帯域）

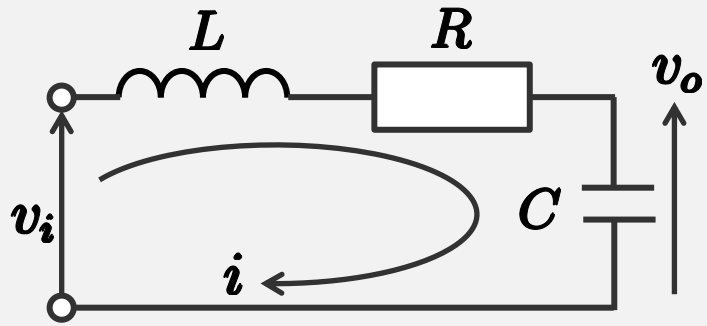
２次遅れ系

垂直アーム



$$\frac{1}{Js^2+cs+mg\ell}$$

RCL回路



$$\frac{1}{CLs^2+CRs+1}$$

２次遅れ系：$G(s)=\dfrac{K\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2}$

ζ ：減衰係数

ω_n：固有角周波数

K ：ゲイン

2次遅れ系

マスバネダンパ

$\omega_n = \sqrt{k/m}$

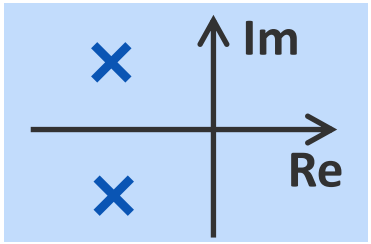
$\zeta = c/(2\sqrt{mk})$

$K = 1/k$

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{p_1 p_2}{(s-p_1)(s-p_2)}$$

減衰係数　固有角周波数

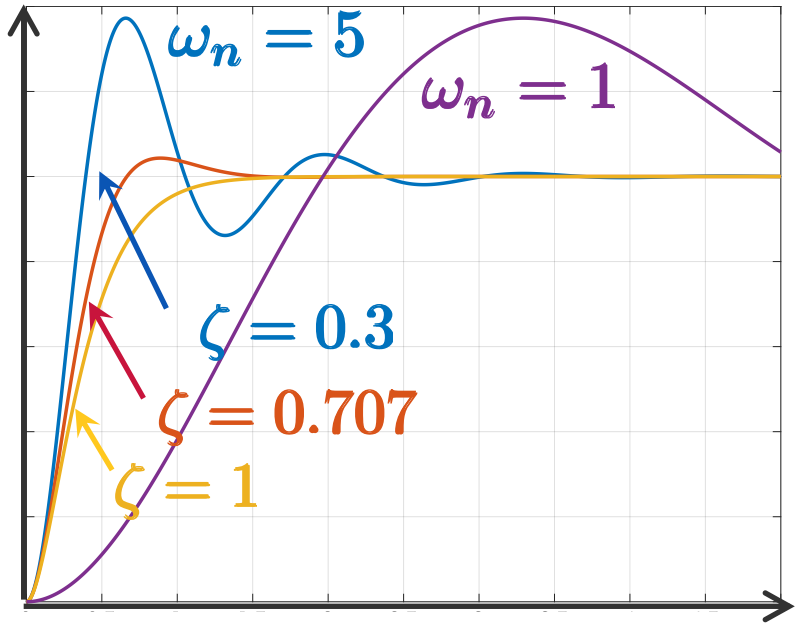
$p_1, p_2 = \alpha \pm \beta j$ のとき



$$G(s) = \frac{\alpha^2 + \beta^2}{s^2 - 2\alpha s + \alpha^2 + \beta^2}$$

$$\begin{cases} K = 1 \\ \zeta = -\dfrac{\alpha}{\sqrt{\alpha^2+\beta^2}} \quad 0 \le \zeta \le 1 \\ \omega_n = \sqrt{\alpha^2+\beta^2} \end{cases}$$

ステップ応答



虚部が大きい ＝ ζ が小さい ＝ 振動的になる

実部or虚部が大きい ＝ ω_n が大きい ＝ 応答が速い

2次遅れ系

マスバネダンパ

$\omega_n = \sqrt{k/m}$

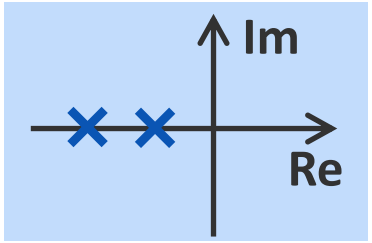
$\zeta = c/(2\sqrt{mk})$

$K = 1/k$

$$G(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} = \frac{p_1 p_2}{(s-p_1)(s-p_2)}$$

減衰係数　固有角周波数

$p_1, p_2 = \alpha_1, \alpha_2$ のとき



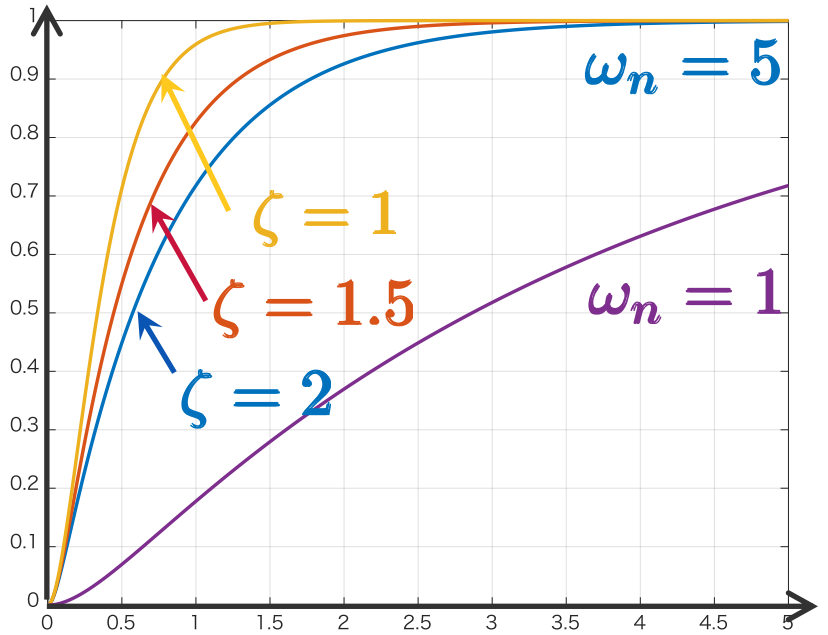
$$G(s) = \frac{\alpha_1\alpha_2}{s^2 - (\alpha_1+\alpha_2)s + \alpha_1\alpha_2}$$

$$\begin{cases} K = 1 \\ \zeta = -\dfrac{\alpha_1+\alpha_2}{2\sqrt{\alpha_1\alpha_2}} \\ \omega_n = \sqrt{\alpha_1\alpha_2} \end{cases}$$

←相加>相乗

$\zeta \geq 1$ で，振動しない．極が負側に大きくなると，ζ と ω_n が大きくなる = 制動が強くなる & 応答が速くなる

ステップ応答

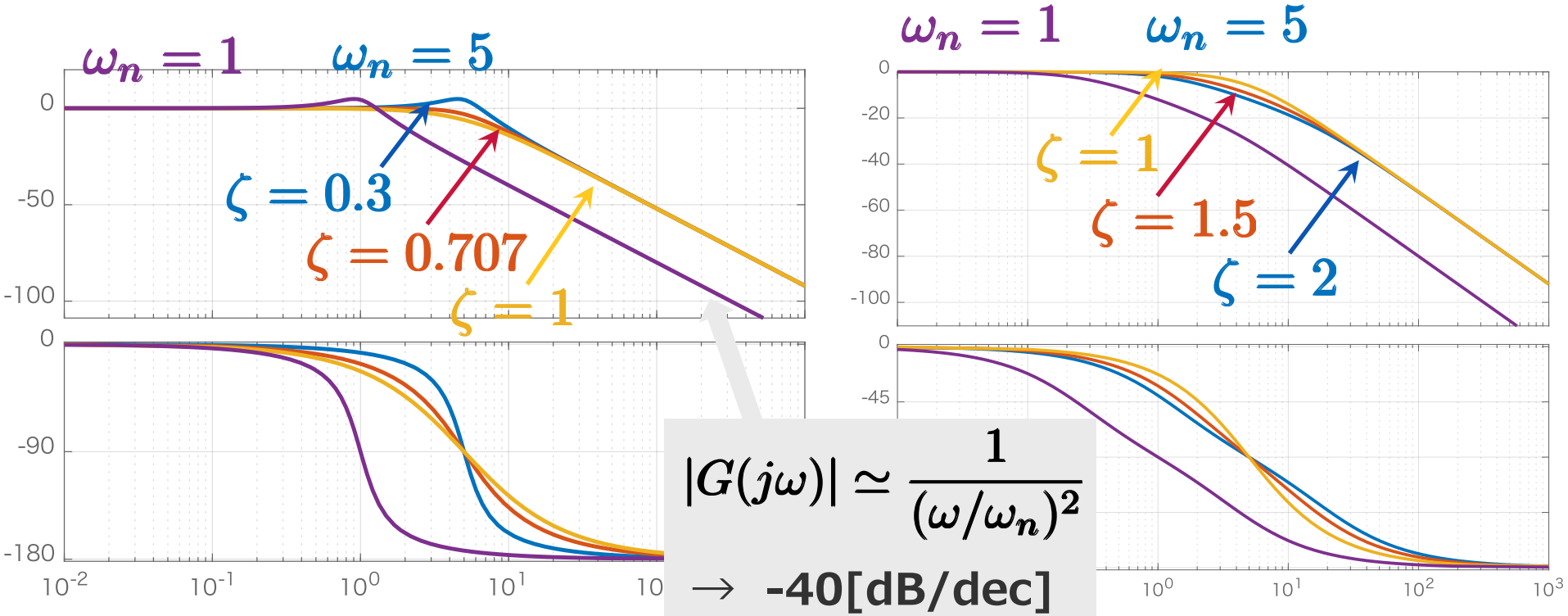


2次遅れ系

周波数応答

$$G(j\omega) = \frac{\omega_n^2}{\omega_n^2 - \omega^2 + j2\zeta\omega_n\omega}$$

$$|G(j\omega)| = \frac{\omega_n^2}{\sqrt{(\omega_n^2 - \omega^2)^2 + (2\zeta\omega_n\omega)^2}} \quad \angle G(j\omega) = -\tan^{-1}\frac{2\zeta\omega_n\omega}{\omega_n^2 - \omega^2}$$



$$|G(j\omega)| \simeq \frac{1}{(\omega/\omega_n)^2}$$

$\rightarrow$ **-40[dB/dec]**

２次遅れ系

- **２次遅れ系のステップ応答は，振動する場合がある**

- **減衰係数が減衰性（安定度）に関するパラメータ**

$\zeta = 0$	安定限界＝持続振動
$0 < \zeta < 1$	不足制動＝振動しながら定常値に収束
$\zeta = 1$	臨界制動＝ぎりぎり振動しない
$\zeta > 1$	過制動＝まったく振動せずに定常値に収束

→ $0 < \zeta < 1/\sqrt{2}$ のとき，周波数特性で共振を生じる

- **固有角周波数は速応性に関するパラメータ**

→ 周波数応答におけるカットオフ周波数

勉強会の概要

Part 1

制御工学の基礎
- 制御とは，フィードバック制御，制御系設計

Pythonプログラミングの基礎
- Jupyter Notebook の使い方
- Pythonプログラミング　★ Python演習 1

制御系設計のためのモデル
- 伝達関数モデル，ブロック線図，時間応答，周波数応答　★ Python演習 2

制御系設計のためのモデル（復習）
- 伝達関数モデル，時間応答，周波数応答

伝達関数モデルを用いた制御系設計
- 閉ループ系の設計仕様，PID制御，モデルマッチング　★ Python演習 3

Part 3

ループ整形による制御系設計
- 開ループ系の設計仕様，位相進み・遅れ補償　★ Python演習 4

状態空間モデルを用いた制御系設計
- 状態空間モデル，状態フィードバック　★ Python演習 5

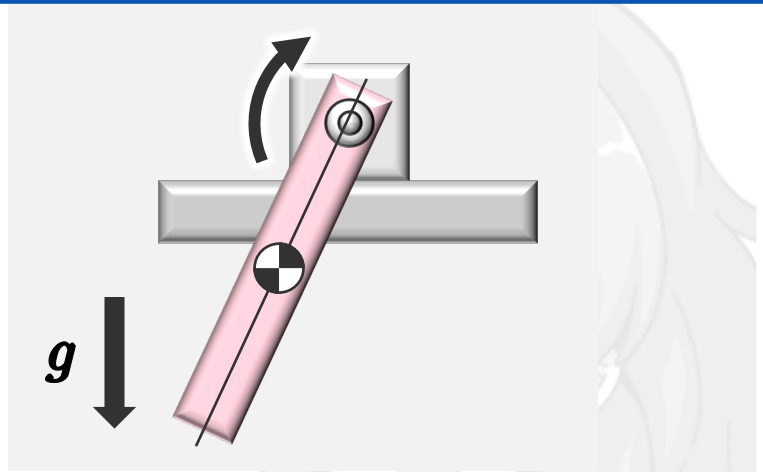
姉妹のよくある日常会話

姉：希波（きなみ）　２３歳

妹：望結（みゆう）　１８歳

姉妹のよくある日常会話

伝達関数モデルがあるから
これでいい感じにして！



うぅー，授業で習ったけど，
よく覚えていない・・・

いい感じってなに？
具体的な仕様は？

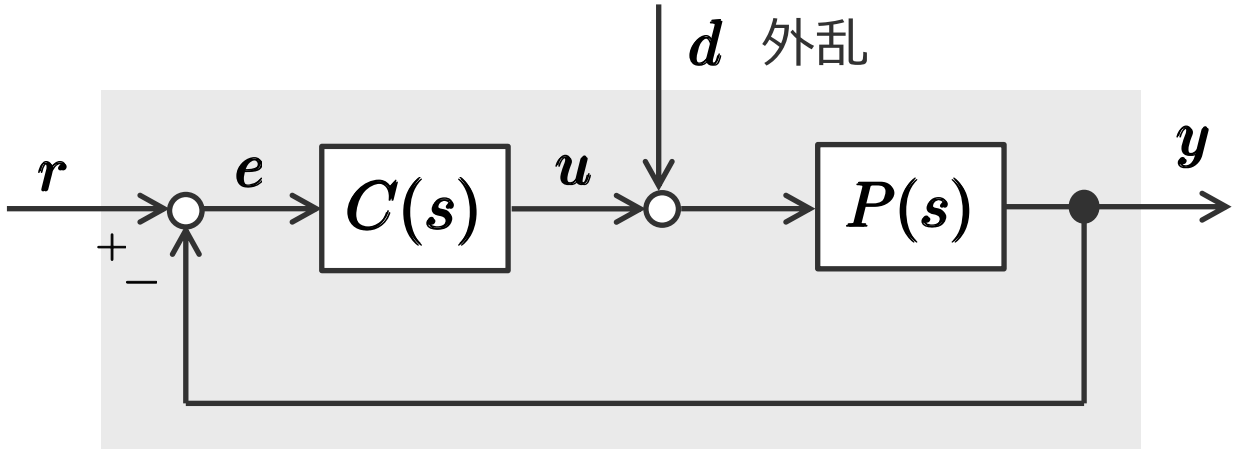
例えば，ステップ応答の整定時間とか
オーバーシュートとか

じゃぁ，閉ループ系の仕様を復習して
からPID制御の設計をやってみよう

★実現したい仕様は何か？PID制御の設計はどのようにするか？

閉ループ系の設計仕様

フィードバック制御系を構築



閉ループ系の特性に注目

・どんな時間応答？
・どんな周波数応答？
・閉ループ極はどこに配置？

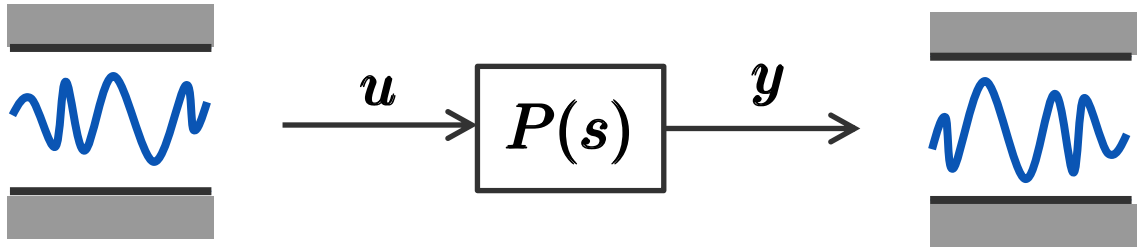
注目するポイント（設計仕様）

1）安定性

2）速応性 ＋ 減衰性

3）定常偏差

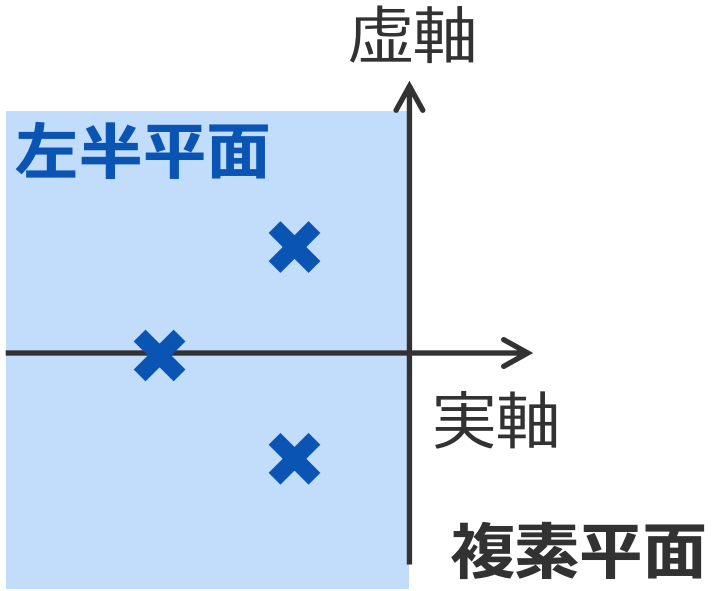
制御対象の安定性



有界な入力を入れたときに，出力も有界になる
（入出力安定，BIBO安定）

⟷ **伝達関数の極の実部がすべて負**　$s = \sigma + j\omega$

↳ $P(s) = \infty$ となる s
$P(s)$ の分母多項式 $D(s)$ の根



$P(s) = 0$ となる s を零点という

$$\frac{1}{s+1}$$

極：-1
零点：無限遠点

Python ：極，零点，直流ゲイン

pole(モデル)	モデル.pole()
zero(モデル)	モデル.zero()
dcgain(モデル)	モデル.dcgain()

【補足】安定判別法（フルビッツ）

$a_n s^2 + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0$ のすべての根の実部が負

⇔ すべての係数 $a_n,\ a_{n-1},\ \cdots,\ a_1,\ a_0$ が正（同符号）
＋ $H_1,\ H_2, \cdots, H_n$ がすべて正

$$H = \begin{bmatrix} a_{n-1} & a_{n-3} & a_{n-5} & \cdots & 0 \\ a_n & a_{n-2} & a_{n-4} & \cdots & 0 \\ 0 & a_{n-1} & a_{n-3} & \cdots & 0 \\ 0 & a_n & a_{n-2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & a_2 & a_0 \end{bmatrix}$$

$$H_2 = \begin{vmatrix} a_{n-1} & a_{n-3} \\ a_n & a_{n-2} \end{vmatrix}$$

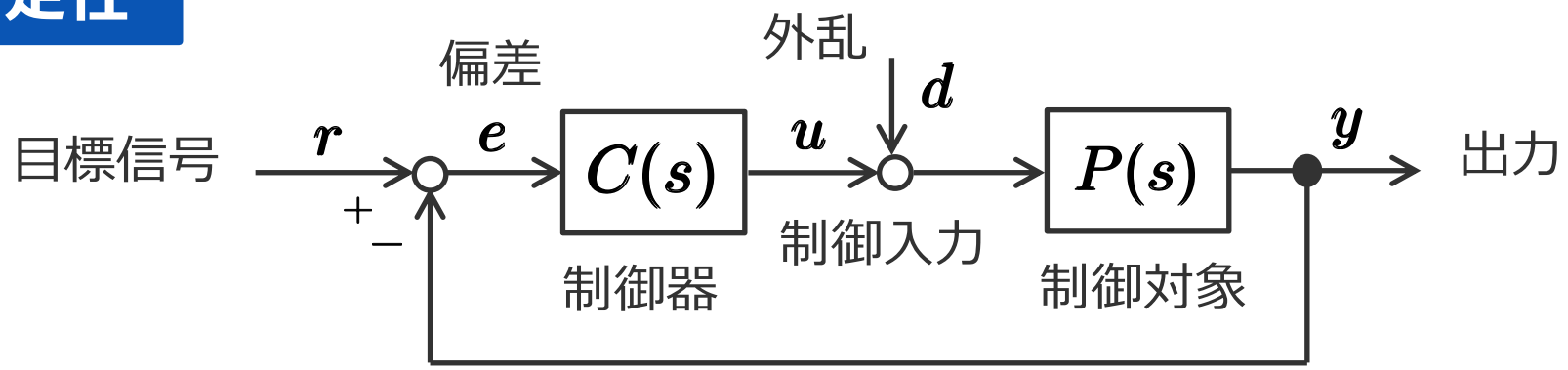
$$H_3 = \begin{vmatrix} a_{n-1} & a_{n-3} & a_{n-5} \\ a_n & a_{n-2} & a_{n-4} \\ 0 & a_{n-1} & a_{n-3} \end{vmatrix}$$

$\vdots$

⇒ すべての係数 $a_n,\ a_{n-1},\ \cdots,\ a_1,\ a_0$ が正（同符号）
まずはコレ！同符号でないとき，一目で不安定であることがわかる

閉ループ系の安定性

安定性



全ての伝達関数が入出力安定（極の実部がすべて負）

$$\begin{bmatrix} u \\ y \end{bmatrix} = \begin{bmatrix} G_{ur}(s) & G_{ud}(s) \\ G_{yr}(s) & G_{yd}(s) \end{bmatrix} \begin{bmatrix} r \\ d \end{bmatrix}$$

内部安定

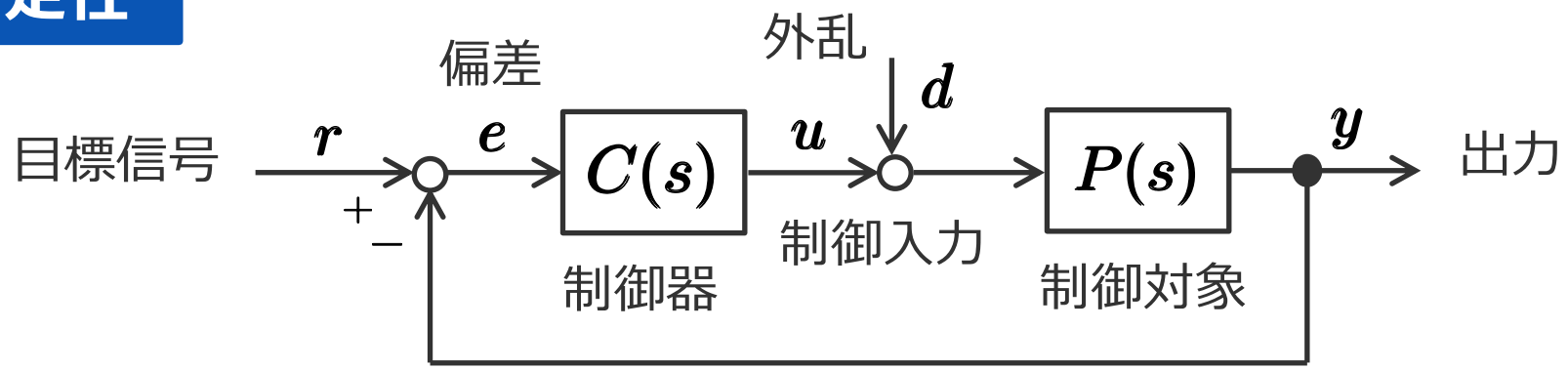
特性多項式が安定多項式　$\phi(s) = D_P(s)D_C(s) + N_P(s)N_C(s)$

根の実部がすべて負

$$P(s) = \frac{N_P(s)}{D_P(s)} \quad C(s) = \frac{N_C(s)}{D_C(s)}$$

閉ループ系の安定性

安定性



$$
\begin{aligned}
y(s) &= P(s)(d(s)+u(s)) \\
&= P(s)d(s)+P(s)u(s) \\
&= P(s)d(s)+P(s)C(s)e(s) \\
&= P(s)d(s)+P(s)C(s)(r(s)-y(s))
\end{aligned}
$$

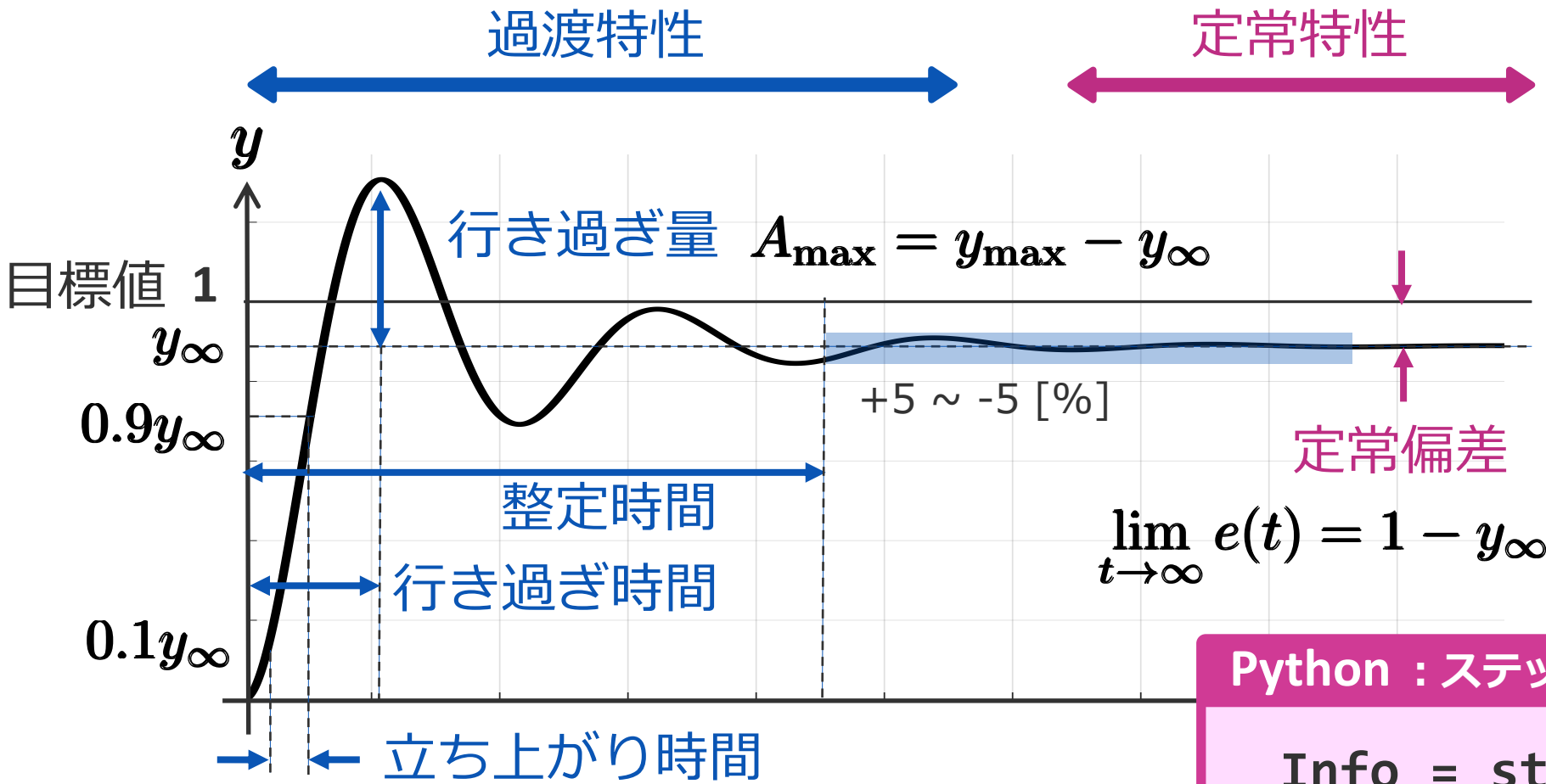
$$
\begin{aligned}
u(s) &= C(s)e(s) \\
&= C(s)r(s)-C(s)y(s) \\
&= C(s)r(s)-C(s)P(s)(d(s)+u(s))
\end{aligned}
$$

$$
y(s)=\frac{P(s)}{1+P(s)C(s)}d(s)+\frac{P(s)C(s)}{1+P(s)C(s)}r(s)
$$

$$
u(s)=\frac{C(s)}{1+P(s)C(s)}r(s)-\frac{P(s)C(s)}{1+P(s)C(s)}d(s)
$$

閉ループ系の時間応答

A）時間応答　$G_{yr}(s)$ のステップ応答

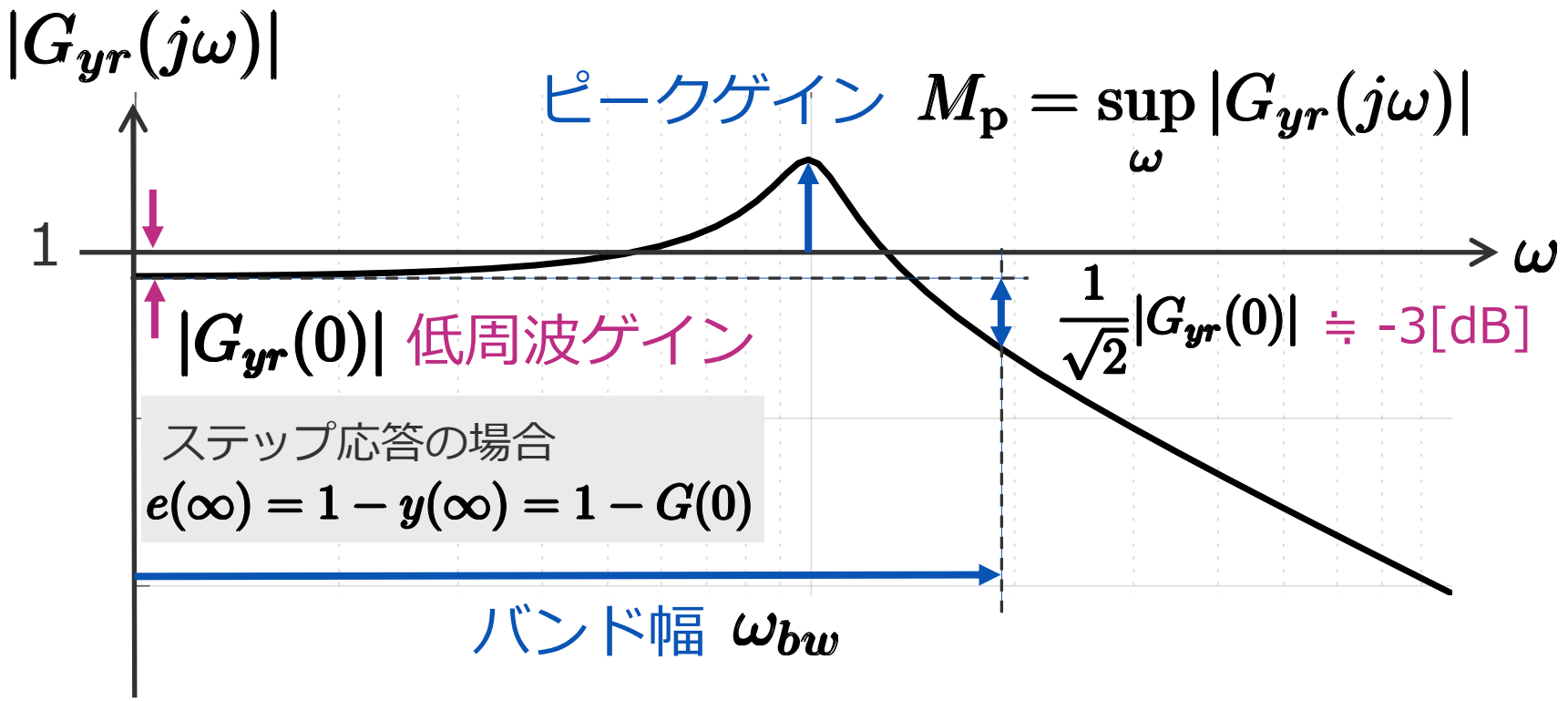


Python ：ステップ応答の情報

```
Info = stepinfo(モデル)
```

閉ループ系の周波数応答

B）周波数応答　$G_{yr}(j\omega)$ のゲイン線図

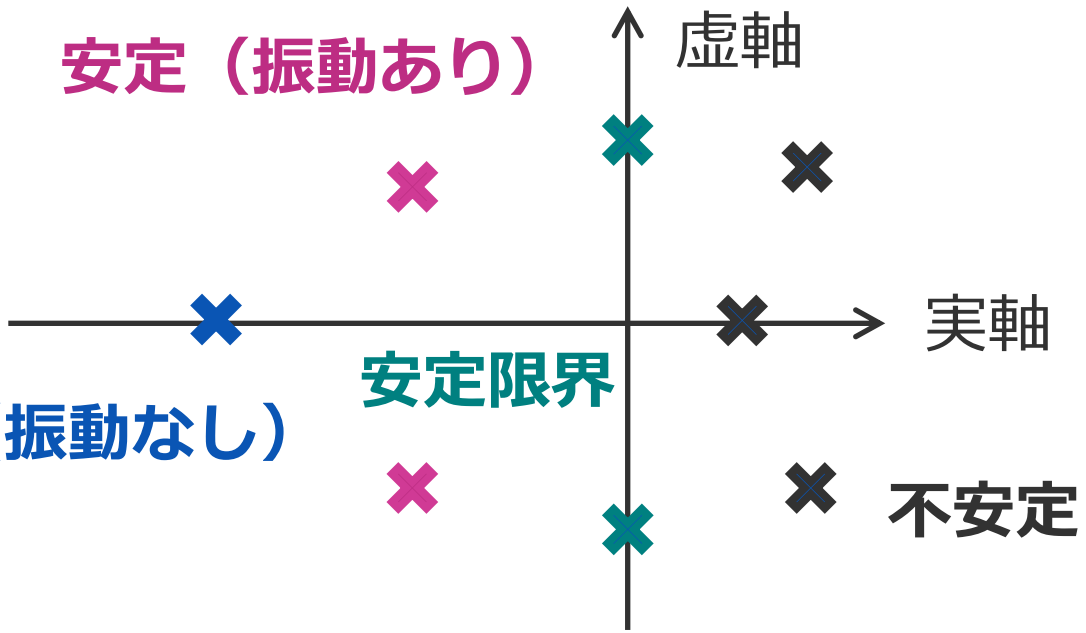


振幅表示	$\lvert G_{yr}(j\omega)\rvert$	0.01	0.1	$\frac{1}{\sqrt{2}}$	1	10	100
dB表示	$20\log_{10}\lvert G_{yr}(j\omega)\rvert$	-40	-20	-3	0	20	40

閉ループ極

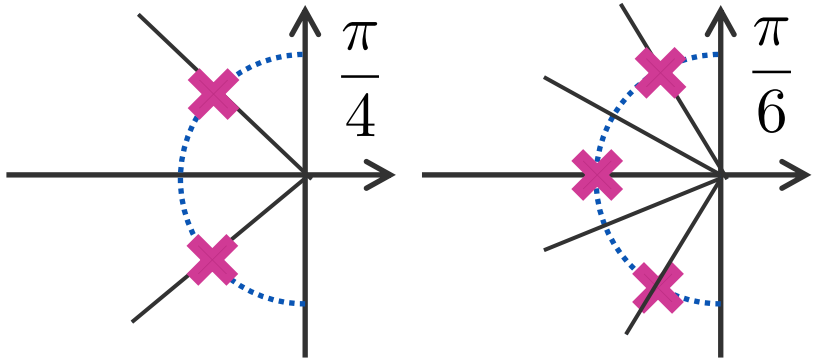
C）閉ループ極

$$s = \sigma + j\omega$$

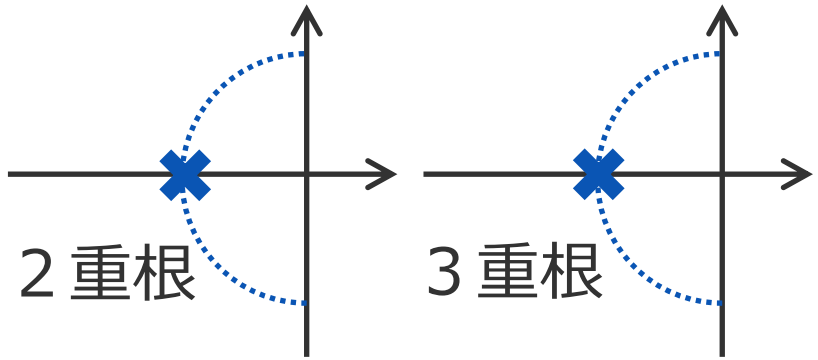


原点に近い極が支配的 → 主要極　主要極を仕様に合わせる

バターワース標準形



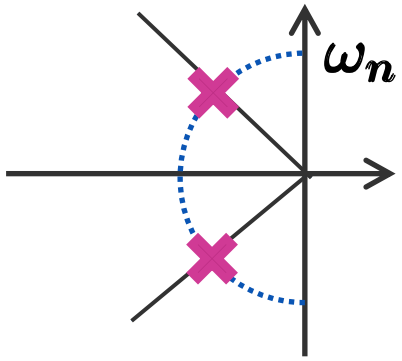
二項係数標準形

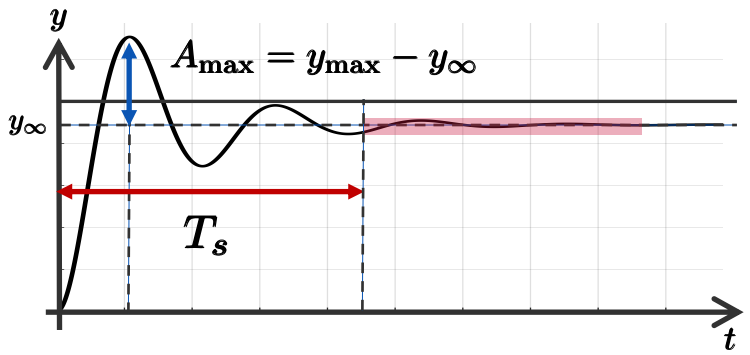


極と応答の関係

２次遅れ系（安定，$0 < \zeta < 1$ ）

$$G_{yr}(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$





極： $s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$

定常値： $y_\infty = K$

$$\zeta = \frac{1}{\sqrt{2}}$$

行き過ぎ量： $A_{\max} = K\exp\left(-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}\right) = K\exp(-\pi) = 0.0432K$

4.3%

極の実部と虚部の比×（- π ）

5%整定時間： $T_s \simeq \frac{3}{\zeta\omega_n} \longrightarrow \zeta\omega_n > \frac{3}{T_s}$

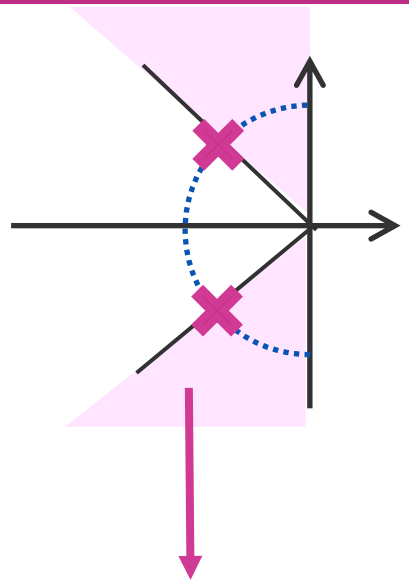
極の実部が -3/Ts より小さい

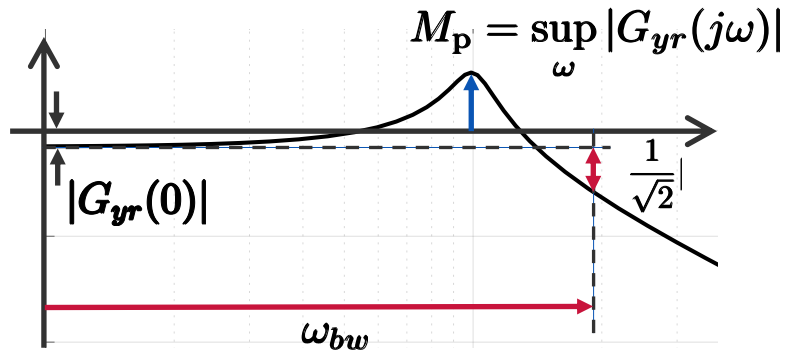
極と応答の関係

２次遅れ系（安定，$0 < \zeta < 1$ ）

$$G_{yr}(s) = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

極： $s = -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2}$





ピークゲイン： $M_p = \dfrac{1}{2\zeta\sqrt{1-\zeta^2}}$ $0 < \zeta < \dfrac{1}{\sqrt{2}}$ $\omega_p = \omega_n\sqrt{1-2\zeta^2}$

減衰係数が小さいほど（極が虚軸に近いほど）
ピークゲインが大きい

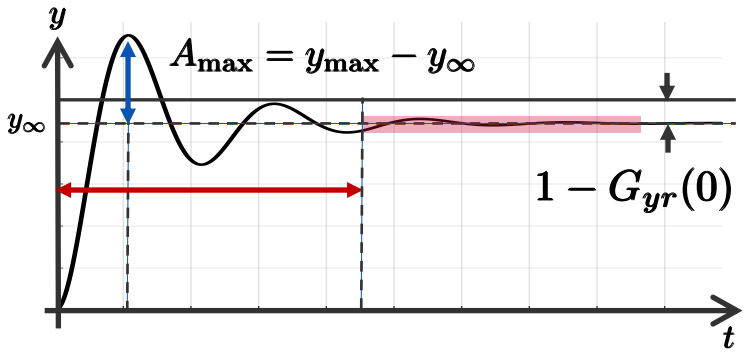
バンド幅： $\omega_{bw} \propto \omega_n$ $\omega_{bw} = \omega_n$ $(\zeta = 1/\sqrt{2})$

$|G_{yr}(j\omega_{bw})| = \dfrac{1}{\sqrt{2}}|G_{yr}(0)|$

固有角周波数が大きいほど（極が原点から遠いほど）
バンド幅が大きい

閉ループ系の設計仕様

閉ループ系の設計仕様



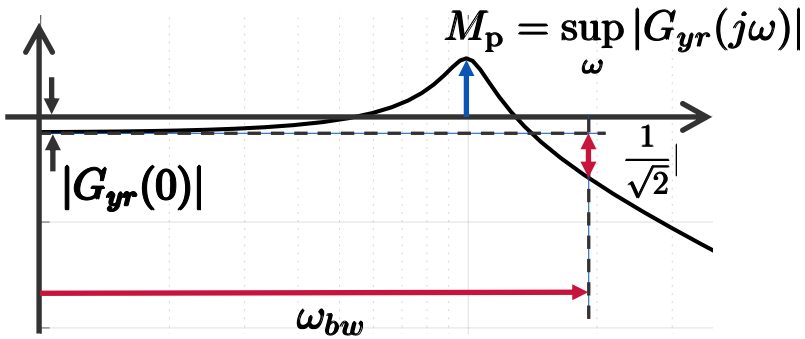
1）安定性 ➡ 内部安定

2）速応性，減衰性（≒ロバスト安定性）

- ➡ G_{yr} のステップ応答において整定時間が小さい
- ➡ 行き過ぎ量（オーバーシュート）が小さい
- ➡ G_{yr} のゲイン線図：バンド幅が十分大きい
- ➡ ピークゲイン $M_{\mathrm{p}} = \sup_{\omega} |G_{yr}(j\omega)|$ が小さい

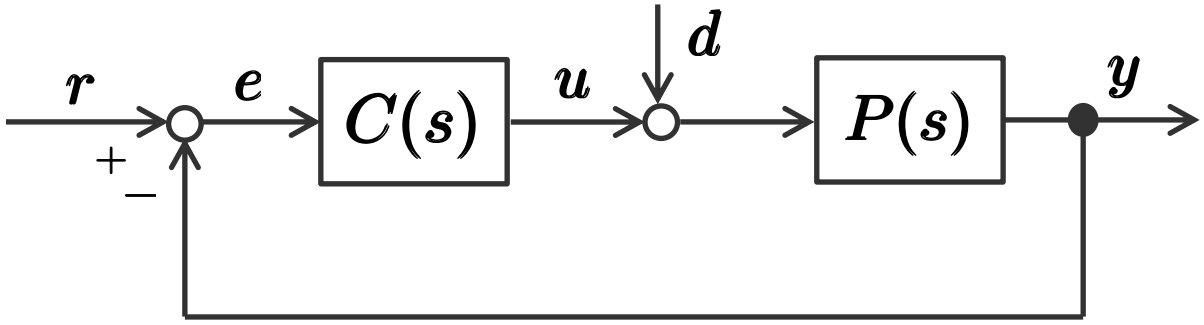
3）定常偏差

- ➡ $|G_{yr}(0)| = 1$



Python演習３

フィードバック系の安定性と応答特性



$$P(s) = \frac{1}{s^2 + s + 1} \qquad C(s) = \frac{2s^2 + 10s + 3}{s} \qquad C = \frac{k_{\mathrm{D}} s^2 + k_{\mathrm{P}} s + k_{\mathrm{I}}}{s}$$

のとき，フィードバック系の内部安定性をチェックせよ．また，

$$G_{yr}(s) = \frac{P(s)C(s)}{1 + P(s)C(s)}$$

のステップ応答と周波数応答のグラフを描け

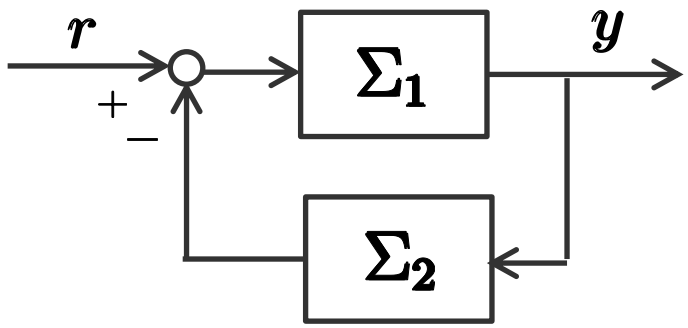
Python演習３

フィードバック系の内部安定性

```
from control.matlab import *

P = tf([0, 1],[1, 1 ,1])
C = tf([2, 10, 3],[1, 0])

Gyr = feedback(P*C, 1)
print(Gyr.pole())

Gud = -feedback(1 ,P*C)
print(Gud.pole())

Gyd = minreal(P*feedback(1, P*C))
print(Gyd.pole())

#Gur = C*feedback(1, P*C) #インプロパー
#print(Gud.pole())
```

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

$$C(s) = \frac{2s^2+10s+3}{s}$$



$$G_{yr} = \mathrm{feedback}(\Sigma_1, \Sigma_2)$$

```
minreal(P*C/(1+P*C))
```

特性多項式の根を計算する →

```
[[Np]], [[Dp]] = tfdata(P)
[[Nc]], [[Dc]] = tfdata(C)

f1 = np.poly1d(Np)
f2 = np.poly1d(Nc)
g1 = np.poly1d(Dp)
g2 = np.poly1d(Dc)

print(np.roots(f1*f2 + g1*g2))
```

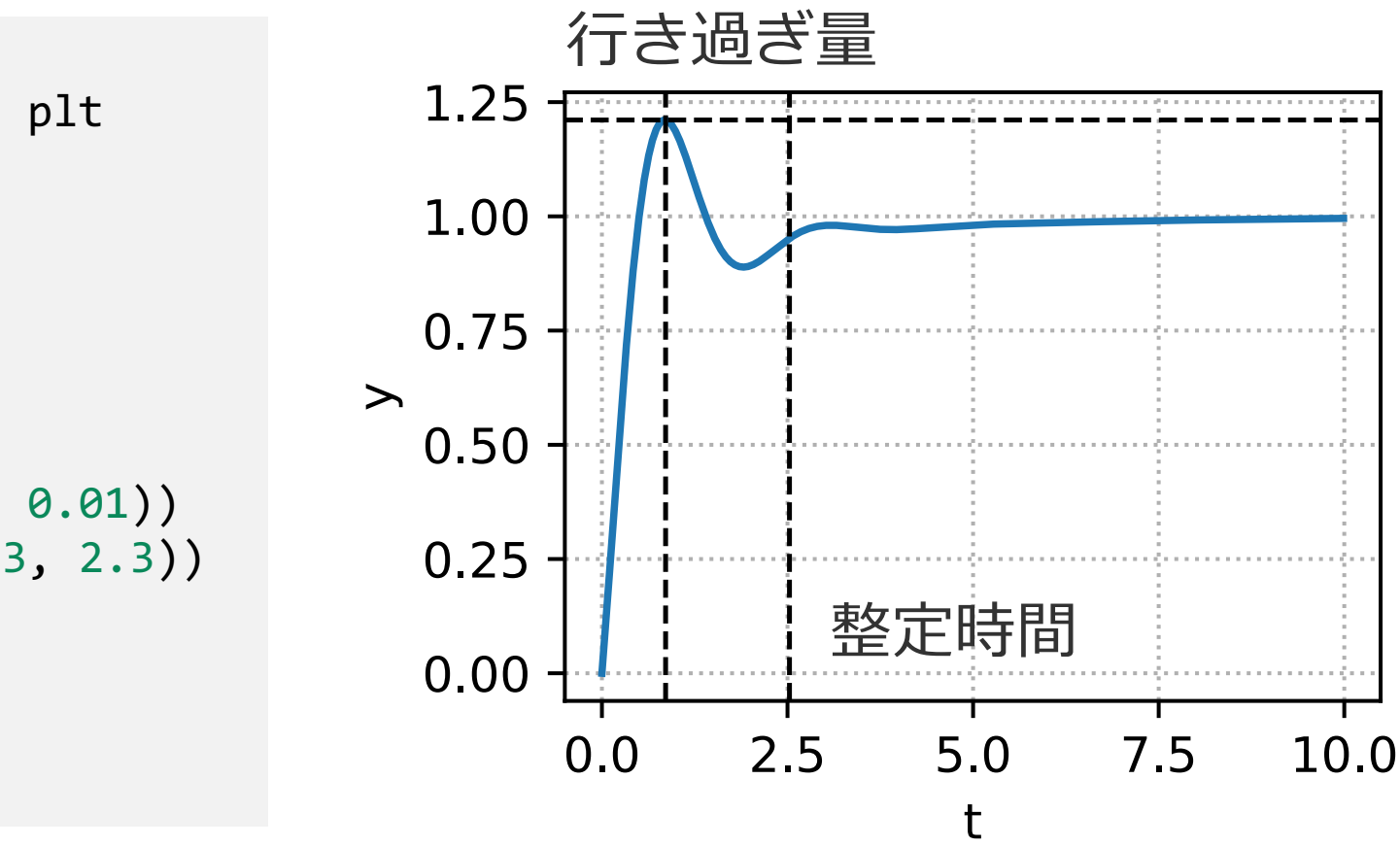
分子・分母多項式の係数を取得

多項式の生成

Python演習３

フィードバック系の応答特性（ステップ応答）

```
from control.matlab import *
from matplotlib import pyplot as plt

P = tf([0, 1],[1, 1 ,1])
C = tf([2, 10, 3],[1, 0])

Gyr = feedback(P*C, 1)

y,t = step(Gyr, np.arange(0, 10, 0.01))
fig, ax = plt.subplots(figsize=(3, 2.3))
ax.plot(t,y)
ax.set_xlabel('t')
ax.set_ylabel('y')
ax.grid(ls=':')
```



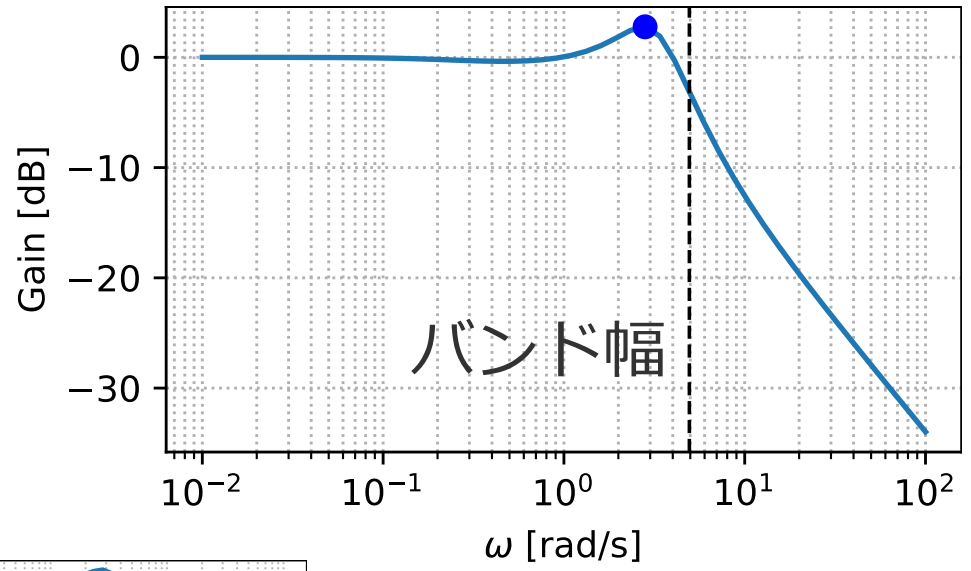

```
Info = stepinfo(Gyr, SettlingTimeThreshold=0.05)
print(Info)
```

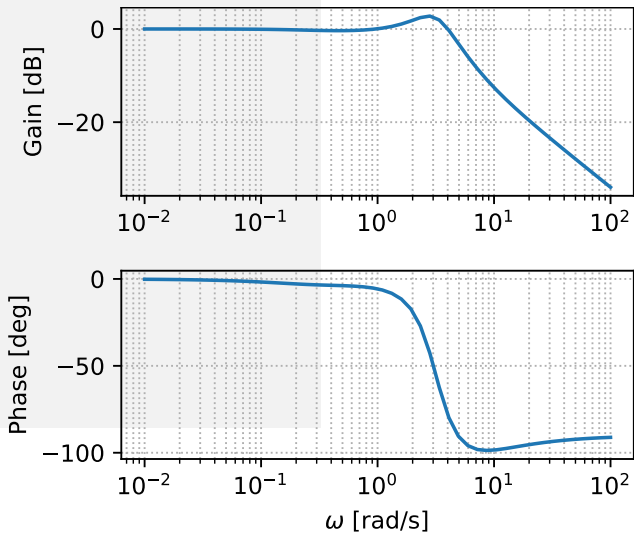
Python演習３

フィードバック系の応答特性（周波数応答）

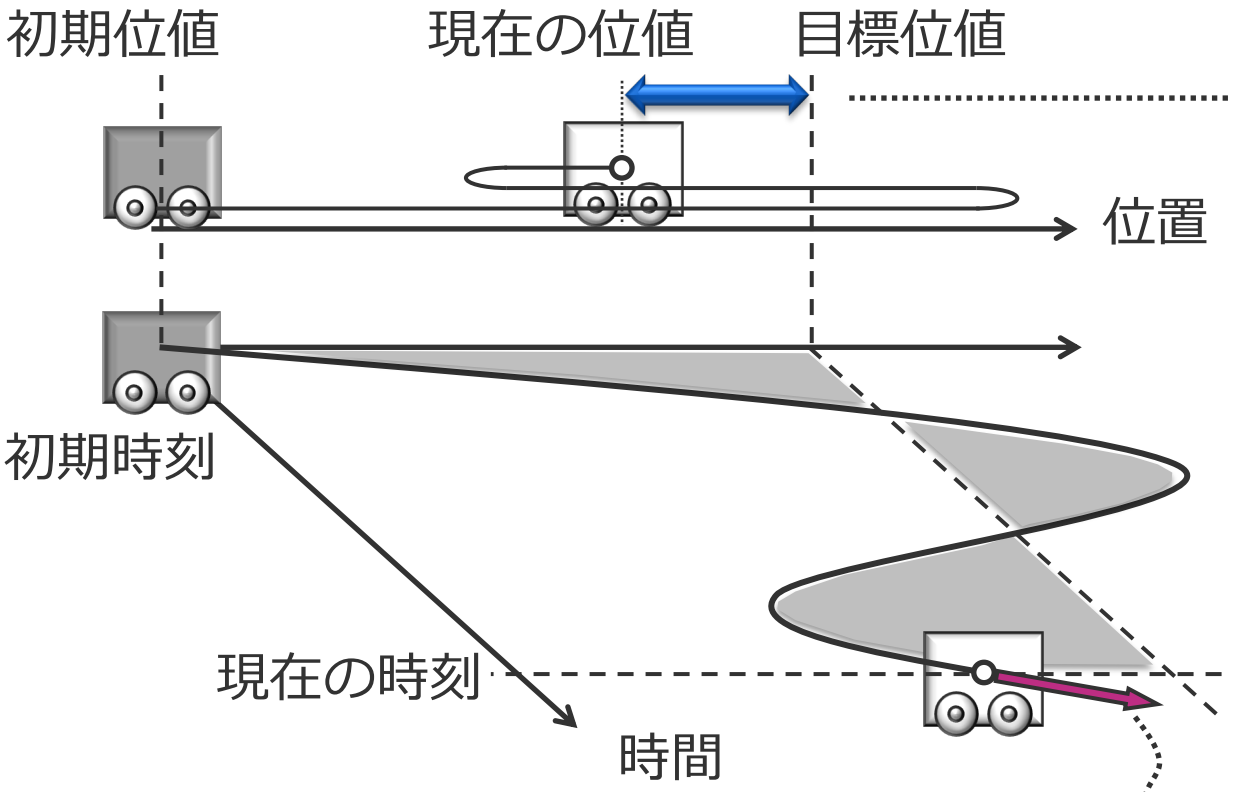
```
P = tf([0, 1],[1, 1 ,1])
C = tf([2, 10, 3],[1, 0])
Gyr = feedback(P*C, 1)

gain, phase, w = bode(Gyr, logspace(-2,2), Plot=False)

fig, ax = plt.subplots(2,1, figsize=(4, 3.5))
ax[0].semilogx(w, 20*np.log10(gain))
ax[1].semilogx(w, phase*180/np.pi)

ax[0].grid(which="both", ls=':')
ax[0].set_ylabel('Gain [dB]')
ax[1].grid(which="both", ls=':')
ax[1].set_xlabel('$¥omega$ [rad/s]')
ax[1].set_ylabel('Phase [deg]')
fig.tight_layout()
```

ピークゲイン





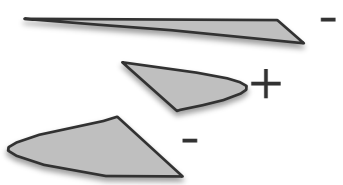
PID制御



P制御
（現在の情報）

偏差 × k_P

I制御
（過去の情報）



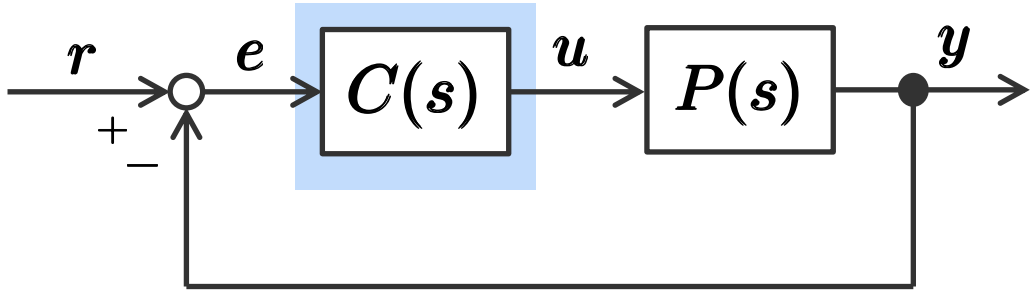
偏差の積分値 × k_I

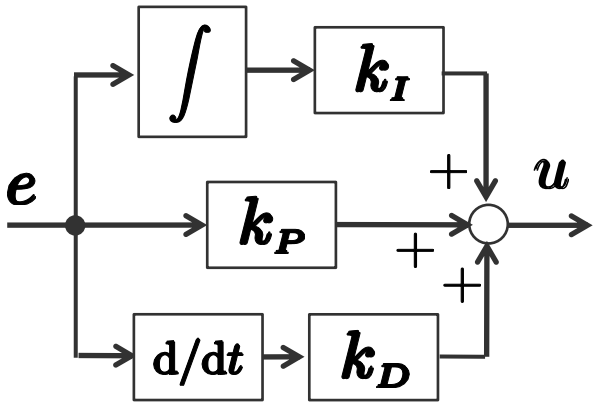
D制御
（未来の情報）

傾き

偏差の微分値 × k_D

PID制御





$$u(t) = k_P e(t) + k_I \int_0^t e(t)\mathrm{d}t + k_D \frac{\mathrm{d}}{\mathrm{d}t} e(t)$$

$$u(s) = C(s)e(s) \quad C(s) = k_P + \frac{k_I}{s} + k_D s$$

P　　I　　D

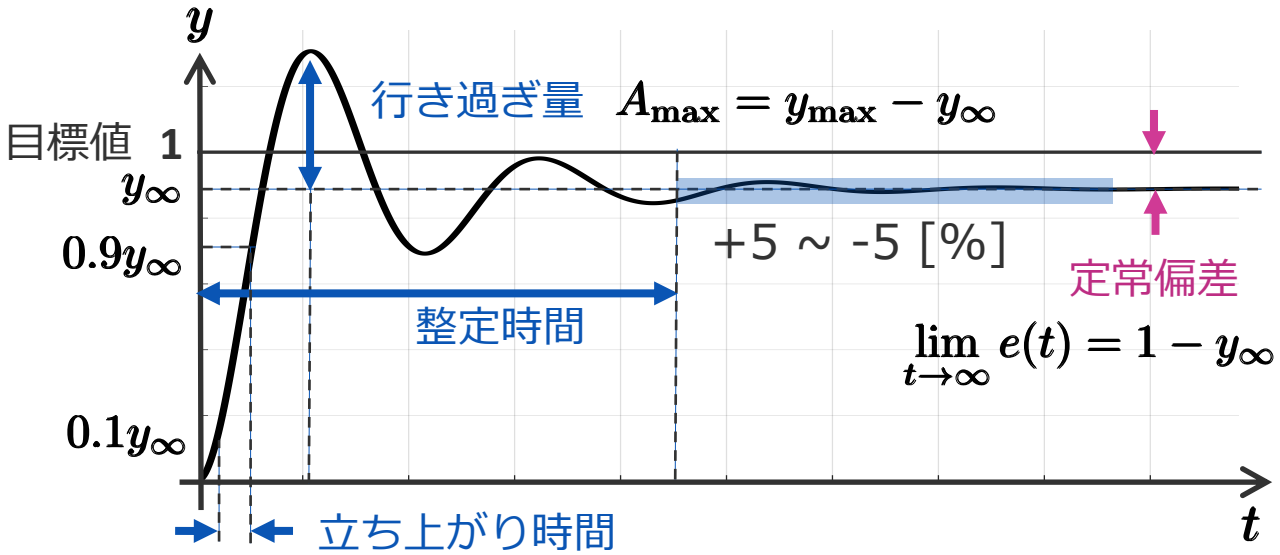
Proportional　　P制御：速応性に寄与

Integral　　I制御：定常偏差改善，位相が遅れる

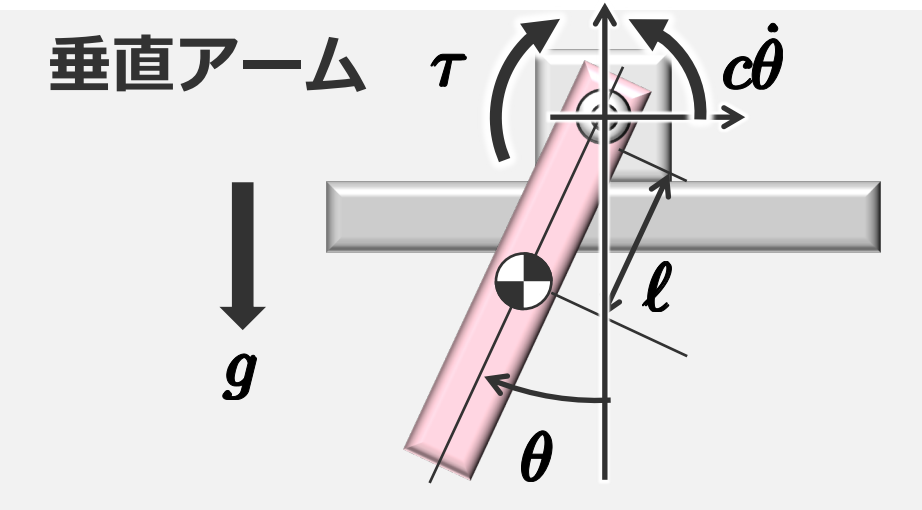
Derivative　　D制御：位相が進む，ノイズに敏感

PID制御

	立ち上がり時間	オーバーシュート	整定時間	定常偏差
P制御	小さくなる	大きくなる	あまり変わらない	小さくなる
I制御	小さくなる	大きくなる	大きくなる	小さくなる（零になる）
D制御	あまり変わらない	小さくなる	小さくなる	変わらない

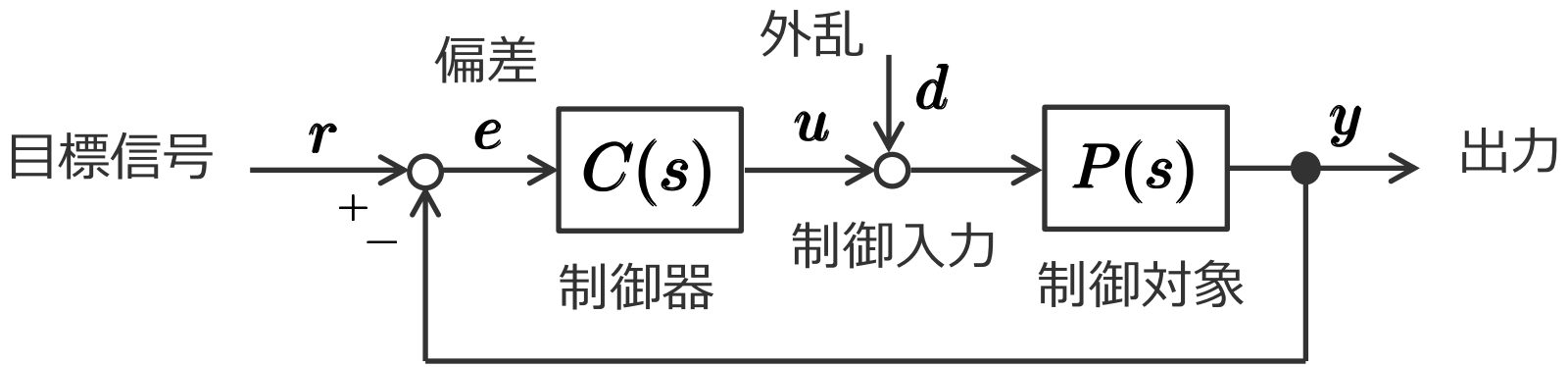


垂直アームのPID制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

```
g = 9.81      # 重力加速度[m/s^2]
l = 0.2       # アームの長さ[m]
m = 0.5       # アームの質量[kg]
c = 1.5e-2    # 粘性摩擦係数
J = 1.0e-2    # 慣性モーメント
```



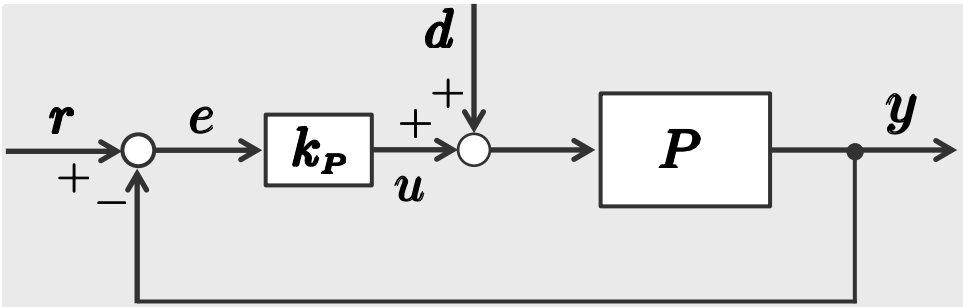
目標値追従

$$G_{yr}(s) = \frac{P(s)C(s)}{1 + P(s)C(s)}$$

外乱抑制

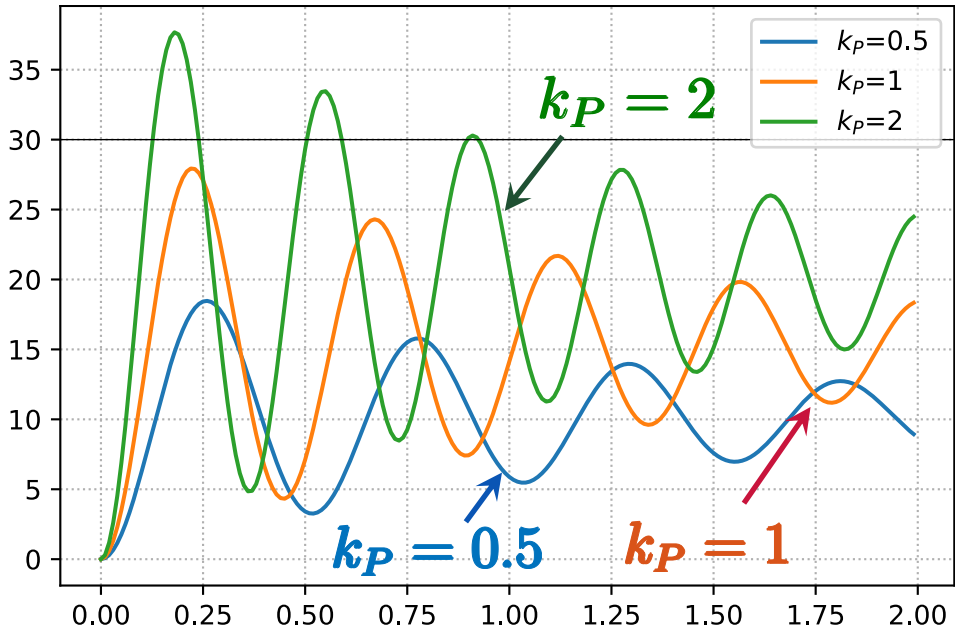
$$G_{yd}(s) = \frac{P(s)}{1 + P(s)C(s)}$$

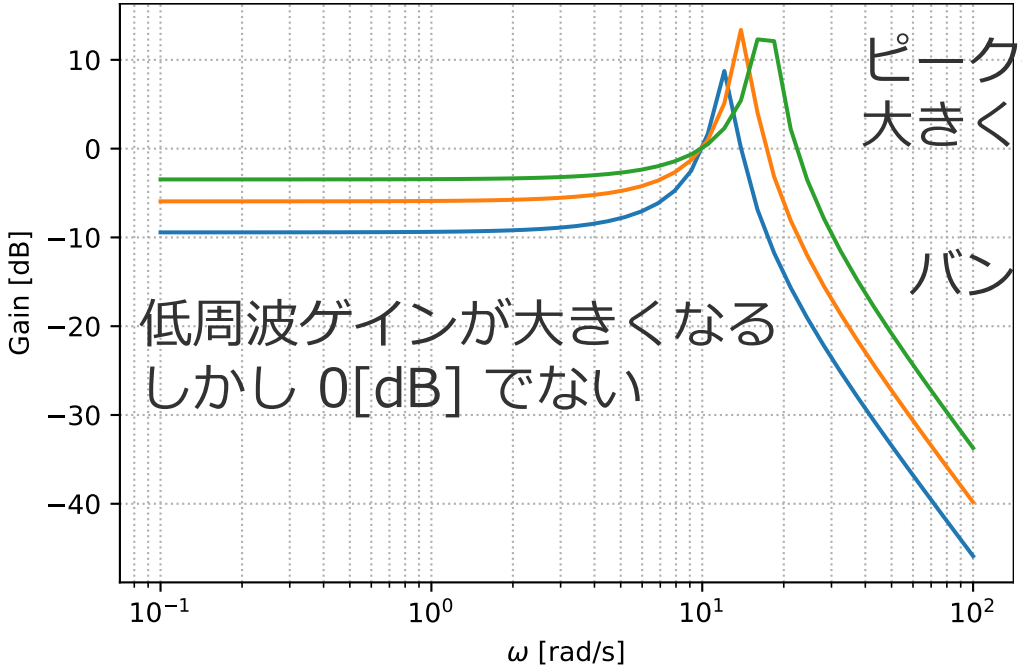
P制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

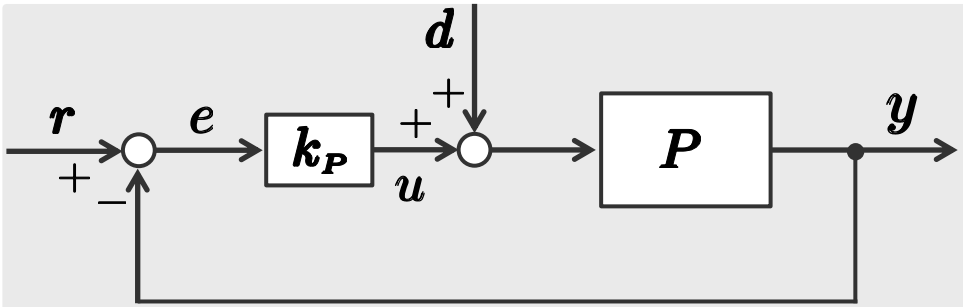
$$C(s) = k_P$$





k_P を大きくすると，
立ち上がりが速くなるが，振動的になる．また，定常偏差が小さくなる

P制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

$$C(s) = k_P$$

$$G_{yr}(s) = \frac{P(s)k_P}{1 + P(s)k_P} = \frac{k_P}{Js^2 + cs + mg\ell + k_P} = \frac{K\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

$$\omega_n = \sqrt{\frac{mg\ell + k_P}{J}} \qquad \zeta = \frac{c}{2\sqrt{J(mg\ell + k_P)}} \qquad K = \frac{k_P}{mg\ell + k_P}$$

$$s = \frac{-c \pm \sqrt{c^2 - 4J(mg\ell + k_P)}}{2J}$$

k_P を大きくすると，応答は速くなるが，振動的になる

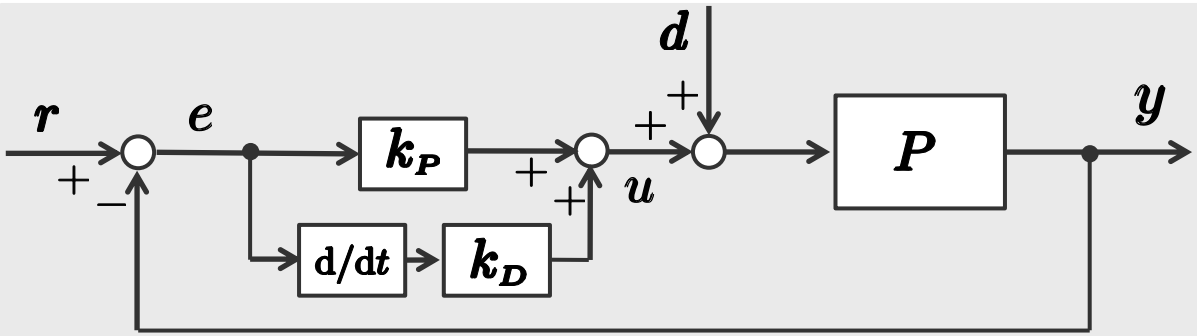
速応性と減衰性を同時に改善できない

$G_{yr}(0) = K \neq 1$ なので，定常偏差が残る

最終値の定理

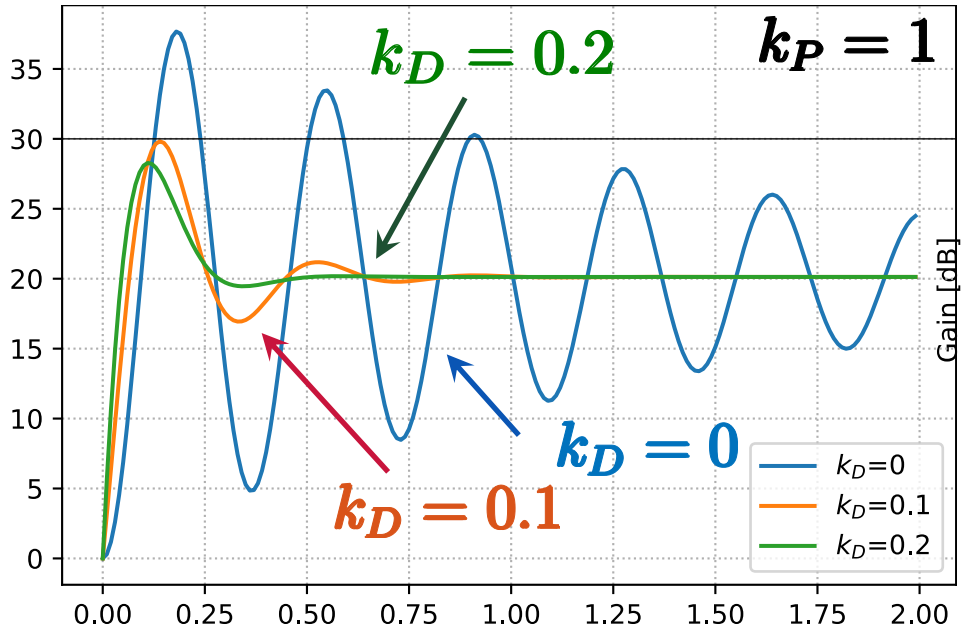
$$y(\infty) = \lim_{s \to 0} G_{yr}(s)$$

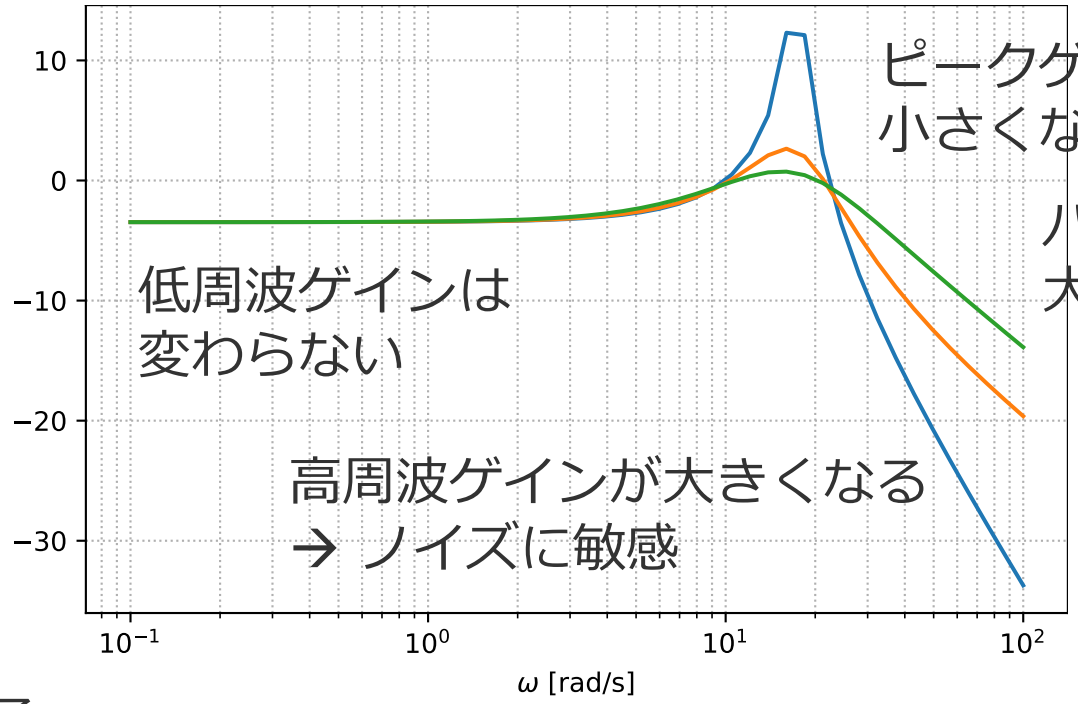
PD制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

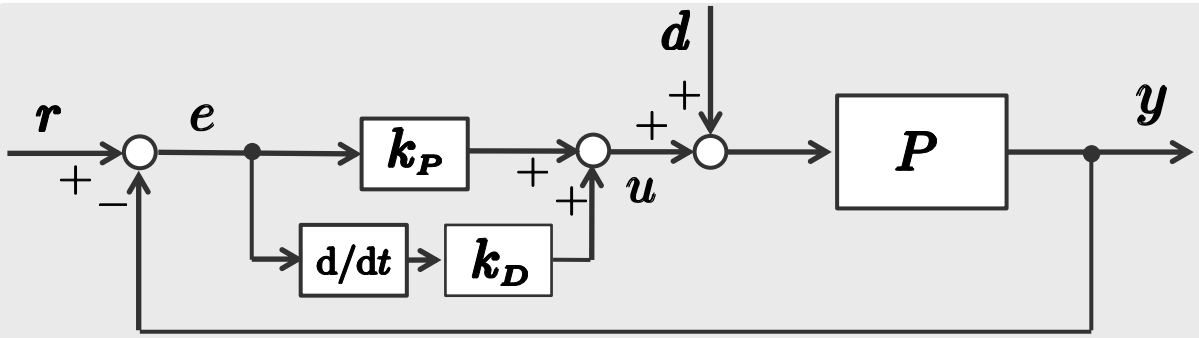
$$C(s) = k_P + k_D s$$





k_D を大きくすると，振動が抑えられる
しかし，定常偏差は残る

PD制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

$$C(s) = k_P + k_D s$$

$$G_{yr}(s) = \frac{P(s)(k_P + k_D s)}{1 + P(s)(k_P + k_D s)} = \frac{k_D s + k_P}{Js^2 + (c + k_D)s + mg\ell + k_P}$$

$$G_{yr}(0) = \frac{k_P}{mg\ell + k_P} \neq 1 \quad \boldsymbol{G_{yr}(0) = K \neq 1}$$ なので，定常偏差が残る

極 $$s = \frac{-(c + k_D) \pm \sqrt{(c + k_D)^2 - 4J(mg\ell + k_P)}}{2J}$$

零点 $$s = -\frac{k_P}{k_D}$$

速応性と減衰性を同時に改善できる！

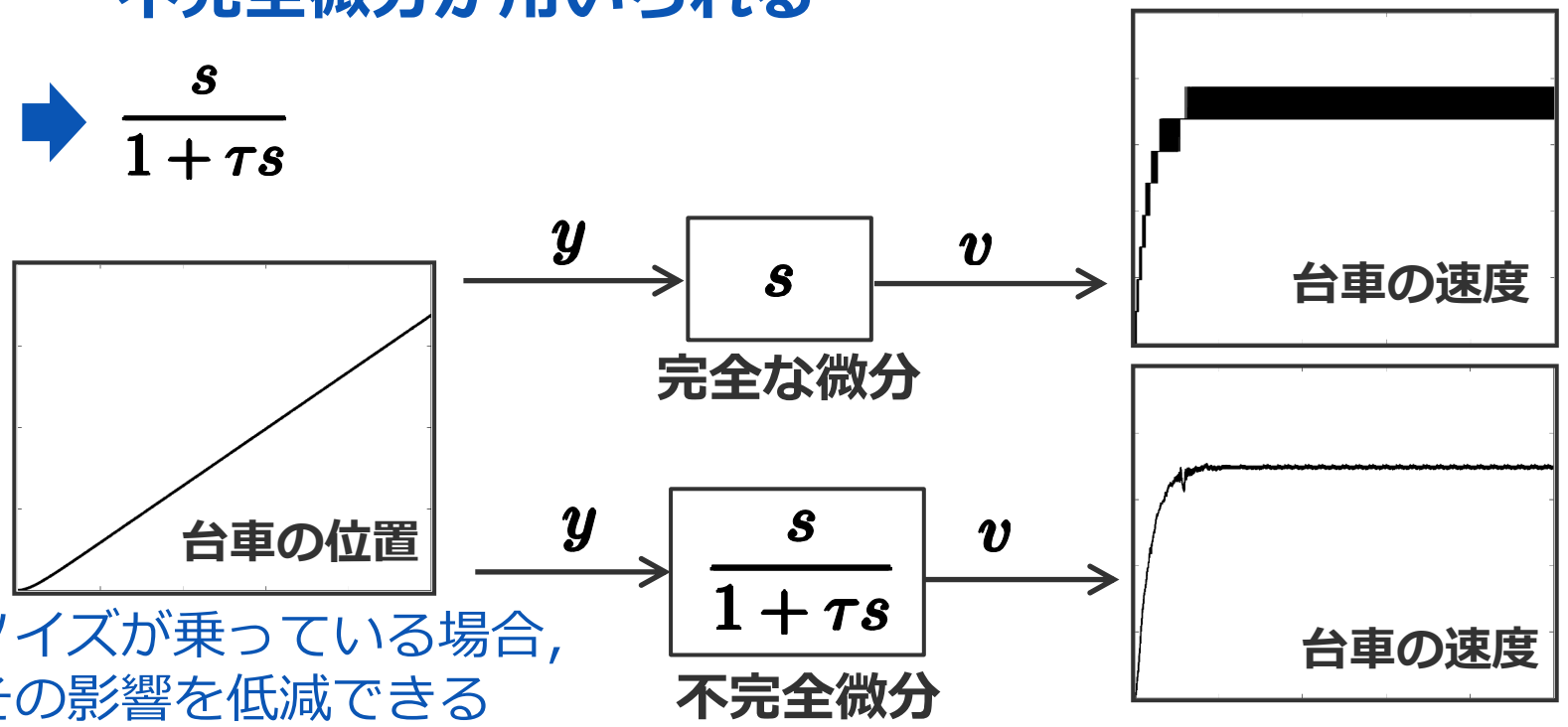
【補足】不完全微分

PD制御では，偏差の微分情報を用いている
しかし，完全な微分を実現することは困難である

$C(s) = k_P + k_D s$ はプロパーでない

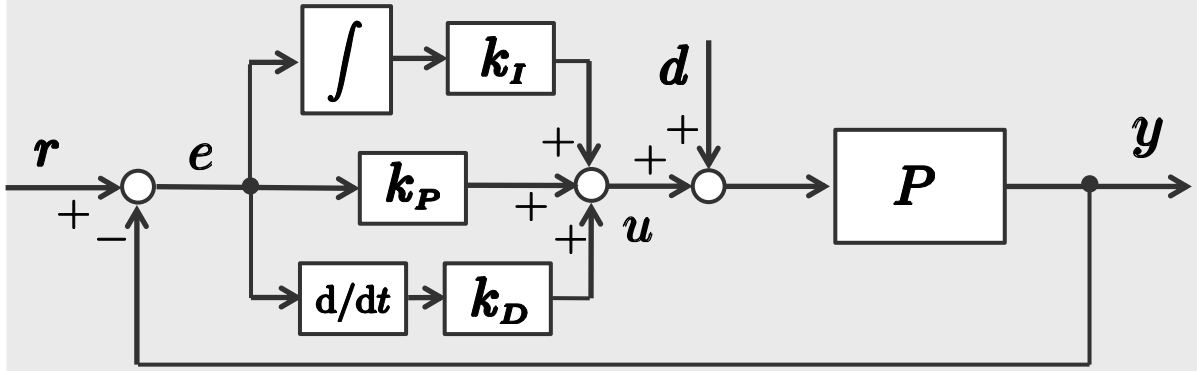
➡ **実際は，１次のフィルタを付加した不完全微分が用いられる**

s ➡ $\dfrac{s}{1+\tau s}$



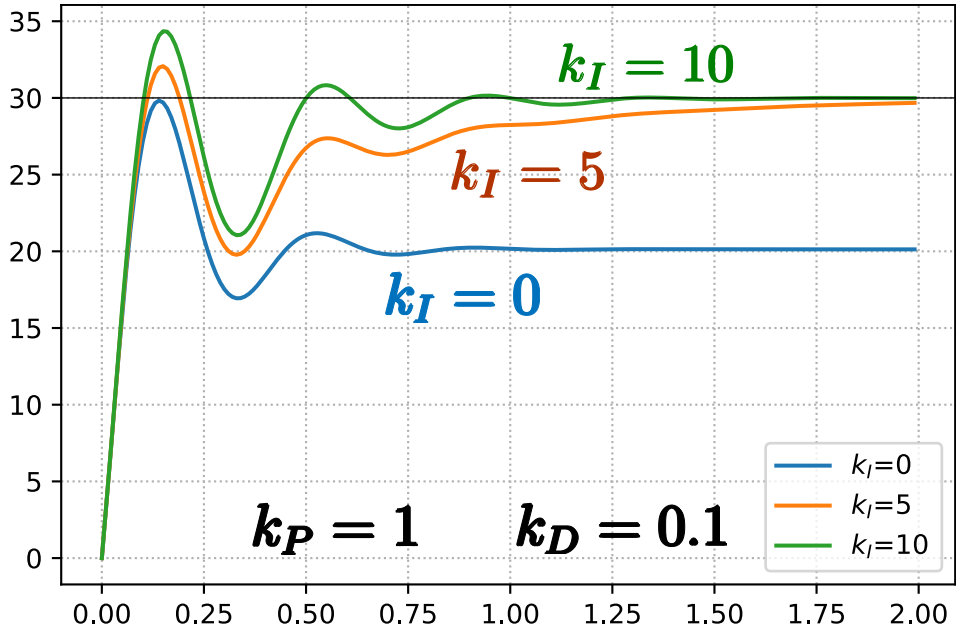
ノイズが乗っている場合，
その影響を低減できる

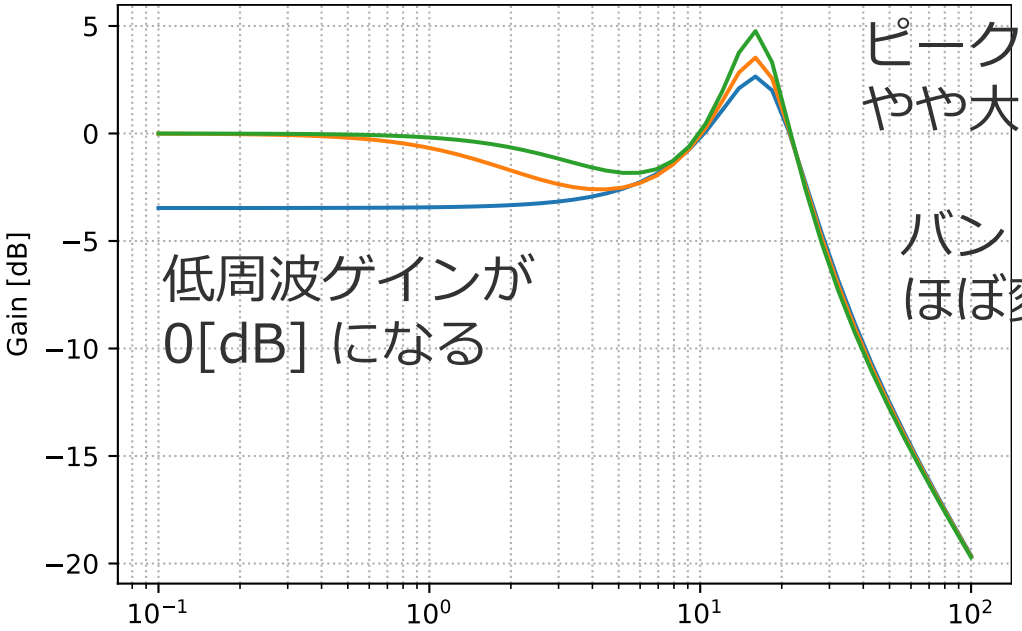
PID制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

$$C(s) = k_P + k_D s + \frac{k_I}{s}$$





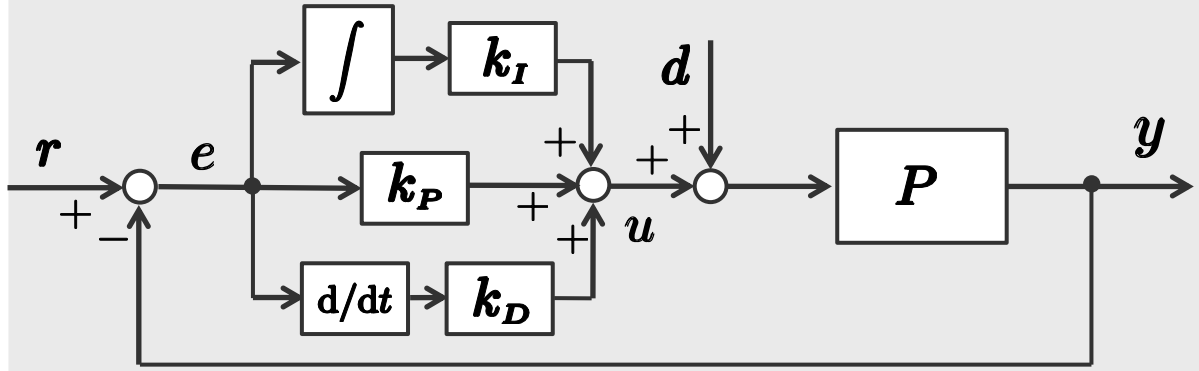
ピークゲインが
やや大きくなる

バンド幅は
ほぼ変わらない

低周波ゲインが
0[dB] になる

定常偏差（ステップ目標値に対する偏差）をゼロにできる
しかし，k_I を大きくすると振動的になる

PID制御



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

$$C(s) = k_P + k_D s + \frac{k_I}{s}$$

$$G_{yr}(s) = \frac{P(s)\frac{k_I + k_P s + k_D s^2}{s}}{1 + P(s)\frac{k_I + k_P s + k_D s^2}{s}} = \frac{k_D s^2 + k_P s + k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$$

$G_{yr}(0) = 1$ → 定常偏差がゼロ！

速応性と減衰性を同時に改善できる！

$$G_{yd}(s) = \frac{P(s)}{1 + P(s)\frac{k_I + k_P s + k_D s^2}{s}} = \frac{s}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$$

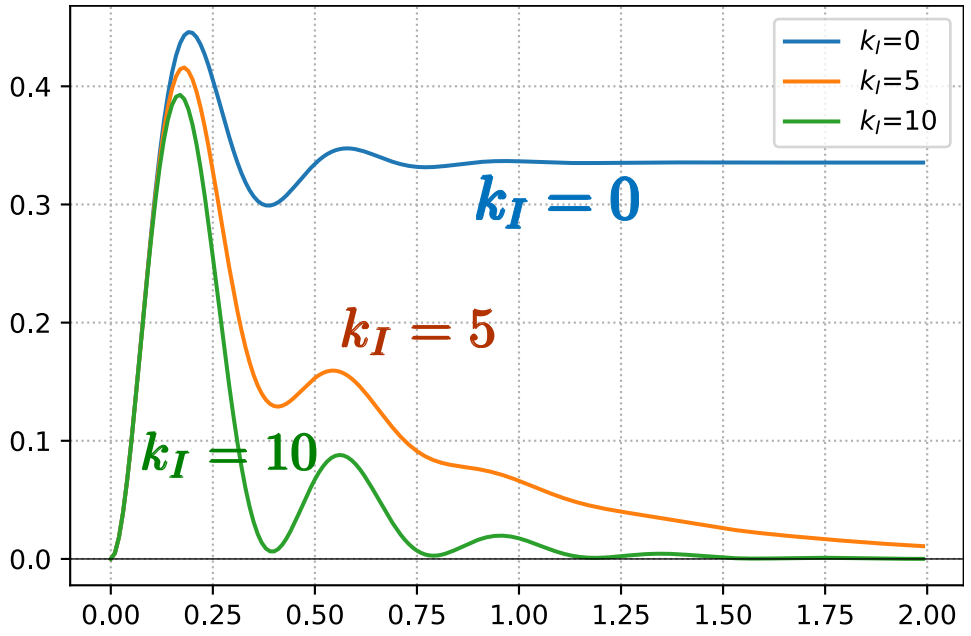
$G_{yd}(0) = 0$ → 定値外乱があっても，偏差がゼロになる！

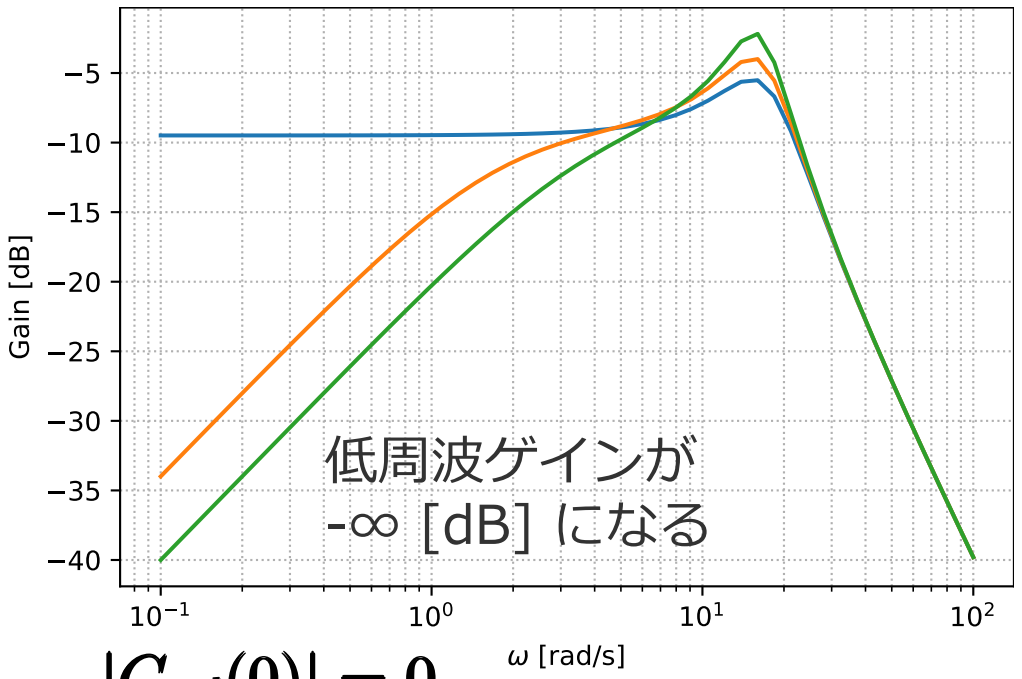
PID制御

定値外乱の抑制特性

$$G_{yd}(s) = \frac{s}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$$

$k_P = 1 \quad k_D = 0.1$





$|G_{yd}(0)| = 0$

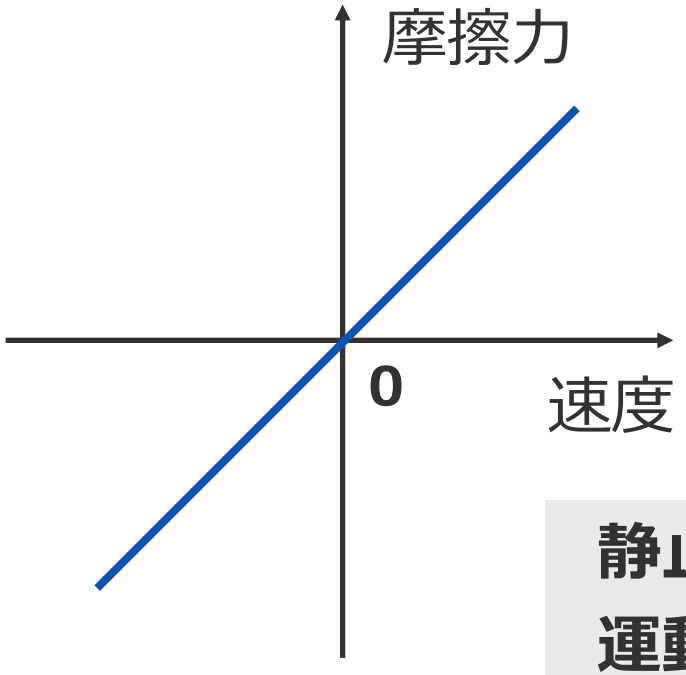
I制御によって，外乱抑制が可能

【補足】現実の定値外乱

線形摩擦

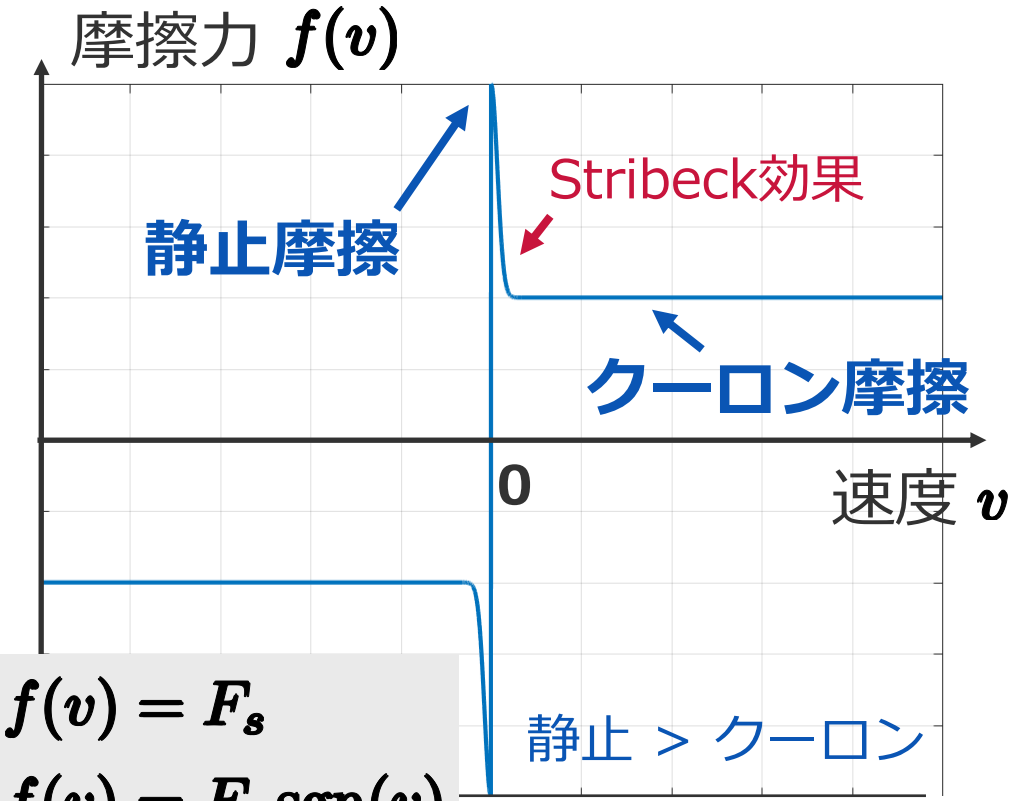
→ モデルに
組み込まれている

粘性摩擦



非線形摩擦 →定値外乱の例

$$f(v) = \left(F_c + (F_s - F_c)e^{-(v/v_s)^2}\right)\mathrm{sgn}(v)$$



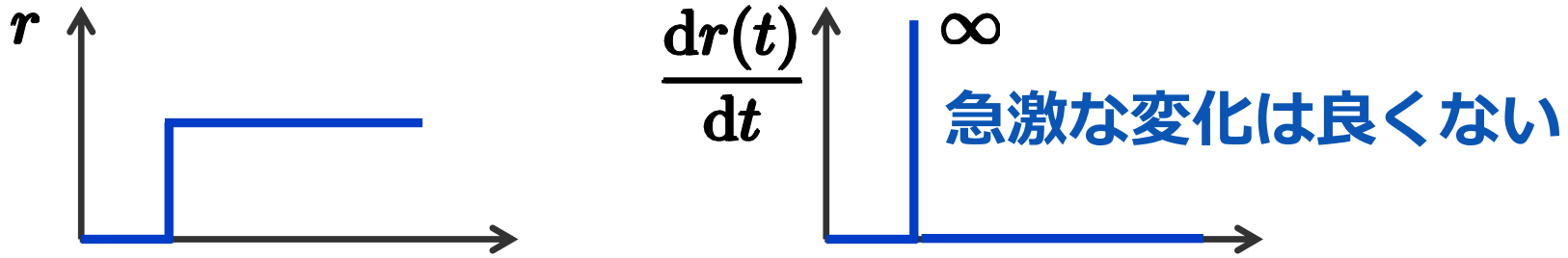
静止： $f(v) = F_s$

運動： $f(v) = F_c\,\mathrm{sgn}(v)$

2自由度制御系（PI-D制御）

PID制御では，偏差の微分情報を利用する
ステップ上に変化する目標値を微分すると∞の値となる
（不完全微分を用いても，その値は過大になる）

→ 制御入力が過大になる

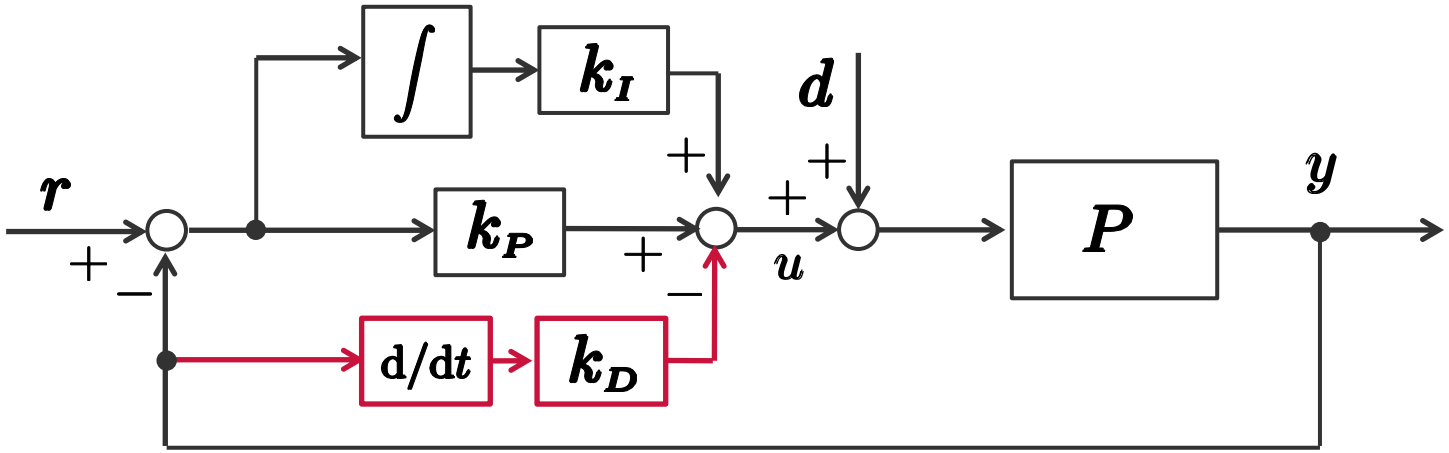


解決策：目標値の微分情報を使わない！

$$\frac{de(t)}{dt} = \frac{dr(t)}{dt} - \frac{dy(t)}{dt} \quad \Rightarrow \quad -\frac{dy(t)}{dt}$$

2自由度制御系（PI-D制御）

微分先行型PID制御（PI-D制御）

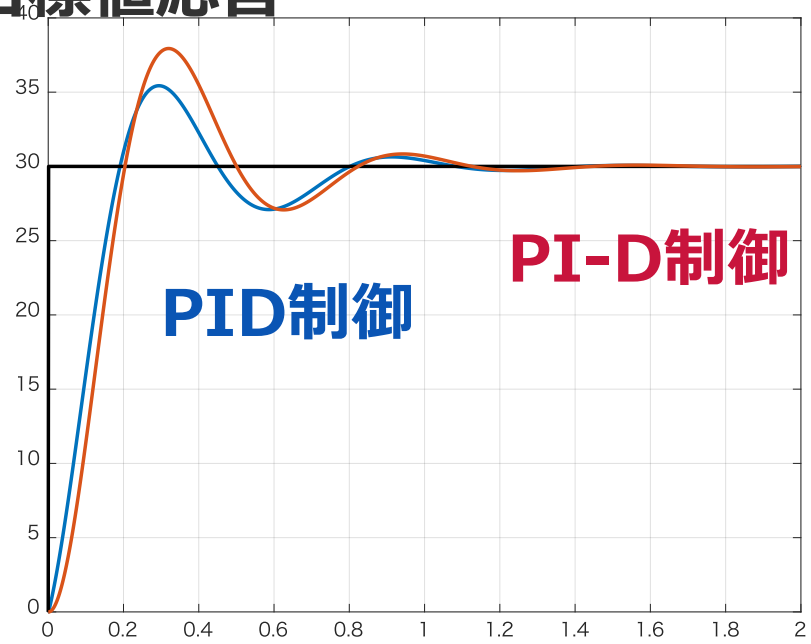


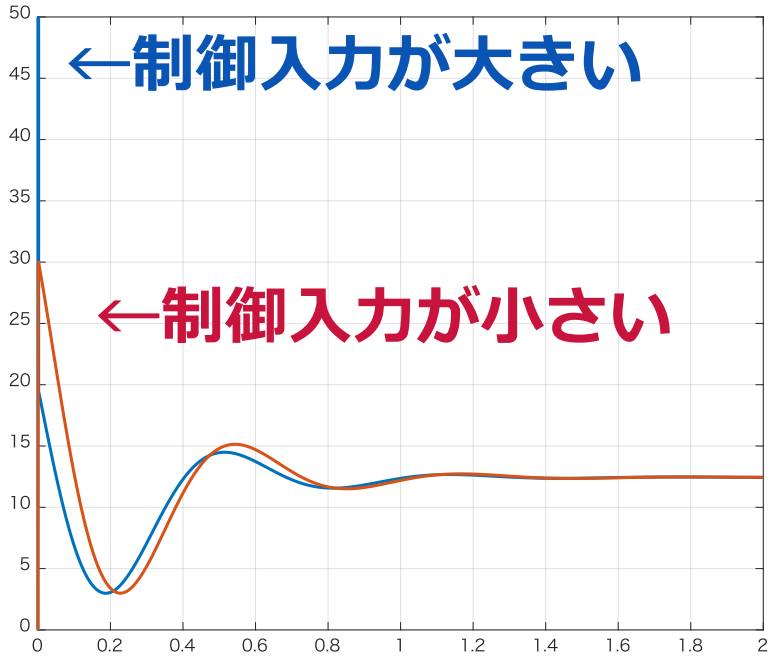
➡ **解決策：目標値の微分情報を使わない！**

$$\frac{\mathrm{d}e(t)}{\mathrm{d}t} = \frac{\mathrm{d}r(t)}{\mathrm{d}t} - \frac{\mathrm{d}y(t)}{\mathrm{d}t} \quad \Rightarrow \quad -\frac{\mathrm{d}y(t)}{\mathrm{d}t}$$

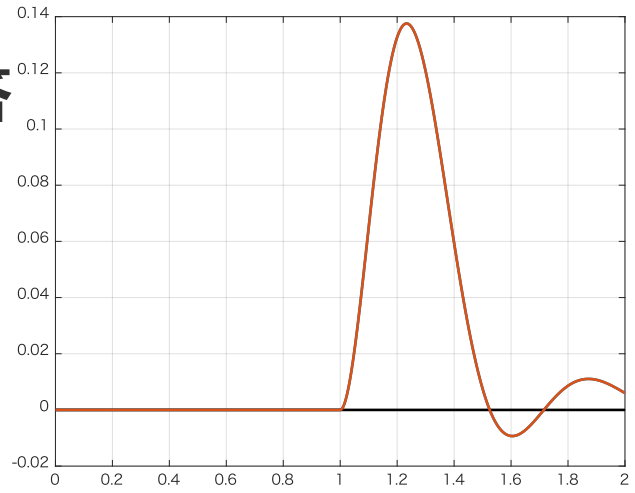
2自由度制御系（PI-D制御）

目標値応答





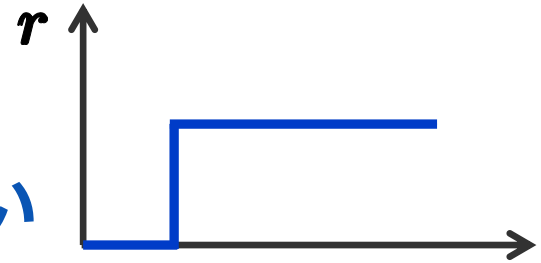
外乱抑制応答



PID制御とPI-D制御は同じ

2自由度制御系（I-PD制御）

同様に．．．ステップ目標値も良くない？

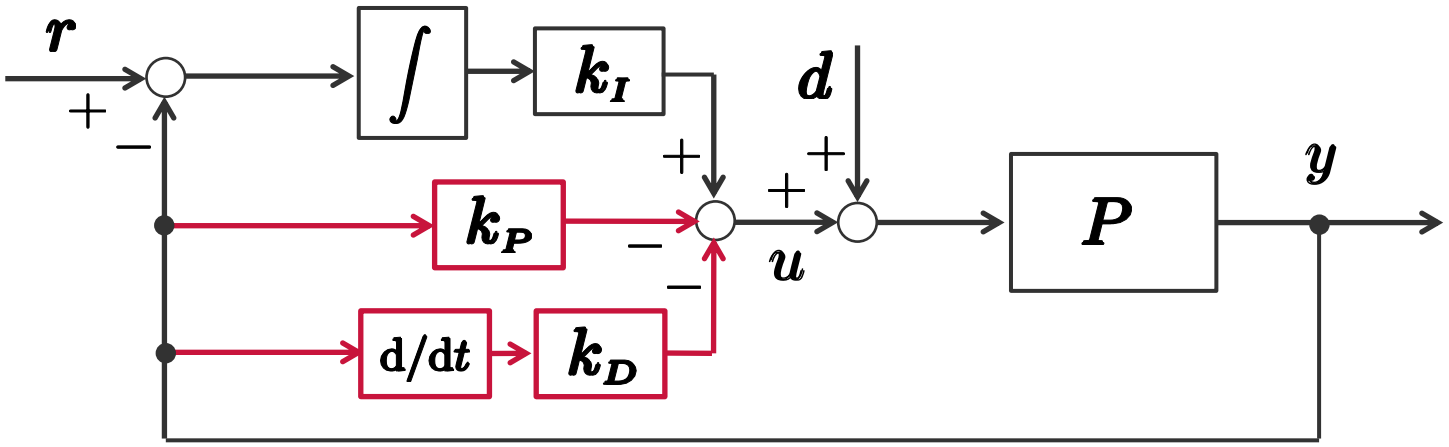


急激な変化は良くない

➡ 解決策：P制御において，目標値の情報を使わない！

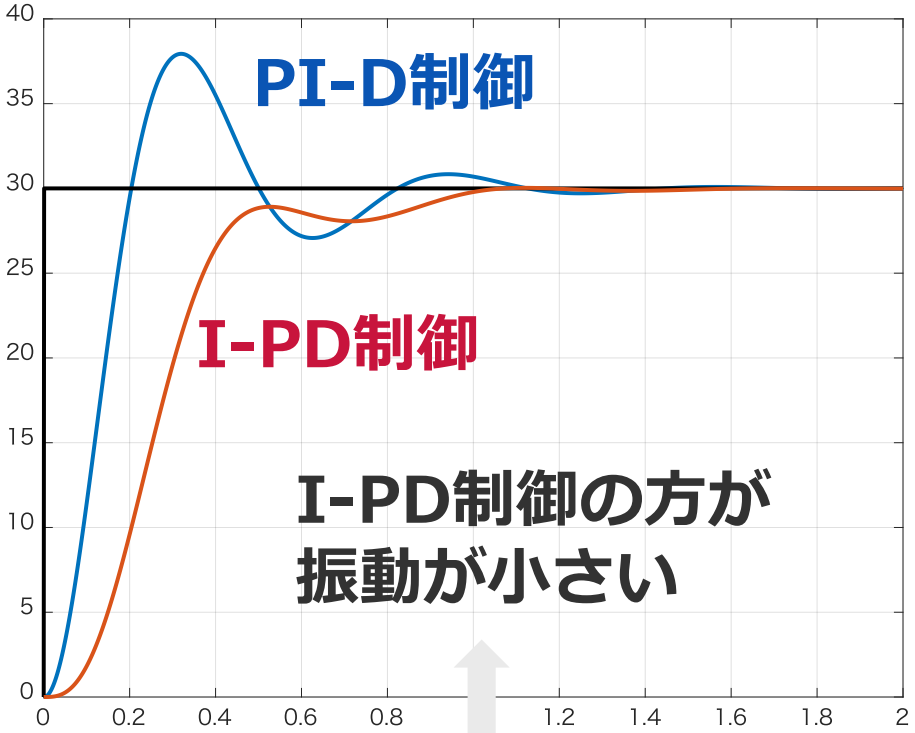
$$e(t) = r(t) - y(t) \quad \Rightarrow \quad -y(t)$$

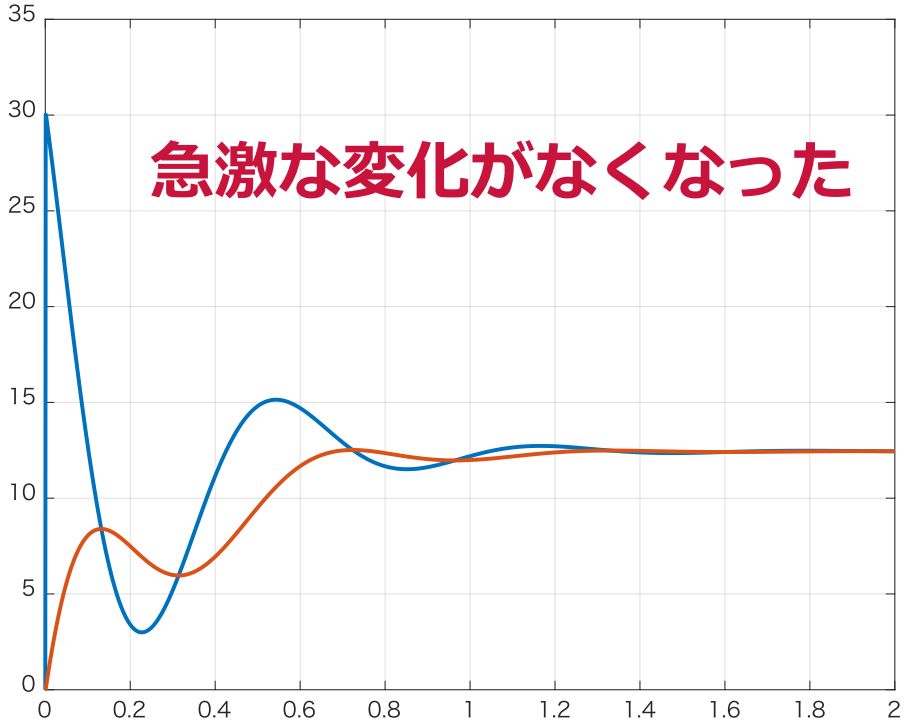
比例
微分
先行型
PID制御



2自由度制御系（I-PD制御）

目標値応答



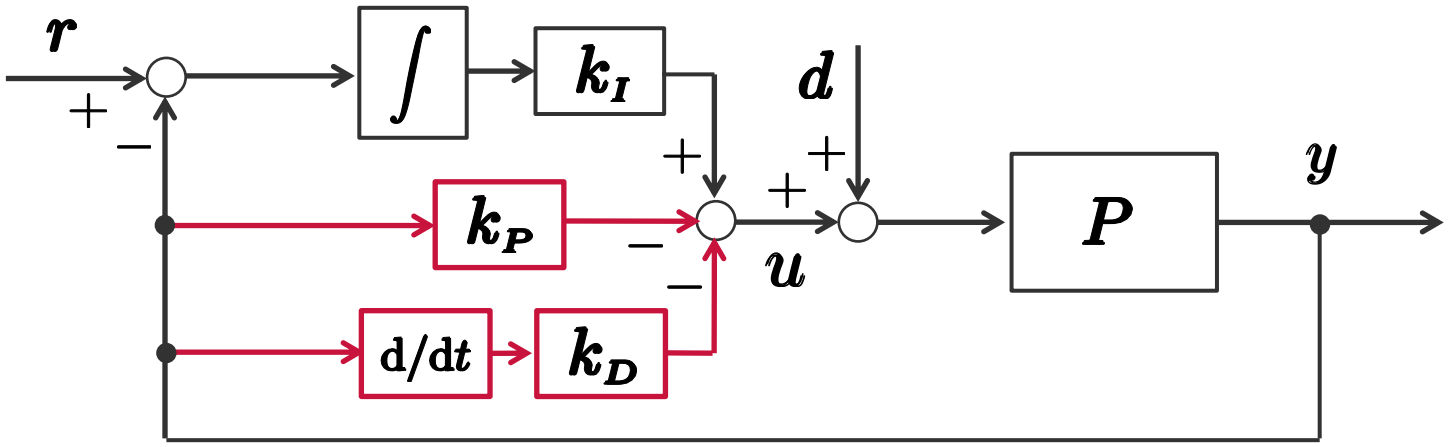


零点の影響

PI-D制御：$G_{yr}(s) = \dfrac{k_P s + k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$

I-PD制御：$G_{yr}(s) = \dfrac{k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$

2自由度制御系



$$u(s) = \frac{k_I}{s} r(s) - \frac{k_D s^2 + k_P s + k_I}{s} y(s)$$

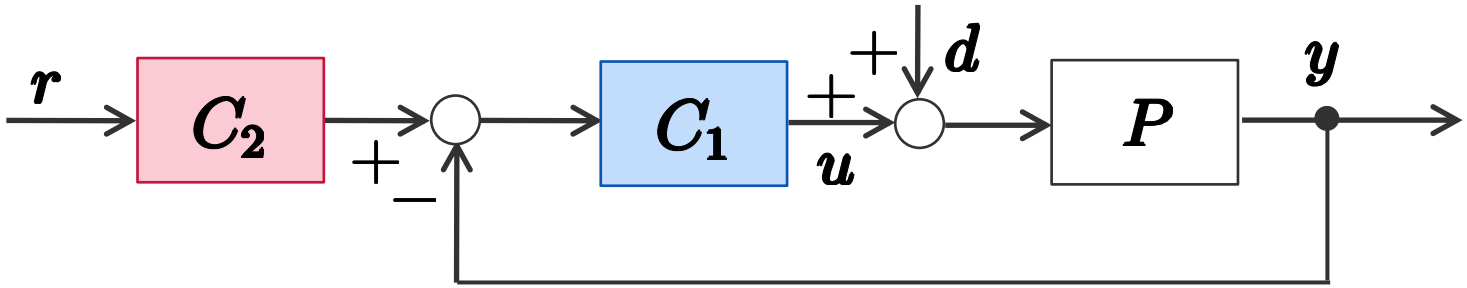
$$= \frac{k_D s^2 + k_P s + k_I}{s} \left(\frac{k_I}{k_D s^2 + k_P s + k_I} r(s) - y(s) \right)$$

$$= C_1(s) \left(C_2(s) r(s) - y(s) \right)$$

2自由度制御系

I-PD制御は，以下のブロック線図で表現できる

$$u(s) = C_1(s)\left(C_2(s)r(s) - y(s)\right)$$



$$C_1(s) = \frac{k_D s^2 + k_P s + k_I}{s}$$

PID制御器

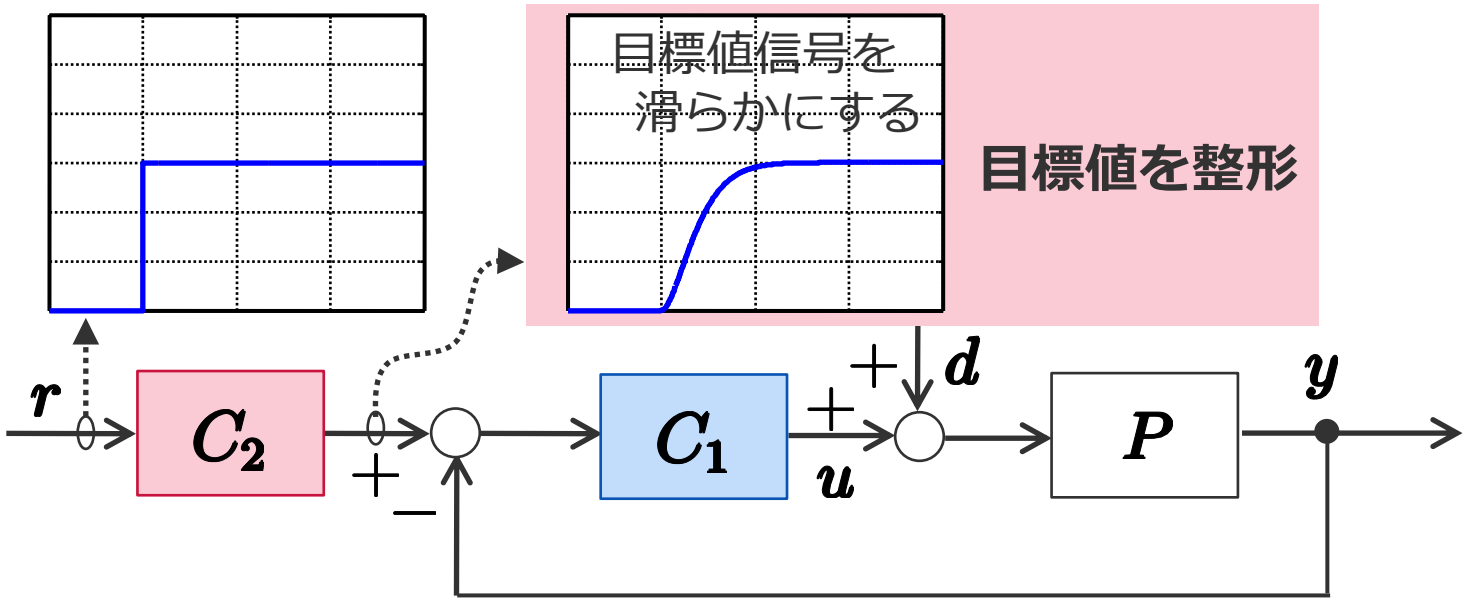
$$C_2(s) = \frac{k_I}{k_D s^2 + k_P s + k_I}$$

2次遅れ要素

通常のPID制御器になっているので，
外乱抑制の性能は変わらない

急激な変化が
緩和される

2自由度制御系



$$C_1(s) = \frac{k_D s^2 + k_P s + k_I}{s}$$

PID制御器

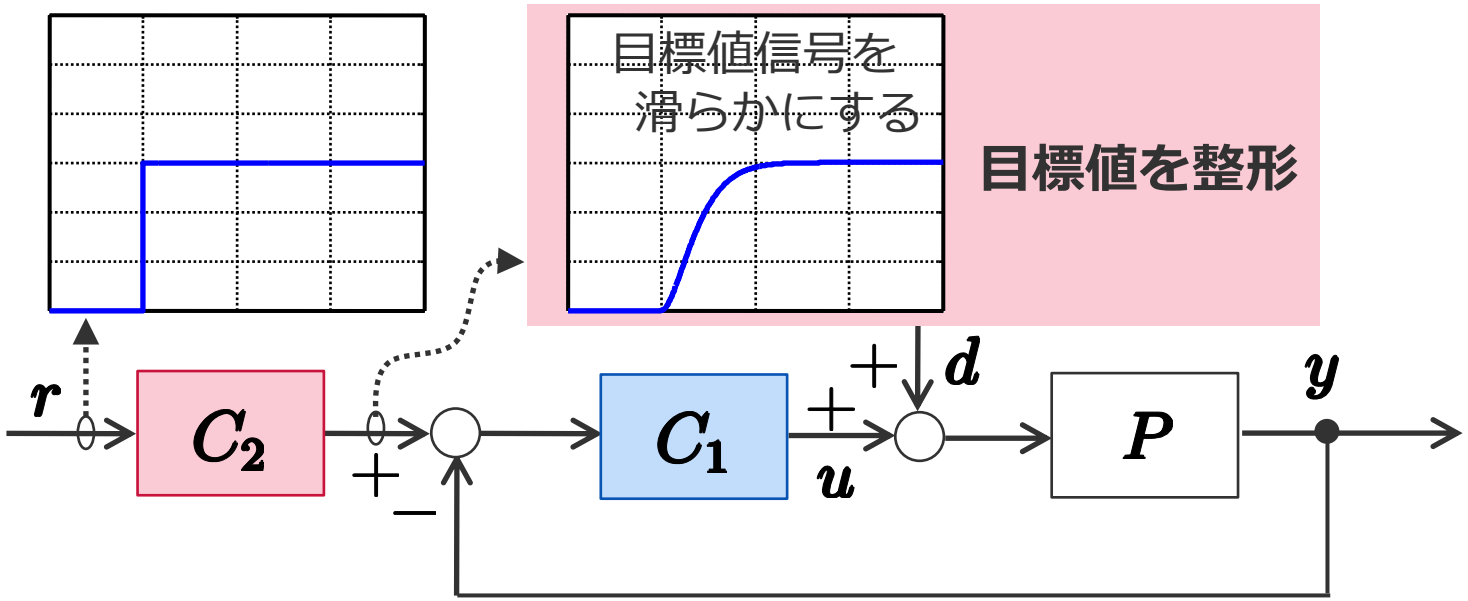
$$C_2(s) = \frac{k_I}{k_D s^2 + k_P s + k_I}$$

2次遅れ要素

通常のPID制御器になっているので，
外乱抑制の性能は変わらない

急激な変化が
緩和される

2自由度制御系



$$C_1(s) = \frac{k_D s^2 + k_P s + k_I}{s}$$

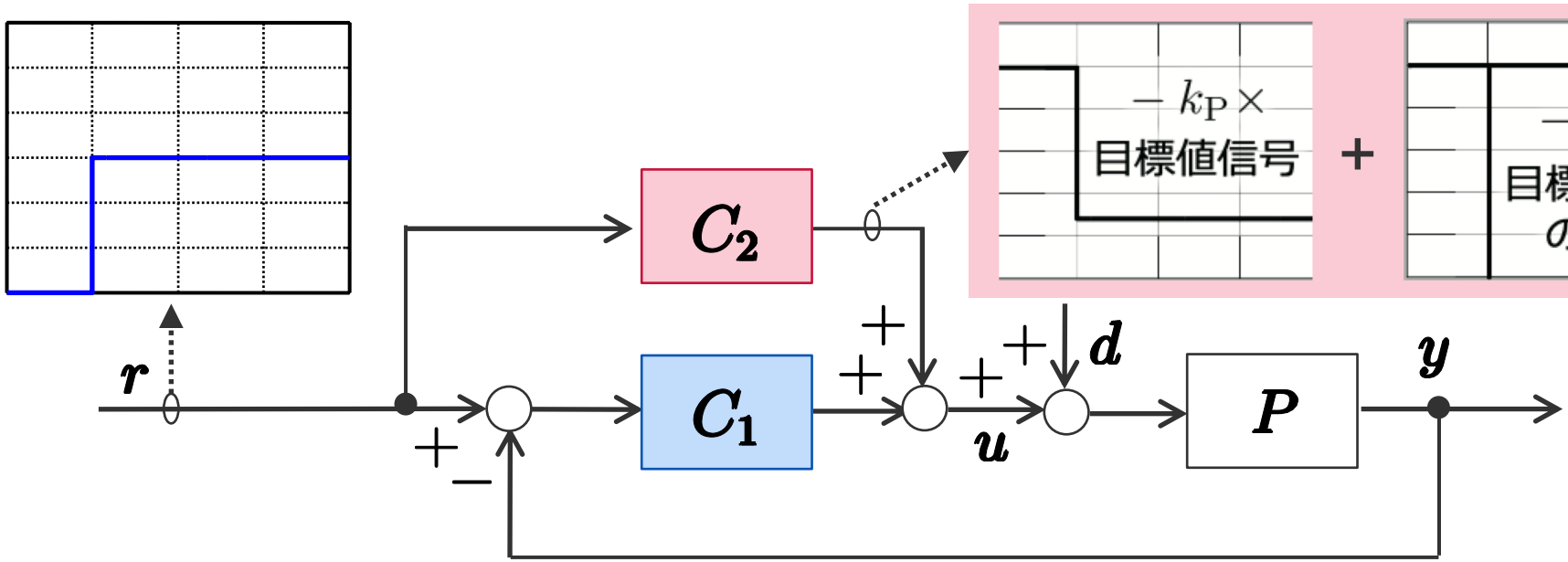
$$C_2(s) = \frac{k_I}{k_D s^2 + k_P s + k_I}$$

同様に，PI-D制御も

$$C_1(s) = \frac{k_D s^2 + k_P s + k_I}{s}$$

$$C_2(s) = \frac{k_P s + k_I}{k_D s^2 + k_P s + k_I}$$

【補足】別の解釈



$$C_1(s) = \frac{k_D s^2 + k_P s + k_I}{s}$$

$$C_2(s) = -k_P - k_D s$$

解釈はこれでよいが，シミュレーションはしにくい（ $C_2(s)$ がインプロパー）

ゲインチューニング

ヒューリスティックな方法による設計

$$u(t) = k_P e(t) + k_I \int_0^t e(t)\mathrm{d}t + k_D \frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

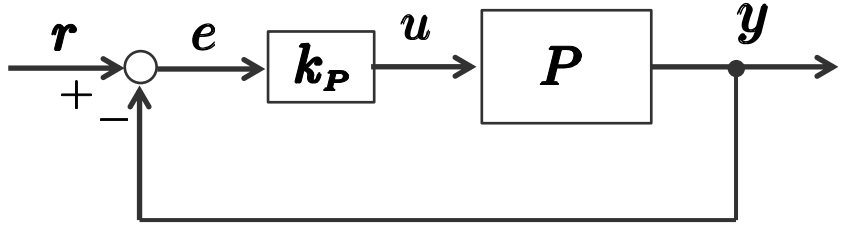
$$= k_P \left(e(t) + \frac{1}{T_I} \int_0^t e(t)\mathrm{d}t + T_D \frac{\mathrm{d}e(t)}{\mathrm{d}t} \right)$$

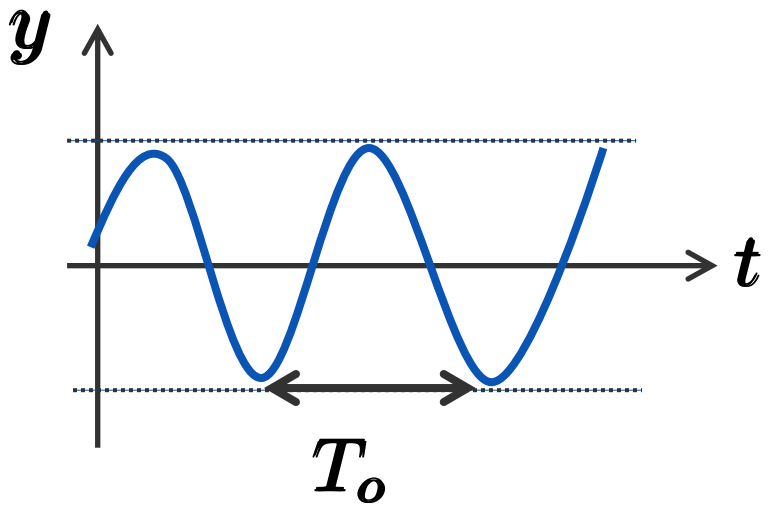
制御対象のモデルが不要

- Ziegler & Nichols の限界感度法
- Ziegler & Nichols のステップ応答法

限界感度法

限界感度法





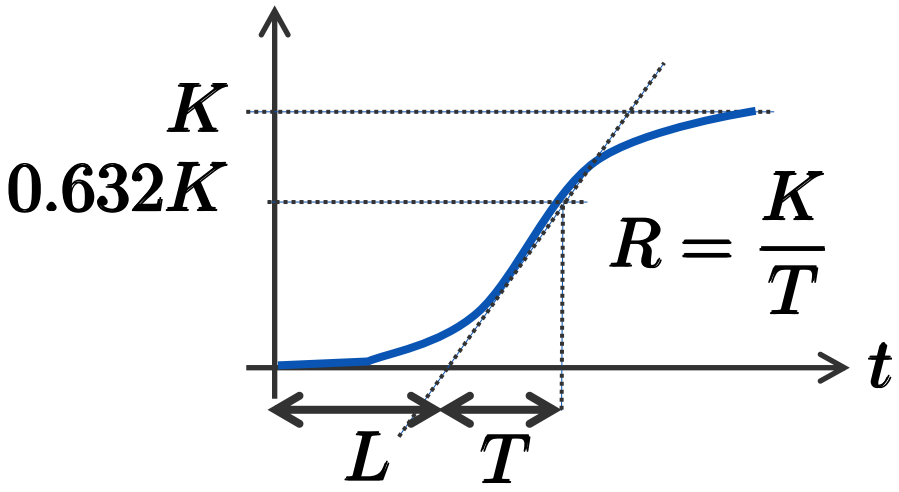
P制御のみで安定限界となる
$k_P = k_o$ を求め，持続振動の周期 T_o を測定する

	比例ゲイン k_P	積分時間 T_I	微分時間 T_D
P制御	$0.5k_o$		
PI制御	$0.45k_o$	$0.83T_o$	
PID制御	$0.6k_o$	$0.5T_o$	$0.125T_o$
改良版	$0.2k_o$	$0.5T_o$	$0.33T_o$

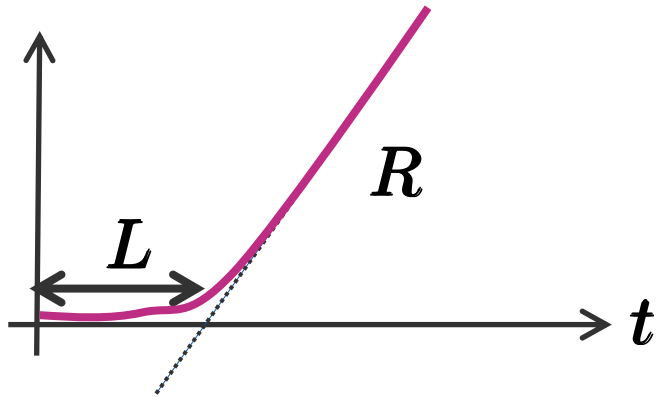
ステップ応答法

ステップ応答法

定位系 $\dfrac{Ke^{-Ls}}{1+Ts}$



無定位系 $\dfrac{R}{s}e^{-Ls}$



未知の制御対象を近似的に
１次遅れ＋むだ時間（定位系）
積分器＋むだ時間（無定位系）とみなし，
ステップ応答の結果からPIDゲインを決定する

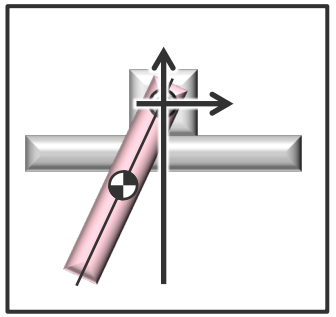
ステップ応答法

ステップ応答法

	比例ゲイン k_P	積分時間 T_I	微分時間 T_D
P制御	$\frac{1}{RL}$		
PI制御	$\frac{0.9}{RL}$	$3.3L$	
PID制御	$\frac{1.2}{RL}$	$2L$	$0.5L$

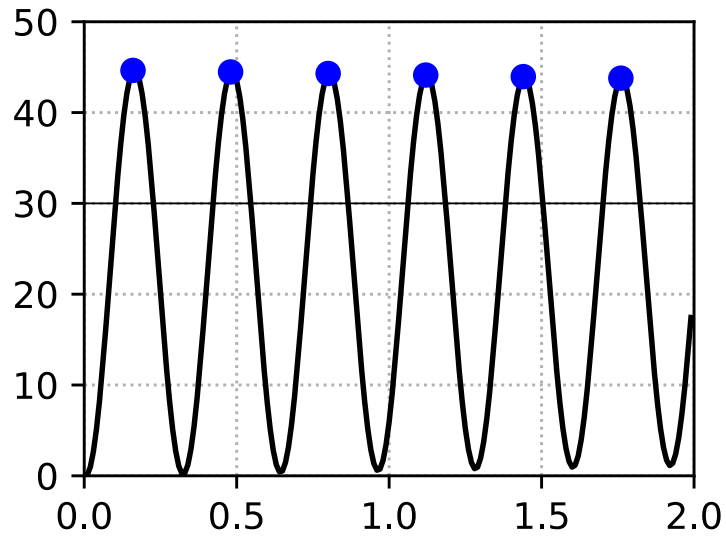
限界感度法の例題


```
from scipy import signal

fig, ax = plt.subplots(figsize=(3, 2.3))

kp0 = 2.9
C = tf([0, kp0], [0, 1])
Gyr = feedback(Pdelay*C, 1)
y, t = step(Gyr, np.arange(0, 2, 0.01))

[maxId] = signal.argrelmax(y)

ax.plot(t, y*ref, color='k')
ax.plot(t[maxId], y[maxId]*ref, 'bo')

ax.axhline(ref, color='k', linewidth=0.5)
ax.set_xlim(0, 2)
ax.set_ylim(0, 50)
ax.grid(ls=':')
```



```
g = 9.81      # 重力加速度[m/s^2]
l = 0.2       # アームの長さ[m]
m = 0.5       # アームの質量[kg]
c = 1.5e-2    # 粘性摩擦係数
J = 1.0e-2    # 慣性モーメント
```

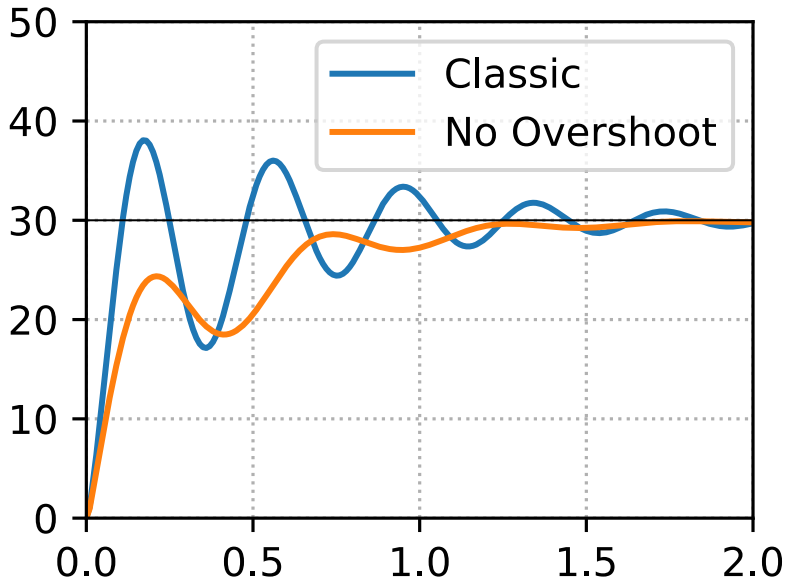
```
ref = 30
P = tf([0, 1], [J, c, m*g*l])
num_delay, den_delay = pade(0.005, 1)
Pdelay = P * tf(num_delay, den_delay)
```



限界感度法の例題


```
kp = [0, 0]
ki = [0, 0]
kd = [0, 0]
Rule = ['', '']

T0 = t[maxId[1]]-t[maxId[0]]
print('T0=', T0)
print('kP0=', kp0)

# Classic ZN
Rule[0] = 'Classic'
kp[0] = 0.6 * kp0
ki[0] = kp[0] / (0.5 * T0)
kd[0] = kp[0] * (0.125 * T0)

# No overshoot
Rule[1] = 'No Overshoot'
kp[1] = 0.2 * kp0
ki[1] = kp[1] / (0.5 * T0)
kd[1] = kp[1] * (0.33 * T0)
```

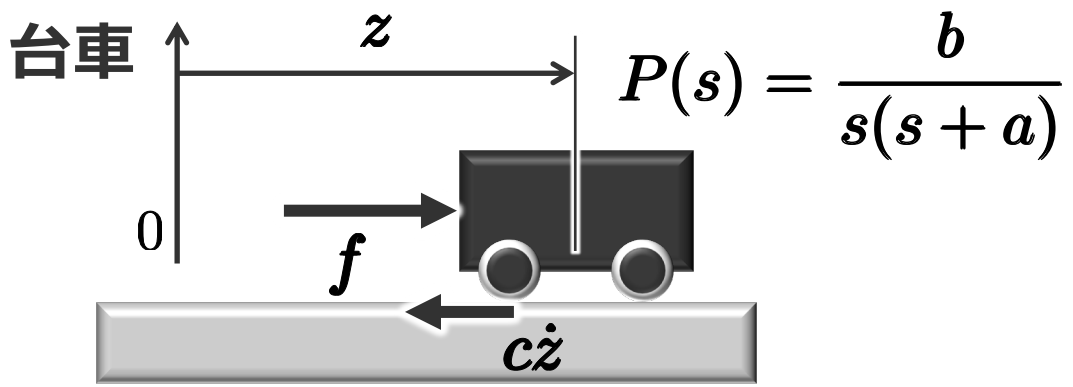
```
fig, ax = plt.subplots(figsize=(3, 2.3))

for i in range(2):
    C = tf([kd[i], kp[i], ki[i]], [1, 0])
    Gyr = feedback(Pdelay*C, 1)
    y, t = step(Gyr, np.arange(0, 2, 0.01))
    ax.plot(t, y*ref, label=Rule[i])

ax.axhline(ref, color="k", linewidth=0.5)
ax.legend()
ax.grid(ls=':')
```

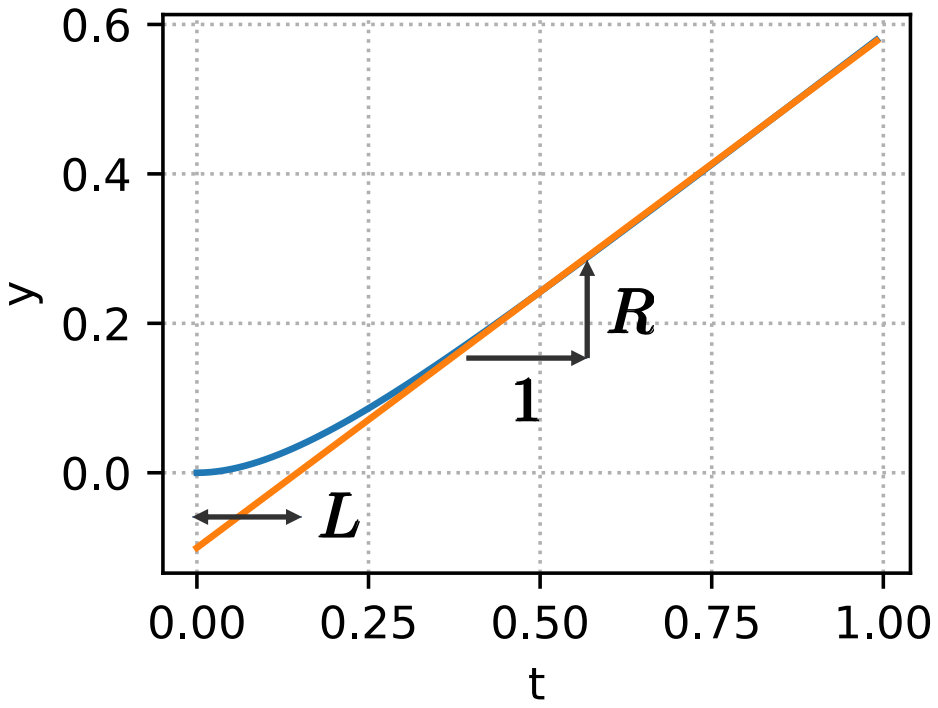


ステップ応答法の例題


```
# 台車系のモデル
a = 6.25
b = 4.36
P = tf([0, b], [1, a, 0])

y, t = step(P, np.arange(0, 1, 0.01))

# ある程度時間が経過したあとの応答を1次関数で近似する
yslice = y[40::]
tslice = t[40::]

p = np.polyfit(tslice, yslice, 1)
```

```
fig, ax = plt.subplots(figsize=(3, 2.3))
y, t = step(P, np.arange(0, 1, 0.01))
ax.plot(t, y)
ax.plot(t, np.poly1d(p)(t))
ax.set_xlabel('t')
ax.set_ylabel('y')
ax.grid(ls=':')
```



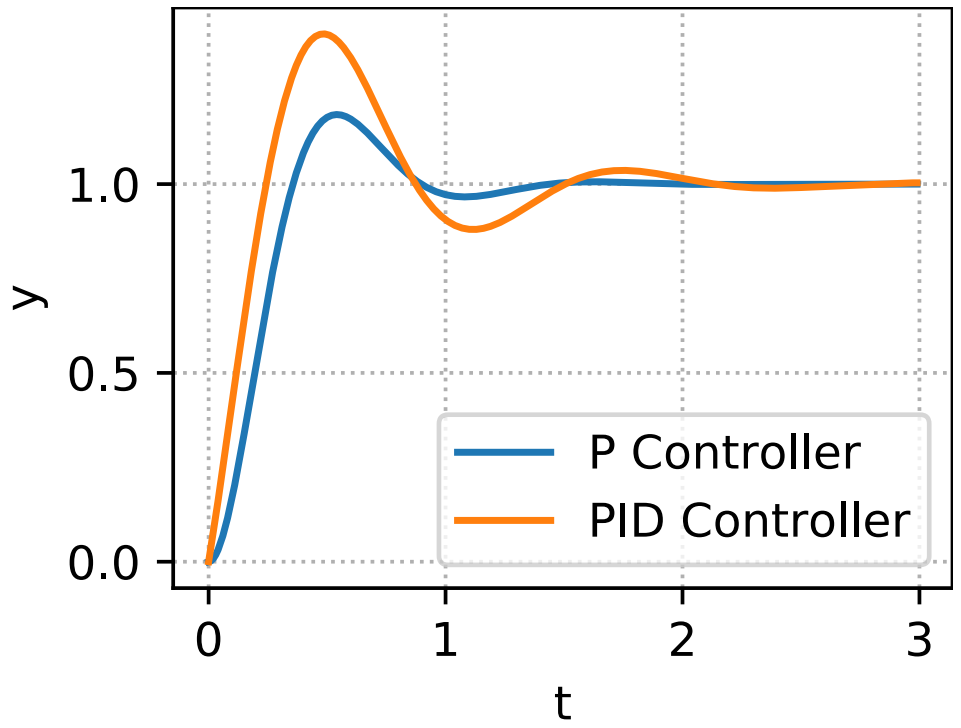


ステップ応答法の例題


```
uref = 1
R = p[0]/uref
L = -p[1]/p[0]
print(R, L)

kp = [0, 0]
ki = [0, 0]
kd = [0, 0]
Rule = ['', '']

# P Controller
Rule[0] = 'P Controller'
kp[0] = 1/(R*L)

# PID Controller
Rule[1] = 'PID Controller'
kp[1] = 1.2/(R*L)
ki[1] = kp[1] / (2 * L)
kd[1] = kp[1] * (0.5 * L)
```

$$y = p[0]t + p[1]$$

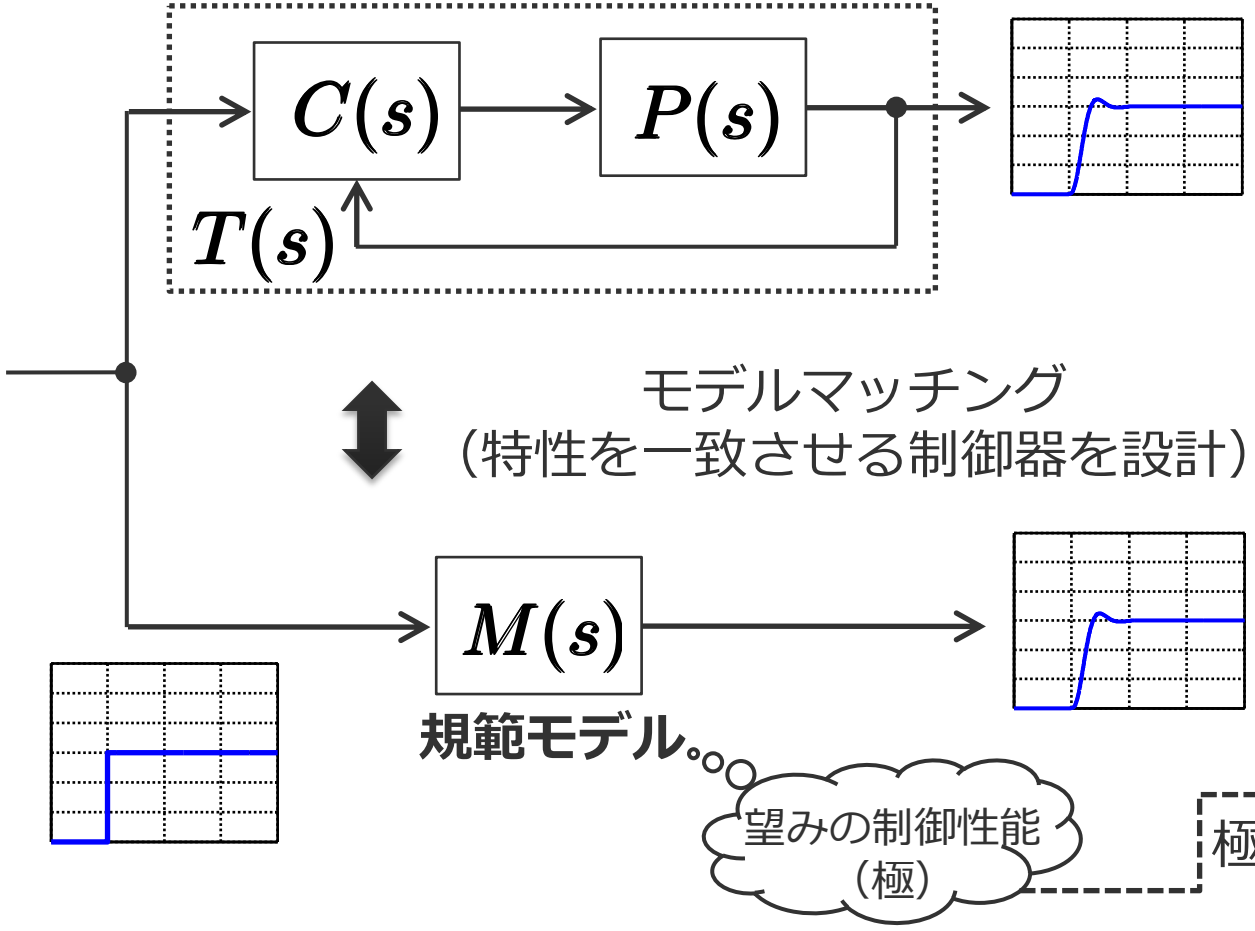
```
fig, ax = plt.subplots(figsize=(3, 2.3))
for i in range(2):
    C = tf([kd[i], kp[i], ki[i]], [1, 0])
    Gyr = feedback(P*C, 1)
    y, t = step(Gyr, np.arange(0, 3, 0.01))
    ax.plot(t, y, label=Rule[i])
```



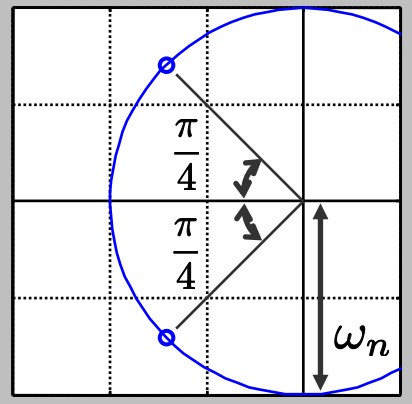
モデルマッチング

閉ループ系の特性を”規範モデル”の特性に近づける

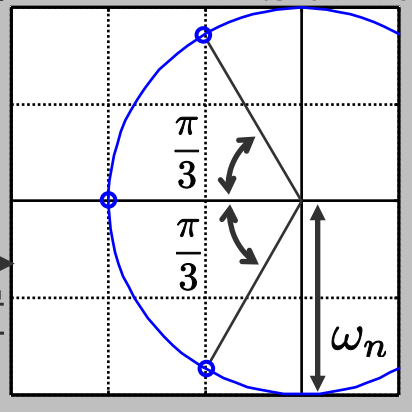
➡ $\frac{1}{T(s)}$ と $\frac{1}{M(s)}$ のマクローリン展開の低次項を一致させる



例）バターワース標準形（2次）



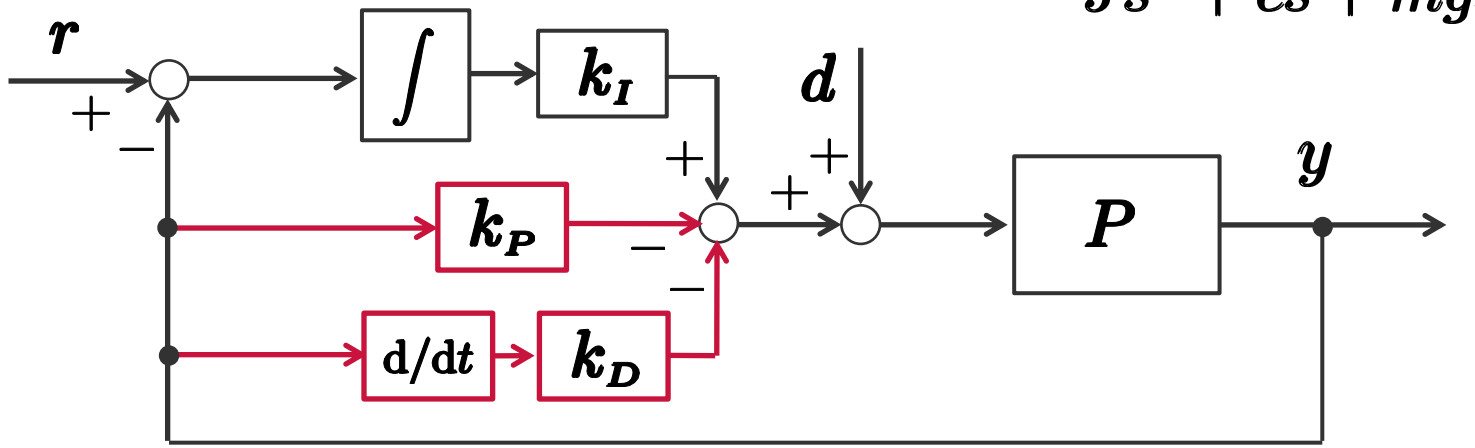
例）バターワース標準形（3次）



I-PD制御

モデルマッチングによる設計

$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$



$$T(s) = \frac{k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$$

規範モデル（３次）

$$M(s) = \frac{\omega_n^3}{s^3 + \alpha_2\omega_n s^2 + \alpha_1\omega_n^2 s + \omega_n^3}$$

２項係数標準形

$$(\alpha_1, \alpha_2) = (3, 3)$$

バターワース標準形

$$(\alpha_1, \alpha_2) = (2, 2)$$

I-PD制御

モデルマッチングによる設計

$$\frac{1}{M(s)} = 1 + \frac{\alpha_1}{\omega_n}s + \frac{\alpha_2}{\omega_n^2}s^2 + \frac{1}{\omega_n^3}s^3$$

$$\frac{1}{T(s)} = 1 + \frac{mg\ell + k_P}{k_I}s + \frac{c + k_D}{k_I}s^2 + \frac{J}{k_I}s^3$$

$$k_I = J\omega_n^3 \quad k_P = J\alpha_1\omega_n^2 - mg\ell$$

$$k_D = J\alpha_2\omega_n - c$$

ITAE最小標準形

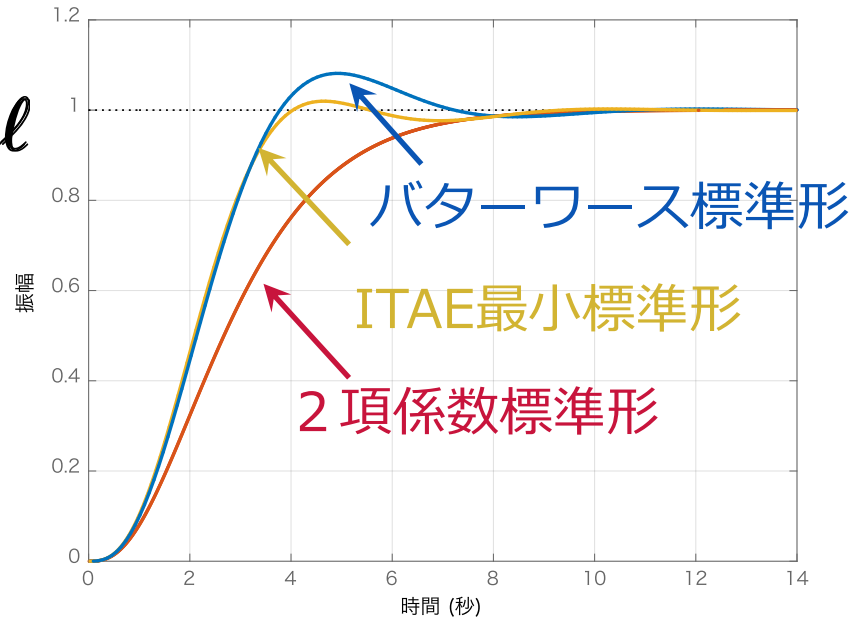
$$(\alpha_1, \alpha_2) = (2.15, 1.75)$$

２項係数標準形

$$(\alpha_1, \alpha_2) = (3, 3)$$

バターワース標準形

$$(\alpha_1, \alpha_2) = (2, 2)$$



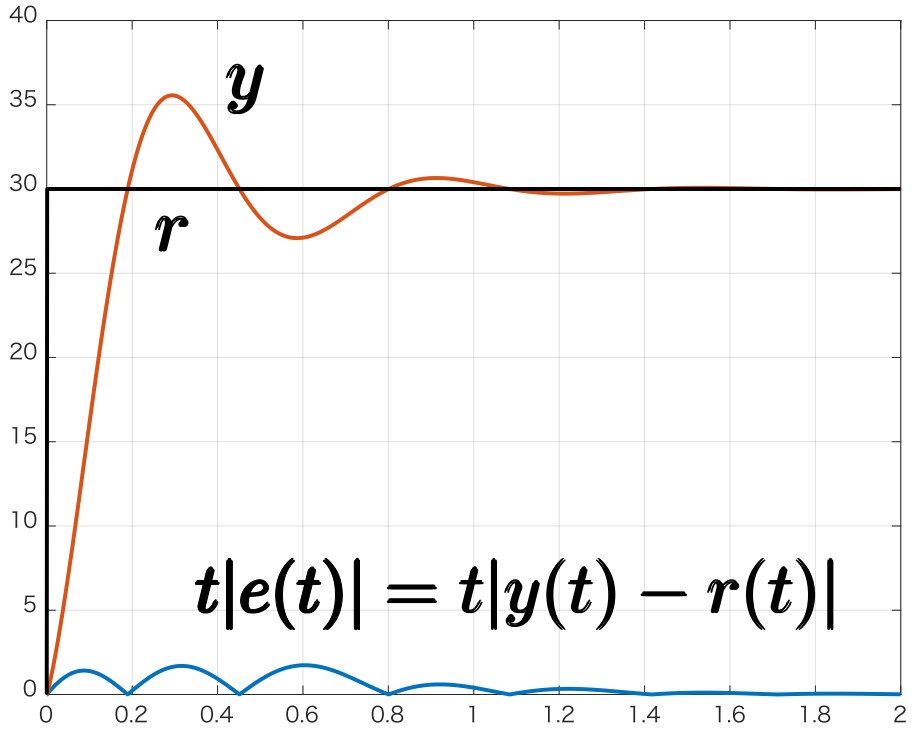
【補足】ITAE最小

$$M(s) = \frac{\omega_n^3}{s^3 + \alpha_2\omega_n s^2 + \alpha_1\omega_n^2 s + \omega_n^3} \qquad (\alpha_1, \alpha_2) = (2.15, 1.75)$$

ITAE: Integral of Time weighted Absolute Error

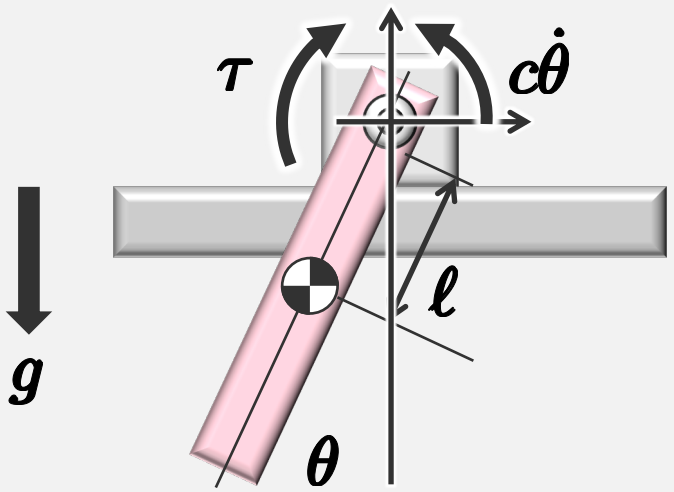
$$J = \int_0^\infty t|e(t)|\mathrm{d}t$$

十分時間が経過
したときの偏差も
しっかり考慮



制御系設計の例題

２次遅れ系（アーム，RCL回路）が対象



$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

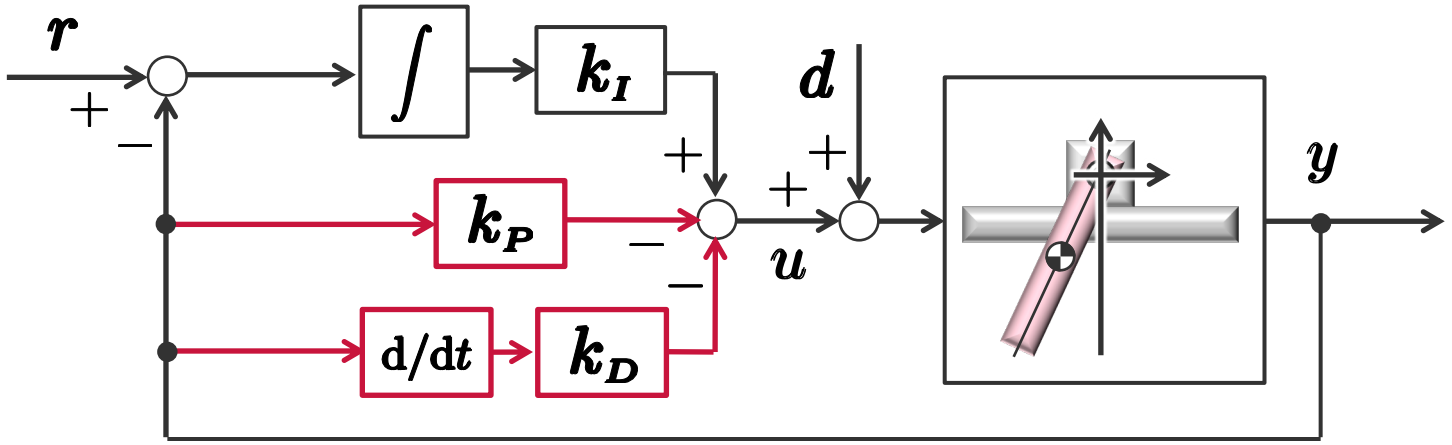
```
g = 9.81      # 重力加速度[m/s^2]
l = 0.2       # アームの長さ[m]
m = 0.5       # アームの質量[kg]
c = 1.5e-2    # 粘性摩擦係数
J = 1.0e-2    # 慣性モーメント
```

制御目標：ステップ目標値への追従

整定時間を0.5秒以下，オーバーシュートを10%以下にし，ステップ目標値＆外乱に対して，定常偏差が生じないPID制御系を構築しなさい

制御系設計の例題

制御器として，I-PD制御を考える（定常偏差は 0 になる）



閉ループ系 $G_{yr}(s) = \dfrac{k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$

を**３次の規範モデル**に一致させるゲインをもとめる

$$\frac{\omega_n^3}{s^3 + \alpha_2\omega_n s^2 + \alpha_1\omega_n^2 s + \omega_n^3}$$

$$k_P = J\alpha_1\omega_n^2 - mg\ell$$

$$k_I = J\omega_n^3 \quad k_D = J\alpha_2\omega_n - c$$

制御系設計の例題

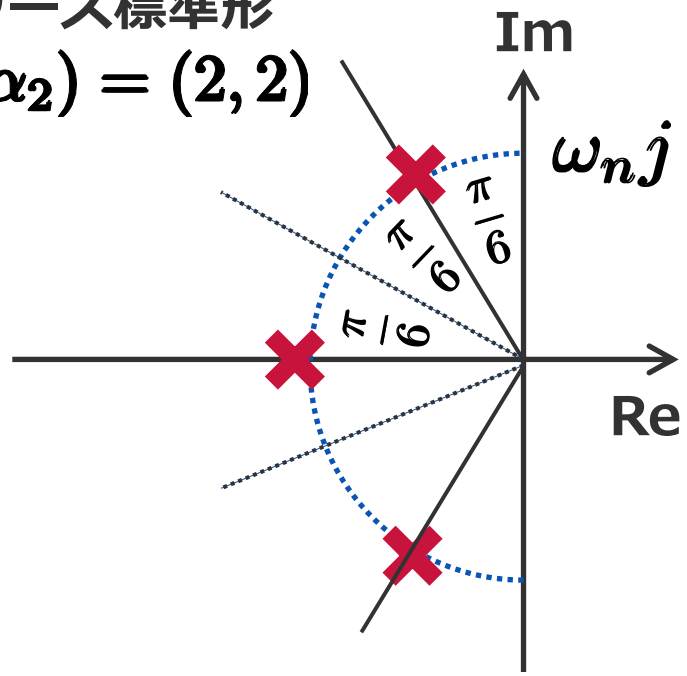
規範モデル$(\alpha_1, \alpha_2, \omega_n)$ を仕様をもとに決める

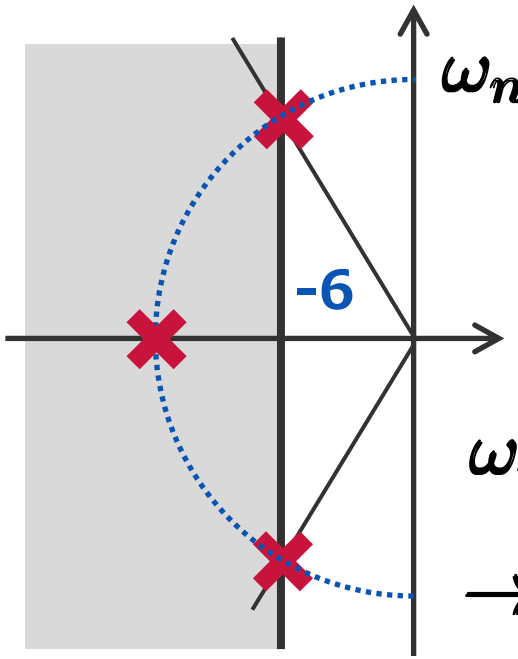
整定時間を0.5秒以下，オーバーシュートを10%以下にし，

ステップ目標値に対して，定常偏差が生じない→$|G_{yr}(0)| = 1$

バターワース標準形

$(\alpha_1, \alpha_2) = (2, 2)$





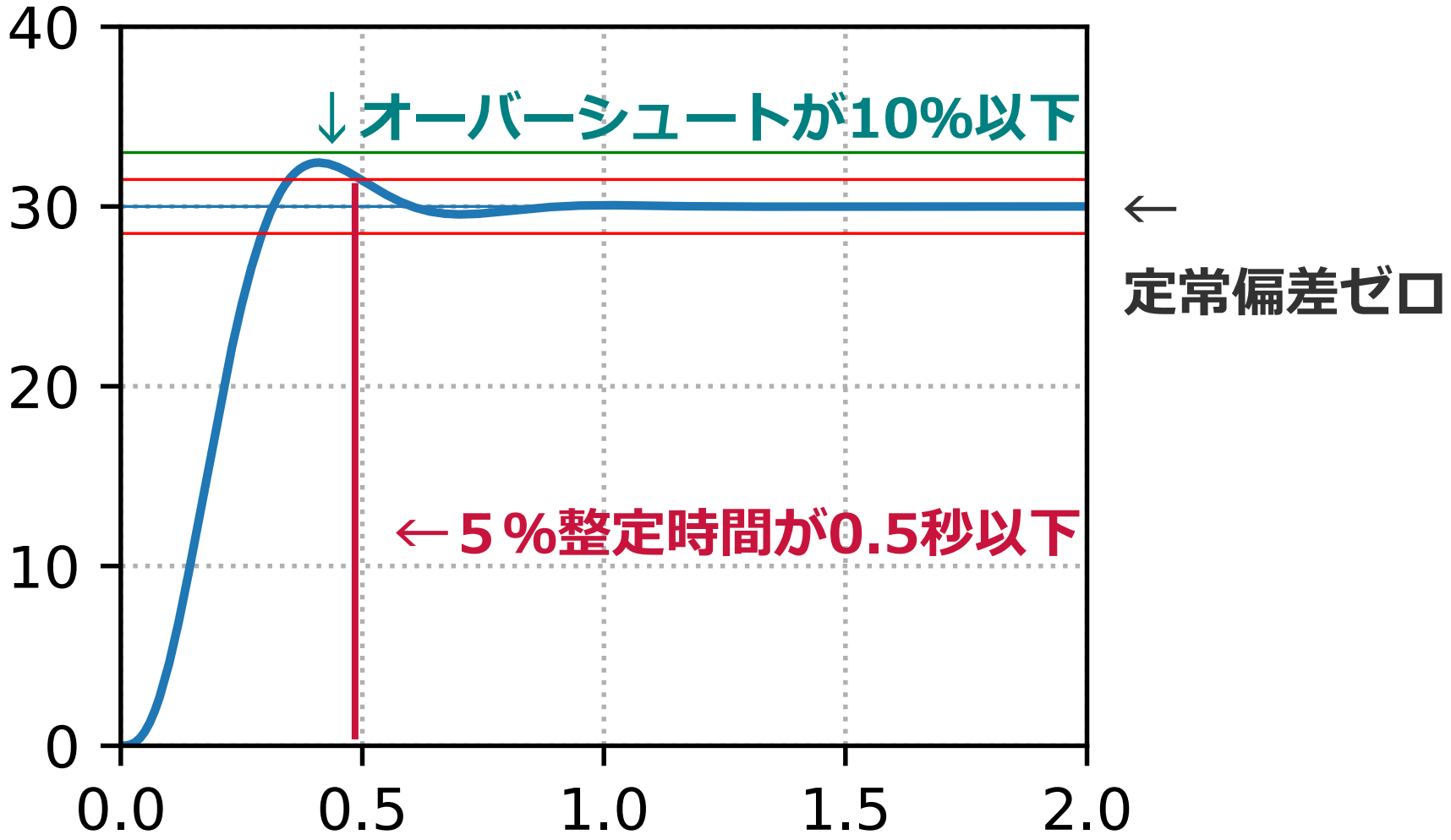
$$\omega_n \sin\frac{\pi}{6} = 6$$

$$\rightarrow \omega_n = 12$$

オーバーシュート8.2%　　実部が -3/Ts 以下で ↑ 整定時間が Ts 秒以下

制御系設計の例題

目標値を３０[deg]にしたときのステップ応答



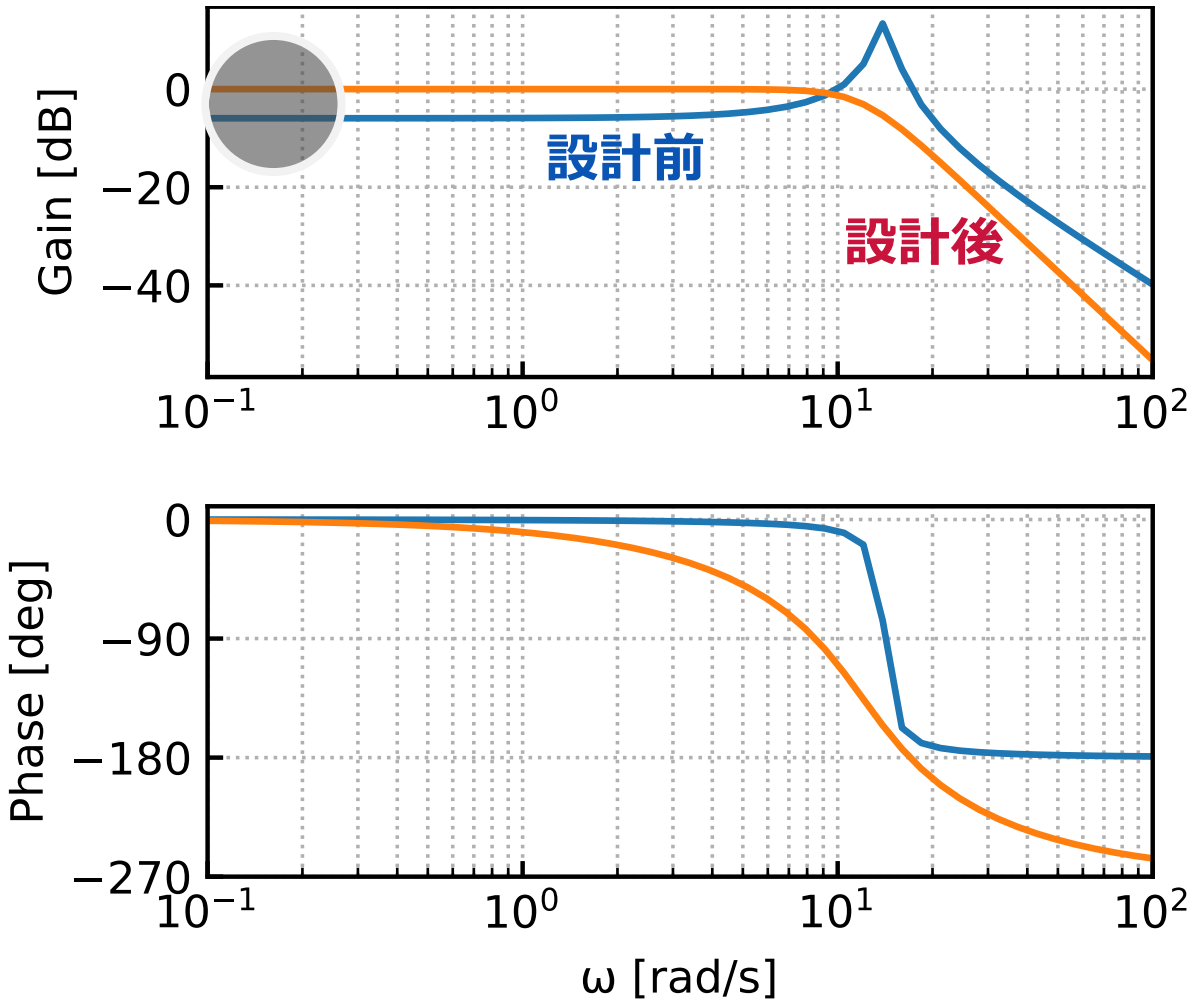
←

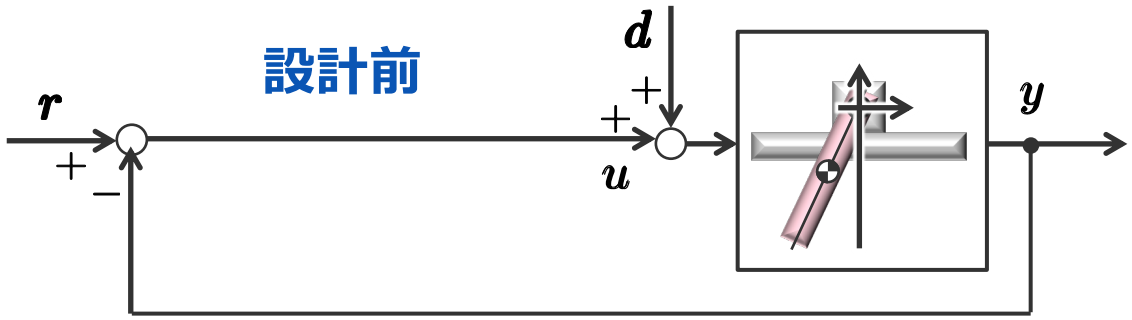
定常偏差ゼロ

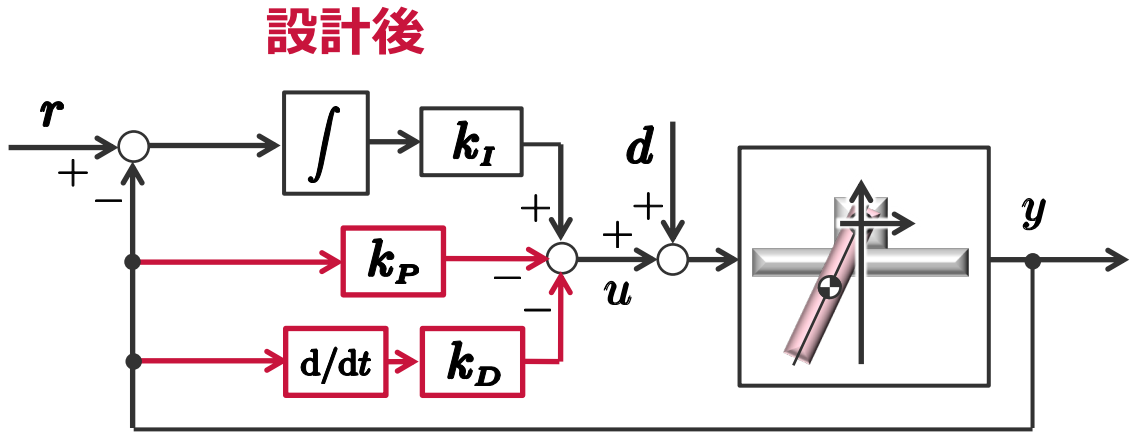
制御系設計の例題

閉ループ系の周波数特性

低周波ゲインが０dB　　ピークゲインが小さい



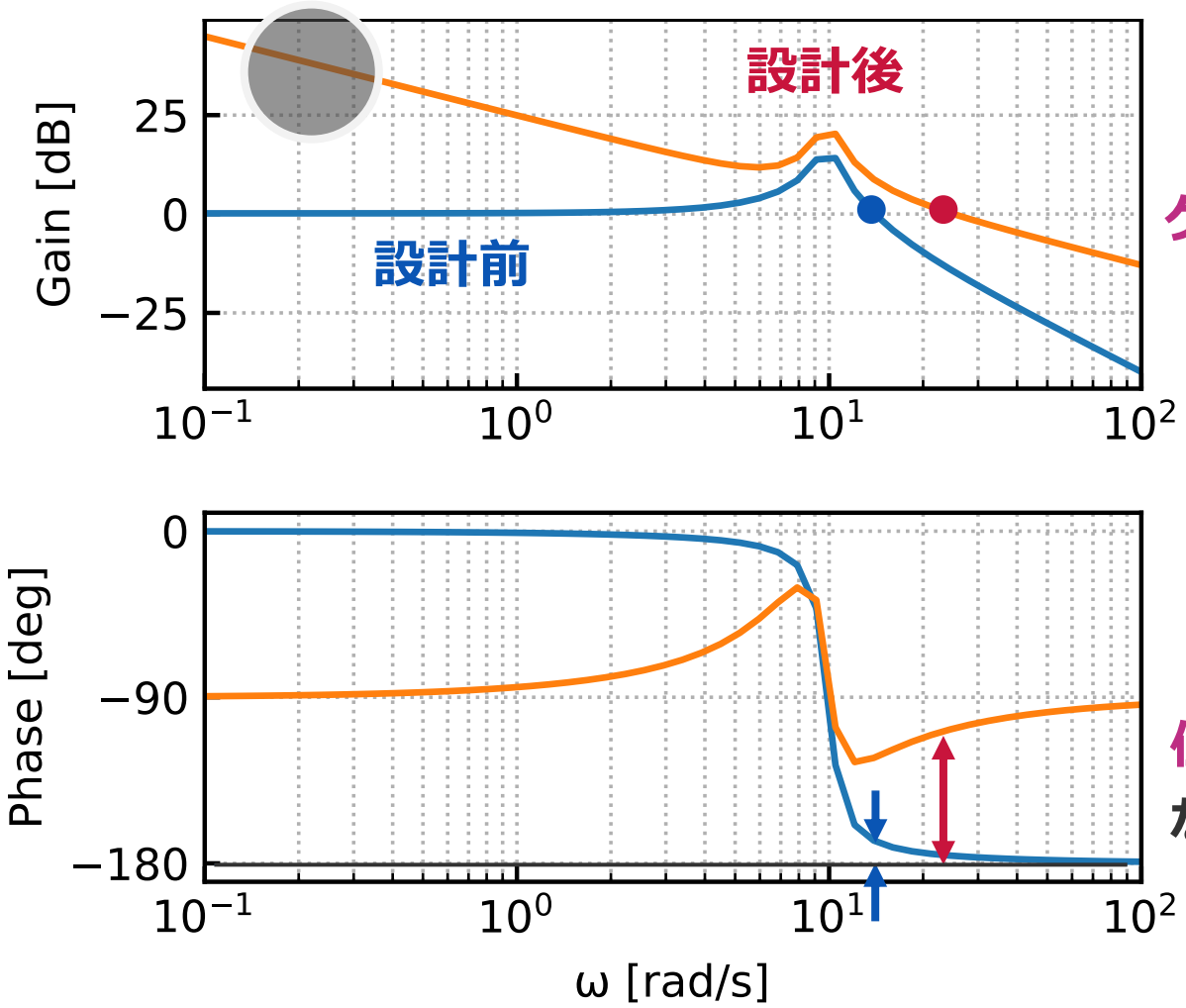




制御系設計の例題（次回予告）

開ループ系の周波数特性（ループ整形のところで登場）

↓-20dB/dec なので直流ゲインは∞になっている



ゲイン交差周波数が
大きくなっている

位相余裕が大きく
なっている

開ループ系の特性を見ながら
でも設計できそう！
→ ループ整形

HomeWork

★グラフを描いて考察しよう

$$P(s) = \frac{1}{Js^2 + cs + mg\ell}$$

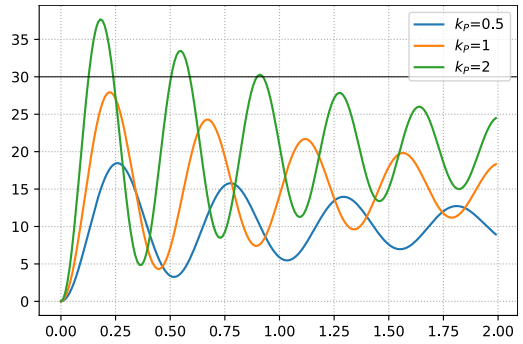
```
g = 9.81      # 重力加速度[m/s^2]
l = 0.2       # アームの長さ[m]
m = 0.5       # アームの質量[kg]
c = 1.5e-2    # 粘性摩擦係数
J = 1.0e-3    # 慣性モーメント
```

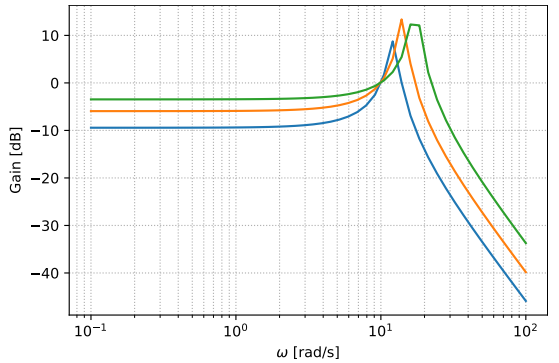
閉ループ系のステップ応答
閉ループ系のゲイン線図

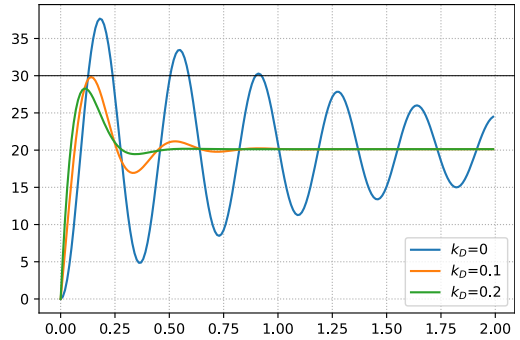
P制御（Pゲインを変化）

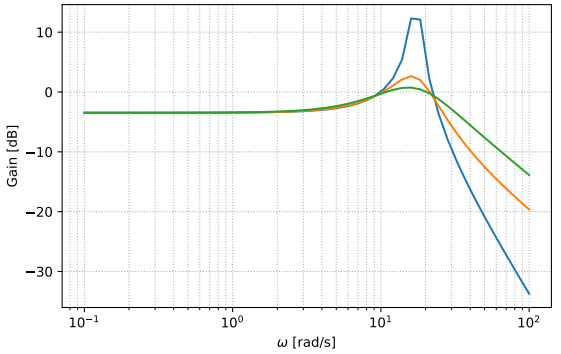
PD制御（Dゲインを変化）

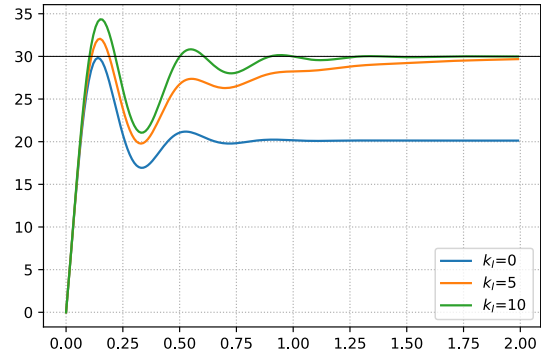
PID制御（Iゲインを変化）

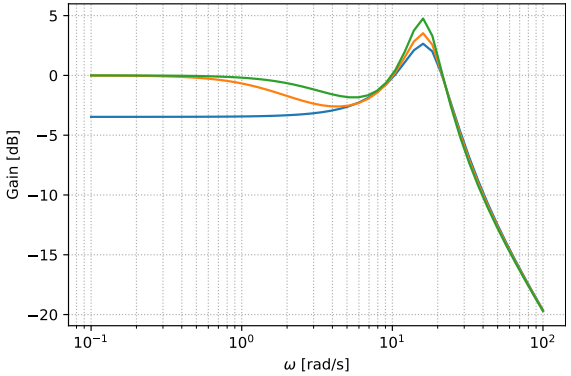






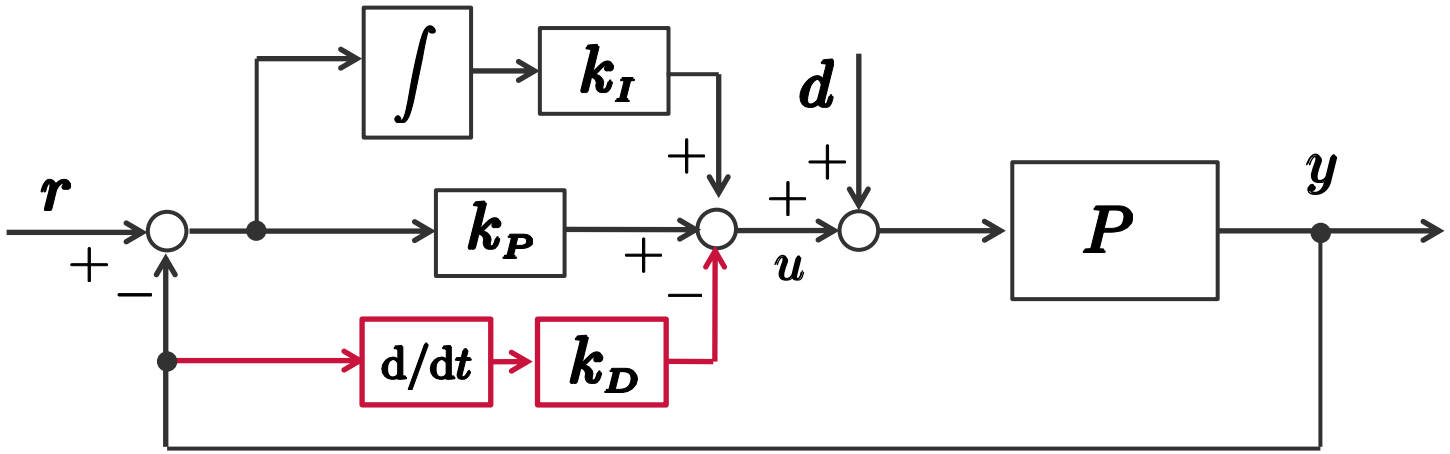






HomeWork

★モデルマッチングによる設計をしてみよう



$$T(s) = \frac{k_P s + k_I}{Js^3 + (c + k_D)s^2 + (mg\ell + k_P)s + k_I}$$

規範モデル（２次）

$$M(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

（１）k_P, k_D, k_I を
　　ζ, ω_n を用いて表せ（モデルマッチング）

（２）$\zeta = 0.707$, $\omega_n = 1$ のときの
　　M と Tのステップ応答を確認せよ

HomeWork

★モデルマッチングによる設計をしてみよう

$$\frac{1}{T(s)} = 1 + \frac{mg\ell}{k_I}s + \left(\frac{c+k_D}{k_I} - mg\ell\frac{k_P}{k_I^2}\right)s^2 + \left(\frac{J}{k_I} - \frac{k_P(c+k_D)}{k_I^2} + mg\ell\frac{k_P^2}{k_I^3}\right)s^3 + \cdots$$

$$\frac{1}{M(s)} = 1 + \frac{2\zeta}{\omega_n}s + \frac{1}{\omega_n^2}s^2$$

3次の項まで一致させるゲイン

$$k_P = \omega_n^2 J \qquad k_I = \frac{\omega_n mg\ell}{2\zeta}$$

$$k_D = 2\zeta\omega_n J + \frac{mg\ell}{2\zeta\omega_n} - c$$

```
import sympy as sp
z, wn = sp.symbols('zeta omega_n')
kp, kd, ki = sp.symbols('k_p k_d k_i')
mgl,c,J = sp.symbols('mgl c J')
sp.init_printing()

f1 = mgl/ki-2*z/wn
f2 = (c+kd)/ki-mgl*kp/(ki**2)-1/(wn**2)
f3 = J/ki-kp*(c+kd)/(ki**2)+mgl*kp**2/(ki**3)
sp.solve([f1, f2, f3],[kp, kd, ki])
```

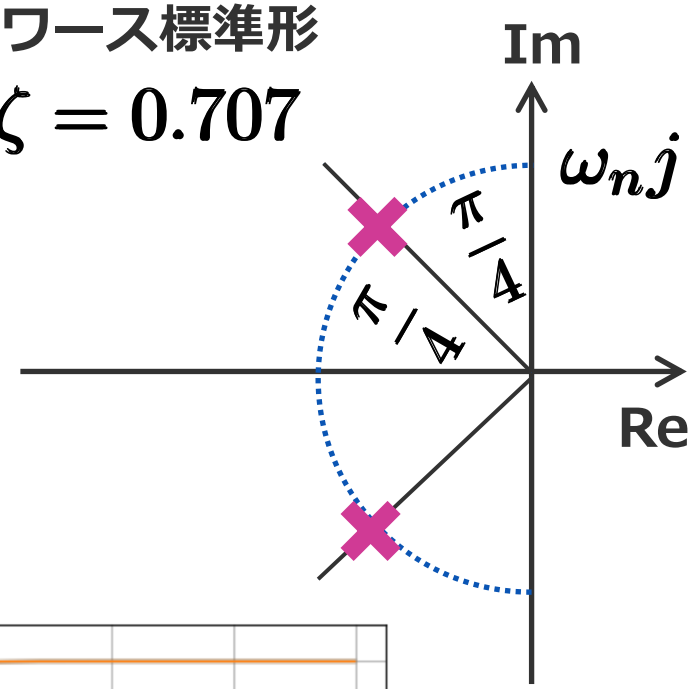
HomeWork

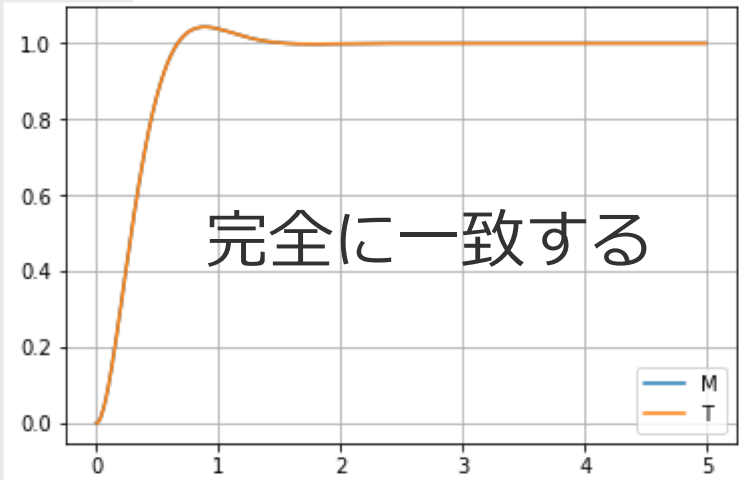
モデルマッチングによる設計

```
g = 9.81   # 重力加速度[m/s^2]
l = 0.2    # アームの長さ[m]
m = 0.5    # アームの質量[kg]
c = 1.5e-2 # 粘性摩擦係数
J = 1.0e-2 # 慣性モーメント

omega_n = 5
zeta = 0.707
M = tf([0,omega_n**2],[1,2*zeta*omega_n,omega_n**2])

kP = omega_n**2*J
kI = omega_n*m*g*l/(2*zeta)
kD = 2*zeta*omega_n*J + m*g*l/(2*zeta*omega_n) - c
T = tf([kP,kI],[J, c+kD, m*g*l+kP, kI])

yM, tM = step(M, np.arange(0, 5, 0.01))
yT, tT = step(T, np.arange(0, 5, 0.01))

fig, ax = plt.subplots(figsize=(3, 2.3))
ax.plot(tM, yM, label='M', ls = '-.')
ax.plot(tT, yT, label='T')
ax.legend()
ax.grid()
```

バターワース標準形

$\zeta = 0.707$





今日の目標

Level☆☆★

・閉ループ系の設計仕様の用語を説明できる
・閉ループ系の安定性をPythonでチェックできる
・PID制御系の応答特性(時間・周波数)のグラフをPythonで描ける

Level☆★★

・限界感度法，ステップ応答法でPIDゲインを設計できる
・モデルマッチング法でPIDゲインを設計できる

Level★★★

・２次遅れ系の行き過ぎ量やバンド幅，ピークゲインを手計算できる
・不完全微分が必要な理由を説明でき，それを用いた解析・設計ができる

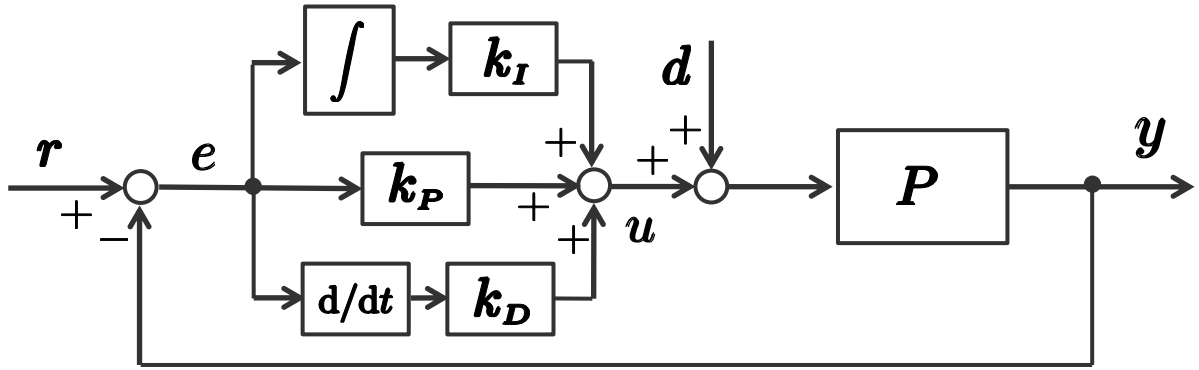
次回予告（6/23）

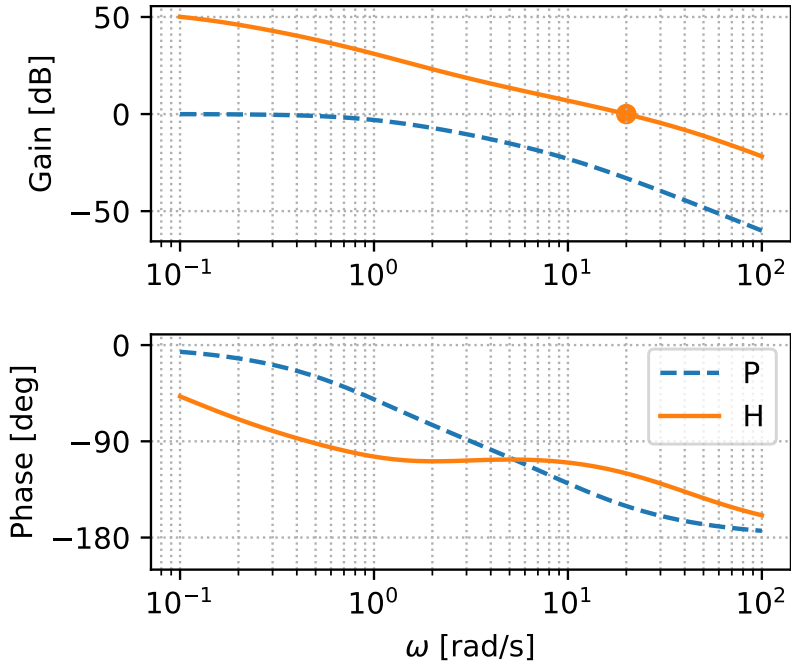
6月 23

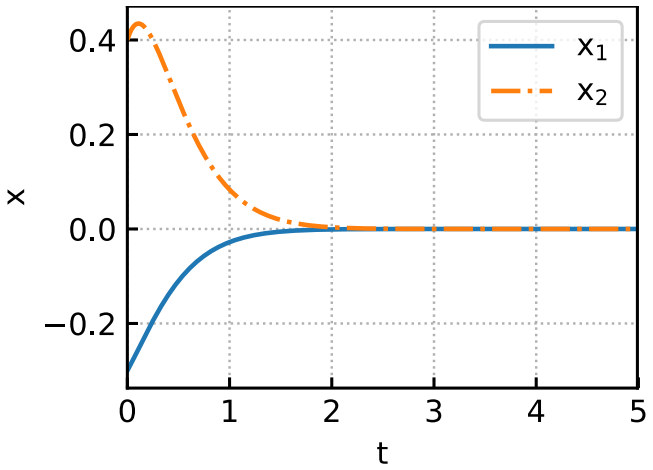
【オンライン勉強会】Pythonで学ぶ制御工学 Part3

YouTubeLive によるオンライン勉強会．初学者歓迎！

主催：SICE関西支部見学会委員会







各種仕様の推奨値

閉ループ系の仕様

	サーボ系	プロセス系
減衰係数	0.6～0.8	0.2～0.5
行き過ぎ量	0～25%	
ピークゲイン	1.1～1.5 (1.3が標準)	

開ループ系の仕様

	サーボ系	プロセス系
位相余裕	40°～60°	20°以上
ゲイン余裕	10～20dB	3～10dB