ディジタル制御 第7回

伝達関数に基づく
ディジタル制御系の設計 (2)

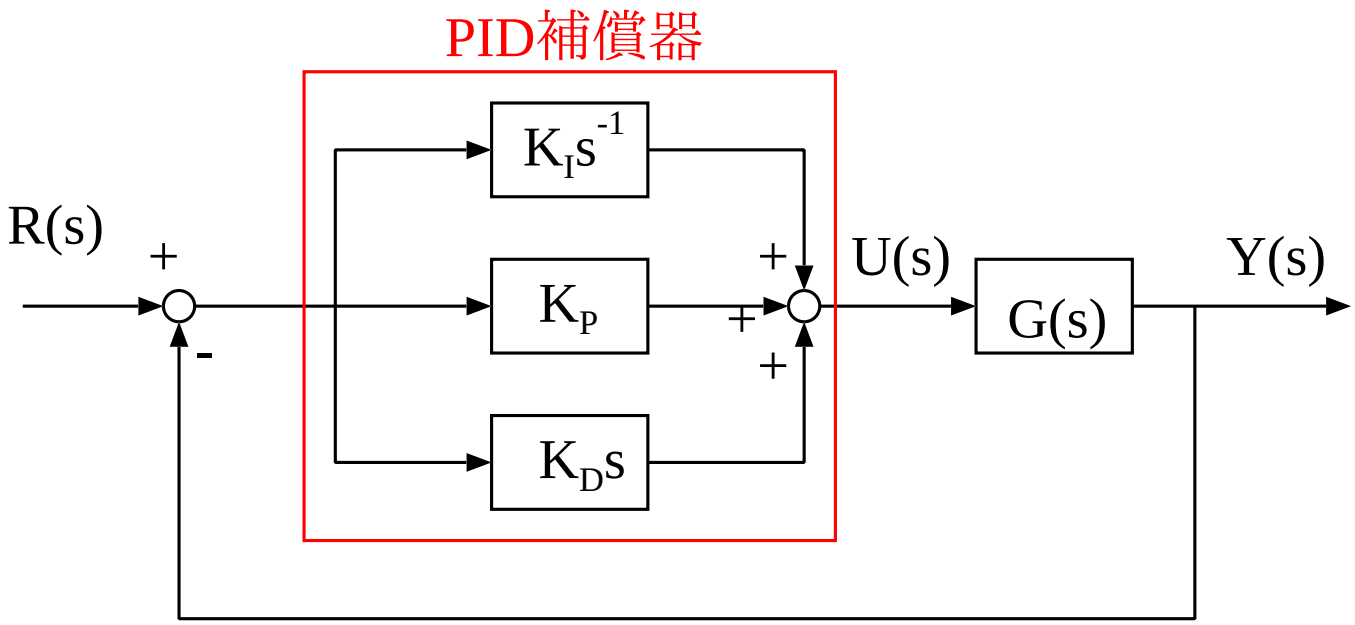
ディジタル PID 制御

- 制御系設計の方法は大別すると
  - ▷ 制御対象の数学モデルを用いるもの
  - ▷ 制御対象の数学モデルを用いないもの

  の 2 種類に分類されるが，これから述べる PID 制御は後者の代表格である．

- PID 制御にはアナログ PID 制御とディジタル PID 制御がある.
- 連続時間の PID 制御 (アナログ PID 制御) はアナログ方式の PID 補償器を用いた制御である.
- ディジタル PID 制御は, アナログ方式の PID 補償器を離散化したものである.

- PID 補償器は 比例 (Proportional), 積分 (Integral), 微分 (Differential) 要素を並列結合したもので, これらの頭文字を取って PID 補償器と呼ばれる. 典型的には, 制御対象の出力 $Y(s)$ を参照入力 $R(s)$ に追従させる目的で用いられる.


PID補償器
R(s)
+
-
$K_I s^{-1}$
K_P
$K_D s$
+
+
+
U(s)
G(s)
Y(s)

- アナログ PID 補償器は, 前ページの図のうち, 以下の伝達関数であらわされる部分である.

$$K_P + K_I \frac{1}{s} + K_D s$$

ただし, K_P, K_I および K_D はすべて正の定数である.

- K_P, K_I および K_D は, 調整可能なパラメータである. これらは, たとえば制御系が望ましいステップ応答を持つなどの目的を実現するために, オペレータによって調整される.

- PID 制御は, 古典制御の範疇に属する古い方法ではあるが (歴史上初の PID 補償器は 1936 年), プロセス制御の分野では今日でも広く使われている制御方式である.

- PID 制御は, 典型的には, 目標値としてステップ状の信号 $R(z)$ を想定し, 積分動作で $R(z)-Y(z)$ (追従誤差) を減らし, 比例動作と微分動作で速応性を改善するという考え方の補償をおこなう.

- 多くの場合, PID 制御の主目的はステップ状の目標値への応答特性の改善であり, 制御系がすでに安定化されていることは暗黙の前提になっている.

- プロセス制御の分野では, 多くの制御対象は, 積分動作によって制御対象の出力をステップ状の目標値に追従させることができるという性質を持つ (制御対象がそのように作られている) が…

- 厳密に言うと, 積分動作でステップ状の目標への追従が達成されるか否かは制御対象の特性に依存する.

- よって, 理論的な観点からは,
  - ▷ 安定性の確認
  - ▷ 最終値定理を用いたステップ状の目標への追従の確認

  が別途必要である.

- PID 制御には, 制御対象のステップ応答波形があれば, 制御対象の数学モデルを用いなくても補償器の設計 (パラメータ設定) ができるという利点がある.

- 制御系が一次遅れ系と遅延の組み合わせで大まかに近似でき, かつ数学モデルの構築が困難な場合には, PID 制御は有力な選択肢であり, それゆえ産業界の多くの分野で用いられている.

- 一方で, 制御系の安定性を理論的に保証するものではないので, 安定性に関する十分な注意が必要となる. また, 最適性を追及するものではないので, 制御性能を限界まで突き詰める必要がある状況には適さない.

- 歴史的にはアナログ PID 補償器がそのまま用いられていたが, 今日では, アナログ方式のプロセス制御装置はほとんど使われなくなり, ディジタル方式に置き換わっている.

- 一方で, パラメータ (K_P, K_I および K_D) の調整の時点では, ノウハウが蓄積された連続時間表現を用いることが多い.

- この講義では, まずアナログ PID 補償器の概要を述べてから, その離散化について述べる.

- PID 制御に関する記述の典拠は以下の通り.

[1] 広井, 宮田, シミュレ– ションで学ぶ自動制御技術入門, 第 2 版, CQ 出版, 2005.

[2] 山本, 加藤, PID 制御の基礎と応用, 朝倉書店, 2005.

[3] 中野, 松尾, ディジタル制御, 昭晃堂, 2001.

[4] K. J. Åström and B. Wittenmark, Computer-Controlled Systems, 3/e, Prentice-Hall, 1997.

[5] M. Sami Fadali and A. Visioli, Digital Control Engineering, 2/e, Academi Press, 2013.

アナログ PID 補償器

- アナログ PID 補償器は以下の形であった:

$$U(s) = \left(K_P + K_I \frac{1}{s} + K_D s \right) E(s)$$

$$E(s) = R(s) - Y(s)$$

- $E(s) = R(s) - Y(s)$ は, 目標信号 $R(s)$ と制御対象の出力信号 $Y(s)$ の偏差.
- PID 制御が適用される制御系では, 多くの場合は, 目標信号 $R(s)$ は, プロセスの運転条件を変えない限りは一定値である.

- プロセスの運転条件を変更すると, 目標値が変動する.
- 目標値が変更された場合に, なるべく迅速かつ振動的でない形で新しい目標値に制御対象の出力を追従させるということが, PID 制御の目的である.

- PID 制御の各項の意味について説明する.

- 比例動作:　比例動作

$$K_P(R(s) - Y(s))$$

の考え方は次ページの通り:

- ▷ $r(t) > y(t)$ (出力過少) の場合には正の入力を制御対象に印加
- ▷ $r(t) < y(t)$ (出力過大) の場合には負の入力を制御対象に印加
- ▷ 印加する入力の振幅はずれの絶対値に比例

- 比例動作は直感的にはわかりやすいが, プロセス制御の多くの分野では制御対象の主要な部分は安定な一次遅れ系の形になっており, この形の制御対象に比例制御を適用しても, 追従誤差を零にすることはできない.

- 制御対象が 1 次遅れ系

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

の場合にこれを確認する (ただし $\alpha > 0$ で, この対象は安定であるものとする).

- 目標信号は単位ステップとする. よって,

$$R(s) = \frac{1}{s}.$$

- $Y(s) = \dfrac{1}{s+\alpha} U(s)$, $U(s) = K_p(R(s) - Y(s))$
  だから, これを整理すると,

$$(s + \alpha + K_P)Y(s) = K_P R(s).$$

よって, $R(s) = 1/s$ を代入すると,

$$Y(s) = \frac{K_P}{s + \alpha + K_P} \frac{1}{s}.$$

- 伝達関数

$$\frac{K_P}{s + \alpha + K_P}$$

は安定だから, これに単位ステップを印加したものは有限の値に収束する. よって, ラプラス変換の最終値定理を用いて, この値を計算することができる.

- ラプラス変換の最終値定理は, 次のようなものだった:

$$\lim_{t\to\infty} y(t) = \lim_{s\to 0} sY(s)$$

- ラプラス変換の最終値定理を使うと,

$$\lim_{t\to\infty} y(t) = \lim_{s\to 0} sY(s)$$

$$= \lim_{s\to 0} \frac{K_P}{s+\alpha+K_P} = \frac{K_p}{\alpha+K_P}$$

- 目標値は単位ステップだったから, $y(t)$ が 1 に収束することが望ましかったのだが, $y(t)$ は 1 より小さい値

$$\frac{K_p}{\alpha + K_P}$$

に収束し,

$$\frac{\alpha}{\alpha + K_P}$$

の偏差 (定常偏差と呼ぶ) が残る.

- 次に述べる積分動作によって定常偏差を零にできるのだが, 積分動作には目標値への追従が遅いという欠点もある. 比例制御の役割は, この欠点の解消, すなわち速応性の改善であると解釈されている.

- **積分動作:** 積分動作

$$\frac{K_I}{s}(R(s) - Y(s))$$

は, 偏差を積分してからフィードバックする動作である.

- 比例動作で定常偏差が残る場合に, 補償器に積分動作が含まれていると, 時間とともに定常偏差の影響が蓄積され, それが制御対象にフィードバックされるから, 最終的に偏差が零になるであろうというのが積分動作の発想.

- 制御対象が 1 次遅れ系

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

の場合にこれを確認する (ただし $\alpha > 0$ で, この対象は安定であるものとする).

- 再び, 目標信号は単位ステップとする. よって,

$$R(s) = \frac{1}{s}.$$

- $Y(s) = \dfrac{1}{s+\alpha}U(s)$, $U(s) = \left(K_p + K_I\dfrac{1}{s}\right)$ だから,

$$(s+\alpha)Y(s) = (K_P + K_I\frac{1}{s})(R(s) - Y(s)),$$

となる.

前ページの式の両辺に s を乗じれば

$$(s^2 + \alpha s)Y(s) = (K_P s + K_I)(R(s) - Y(s)).$$

となり, これを整理して次式を得る.

$$(s^2+(\alpha+K_P)s+K_I)Y(s) = (K_P s+K_I)R(s).$$

- K_P と K_I を, 多項式 $s^2 + (\alpha + K_P)s + K_I$ が安定となるように選ばなければならないことに注意する. このようにしないと, 制御系は不安定となり, ラプラス変換の最終値定理は適用できない.

- 以下では, 上記の条件が満たさるよう K_P と K_I が選択されているものとする. このとき, $R(s) = 1/s$ だったから,

$$Y(s) = \frac{K_P s + K_I}{s^2 + (\alpha + K_P)s + K_I} \frac{1}{s}$$

である.

- よって,

$$\lim_{t\to\infty} y(t) = \lim_{s\to 0} sY(s)$$
$$= \lim_{s\to 0} \frac{K_P s + K_I}{s^2 + (\alpha + K_P)s + K_I} = 1.$$

以上により, 確かに定常偏差は零になっている.

- 制御系が安定になっていることを別途確認する必要があったことに注意. また, 制御対象が 1 次遅れ系でない場合には, 積分動作によって定常偏差が零となることは保証されない.

- **微分動作:** 微分動作

$$K_D s(R(s) - Y(s))$$

は, 偏差を微分してからフィードバックする動作である.

- 比例動作の役割が目標値の変動への追従への応答測度の改善であったことを踏まえ, 微分動作は, この応答測度をさらに速くする目的で用いられる.

- **アナログ PID 補償器の別の表現**　PID 補償器の伝達関数は以下の形であったが:

$$K_P + K_I \frac{1}{s} + K_D s$$

これを (数学的には同じことなのだが) 次の形で書き直すことも多い.

$$K_P \left(1 + \frac{1}{T_I s} + T_D s \right)$$

- 前ページ後半の表現は, $K_I = \frac{K_P}{T_I}$, $K_D = K_P T_D$ という記号の置き換えをおこなっているだけなのだが, この表現を使い, さらに T_I を積分時間, T_D を微分時間と呼ぶことがある.

- **不完全微分**　比例動作, 積分動作, 微分動作のうち, 微分動作は外乱の影響を受けやすいため, この部分をこのままの形で用いることはあまりない. かわりに, 以下の形の項を用いる.

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

これを不完全微分と呼ぶ.

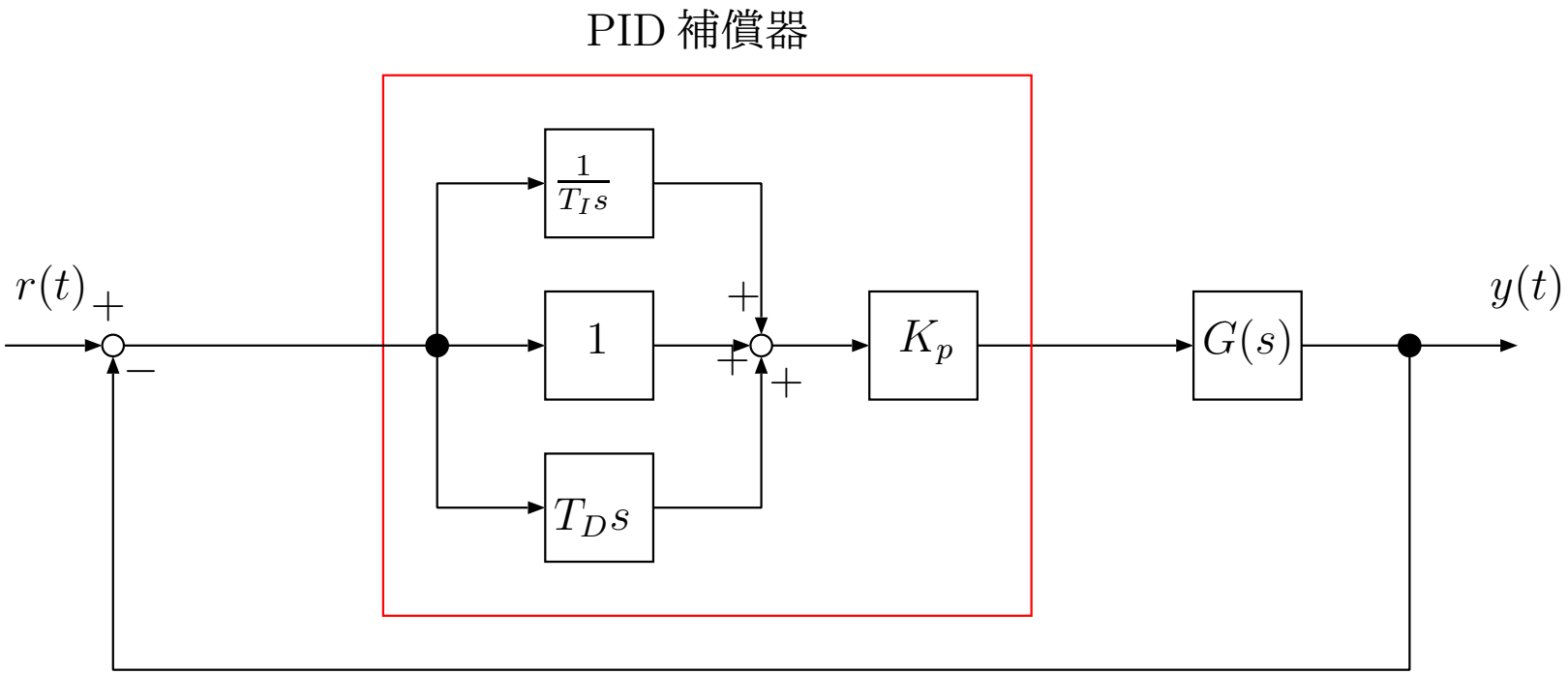
- 不完全微分は微分を低域通過フィルタに通した形となっており, 低周波域での動作が微分に近い一方で, 高周波域での過剰な反応は抑制される. この理由から, PID 制御における微分動作としては, この形が採用されることが多い. 微分をこのような不完全微分で置き換えた PID 制御を実用非干渉型 PID 制御と呼ぶことがある.

- これとは別に, PID 補償器の伝達関数全体の前に低域通過フィルタを配置したもの:

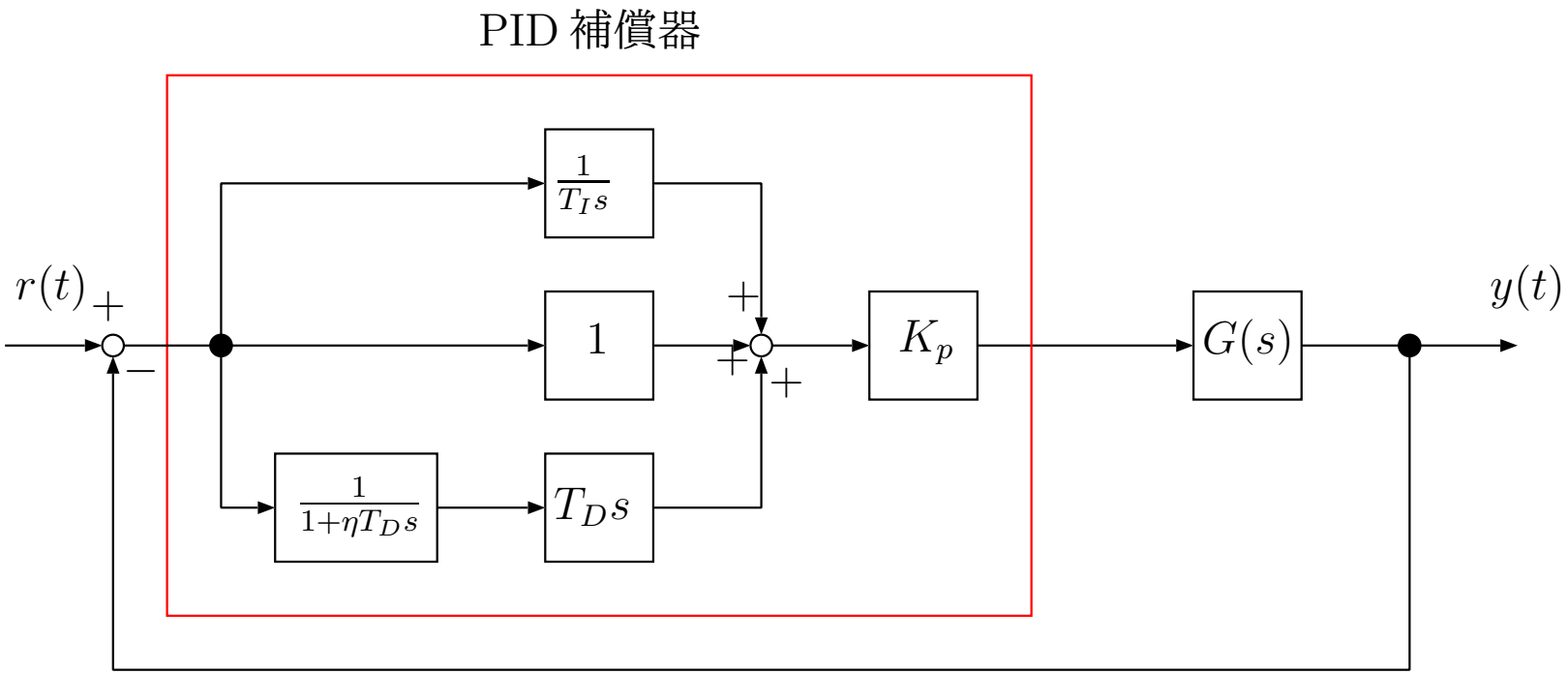
$$\frac{K_P}{1+\eta T_D s}\left(1+\frac{1}{T_I s}+T_D s\right)$$

が用いられることもある. これを, 実用干渉型 PID 制御と呼ぶことがある.

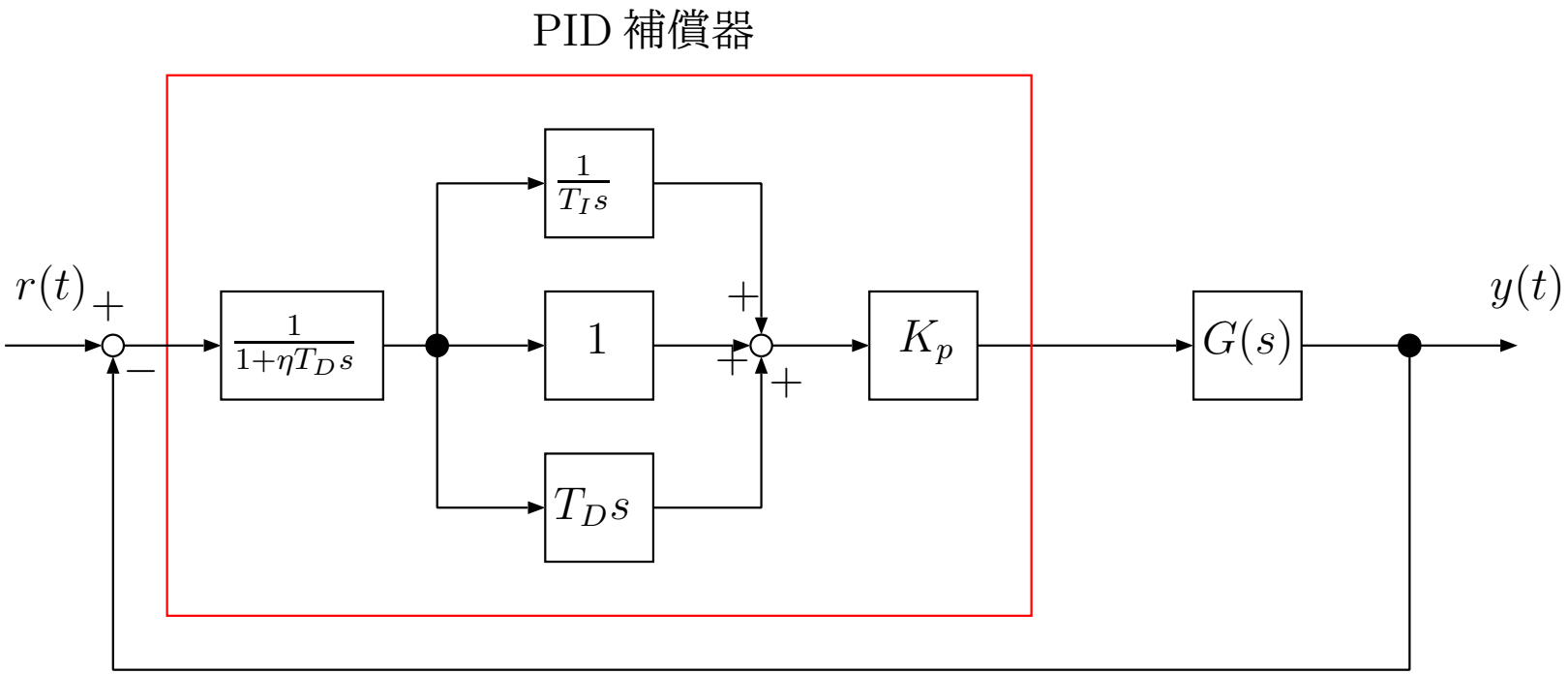
PID 補償器 (基本形):



PID 補償器 (実用非干渉型):



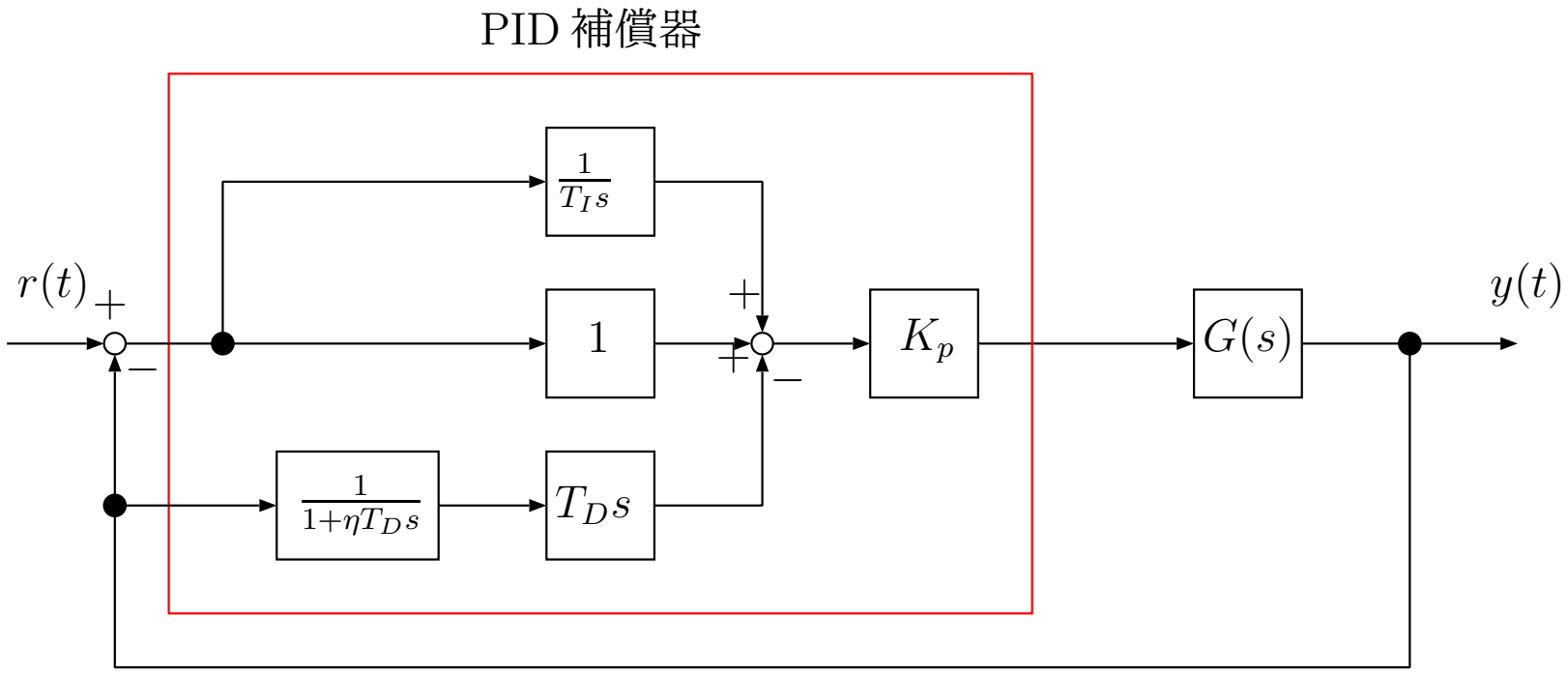
PID 補償器 (実用干渉型):



- 更に別の問題であるが, 目標値はステップ状に変化することが想定されているため, 目標値を微分器に通すと大きな制御信号が発生し, 実用上好ましくないことがある.

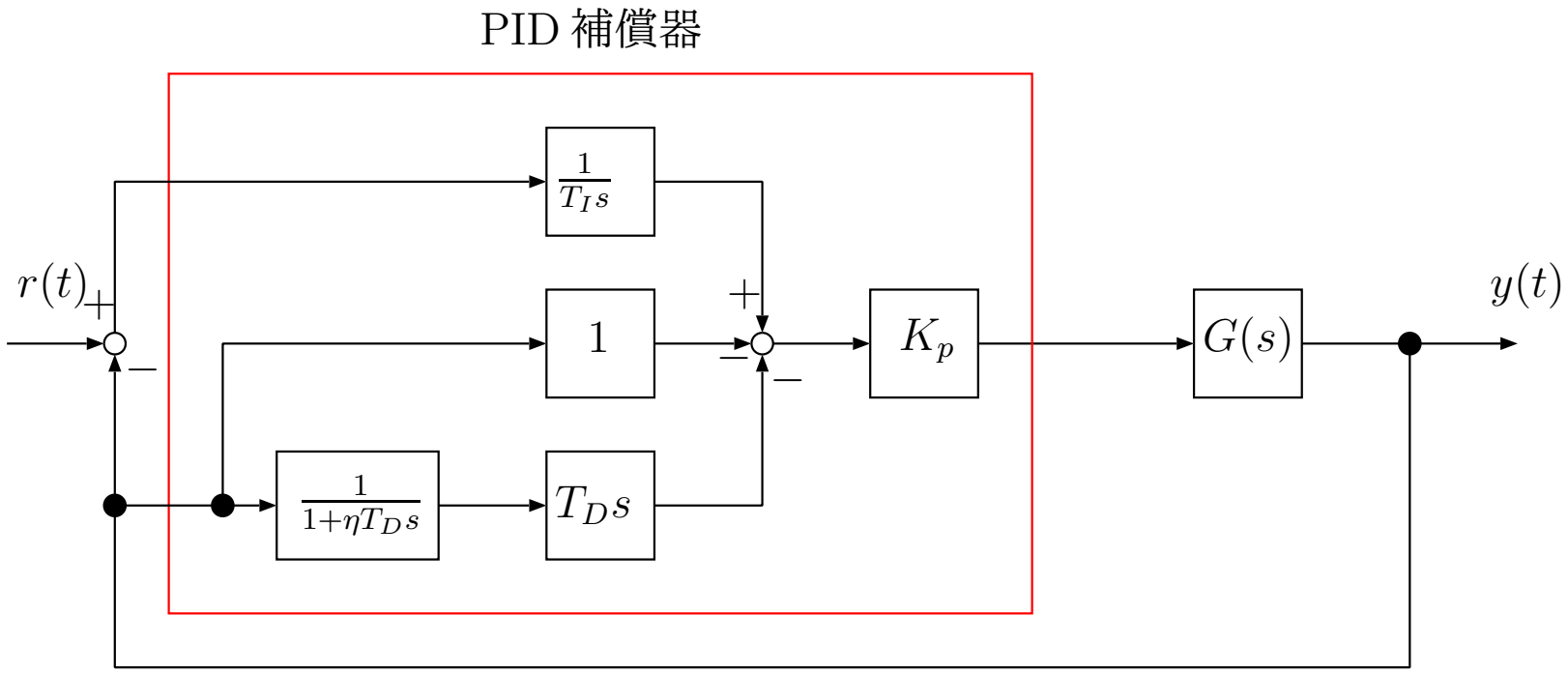
- この問題を解消するため, 次ページの図のように, $r(t)$ を微分器に通さず, $y(t)$ だけを微分器に通す構成が使われることがある (可算ブロックの部分の符号に注意).
- これを, 測定値微分先行型 PID 制御, あるいは PI-D 制御と呼ぶことがある.

測定値微分先行型 PID 補償器:



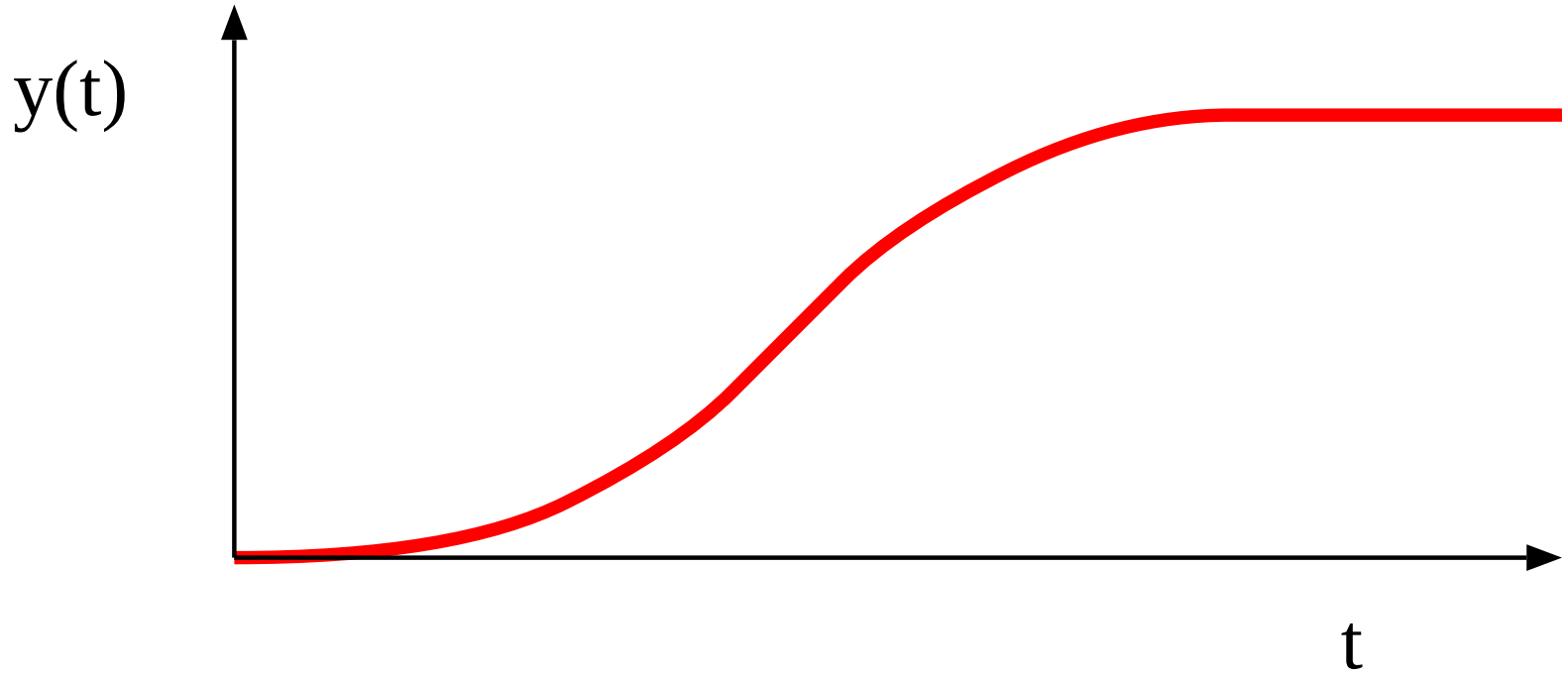
- 同じ考えを比例項にも適用し, 目標信号が比例器および微分器の双方を通過しない, 次ページのような構成が用いられることもある.
- これを, 測定値比例微分先行型 PID 制御, あるいは I-PD 制御と呼ぶことがある.

測定値比例微分先行型 PID 補償器:



- **PID パラメータの調整:**　PID 制御は, 多くの場合, 制御対象の応答が 1 次遅れとむだ時間 (遅延) の組み合わせであることを想定している. そして, このような場合に, ステップ応答の波形からいくつかのパラメータを決定し, それを使って PID 補償器の各数値を決める経験的な方法が知られている. これを紹介する.

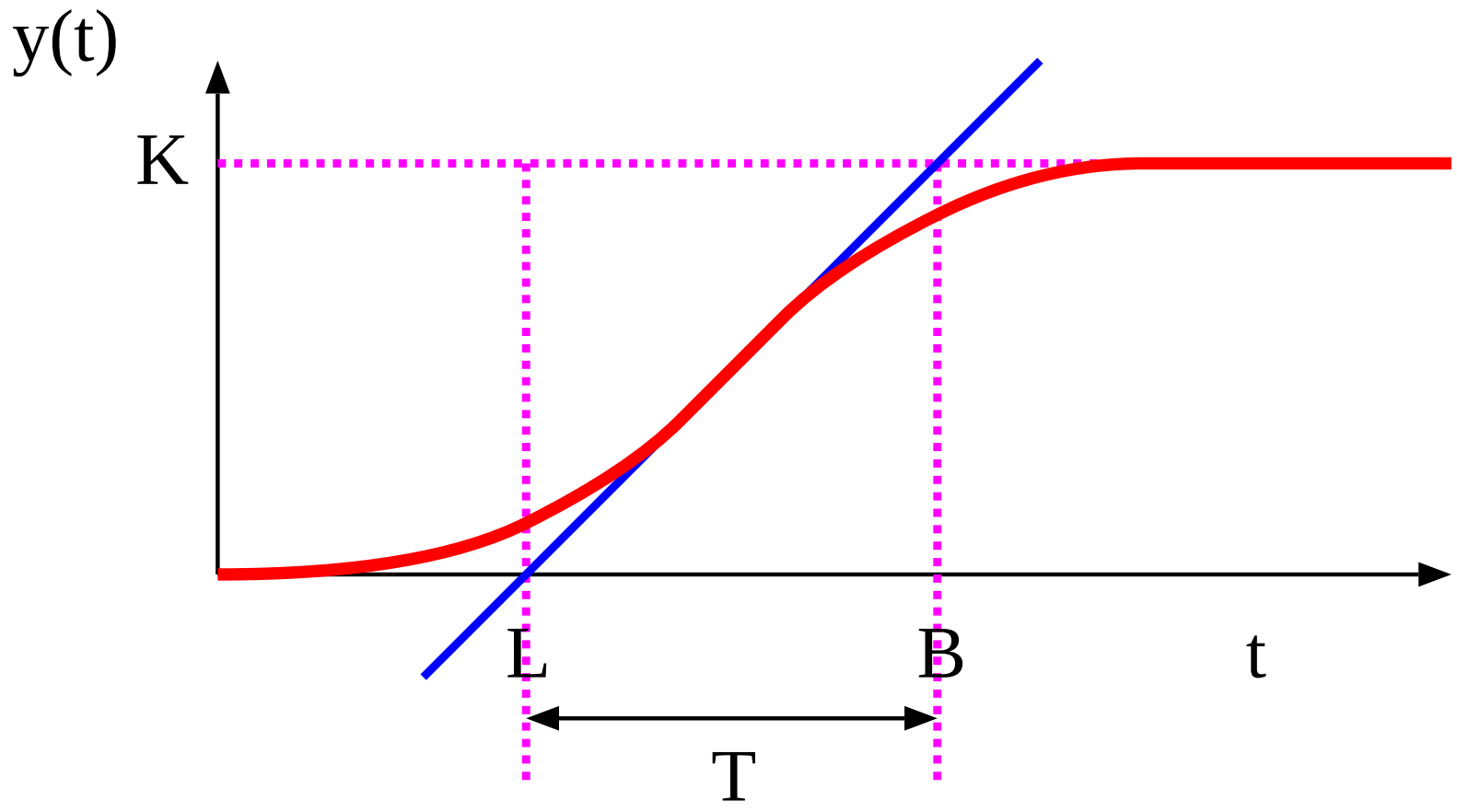
- 制御対象に単位ステップを印加したときの応答波形が得られているものとする．この応答波形は, 1 次遅れとむだ時間 (遅延) が組み合わされた, 図のようにおおむね S 字型で, 時間とともに一定値に収束する波形となっているものとする．



想定しているステップ応答の波形

- 制御対象はおおむね遅延と 1 次遅れ系の組み合わせで, ステップ応答がおおむね S 字型で時間とともに一定値に収束するという想定のもとで, 遅延の量と, 1 次遅れ系のパラメータを, 図から読み取りたい.
- 説明の便宜上, 横軸を x 軸, 縦軸を y 軸と呼ぶ.

- ステップ応答が一定に収束した値を K とする.
- ステップ応答の変曲点を見付け, 変曲点でステップ応答に接線を引く. 接線が x 軸と交わる点を L とする. これを遅延と見做す. また, 接線が直線 $y = K$ と交わる点の x 座標を B とし, $T = B - L$ とおく. この T を制御対象の時定数と見做す.



グラフからの K, L, T の読み取り

- 読み取ったパラメータから, P 動作のみ, PI 動作, PID 動作のそれぞれの場合について, 以下の表にしたがって K_P, T_I および T_D を決める. これは, Ziegler-Nochols の方法と呼ばれるものの 1 種で, ステップ応答法と呼ばれる. (典拠は [広井, 宮田]).

ステップ応答法によるパラメータの決定:

動作モード	K_P	T_I	T_D
P	$\frac{T}{KL}$	$-$	$-$
PI	$0.9\frac{T}{KL}$	$3.3L$	$-$
PID	$1.2\frac{T}{KL}$	$2.0L$	$0.5L$

典拠: [広井, 宮田], p.99

- 上記とは別に, CHR 法と呼ばれる方法も知られている. CHR は提案者 (Chein, Hrones, Rwswich) の頭文字である. 制御の目的が目標値追従と外乱抑制の場合, 応答波形のオーバーシュートを許容するか否かに応じて, 以下の 4 通りのパラメータが挙げられている (典拠は [広井, 宮田]).

CHR 法によるパラメータの決定 (1):

目標値追従, オーバーシュートなし

動作モード	K_P	T_I	T_D
P	$0.3\frac{T}{KL}$	$-$	$-$
PI	$0.35\frac{T}{KL}$	$1.2T$	$-$
PID	$0.6\frac{T}{KL}$	T	$0.5L$

典拠: [広井, 宮田], p.101

CHR 法によるパラメータの決定 (2):

目標値追従, オーバーシュートあり

動作モード	K_P	T_I	T_D
P	$0.7\frac{T}{KL}$	–	–
PI	$0.6\frac{T}{KL}$	T	–
PID	$0.95\frac{T}{KL}$	$1.35T$	$0.47L$

典拠: [広井, 宮田], p.101

CHR 法によるパラメータの決定 (3):

外乱抑制, オーバーシュートなし

動作モード	K_P	T_I	T_D
P	$0.3\frac{T}{KL}$	$-$	$-$
PI	$0.6\frac{T}{KL}$	$4L$	$-$
PID	$0.95\frac{T}{KL}$	$2.4L$	$0.4L$

典拠: [広井, 宮田], p.101

CHR 法によるパラメータの決定 (4):

外乱抑制, オーバーシュートあり

動作モード	K_P	T_I	T_D
P	$0.7\frac{T}{KL}$	$-$	$-$
PI	$0.7\frac{T}{KL}$	$2.3L$	$-$
PID	$1.2\frac{T}{KL}$	$2L$	$0.42L$

典拠: [広井, 宮田], p.101

- 今まで述べてきたパラメータ決定法は $L = 0$ の場合は使えないことに注意.

- **限界感度法**これとは別に, 限界感度法と呼ばれる方法も知られている. これは $L = 0$ の場合にも使える. この方法は以下の通り.

  - ▷ PID 補償器の比例項以外をオフにする
  - ▷ K_P を徐々に大きくしてゆき, 持続的発振が始まったら, その時の K_P^* と, 振動の周期 T^* を記録する.

▷ 以下の表にしたがってパラメータを決定する.

動作モード	K_P	T_I	T_D
P	$0.5K_P^*$	$-$	$-$
PI	$0.45K_P^*$	$\frac{T^*}{1.2}$	$-$
PID	$0.6K_P^*$	$\frac{T^*}{2}$	$\frac{T^*}{8}$

- これらの方法によって決まるPIDパラメータの値は参考値であって, 一般には, これらのパラメータを仮に設定して制御装置を動作させてから, 試行錯誤によって微調整をおこなう必要がある.

- PID パラメータの決め方には, 他にも様々なものがあるが, この講義ではこれ以上は述べない.
- 上述のパラメータは制御対象がこれで安定化されることを保証するわけではないので, 安定性については別途検討が必要.

- 制御対象を安定化するPIDパラメータを求める方法もあるが(たとえばS. P. Bhattacharyya, A. Datta and L. H. Keel, Linear Control Theory: Structure, Robustness, and Optimization, CRC Press, 2009), この講義では「そのような方法もある」という紹介に留める.

- **飽和とその対策:**　制御対象に印加できる入力には上限や下限があることが多く, PID 制御では, 積分器の部分でこれに抵触し, 入力が飽和するリスクが高い. 積分器の 飽和現象を integrator windup と呼ぶ. Integrator windup を回避する仕組みを anti-windup と呼び, 様々な技法が知られているが, この講義では立ち入らない.

アナログ PID 補償器の離散化

- アナログ PID 補償器の離散化してディジタル PID 補償器を求めるための典型的な方法は, 双一次変換法と (後退) 差分法である.

- **双一次変換による方法:** すでにアナログPID補償器が求められているとき, その補償器に次の双一次変換

$$s = \frac{2}{T_s}\frac{z-1}{z+1}$$

を適用することにより, ディジタルPID補償器を求めることができる.

- 前回の講義で見たように,

$$s = \frac{2}{T_s}\frac{z-1}{z+1}$$

は, $z = e^{sT_s}$ の逆関数

$$s = \frac{1}{T_s}\log z$$

の近似から誘導される.

- 双一次変換によって離散化したディジタルPID補償器は以下の通り.

$$K_P \left(1 + \frac{1}{T_I} \frac{T_s}{2} \frac{z+1}{z-1} + T_D \frac{2}{T_s} \frac{z-1}{z+1} \right)$$

- 実用上の工夫として, 微分動作の $z = -1$ における極は安定性の観点から好ましくないため, これを $z = 0$ に変更することがある. すなわち, 微分項の分母多項式 $\dfrac{1}{z+1}$ を $\dfrac{1}{z}$ で置き換える.

上記のようにすると, 周波数零 (z 領域では $z = 1$ に対応) における分母多項式の値が $\frac{1}{2}$ から 1 に変わる. これは低周波域で微分項の影響が倍増することを意味し, 好ましくないため, これを抑制するため, このようにした場合は, 微分ゲインを半減する.

$$K_P \left(1 + \frac{1}{T_I} \frac{T_s}{2} \frac{z+1}{z-1} + T_D \frac{1}{T_s} \frac{z-1}{z} \right)$$

- **後退差分による方法:** 差分による離散化の方法では, サンプリング周期を T_s としたとき, 微分を後退差分

$$\frac{dy}{dt} \simeq \frac{y(t) - y(t-1)}{T_s}$$

で置き換える.

- 先の近似は,

$$sY(s) \simeq \frac{Y(z) - z^{-1}Y(z)}{T_s}$$

と書けるので, 後退差分近似は, 演算子 s を次の演算子で置き換えることに対応する.

$$\frac{z-1}{zT_s}$$

- **位置型と速度型:** アナログ PID 補償器を素直に離散化すると, 各時刻で制御信号 $u(n)$ を計算する形になる. これを位置型と呼ぶのであるが, ディジタル PID 補償器の実装では, $u(n)-u(n-1)$ (制御信号の差分) を計算する形が用いられることがある. $u(n)-u(n-1)$ を計算する形の補償器を速度型と呼ぶ.

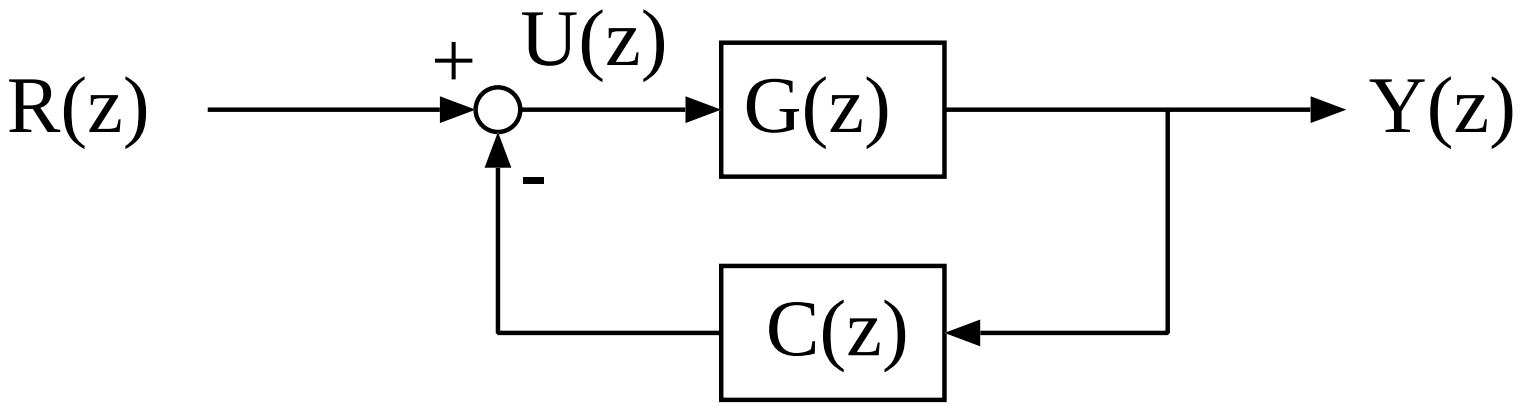
極配置

- 古典制御的な設計では, 少数の補償器のパラメータを調整して制御系の特性を改善することが多いのだが, 安定性については, より積極的に 制御系の閉ループ伝達関数の極を 任意に指定することも可能. これを極配置と呼ぶ.

- 出力信号のフィードバックに基づいて制御系の極配置をおこなう方法は, 1) 補償器の伝達関数を直接設計する, 2) 状態フィードバックとオブザーバを併用する, という 2 種類の方法に 分類される. 今回は第一の方法について述べる (第二の方法については次回).

- PID 制御と異なり, 極配置には制御対象の数学モデルが必要となる.

- 以下の図のような制御系を考える.



- $G(z) = \dfrac{B(z)}{A(z)}$, $C(z) = \dfrac{N(z)}{D(z)}$ で, $G(z)$ はプロパー, $A(z)$ と $B(z)$ は既約とする. $N(z)$, $D(z)$ は $C(z)$ がプロパーとなる限り自由に設定できる.

- $R(z)$ から $Y(z)$ への伝達関数は:

$$
\begin{aligned}
Y(z) &= \frac{G(z)}{1 + G(z)C(z)} R(z) \\
&= \frac{B(z)D(z)}{A(z)D(z) + B(z)N(z)} R(z)
\end{aligned}
$$

- $M(z)$ を希望するフィードバック系の分母多項式とする．$A(z)D(z) + B(z)N(z) = M(z)$ を満たす $D(z)$ と $N(z)$ を求めることができ，かつ $\deg D(z) \geq \deg N(z)$ であれば，$C(z) = N(z)/D(z)$ を補償器に用いることで，$Y(z) = \dfrac{B(z)D(z)}{M(z)} R(z)$ となり，極配置が達成される．
- 後で述べるように，$\deg M(z) \geq 2 \deg A(z)$ ならこの問題は解ける．

- まず, $A(z)$ と $B(z)$ を多項式, $C(z)$ を $A(z)$ と $B(z)$ の最大公約多項式としたとき, $A(z)D_0(z)+B(z)N_0(z) = C(z)$ を満たす多項式 $D_0(z)$ と $N_0(z)$ が存在することを示す (次ページ).
- 上記の結果から, $A(z)$ と $B(z)$ が既約のときには, ある $D_0(z)$ と $N_0(z)$ に対し, $A(z)D_0(z) + B(z)N_0(z) = 1$ となる.

- **定理**　$A(z)$ と $B(z)$ を多項式で $C(z)$ を $A(z)$ と $B(z)$ の最大公約多項式とする．このとき，$A(z)D_0(z) + B(z)N_0(z) = C(z)$ を満たす多項式 $D_0(z)$ と $N_0(z)$ が存在する．

- 証明

  - $\mathbb{K}[X]$ を多項式全体の集合とし,

$$\mathcal{S} = \{A(z)D(z) + B(z)N(z) : D(z), N(z) \in \mathbb{K}[X]\} \setminus \{0\}$$

とする.

- $\deg$ によって多項式の次数をあらわす. 便宜上, 零多項式の次数は $-\infty$ とする.
- $D(z) = 1, N(z) = 0$ あるいは $D(z) = 0, N(z) = 1$ と取ることにより, $A(z), B(z) \in \mathcal{S}$ であることがわかる.

- $\mathcal{S}$ は零でない多項式の集合であるから, 次数が最小の元を含む. それを $C(z)$ とする.
- $P(z) \in \mathcal{S}$ に対し, $\deg P(z) \geq \deg C(z)$ であり, 多項式の除算により,

$$P(z) = C(z)Q(z) + R(z)$$

で, $\deg R(z) < \deg C(z)$ となる多項式 $Q(z)$, $R(z)$ が取れる.

▷ $P(z), C(z) \in \mathcal{S}$ だから,

$$P(z) = A(z)D_P(z) + B(z)N_P(z)$$
$$C(z) = A(z)D_C(z) + B(z)N_C(z)$$

となるような多項式 $D_P(z)$, $N_P(z)$, $D_C(z)$, $N_C(z)$ が取れる.

▷ よって,

$$\begin{aligned} R(z) &= P(z) - C(z)Q(z) \\ &= A(z)(D_P(z) - Q(z)D_C(z)) \\ &\quad + B(z)(N_P(z) - Q(z)N_C(z)) \in \mathcal{S}. \end{aligned}$$

- $\deg R(z) < \deg C(z)$ で, $C(z)$ は次数最小の元だったから, $R(z) = 0$ でなければならない. よって, $R(z) = 0$, すなわち $C(z)$ は $\mathcal{S}$ のすべての元を割り切る. 特に, $C(z)$ は $A(z)$ と $B(z)$ の公約多項式になっている.

▷ $S(z)$ を $A(z)$ と $B(z)$ の公約多項式とすると, $S(z)$ は $\boldsymbol{S}$ のすべての元を割り切るから, $C(z)$ を割り切る. ゆえに, $C(z)$ は $A(z)$ と $B(z)$ の公約多項式で, $A(z)$ と $B(z)$ の任意の公約多項式によって割り切られるから, $A(z)$ と $B(z)$ の最大公約多項式である. (証明終わり)

- 上記の存在証明はやや複雑であるが, 上記の条件を満たす $D(z)$ と $N(z)$ の構成自体は易しい. 具体的には, $(A(z), B(z))$ から出発して, 次に述べる Euclid の互除法を適用すればよい.

- **Euclid の互除法**

  準備 $A_1(z) = A(z)$, $A_2(z) = B(z)$, $k = 1$ とする.

  ループ $A_k(z)$ を $A_{k+1}(z)$ で割った余りを $A_{k+2}(z)$ とおき, $k = k + 1$ とする. $A_{k+2}(z) = 0$ となれば終了.

- **定理** Euclid の互除法は有限回で終了し, 終了時に $A(z)$ と $B(z)$ の最大公約多項式 $C(z)$ が得られる.

- 証明
  - まず有限回で計算が終了することを確認する.

▷ Euclid の互除法が生成する多項式の列は,

$$\deg A_1(z) > \deg A_3(z) > \cdots$$
$$\deg A_2(z) > \deg A_4(z) > \cdots$$

のように多項式の次数が下がるため, 上記のアルゴリズムは有限回で終了する.

- ▷ アルゴリズム終了時点の $A_{k+1}(z)$ が $A(z)$ と $B(z)$ の最大公約多項式になるのであるが, これを確認するのにはひと手間かかる.

▷ $k = K$ においてアルゴリズムが終了したものとする．$A_{K+2}(z) = 0$ より，$A_K(z) = P_K(z)A_{K+1}(z)$ となり，$A_{K+1}(z)$ は $A_K(z)$ を割り切る．

一方, ある $P_{K-1}(z)$ に対し,

$$A_{K-1}(z) = P_{K-1}(z)A_K(z) + A_{K+1}(z)$$

となっているから, $A_{K+1}(z)$ は $A_{K-1}(z)$ を割り切り, 以下同様にすべての $\{A_k(z) : 1 \leq k \leq K\}$ を割り切る. とくに, $A_{K+1}(z)$ は $A(z)$ と $B(z)$ を割り切るから, $A(z)$ と $B(z)$ の公約多項式である.

$A_{K+1}(z)$ は $\mathcal{S}$ に属するから, $A(z)$ と $B(z)$ の最大公約多項式 $C(z)$ で割り切れる. したがって, $A_{K+1}(z)$ は $A(z)$ と $B(z)$ の約多項式で, 最大公約多項式 $C(z)$ で割り切れるから, 定数倍を除き $A(z)$ と $B(z)$ の最大公約多項式 $C(z)$ と一致する.

(証明終わり)

- 我々は, $A(z)$ と $B(z)$ が既約であり, $M(z)$ が希望するフィードバック系の分母多項式であるとき (ただし $\deg M(z) \geq 2 \deg A(z)$), $M(z)$ に極を配置するフィードバック補償器を導出したいのであった. 次にこの方法について述べる.

- $A(z)$ と $B(z)$ は既約であると

$$A(z)D_0(z) + B(z)N_0(z) = 1$$

となる $(N_0(z), D_0(z))$ が存在する．これは既に求められているものとする．

- $N(z) = M(z)N_0(z),\ D(z) = D_0(z)M(z),$ とすれば,

$$\begin{aligned} &A(z)D(z) + B(z)N(z) \\ &\quad = (A(z)D_0(z) + B(z)N_0(z))M(z) \\ &\quad = M(z). \end{aligned}$$

となる.

- 以上によって, 形式的には $C(z) = \dfrac{N(z)}{D(z)}$ によって極配置を達成することができるのだが (この時点では $\deg M(z) \geq 2 \deg A(z)$ という条件は使われていない), この形では $C(z) = \dfrac{N(z)}{D(z)}$ が因果的であることが保証されないため, これを因果的にするために, もう 1 段階の工程が必要である.

- $\deg M(z) \geq 2 \deg A(z)$ という条件を使うのは, この $C(z) = \dfrac{N(z)}{D(z)}$ を因果的な形に変更するための工程である.

- $N(z)$ を $A(z)$ で割った剰余を $R(z)$ とする. ある $Q(z)$ に対し, $N(z) = A(z)Q(z) + R(z)$ であり, よって, 次式が成り立つ.

$$\begin{aligned} M(z) &= A(z)D(z) + B(z)N(z) \\ &= A(z)D(z) \\ &\quad + B(z)N(z)\left(A(z)Q(z) + R(z)\right) \\ &= A(z)(D(z) + B(z)Q(z)) \\ &\quad + B(z)R(z). \end{aligned}$$

- $L(z) = D(z) + B(z)Q(z)$ とおく. これを先の式に代入すると, 次式が得られる.

$$M(z) = A(z)L(z) + B(z)R(z)$$

- $\deg R(z) < \deg A(z)$ で, $\deg B(z) \leq \deg A(z)$ だから, $\deg B(z)R(z) < 2\deg A(z)$. 一方, $\deg M(z) \geq 2\deg A(z)$ であるように取られていたから, $\deg A(z)L(z) \geq 2\deg A(z)$ でなければならない.
- よって, $\deg L(z) \geq \deg A(z)$.

- しがたって,

$$C(z) = \frac{R(z)}{L(z)}$$

は因果的な補償器になる．また, $\deg A(z) + \deg L(z) = \deg M(z)$ になっている.

- $R(z)$ から $Y(z)$ への伝達関数は

$$Y(z) = \frac{G(z)}{1 + G(z)C(z)} R(z)$$

となるのであった. ここに

$$G(z) = \frac{B(z)}{A(z)}, \quad C(z) = \frac{R(z)}{L(z)}$$

を代入すると…

- 以下のようになる.

$$
\begin{aligned}
Y(z) &= \frac{\frac{B(z)}{A(z)}}{1+\frac{B(z)}{A(z)}\frac{R(z)}{L(z)}}R(z) \\
&= \frac{B(z)L(z)}{A(z)L(z)+B(z)R(z)}R(z) \\
&= \frac{B(z)L(z)}{M(z)}R(z)
\end{aligned}
$$

- $M(z)$ は希望する極配置に対応する分母多項式だったから, 極配置が達成されている.

- 念のため, この閉ループ伝達関数が因果的であることを確認しておく. $\deg A(z)+\deg L(z) = \deg M(z)$ だったから, 分子の次数は

$$\deg L(z) + \deg B(z)$$
$$= \deg M(z) - \deg A(z) + \deg B(z) \leq \deg M(z)$$

で, 分母の次数は $\deg M(z)$ なので, $R(z)$ から $Y(z)$ への伝達関数は確かに因果的になっている.

- $M(z)$ の次数が $2\deg A(z)$ 以上で, $B(z)R(z)$ が高々$2\deg A(z)-1$ 次であるために, $\deg L(z) = \deg M(z) - \deg A(z) \geq \deg A(z)$ となるという点が重要. $M(z)$ の次数を上記の条件を満たすように取らないと, 因果的な補償器が構成できないことがある.

$$\underbrace{M(z)}_{2\deg A(z)\text{ 以上}} = \underbrace{A(z)L(z)}_{2\deg A(z)\text{ 以上}} + \underbrace{B(z)R(z)}_{2\deg A(z)\text{ 未満}}$$

- この講義では多項式に基づいて極配置に基づく (安定化) 補償器を求めてきたのだが, より現代的なのは, 有理式を使って同様の計算をおこなう手法 (H_∞ 制御など) である.

- いずれの場合も，上記の方法を拡張して，すべての安定化補償器を特徴付けることができ，それを **Youla parametrization** と呼ぶが，この講義では名前を紹介するに留める．