ディジタル制御 第８回

ディジタル制御系における状態フィードバック・極配置・オブザーバ

状態フィードバック・極配置

- 第 3 回の講義では，分母が N 次のプロパーな伝達関数の実現を取り扱った．厳密にプロパーな場合の可制御正準形による実現について復習する．

- 以下の伝達関数を考える.

$$G(z) = \frac{B(z)}{A(z)} = \frac{b_1 z^{N-1} + \cdots + b_N}{z^N + a_1 z^{N-1} + \cdots + a_N}.$$

- 入力を u, 出力を y, $U(z) = \mathcal{Z}[u]$, $Y(z) = \mathcal{Z}[y]$ とし, $l = 1, \ldots, N$ に対し,

$$X_l(z) = \frac{1}{A(z)} z^{l-1} U(z)$$

と定義すると,

$$Y(z) = b_1 X_N(z) + b_2 X_{N-1}(z) + \cdots + b_N X_1(z)$$

となる.

- $l = 1, \ldots, N-1$ に対し

$$zX_l(z) = X_{l+1}(z),$$

$l = N$ については

$$zX_N(z) = U(z) - a_1 X_N(z) - \cdots - a_N X_1(z).$$

とする.

以下の記号を用いる:
$\boldsymbol{I}_{N-1}$: $N-1$ 次の単位行列 $\mathbf{0}_{N-1}$: $N-1$ 次の零ベクトル $\boldsymbol{B}_c$: N 次で N 番目の単位ベクトル $(=\boldsymbol{e}_N = (0,\ldots,0,1)^T)$ $\boldsymbol{C}_c = (b_N,\ldots,b_1)$(行ベクトル) $\boldsymbol{A}_c = \left(\begin{array}{c\|c} \mathbf{0}_{N-1} & \boldsymbol{I}_{N-1} \\ \hline -a_N & \cdots \quad -a_1 \end{array}\right)$ $\boldsymbol{X}(z) = (X_1(z),\ldots,X_N(z))^T$

- 前ページの記号の先程の式を纏めると, 次式が得られる.

$$
\begin{aligned}
z\boldsymbol{X}(z) &= \boldsymbol{A}_c\boldsymbol{X}(z) + \boldsymbol{B}_c U(z) \\
Y(z) &= \boldsymbol{C}_c\boldsymbol{X}(z).
\end{aligned}
$$

- 全体を逆 z 変換すると, 差分方程式に直すと可制御正準形による実現になる.

$$
\begin{aligned}
\boldsymbol{x}(n+1) &= \boldsymbol{A}_c \boldsymbol{x}(n) + \boldsymbol{B}_c u(n) \\
y(n) &= \boldsymbol{C}_c \boldsymbol{x}(n).
\end{aligned}
$$

- さて, 当面状態 $\boldsymbol{x}(n)$ が直接測定可能であるものとし, $\boldsymbol{K} \in \mathbb{R}^{1\times N}$ として,

$$u(n) = \boldsymbol{K}\boldsymbol{x}(n) + v(n)$$

という制御則を考える. $u(n) = \boldsymbol{K}\boldsymbol{x}(n)$ の部分を 状態フィードバック と呼ぶ. $v(n)$ については 他の設計目的のために残しておく.

- 状態フィードバックで何ができるかを理解するためには, 可制御正準形で, 行列 $\boldsymbol{A}_c$ の最下段に伝達関数の分母多項式の係数が並んでいたことを思い出す必要がある.

- $\boldsymbol{K} = (k_N, \ldots, k_1)$ とすると (添字の順番に注意), 状態フィードバック

$$\boldsymbol{u}(n) = \boldsymbol{K}\boldsymbol{x}$$

によって, 状態方程式は

$$\boldsymbol{x}(n+1) = (\boldsymbol{A}_c + \boldsymbol{B}_c\boldsymbol{K})\boldsymbol{x}(n) + \boldsymbol{B}_c v(n)$$

のように変わる.

- ここで $\boldsymbol{A}_c$ と $\boldsymbol{B}_c$ の具体的な形を思い出すと,

$$
\begin{aligned}
&\boldsymbol{A}_c + \boldsymbol{B}_c\boldsymbol{K} \\
&\qquad = \left(\begin{array}{c|c} \mathbf{0}_{N-1} & \boldsymbol{I}_{N-1} \\ \hline k_N - a_N & \cdots \quad k_1 - a_1 \end{array}\right).
\end{aligned}
$$

- $\boldsymbol{K}$ は自由に設定できるから，状態フィードバック
$$u(n) = \boldsymbol{K}\boldsymbol{x}(n)$$
により，伝達関数の分母多項式の係数，したがって 伝達関数の極を 自由に設定することができる．これを状態フィードバックによる 極配置 と呼ぶ．

- Euclid の互除法に基づく伝達関数の演算でも極配置は可能ではあるが, 状態が直接利用可能な場合には, 状態フィードバックによる極配置の方が簡単.

- 伝達関数から出発して状態フィードバックに基づく極配置をしたいときには, 可制御正準形による実現を選べば, 残りの計算は自明.
- 可制御な状態方程式が与えられている場合, それを可制御正準形に変形することにより, 極配置が可能であることが確認できる.

- 続いて, 可制御な線形時不変システムの状態空間表現を可制御正準形へ変換する方法を検討する.
- まず 1 入力 1 出力系について述べ, 続いた多入力多出力系について述べる.

- 可制御な離散時間線形時不変システムの可制御正準形への変形には逆行列を用いるが，この逆行列を明示的に書き下すことが一般的.
- この講義では，実質的には同等だが，ニュアンスの異なる説明をする.

- 可制御な N 次の 1 入力離散時間線形時不変システム

$$\boldsymbol{x}(n+1) = \boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}u(n)$$

を考える (可制御性の議論には出力方程式は無関係なので省く). 可制御性を仮定したから, $\boldsymbol{M}_c = (\boldsymbol{B}, \boldsymbol{A}\boldsymbol{B}, \ldots, \boldsymbol{A}^{N-1}\boldsymbol{B})$ とすると, $\operatorname{rank} \boldsymbol{M}_c = N$ である.

- $\boldsymbol{\lambda} \in \mathbb{R}^N$ を, 連立一次方程式

$$\boldsymbol{\lambda}^T \boldsymbol{M}_c = (0, \ldots, 0, 1)$$

の解とする. $\boldsymbol{M}_c$ は仮定より正則だから, 上記は一意解を持つ. これを用い,

$$w_k(n) = \boldsymbol{\lambda}^T \boldsymbol{A}^{k-1} \boldsymbol{x}(n)$$

とし $(1 \leq k \leq N)$,
$\boldsymbol{w}(n) = (w_1(n), \ldots, w_N(n))^T$ とする.

- $\boldsymbol{T} = \begin{pmatrix} \boldsymbol{\lambda}^T \\ \boldsymbol{\lambda}^T \boldsymbol{A} \\ \cdots \\ \boldsymbol{\lambda}^T \boldsymbol{A}^{N-1} \end{pmatrix}$ とおくと, 後で見るように, $\boldsymbol{T}$ は正則で, よって $\boldsymbol{w}(n) = \boldsymbol{T}\boldsymbol{x}(n)$ は座標変換である. また, $\boldsymbol{\lambda}^T \boldsymbol{M}_c = (\boldsymbol{T}\boldsymbol{B})^T$ となっている.

- $w_k(n)$ の構成法より, $k < N$ なら $w_k(n+1) = w_{k+1}(n)$.
- $\boldsymbol{\lambda}^T \boldsymbol{M}_c$ の第 N 成分は 1 なので, $w_N(n+1) = \boldsymbol{\lambda}^T \boldsymbol{A}^{N-1}(\boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}u(n)) = \boldsymbol{\lambda}^T \boldsymbol{A}^N \boldsymbol{x}(n) + u(n)$. また, $\boldsymbol{T}$ は正則なので, ある $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_N) \in \mathbb{R}^N$ に対し, $\boldsymbol{\lambda}^T \boldsymbol{A}^N = \boldsymbol{\gamma}^T \boldsymbol{T}$. ゆえに, $w_N(n+1) = \boldsymbol{\gamma}^T \boldsymbol{w}(n) + u(n)$.

- これらをまとめて

$$\boldsymbol{w}(n+1) = \boldsymbol{A}_{cc}\boldsymbol{w}(n) + \boldsymbol{B}_{cc}u(n)$$

という形に書くと, $\boldsymbol{A}_{cc}$ と $\boldsymbol{B}_{cc}$ は, 以下のように可制御正準形になっている.

$$
\boldsymbol{B}_{cc} = \boldsymbol{e}_N = (0, \ldots, 0, 1)^T
$$

$$
\boldsymbol{A}_{cc} = \left(\begin{array}{c|ccc} \mathbf{0}_{N-1} & & \boldsymbol{I}_{N-1} & \\ \hline \gamma_1 & & \cdots & \gamma_N \end{array} \right)
$$

- まだ $\boldsymbol{T}$ が正則であることの証明が残っていた. これを示す.
  - ▷ ある零でない $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_N)^T \in \mathbb{R}^N$ に対し, $\boldsymbol{\theta}^T \boldsymbol{T} = \mathbf{0}^T$ と仮定して矛盾を導く.
  - ▷ $\boldsymbol{\theta}^T \boldsymbol{T} \boldsymbol{B} = \mathbf{0}^T \boldsymbol{B} = 0$ であるが, 一方, $\boldsymbol{T}\boldsymbol{B} = \boldsymbol{e}_N$ なので, $\theta_N = 0$.

▷ $K = \max\{i : \theta_i \neq 0\}$ とすると, $K \leq N-1$ で, $\boldsymbol{T}$ の形を思い出すと,

$$\theta_1 \boldsymbol{\lambda}^T + \cdots + \theta_K \boldsymbol{\lambda}^T \boldsymbol{A}^{K-1} = \mathbf{0}^T$$

となることがわかる.

- 先の式の左辺の最後の項を右辺に移項する:

$$\theta_1 \boldsymbol{\lambda}^T + \cdots + \theta_{K-1} \boldsymbol{\lambda}^T \boldsymbol{A}^{K-2} = -\theta_K \boldsymbol{\lambda}^T \boldsymbol{A}^{K-1}.$$

- 両辺に $\boldsymbol{A}^{N-K}\boldsymbol{B}$ を左から掛けると, 次式が得られる.

$$\begin{aligned}\theta_1 \boldsymbol{\lambda}^T \boldsymbol{A}^{N-K} \boldsymbol{B} + \cdots + \theta_{K-1} \boldsymbol{\lambda}^T \boldsymbol{A}^{N-2} \boldsymbol{B} \\ - \theta_K \boldsymbol{\lambda}^T \boldsymbol{A}^{N-1} \boldsymbol{B} = -\theta_K \neq 0\end{aligned}$$

▷ ここで,

$$\boldsymbol{M}_c = (\boldsymbol{B}, \boldsymbol{A}\boldsymbol{B}, \ldots, \boldsymbol{A}^{N-1}\boldsymbol{B})$$

で, $\boldsymbol{\lambda} \in \mathbb{R}^N$ は, 連立一次方程式

$$\boldsymbol{\lambda}^T \boldsymbol{M}_c = (0, \ldots, 0, 1)$$

の解であったことを思い出す.

- ▷ したがって, $0 \leq j \leq N-1$ に対し,

$$\boldsymbol{\lambda}^T \boldsymbol{A}^j \boldsymbol{B} = 0$$

である.

▷ すでに得られていた

$$\theta_1 \boldsymbol{\lambda}^T \boldsymbol{A}^{N-K} \boldsymbol{B} + \cdots + \theta_{K-1} \boldsymbol{\lambda}^T \boldsymbol{A}^{N-2} \boldsymbol{B} - \theta_K \boldsymbol{\lambda}^T \boldsymbol{A}^{N-1} \boldsymbol{B} = -\theta_K \neq 0$$

と比較すると,

$$0 = -\theta_K \boldsymbol{\lambda}^T \boldsymbol{A}^{N-1} \boldsymbol{B} = -\theta_K \neq 0$$

となり矛盾. (証明終わり)

- 以上によって, 可制御な１入力システムは 可制御正準形に変換でき，従って 状態フィードバックによって その極を任意に指定でき, よって安定化できることがわかった.

- $\boldsymbol{\lambda}^T \boldsymbol{M}_c = (0, \ldots, 0, 1)$ を解くということは, $\boldsymbol{M}_c$ の逆行列の第 N 行を使うということと同じ. よって, 以上の方法は, 通常の可制御正準形の導出と実質的には同じ.

- 連続時間システムと異なり, 離散時間システムでは, すべての極を原点に配置することで, 状態が有限時間で零に収束するような制御をおこなうことができる. これを デッドビート制御と呼ぶことがある. ただし, デッドビート制御という言葉は, 制御対象の出力を有限時間で目標値に誤差なく追従させる制御という意味で用いられることもある.

- 次に, 可制御な N 次の M 入力離散時間線形時不変システム

$$\boldsymbol{x}(n+1) = \boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}\boldsymbol{u}(n)$$

を考える. 可制御性を仮定したから, $\boldsymbol{M}_c = (\boldsymbol{B}, \boldsymbol{A}\boldsymbol{B}, \ldots, \boldsymbol{A}^{N-1}\boldsymbol{B})$ とすると, $\operatorname{rank} \boldsymbol{M}_c = N$ である. これを多入力版の可制御正準形 (**Brunovsky 正準形**と呼ばれる) に変換する. ただし, これは一意的でない.

- $\boldsymbol{B}$ のある列 $\boldsymbol{b}_i$ に対し, $\operatorname{rank}(\boldsymbol{b}_i, \ldots, \boldsymbol{A}^{N-1}\boldsymbol{b}_i) = N$ となっていれば, この列だけを使って もとのシステムを 可制御正準形に変換できる.
- 上記以外, すなわち $\operatorname{rank} \boldsymbol{M}_c = N$ とするために $\boldsymbol{B}$ の複数の行が必要な場合には, $\boldsymbol{B}$ の複数の列を使った操作が必要になる.

- $\boldsymbol{B}$ がフルランクでないときには, 基本変形と入力変換で冗長性を除くことができるので, この処理が既に終了し, $\boldsymbol{B}$ がフルランクになっているという状況から議論を始める. $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M)$ とする.
- まず, 座標変換のための基底を構成する. 基底の取り方は一意的でないため, Brunovsky 正準形の議論は複雑になる.

- 基底の構成
  - ▷ **ステップ 1:** $\mathcal{D}_0 = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M\}$ とする. $\boldsymbol{B}$ はフルランクと仮定したから, $\mathcal{D}_0$ は一次独立なベクトルから成る.

▷ **ステップ** k $(k \geq 1)$**:**

$$\boldsymbol{A}\mathcal{D}_k = \{\boldsymbol{A}\boldsymbol{d} : \boldsymbol{d} \in \mathcal{D}_k\}$$

とする．$\boldsymbol{A}\mathcal{D}_k$ の全要素が $\mathcal{D}_k$ と一次従属であれば終了．そうでなければ，

$$\mathcal{D}_k \cup \boldsymbol{A}\mathcal{D}_k$$

から $\mathcal{D}_k$ を含む一次独立な要素を可能な限り多く選び，それを $\mathcal{D}_{k+1}$ とする．

以上の処理の意味は, $\boldsymbol{M}_c$ から $\boldsymbol{A}$ のべきの次数が小さい極大の一次独立な列を選ぶということ． $\operatorname{rank} \boldsymbol{M}_c = N$ だから, 上記の処理はある $k = K$ で終了し, $\mathcal{D}_K$ が $\mathbb{R}^N$ の基底となる． ■

- $\mathcal{D}_K$ に含まれるベクトルを横に並べた行列 (正方行列である) を $\boldsymbol{D}_K$ としたとき, 多入力システムの可制御正準形は 1 入力システムと同様に $\boldsymbol{D}_K^{-1}$ を使って構成されるのであるが, ここで若干の工夫が必要になる.

- この講義だけの用語であるが, $\mathcal{D}_K$ に

$$\boldsymbol{b}_i, \boldsymbol{A}\boldsymbol{b}_i, \ldots, \boldsymbol{A}^l\boldsymbol{b}_i$$

までが含まれ,

$$\boldsymbol{A}^{l+1}\boldsymbol{b}_i$$

が含まれないとき, $\boldsymbol{b}_i$ はレベル l であるという.

- $\boldsymbol{B}$ の列ベクトルの添字を取り直し, $\boldsymbol{b}_1^{[l]}, \ldots, \boldsymbol{b}_{r_l}^{[l]}$ がレベル l であるようにする (レベル l の列がないときにはこの集合は空集合で $r_l = 0$). 入力についても $u_j^{[l]}(n)$ のように添字を合わせておく. $\mathcal{D}_K$ の構成法から, $\boldsymbol{B}$ の列が取り得るレベルは 0 から K までで, レベル K の列が少なくともひとつは存在する.

- レベル K の $\boldsymbol{B}$ の列をひとつ取り ($\boldsymbol{b}_1^{[K]}$ とする), $\boldsymbol{\lambda}_1^T \boldsymbol{A}^K \boldsymbol{b}_1^{[K]} = 1$ かつ $\boldsymbol{d} \in \mathcal{D}_K \setminus \{\boldsymbol{A}^K \boldsymbol{b}_1^{[K]}\}$ に対して $\boldsymbol{\lambda}_1^T \boldsymbol{d} = 0$ となる $\boldsymbol{\lambda}_1$ を求める ($\boldsymbol{D}_K^{-1}$ から該当する行を選ぶことに相当).

- レベル $l < K$ の $\boldsymbol{b}_i^{[l]}$ に対しては, $\boldsymbol{\lambda}_1^T \boldsymbol{A}^j \boldsymbol{b}_i^{[l]} = 0$ となることが明示的に指定されているのは $j \leq l$ の場合だけなのであるが, 実は任意の $j \leq K$ に対して $\boldsymbol{A}^j \boldsymbol{b}_i^{[l]} \in \mathcal{D}_j$ であるから, $j \leq K-1$ であれば $\boldsymbol{\lambda}_1^T \boldsymbol{A}^j \boldsymbol{b}_i^{[l]} = 0$ である.

- $w_{1,k}^{[K]}(n) = \boldsymbol{\lambda}_1^T \boldsymbol{A}^{k-1} \boldsymbol{x}(n)$ と定義する ($1 \leq k \leq K+1$). $k \leq K$ なら,

$$
\begin{aligned}
w_{1,k}^{[K]}(n+1) &= \boldsymbol{\lambda}_1^T \boldsymbol{A}^{k-1} (\boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}\boldsymbol{u}(n)) \\
&= w_{1,k+1}^{[K]}(n)
\end{aligned}
$$

である ($\boldsymbol{\lambda}_1^T \boldsymbol{A}^{k-1} \boldsymbol{B} = \mathbf{0}$ に注意).

- $w_{1,K+1}^{[K]}(n)$ については,

$$\begin{aligned} w_{1,k}^{[K]}(n+1) &= \boldsymbol{\lambda}_1^T \boldsymbol{A}^K (\boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}\boldsymbol{u}(n)) \\ &= \boldsymbol{\lambda}_1^T \boldsymbol{A}^{K+1} \boldsymbol{x}(n) + u_1^{[K]}(n) \\ &\quad + \sum_{0 \le l < K, 1 \le j \le r_l} c_{l,j} u_j^{[l]} \end{aligned}$$

という形になる.

- レベル K の成分に対応する入力は $u_1^{[K]}(n)$ しか現れないが, レベル K 未満の成分に対応する入力は現れる (この部分は, 状態に関する座標変換の構成が終わってから入力ベクトルの座標変換で消去する).

- $\mathbf{0}_j$ を j 次の行あるいは列ベクトル,

$$\boldsymbol{N}_{j+1} = \left(\begin{array}{c|c} \mathbf{0}_j & \boldsymbol{I}_j \\ \hline 0 & \mathbf{0}_j \end{array}\right),$$

$\boldsymbol{e}_{K+1}$ を 第 $K+1$ 番目の単位ベクトルとし,

$$\boldsymbol{p}_1^{[K]} = \boldsymbol{\lambda}_1^T \boldsymbol{A}^{K+}$$

$\boldsymbol{w}_1^{[K]}(n) = (w_{1,1}^{[K]}(n), \ldots, w_{1,K+1}^{[K]}(n))^T$

とすると⋯

$$
\begin{aligned}
&\boldsymbol{w}_1^{[K]}(n+1) \\
&\qquad = \boldsymbol{N}_{K+1}\boldsymbol{w}_1^{[K]}(n) + \boldsymbol{e}_{K+1}(\boldsymbol{p}_1^{[K]}\boldsymbol{x}(n) \\
&\qquad\qquad + u_1^{[K]}(n) + \phi_1^{[K]}(u^{[l]} : l < K))
\end{aligned}
$$

という形になる ($\phi_1^{[K]}(u^{[l]} : l < K)$ はレベル K 未満の入力の線形結合).

- レベル K の列が複数ある場合には, 同様の処理をおこなう.

- レベル L ($L < K$) の列 $\boldsymbol{b}_j^{[L]}$ についても同様に処理し, 次の形を得る.

$$
\begin{aligned}
&\boldsymbol{w}_j^{[L]}(n+1) \\
&\quad = \boldsymbol{N}_{L+1}\boldsymbol{w}_j^{[L]}(n) + \boldsymbol{e}_{L+1}(\boldsymbol{p}_j^{[L]}\boldsymbol{x}(n) \\
&\qquad + u_j^{[L]}(n) + \phi_j^{[L]}(u^{[l]} : l < L)). \quad (♪)
\end{aligned}
$$

- レベル l の要素は r_l 個あり, それぞれに対して $(l+1)$ 次元のベクトル $\boldsymbol{w}_j^{[l]}$ を生成したから, 以上のようにして作ったベクトル全体の次元は $\sum_{l=0}^{K} r_l(l+1)$ であるが, これは $\mathcal{D}_K$ の次元に一致し, よって $\sum_{l=0}^{K} r_l(l+1) = N$ である.

- 先に述べた $\boldsymbol{w}_j^{[L]}(n)$ に関する状態方程式は $\phi_j^{[L]}(u^{[l]}: l < L))$ の項を除けば可制御正準形になっている. よって, $\boldsymbol{x}(n)$ から $\{\boldsymbol{w}_j^l : 0 \le j \le K, 1 \le j \le r_j\}$ への写像が正則なら, 座標変換によって 状態方程式を 可制御正準形に変換できたことになる.

- 上述の写像の正則性を示すために，レベル l の $\boldsymbol{w}_j^{[l]}(n)$ を構成するときに使った 行ベクトルの全体を，記号を取り直して，以下のように書く．

$$
\begin{array}{cccc}
\boldsymbol{\lambda}_1^{[l]}, & \boldsymbol{\lambda}_1^{[l]}\boldsymbol{A}, & \ldots, & \boldsymbol{\lambda}_1^{[l]}\boldsymbol{A}^l, \\
\ldots & \ldots & \ldots & \ldots \\
\boldsymbol{\lambda}_{r_l}^{[l]}, & \boldsymbol{\lambda}_{r_l}^{[l]}\boldsymbol{A}, & \ldots, & \boldsymbol{\lambda}_{r_l}^{[l]}\boldsymbol{A}^l.
\end{array}
$$

- これらが一次従属, すなわち それらの中に1個以上非零のものがあるスカラーの組 $\{c_{j,p}^{[l]} : 0 \leq l \leq K, 1 \leq j \leq r_l, 0 \leq p \leq l\}$ に対し,

$$\sum_{0 \leq l \leq K, 0 \leq p \leq l, 1 \leq j \leq r_l} c_{j,p}^{[l]} \boldsymbol{\lambda}_j^{[l]} \boldsymbol{A}^p = \boldsymbol{0} \qquad (\text{☆})$$

となったものと仮定して矛盾を導く.

- (☆) の両辺に右から $\boldsymbol{b}_j^{[l]}$ を乗ずると，左辺は $c_{j,l}^{[l]}$, 右辺は零となるから, $c_{j,l}^{[l]} = 0$ である $(0 \leq l \leq K, 1 \leq j \leq r_l)$. 同様に, (☆) の両辺に右から $\boldsymbol{A}^m \boldsymbol{b}_j^{[l]}$ を m の次数を上げながら順に乗ずることで, すべての $c_{j,p}^{[l]}$ が零となり, 矛盾.

- 証明のイメージは次のページの通り．以下のように階段状にベクトルを並べ, 右から $\boldsymbol{A}^k\boldsymbol{b}_j$ を乗ずることで, 右端から順に対応する係数が零であることを示してゆく．

$$\ldots, \quad \boldsymbol{\lambda}_1^{[K]} \boldsymbol{A}^{K-2} \quad \boldsymbol{\lambda}_1^{[K]} \boldsymbol{A}^{K-1} \quad \boldsymbol{\lambda}_1^{[K]} \boldsymbol{A}^{K}$$

$$\ldots, \quad \boldsymbol{\lambda}_{r_K}^{[K]} \boldsymbol{A}^{K-2} \quad \boldsymbol{\lambda}_{r_K}^{[K]} \boldsymbol{A}^{K-1} \quad \boldsymbol{\lambda}_{r_K}^{[K]} \boldsymbol{A}^{K}$$

$$\ldots, \quad \boldsymbol{\lambda}_1^{[K-1]} \boldsymbol{A}^{K-2} \quad \boldsymbol{\lambda}_1^{[K-1]} \boldsymbol{A}^{K-1}$$

$$\ldots, \quad \boldsymbol{\lambda}_{r_{K-1}}^{[K-1]} \boldsymbol{A}^{K-2} \quad \boldsymbol{\lambda}_{r_{K-1}}^{[K-1]} \boldsymbol{A}^{K-1}$$

$$\ldots, \quad \boldsymbol{\lambda}_1^{[K-2]} \boldsymbol{A}^{K-2}$$

$$\ldots, \quad \boldsymbol{\lambda}_{r_{K-1}}^{[K-2]} \boldsymbol{A}^{K-2}$$

- $\boldsymbol{w}(n) = (\boldsymbol{w}_j^{[l]}(n) : 0 \le l \le K, 1 \le j \le r_l)$ (ただし, 上記の成分の並べ方を踏襲する) とおき, $\boldsymbol{T}$ を $\boldsymbol{x}(n)$ から $\boldsymbol{w}(n)$ への変換行列とすると, 以上で見たように $\boldsymbol{T}$ は正則な正方行列. よって, 上記 (♪) に含まれる $\boldsymbol{p}_j^{[L]}\boldsymbol{x}(n)$ という項は, $\boldsymbol{p}_j^{[L]}\boldsymbol{T}^{-1}\boldsymbol{w}(n)$ と書き直せる.

- 以上はわかりにくいと思われるので, 例としてレベル $\boldsymbol{b}_1$ がレベル 2, $\boldsymbol{b}_2$ と $\boldsymbol{b}_3$ がレベル 1 の 3 入力システムに上記を適用する. $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, $\boldsymbol{p}_3$ は適切なベクトルとする.

- もとの式は以下の通り.

$$
\boldsymbol{w}(n+1) = \left(\begin{array}{ccc|cc|cc}
0 & 1 & 0 & & & & \\
0 & 0 & 1 & & & & \\
0 & 0 & 0 & & & & \\
\hline
 & & & 0 & 1 & & \\
 & & & 0 & 0 & & \\
\hline
 & & & & & 0 & 1 \\
 & & & & & 0 & 0
\end{array}\right) \boldsymbol{w}(n) + \left(\begin{array}{c|c|c}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & {\color{red}*} & {\color{red}*} \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
\hline
0 & 0 & 0 \\
0 & 0 & 1
\end{array}\right) \boldsymbol{u}(n)
$$

$$
+ \left(\begin{array}{c|c|c}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
\hline
0 & 0 & 0 \\
0 & 0 & 1
\end{array}\right) \begin{pmatrix} \boldsymbol{p}_1^T \boldsymbol{w}(n) \\ \boldsymbol{p}_2^T \boldsymbol{w}(n) \\ \boldsymbol{p}_3^T \boldsymbol{w}(n) \end{pmatrix}
$$

- $\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3$ を行列の中に入れると (数値は重要でないので $*$ で書いてある)$\cdots$

$$
\boldsymbol{w}(n+1) = \left(\begin{array}{ccc|cc|cc}
0 & 1 & 0 & & & & \\
0 & 0 & 1 & & & & \\
* & * & * & * & * & * & * \\
\hline
 & & & 0 & 1 & & \\
* & * & * & * & * & * & * \\
\hline
 & & & & & 0 & 1 \\
* & * & * & * & * & * & *
\end{array}\right) \boldsymbol{w}(n) + \left(\begin{array}{c|c|c}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & * & * \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
\hline
0 & 0 & 0 \\
0 & 0 & 1
\end{array}\right) \boldsymbol{u}(n)
$$

- 入力側の 変換で レベルが低い入力への依存性を除いて完成. このような形が **Brunovsky 正準形**だが, 文献によって定義にバリエーションがある.

$$
\boldsymbol{w}(n+1) = \left(\begin{array}{ccc|cc|cc}
0 & 1 & 0 & & & & \\
0 & 0 & 1 & & & & \\
* & * & * & * & * & * & * \\
\hline
 & & & 0 & 1 & & \\
* & * & * & * & * & * & * \\
\hline
 & & & & & 0 & 1 \\
* & * & * & * & * & * & *
\end{array}\right) \boldsymbol{w}(n) + \left(\begin{array}{c|c|c}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & \color{red}{0} & \color{red}{0} \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
\hline
0 & 0 & 0 \\
0 & 0 & 1
\end{array}\right) \boldsymbol{u}(n)
$$

- 次に, 多入力システムの極配置について考える. 一般論はわかりにくいので, 先に述べた例を踏襲する.

$*$ の部分に適当な記号を入れて…

$$\boldsymbol{w}(n+1) = \left(\begin{array}{ccc|cc|cc} 0 & 1 & 0 & & & & \\ 0 & 0 & 1 & & & & \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & a_{3,6} & a_{3,7} \\ \hline & & & 0 & 1 & & \\ a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5} & a_{5,6} & a_{5,7} \\ \hline & & & & & 0 & 1 \\ a_{7,1} & a_{7,2} & a_{7,3} & a_{7,4} & a_{7,5} & a_{7,6} & a_{7,7} \end{array}\right) \boldsymbol{w}(n) + \left(\begin{array}{c|c|c} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 1 \end{array}\right) \boldsymbol{u}(n)$$

- $\boldsymbol{k}_1$, $\boldsymbol{k}_2$, $\boldsymbol{k}_2$ を以下のように取り, $u_i(n) = \boldsymbol{k}_i \boldsymbol{w}(n) + v_i(n)$ とすると $(i = 1, 2, 3)\cdots$

$$\boldsymbol{k}_1 = (\alpha_1 - a_{3,1}, \alpha_2 - a_{3,2}, \alpha_3 - a_{3,3}, -a_{3,4}, -a_{3,5}, -a_{3,6}, -a_{3,7})$$

$$\boldsymbol{k}_2 = (-a_{5,1}, -a_{5,2}, -a_{5,3}, \alpha_4 - a_{5,4}, \alpha_5 - a_{5,5}, -a_{5,6}, -a_{5,7})$$

$$\boldsymbol{k}_3 = (-a_{7,1}, -a_{7,2}, -a_{7,3}, -a_{7,4}, -a_{7,5}, \alpha_6 - a_{7,6}, \alpha_7 - a_{7,7})$$

$$
\boldsymbol{w}(n+1) = \left(\begin{array}{ccc|cc|cc}
0 & 1 & 0 & & & & \\
0 & 0 & 1 & & & & \\
\alpha_1 & \alpha_2 & \alpha_3 & & & & \\
\hline
 & & & 0 & 1 & & \\
 & & & \alpha_4 & \alpha_5 & & \\
\hline
 & & & & & 0 & 1 \\
 & & & & & \alpha_6 & \alpha_7
\end{array}\right) \boldsymbol{w}(n) + \left(\begin{array}{c|c|c}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
\hline
0 & 0 & 0 \\
0 & 0 & 1
\end{array}\right) \boldsymbol{v}(n)
$$

- $\boldsymbol{w}(n+1) = \boldsymbol{A}\boldsymbol{w}(n) + \boldsymbol{B}\boldsymbol{v}(n)$ に対応する伝達関数行列の各要素の分母は, $\det(z\boldsymbol{I} - \boldsymbol{A})$ を適当な多項式で割ったものになるが, 上記のように $\det(z\boldsymbol{I} - \boldsymbol{A})$ は状態フィードバックによって 自由に設定できる. よって, この例では状態フィードバックによって 極配置ができている.

- 先の例では 状態フィードバック によって パラメータが変更できる箇所を明示するために, 敢えて小規模なシステムを挙げたのだが, 一般の場合にも, システムを可制御正準形 $\boldsymbol{x} = \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c\boldsymbol{u}$ に変換すると行列 $\boldsymbol{A}_c$ において特性多項式を定める行の成分が 状態フィードバック $\boldsymbol{u}(n) = \boldsymbol{K}\boldsymbol{x}(n)$ によって任意に設定できるという構造が現れる ということは同じ.

- よって，多入力システムでも，状態空間表現を可制御正準形に変換してから適切な状態フィードバック $\boldsymbol{u}(n) = \boldsymbol{K}\boldsymbol{x}(n)$ を施すことにより，極配置が可能で，よってシステムを安定化することができることがわかるのであるが…

- なお, 可制御正準形への変換は, 極配置が可能であることを確認するための手段である. すでに 可制御であるとわかっている 状態方程式を (極配置ではなく) 安定化したいとき, わざわざ 可制御正準形に変換する 必然性はない (別の方法がある).

- 状態方程式

$$\boldsymbol{x}(n+1)=\boldsymbol{A}\boldsymbol{x}(n)+\boldsymbol{B}\boldsymbol{u}(n)$$

が可制御であれば, 状態フィードバック

$$\boldsymbol{u}(n)=\boldsymbol{K}\boldsymbol{x}(n)$$

によって安定化もできるというのが以上の結論.

- この状態フィードバックによって $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$ の特性多項式の根は任意に設定できるが, $\boldsymbol{K}$ をどのような手段で求めても構わない.

- 文献によっては, 状態フィードバックを $\boldsymbol{u}(n) = \boldsymbol{K}\boldsymbol{x}(n)$ ではなく $\boldsymbol{u}(n) = -\boldsymbol{K}\boldsymbol{x}(n)$ と定義しているので注意.

- システムが単に安定化できれば良い場合には $\boldsymbol{K}$ には無数の取り方があるが, 最適制御 (第 10 回) ではある評価関数に関して最適な $\boldsymbol{K}$ を求める. また, ロバスト性を考慮して $\boldsymbol{K}$ を定めることもある.

オブザーバ

- 伝達関数の実現により可観測正準形の状態空間表現が得られているという状況を考える. この場合, システムの挙動に影響を及ぼす信号の中で, センサなどによって直接測定されているのは, 大抵は入力と出力のみであり, 数学的な手段で導出されたすべての状態変数に対応するセンサが揃っているということは稀.

- このような理由から, 状態空間表現された制御対象に基づいて制御系設計をおこなうとき, 直接測定可能な信号は 入力と 出力のみ, という条件を課すことが多い.
- このような状況下で, 状態フィードバックと同等な制御をおこないたいときには, 何らかの手段で 入力と出力から 状態 (の推定値) を生成しなければならない.

- 状態推定のための 最もポピュラーな方法は, **(Luenberger) オブザーバ**と呼ばれる動的システムを利用する手法. Luenberger オブザーバは 確定的なシステムを想定しているが, 対象が確率システムの場合には, 手法自体は同一なのだが, 名前が **Kalman フィルタ** に変わる.

- オブザーバのもっとも素朴な構成では, 状態フィードバックと類似した機構が利用される.
- 再び, 分母が N 次, 分子が $N-1$ 次以下の厳密にプロパーな伝達関数を考える.

$$G(z) = \frac{B(z)}{A(z)} = \frac{b_1 z^{N-1} + \cdots + b_N}{z^N + a_1 z^{N-1} + \cdots + a_N}.$$

記号の準備

$\boldsymbol{a} = (a_1, \ldots, a_N)^T$

$\boldsymbol{B}_o = (b_1, \ldots, b_N)^T$

$\boldsymbol{C}_o = (1, 0, \ldots, 0)$:

(N 次の１番目の単位ベクトル)

$$\boldsymbol{A}_o = \left(\begin{array}{c|c} -\boldsymbol{a} & \begin{array}{c} \boldsymbol{I}_{N-1} \\ \hline \boldsymbol{0}_{N-1}^T \end{array} \end{array} \right)$$

$\boldsymbol{x}(n) = (x_1(n), \ldots, x_n(z))^T$

- 以前の講義で, $G(z)$ の可観測正準形による実現は次のようになることを見た.

$$
\begin{aligned}
\boldsymbol{x}(n+1) &= \boldsymbol{A}_o \boldsymbol{x}(n) + \boldsymbol{B}_o u(n) \\
y(n) &= \boldsymbol{C}_o \boldsymbol{x}(n)
\end{aligned}
\qquad (\bigstar)
$$

- ($\bigstar$) に対する もっとも標準的なオブザーバ (Luenberger オブザーバ) は, ($\bigstar$) の同一の構造の動的システムによって状態 $\boldsymbol{x}$ の推定値 $\hat{\boldsymbol{x}}$ を生成し, 状態推定値から得られる出力と ($\bigstar$) の出力の偏差を その動的システムにフィードバックすることで, 状態推定誤差 $\boldsymbol{e} = \boldsymbol{x} - \widehat{\boldsymbol{x}}$ の安定化をはかるというもの.

- Luenberger オブザーバの構成は以下の通り.

$$\begin{aligned} \hat{\boldsymbol{x}}(n+1) &= \boldsymbol{A}_o \hat{\boldsymbol{x}}(n) + \boldsymbol{B}_o u(n) \\ &\quad - \boldsymbol{F}(y(n) - \hat{y}(n)) \qquad (\star) \\ \hat{y}(n) &= \boldsymbol{C}_o \hat{\boldsymbol{x}}(n) \end{aligned}$$

- $\boldsymbol{F}(y(n) - \hat{y}(n))$ の部分は **Output Injection** と呼ばれることがあり, これを使って 状態推定誤差を安定化する. $\boldsymbol{F} = (f_1, \ldots, f_N)^T$ は設計者が自由に設定できる.

- 状態推定誤差 $\boldsymbol{e}$ に関する方程式は, 次のようになる. ただし, $y(n) = \boldsymbol{C}\boldsymbol{x}(n)$ および $\hat{y}(n) = \boldsymbol{C}\hat{\boldsymbol{x}}(n)$ を代入している.

$$\boldsymbol{e}(n+1) = \boldsymbol{A}_o\boldsymbol{e}(n) + \boldsymbol{F}\boldsymbol{C}_o(\boldsymbol{x}(n) - \hat{\boldsymbol{x}}(n)) \quad (♪)$$

- ここで, $\boldsymbol{C}_o = (1, 0, \ldots, 0)$ であったことを思い出し, (♪) に含まれる行列を明示的に書

くと…

$$
\begin{pmatrix} e_1(n+1) \\ \vdots \\ e_{N-1}(n+1) \\ e_N(n+1) \end{pmatrix}
= \left(\begin{array}{c|ccc} -a_1 + f_1 & & & \\ \vdots & & \boldsymbol{I}_{N-1} & \\ -a_{N-1} + f_{N-1} & & & \\ \hline -a_N + f_N & 0 & \cdots & 0 \end{array}\right)
\begin{pmatrix} e_1(n) \\ \vdots \\ e_{N-1}(n) \\ e_N(n) \end{pmatrix}
$$

- 上記右辺の行列の 左端に 特性多項式の係数が並んでいるから, これが安定多項式となるように $(f_1, \ldots, f_N)$ を設定すれば, 状態推定誤差 $\boldsymbol{e}(n)$ は 時間とともに零に収束する.

- Luenberger オブザーバの設計問題は, 状態推定誤差が零に収束するような $\boldsymbol{F}$ を設定する問題に帰着され, このような $\boldsymbol{F}$ を見付けられた場合に, オブザーバが構成できたということになる.

- 以上で確認したのは, 状態空間表現が可観測正準形になっているときには, いつでもオブザーバが構成可能であるということ.

- 実は, 可観測な線形時不変システムは, 可制御正準形への変換と同様の手順によって, 可観測正準形に変換できることが証明できる. MIMO システムへの拡張についても同様.

- 上記の事実を改めて証明することも可能だが，次回の講義で述べる双対性という性質を使うと，システムの可制御正準形への変換が (若干変更すれば) 使える．ここで，今回の講義では，可観測正準形への変換についてはこれ以上述べない．

- 次に, 多出力システムに対するオブザーバの構成を考える.

- 多出力版の可観測正準形への変換は既になされているものとする.

$$\boldsymbol{x}(n+1) = \boldsymbol{A}_o\boldsymbol{x}(n) + \boldsymbol{B}_o\boldsymbol{u}(n)$$
$$\boldsymbol{y}(n) = \boldsymbol{A}_o\boldsymbol{x}(n)$$

- このシステムに対するオブザーバの構造は次の通りで, $\boldsymbol{F}$ を適切に設定することで

$$\boldsymbol{A}_o + \boldsymbol{C}_o\boldsymbol{F}$$

の特性多項式の根を任意に設定できる．その理由は状態フィードバックによる 安定化と同じ.

$$
\begin{aligned}
\hat{\boldsymbol{x}}(n+1) &= \boldsymbol{A}_o\hat{\boldsymbol{x}}(n) + \boldsymbol{B}_o\boldsymbol{u}(n) \\
&\quad - \boldsymbol{F}(\boldsymbol{y}(n) - \hat{\boldsymbol{y}}(n)) \\
\hat{\boldsymbol{y}}(n) &= \boldsymbol{A}_o\hat{\boldsymbol{x}}(n)
\end{aligned}
$$

- 先と同様に, 可観測正準形への変換は, 状態推定誤差に関する極配置が可能であることを確認するための手段であって, すでに 可観測であるとわかっている 状態方程式に対して, オブザーバを構成するために, 改めて可観測正準形に変換する 必然性はない.

- 以下の可観測な状態空間表現に対して⋯

$$\boldsymbol{x}(n+1) = \boldsymbol{A}\boldsymbol{x}(n) + \boldsymbol{B}\boldsymbol{u}(n)$$
$$\boldsymbol{y}(n) = \boldsymbol{C}\boldsymbol{x}(n)$$

- 次の形の オブザーバが構成可能で, 行列 $\boldsymbol{F}$ を適切に設定することで状態推定誤差 $\boldsymbol{x}(n) - \hat{\boldsymbol{x}}(n)$ が 零に収束するようにできるというのが上記の結論. 行列 $\boldsymbol{F}$ をどのような手段で求めてもよい.

$$
\begin{aligned}
\hat{\boldsymbol{x}}(n+1) &= \boldsymbol{A}\hat{\boldsymbol{x}}(n) + \boldsymbol{B}\boldsymbol{u}(n) \\
&\quad - \boldsymbol{F}(\boldsymbol{y}(n) - \hat{\boldsymbol{y}}(n)) \\
\hat{\boldsymbol{y}}(n) &= \boldsymbol{C}\hat{\boldsymbol{x}}(n)
\end{aligned}
$$

- 状態推定誤差を $\boldsymbol{e}(n) = \boldsymbol{x}(n) - \hat{\boldsymbol{x}}(n)$ とすると, $\boldsymbol{e}(n)$ は次式を満たすのだが:

$$\boldsymbol{e}(n+1) = (\boldsymbol{A} + \boldsymbol{F}\boldsymbol{C})\boldsymbol{e}(n)$$

  $\boldsymbol{A} + \boldsymbol{F}\boldsymbol{C}$ の固有値をオブザーバの極と呼ぶことがある.

- オブザーバの極をすべて原点に配置すると，雑音等がなければ，状態推定誤差は有限時間で零に収束する．このようなオブザーバを有限整定オブザーバないしデッドビートオブザーバと呼ぶことがある．ただし，有限整定オブザーバの構成法は他にもある．

- 文献によっては, $-\boldsymbol{F}(\boldsymbol{y}(n)-\hat{\boldsymbol{y}}(n))$ のかわりに $+\boldsymbol{F}(\boldsymbol{y}(n)-\hat{\boldsymbol{y}}(n))$ を用いることがあるので注意.
- 状態推定誤差が零に減衰しさえすれば良いという場合には $\boldsymbol{F}$ には無数の取り方がある. 一方, 何らかの評価関数に関し最適な $\boldsymbol{F}$ を求めたり, ロバスト性を考慮して $\boldsymbol{F}$ を定めたりすることもある.

- オブザーバについては他にも述べるべきことがあるが, 次回に回す.

Scilabにおける可制御正準形・可観測正準形・極配置

- Scilabにおいて可制御正準形を求める関数は `canon` である．ただし，Scilabの `canon` が生成する可制御正準形は，講義資料で述べた可制御正準形とは，パラメータの並べ方が異なる．

- 講義資料の可制御正準形は…

$$\left(\begin{array}{c|c} \mathbf{0} & \boldsymbol{I}_{n-1} \\ \hline \multicolumn{2}{c}{* \cdots *} \end{array}\right)$$

- Scilab の可制御正準形は…

$$\left(\begin{array}{c|c} \multicolumn{2}{c}{* \cdots *} \\ \hline \boldsymbol{I}_{n-1} & \mathbf{0} \end{array}\right)$$

- Scilab において, 線形システムの状態空間表現に対応する `A`, `B` がすでに定義されているという前提の下で,

  ```
  [Ac,Bc,U,ind]=canon(A,B)
  ```

  とすると, (`Ac`,`Bc`) に可制御正準形に変換された行列が格納される.

- Scilab で 状態フィードバックに基づく極配置をおこなう関数は `ppol` である.

- Scilab において, 線形システムの状態空間表現に対応する `A`, `B` がすでに定義されているという前提の下で, 適切な変数 (ここでは `poles` とするが, 何でもよい), に指定する極を並べたものをベクトルとして格納しておき,

```
K=ppol(A,B,poles)
```

とすると, `K` に極配置を実現する状態フィードバックゲイン行列が格納される.

- ただし, ここでも注意が必要で, Scilab は

$$\texttt{A} - \texttt{B} * \texttt{K}$$

が指定された極を持つように K を定める.

$$\texttt{A} + \texttt{B} * \texttt{K}$$

ではないので注意せよ.

- Scilabには可観測正準形とオブザーバの設計のための関数は用意されていない.

- Scilab において, 線形システムの状態空間表現に対応する `A`, `B`, `C` がすでに定義されているという条件の下で, Scilab でこのシステムの可観測正準形を求めるときには, `A` と `C` を転置してからこれらの可制御正準形を求め, その結果を転置する.

- 同様に, オブザーバの設計のためには, `A` と `C` を転置してから, これらに `ppol` を適用し, 得られた結果を転置する.