並列学習制御系の安定性*

磯 村 修 —*1

Stability of Parallel Learning Control Systems

Shuichi ISOMURA

This paper deals with the eigenvalue problem of the multiple eigenvalue filter which was proposed in the previous paper, "Parallel Learning in Control Systems (Derivation of Multiple Eigenvalue Filter)". The system with proposed filter derived by means of parallel learning not only reduces control variable deviation without decreasing stability, but also has PID (Proportional, Integral and Derivative) control action although the original control system before learning has just proportional control action. In this paper, the reason is shown from the view point of eigenvalue problem and Nyquist stability theory.

Key Words: Paralle Learning, Model, Linear Feedback Control, Multiple Eigenvalue, Stability, Nyquist

1. ま え が き

電力系統における需給のアンバランスはますます大幅かつ急峻となり、原動機の制御はこのような要求に応えるため、従来のフィードバック制御に加え、負荷変動外乱に応じた先行動作を与えるフィードフォワード制御を採用してきた。現在、フィードフォワードとフィードバックの割合は7:3位になっている⁽¹⁾。この3割のフィードバック制御は予期せぬ外乱や、対象の動特性の変化に対処するために残されている。フィードフォワード制御は、種々の負荷変動外乱に対する操作量変化の長年にわたる経験の積み重ねであったが、最近では、さらに、さまざまな補償回路や⁽²⁾最適制御⁽³⁾の実用化が進んでいる。

本論文では、外乱を受けるフィードバック制御系において、フィードフォワード制御を求める手段として、フィードバック制御による操作量の変化を繰返し学習することを提案した。これまで、制御偏差の逐次学習によって制御性の改善を試みた報告がなされているが^{(4)~(6)}、本論文では、操作量を記憶して、次に外乱を

受けたとき記憶した操作量を先行動作として加える時系列的な逐次学習ではなく、リアルタイムシミュレーションモデルを学習の回数分、例えば、 k 個並べて、外乱をすべてのモデルに一齐に与え、 $i-1$ 番めのモデルの操作量を i 番めのモデルの操作量に加える方法を提案している。この方法によれば k を増加させていくことにより、制御系の安定性を損なうことなく制御偏差を限りなく小さくすることができるのみならず、瞬時に学習が完了するので、動的システムの学習過程を一括して連立微分方程式系で表現でき、学習の効果や安定性を解明しやすいからである。

前報の「制御系の並列学習(重根フィルタの提案)」では、このような並列学習を提案し、その構成方法と効果について述べ、フィードバック制御が比例動作のみでも、並列学習の最終過程ではPID制御と同じような効果が得られることを示した。さらに、フィードバック制御系の操作部を k 回並列学習することは、制御対象伝達関数を P 、制御伝達関数を C とするとき、特性方程式の左辺 $(I+PC)$ を i 乗($i=0, 1, 2, \dots, k$)したもの合計して、制御伝達関数 C に直列に挿入したものと同等であることから、これを重根フィルタと名付けた。このフィルタでは、特性方程式の根は重複するだけなので、制御系は不安定とはならないことが

* 原稿受付 平成5年5月20日。

*1 正員、三菱重工業(株)原動機技術統括室 (〒108 東京都港区芝5-34-6)。

わかったが、本論文では、なぜPIDの機能が生じるのかについて調べる。この並列学習では、偏差がある限り、繰返しフィードバック制御が働くので、積分的な効果(I動作)があるが、それでも不安定にならないのは、微分的な要素(D動作)もあることが予想される。

並列学習された系の特性改善に影響を及ぼすこれらの機能について、固有値、固有ベクトルの計算結果をもとに明らかにするとともに、ナイキストの判別法を用いて、より定量的に安定性を検討する。

2. 並列学習システムの構成

並列学習の固有値問題を考えるため、連立状態方程式で表された線形ダイナミカルシステムを構成する。状態方程式および出力の式は次式で与えられる。

$$dx/dt = Ax + Bu + Hv \quad (1)$$

$$y = Gx \quad (2)$$

操作量の式を

$$u = w + z \quad (3)$$

フィードバック制御を

$$z = C(y_r - y) \quad (4)$$

出力ベクトルと状態ベクトルの関係を

$$y_r = Gx_r \quad (5)$$

と表すことにする。

ここに、 x は n 次元状態ベクトル、 u は m 次元操作ベクトル、 v は 1 次元外乱ベクトル、 y は m 次元出力ベクトル、 y_r は m 次元目標出力ベクトル、 w は m 次元フィードフォワード制御ベクトル、 z は m 次元フィードバック制御の動作信号ベクトル、 x_r は n 次元目標状態ベクトルであり、 A は $n \times n$ 定数マトリックス、 B は $n \times m$ 定数マトリックス、 H は $n \times 1$ 定数マトリックス、 G は $m \times n$ 定数マトリックス、 C は $m \times m$ 定数マトリックスである。式(1)~(5)から y 、 u 、 z 、 y_r を消去すると状態方程式は次式となる。

$$dx/dt = (A - BCG)x + BCGx_r + Bw + Hv \quad (6)$$

式(1)~(5)で表される制御系を k 組コピーしたとき、 i 番目のモデルは、組の番号を変数の右肩に () 付きで表すと、

$$dx^{(i)}/dt = Ax^{(i)} + Bu^{(i)} + Hv^{(i)} \quad (7)$$

$$y^{(i)} = Gx^{(i)} \quad (8)$$

$$u^{(i)} = w^{(i)} + z^{(i)} \quad (9)$$

$$z^{(i)} = C(y_r^{(i)} - y^{(i)}) \quad (10)$$

$$y_r^{(i)} = Gx_r^{(i)} \quad (11)$$

となる。フィードフォワード制御 $w^{(i)}$ として、前回までのすべてのフィードバック制御信号 $z^{(j)}$ ($j=1, 2, \dots, i-1$) を取り込むと、

$$w^{(i)} = z^{(1)} + z^{(2)} + z^{(3)} + \dots + z^{(i-1)} \quad (12)$$

である。ただし、 $i=1$ のときは学習されていないので、 $w^{(1)}=0$ である。 y 、 u 、 w 、 z 、 y_r を消去すると

$$dx^{(i)}/dt = -BCG(x^{(1)} + x^{(2)} + \dots + x^{(i-1)}) + (A - BCG)x^{(i)} + Hv^{(i)} + BCG(x_r^{(1)} + x_r^{(2)} + \dots + x_r^{(i)}) \quad (13)$$

を得る。 $i=1, 2, \dots, k$ として、 $k-1$ 回学習した結果は

$$\begin{aligned} dx^{(1)}/dt &= (A - BCG)x^{(1)} + Hv^{(1)} + BCGx_r^{(1)} \\ dx^{(2)}/dt &= -BCGx^{(1)} + (A - BCG)x^{(2)} + Hv^{(2)} + BCG(x_r^{(1)} + x_r^{(2)}) \\ dx^{(k)}/dt &= -BCG(x^{(1)} + x^{(2)} + \dots + x^{(k-1)}) + (A - BCG)x^{(k)} + Hv^{(k)} + BCG(x_r^{(1)} + x_r^{(2)} + \dots + x_r^{(k)}) \end{aligned} \quad (14)$$

となる。 $\alpha = -A + BCG$ 、 $\beta = -BCG$ とおくと、式(14)は行列で次のように表すことができる。

$$dx/dt = \begin{bmatrix} -\alpha & 0 & \dots & 0 \\ \beta & & & \\ \vdots & & & \\ \beta & \dots & \beta & -\alpha \end{bmatrix} x - \begin{bmatrix} \beta & 0 & \dots & 0 \\ \vdots & & & \\ \beta & \dots & \dots & \beta \end{bmatrix} x_r + \begin{bmatrix} H & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & & \\ 0 & \dots & 0 & H \end{bmatrix} v \quad (15)$$

となる。ここに、 $x = [x^{(1)} x^{(2)} \dots x^{(k)}]'$ 、 $x_r = [x_r^{(1)} x_r^{(2)} \dots x_r^{(k)}]'$ 、 $v = [v^{(1)} v^{(2)} \dots v^{(k)}]'$ である。' は転置を示す。

3. 固有値および固有ベクトルによる検討

3.1 固有値による安定判別 $n \times n$ 行列 $A - BCG = -\alpha$ が相異なる固有値 λ_i ($i=1, 2, \dots, n$) をもつとすると、 $(k \times n) \times (k \times n)$ 行列

$$\Gamma = \begin{bmatrix} -\alpha & 0 & \dots & 0 \\ \beta & & & \\ \vdots & & & \\ \beta & \dots & \beta & -\alpha \end{bmatrix} \quad (16)$$

の λ 行列 $\Gamma - \lambda I$ は、 λ_i ($i=1, 2, \dots, n$) 以外の固有値はもたず、 k 重複するのみである。したがって、 $\Gamma - \lambda I$ の行列式因子 $d^*(\lambda)$ は

$$d^*(\lambda) = (\lambda_1 - \lambda)^k (\lambda_2 - \lambda)^k (\lambda_3 - \lambda)^k \dots (\lambda_n - \lambda)^k \quad (17)$$

となり、 λ 行列 $\Gamma - \lambda I$ は $k \times n$ 本の固有ベクトルを列とする変換行列によって、 k 次のジョルダン細胞を n 個 (n ブロック) もつ標準形に変換される。もし、 $n \times n$ 行列 $A - BCG = -\alpha$ で表される状態方程式(6)が安定なら、 k 回の学習によって得られた $(k \times n) \times (k \times n)$ 行列 Γ で表される状態方程式(15)も安定である。

3.2 固有ベクトルによる検討 外乱のない場合、

式(15)は,

$$dx/dt = \begin{bmatrix} -\alpha & 0 & \cdots & 0 \\ \beta & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ \beta & \cdots & \beta & -\alpha \end{bmatrix} x - \begin{bmatrix} \beta & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots \\ 0 & \cdots & \ddots & 0 \\ \beta & \cdots & \cdots & \beta \end{bmatrix} x_r \quad \dots\dots\dots (18)$$

となる。重根フィルタのモードを調べるために、行列 Γ をジョルダン標準形に変換する変換行列の列ベクトル、つまり固有ベクトル $p^{(i)} (i=1, 2, \dots, k)$ を求める。まず、 $n=1$ の一変数系で見通しをたてる。

式(18)の右辺第1項の行列の固有値を λ とすると、対応する k 本の固有ベクトル $p^{(i)}$ は同次方程式

$$\begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \beta & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ \beta & \cdots & \beta & 0 \end{bmatrix} p^{(i)} = p^{(i-1)} \quad \dots\dots\dots (19)$$

の解である。ただし、 $i=1$ のときのみ、 $p^{(0)} = p^{(0)} = \dots = p^{(0)} = 0$ である。一般的には、 λ および $p^{(i)}$ は $n \times n$ 固有値行列および $(k \times n) \times n$ 行列というべきであろう。式(19)から固有ベクトルを求め、これらを列とする変換行列は

$$T = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & \beta^{-3} & \cdots \\ 0 & 0 & \beta^{-2} & \beta^{-2}(I-2\beta^{-1}) & \cdots \\ 0 & \beta^{-1} & \beta^{-1}(I-\beta^{-1}) & \beta^{-1}(I-\beta^{-1}+\beta^{-2}) & \cdots \\ I & I & I & I & \cdots \end{bmatrix} \quad \dots\dots\dots (20)$$

となる。式(18)はこの T によってジョルダン標準形に変換される。逆行列 T^{-1} は

$$T^{-1} = \begin{bmatrix} \cdots & -\beta & -\beta & -\beta & I \\ \cdots & \beta(I-2\beta) & \beta(I-\beta) & \beta & 0 \\ \cdots & \beta^2(2I-\beta) & \beta^2 & 0 & 0 \\ \cdots & \beta^3 & 0 & 0 & 0 \\ \cdots & 0 & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 0 & 0 & 0 & 0 \end{bmatrix} \quad \dots\dots\dots (21)$$

となる。 x, x_r の固有関数をそれぞれ x^*, x_r^* と表し、 $x = Tx^*, x_r = Tx_r^*$ と変換すると、式(18)は

$$dx^*/dt = \begin{bmatrix} -\alpha & I & 0 & \cdots & 0 \\ 0 & \ddots & & \vdots & \vdots \\ \vdots & & \ddots & I & 0 \\ 0 & \cdots & \cdots & 0 & -\alpha \end{bmatrix} x^* - \begin{bmatrix} \beta & I & 0 & \cdots & 0 \\ \vdots & \ddots & & \vdots & \vdots \\ 0 & \cdots & \ddots & I & 0 \\ 0 & \cdots & \cdots & 0 & \beta \end{bmatrix} x_r^* \quad \dots\dots\dots (22)$$

となり、式(18)の右辺第1項の行列 Γ はジョルダン

標準形になる。次に、 i 番めのモードに相当する入力 x_r^* の i 番めの成分 $x_r^{*(i)}$ に対する応答、すなわち固有のモードを調べるには、これを式(22)に代入して $x^{*(1)}, x^{*(2)}, \dots, x^{*(i-1)}, x^{*(i)}, x^{*(i+1)}, \dots, x^{*(k)}$ を求めればよい。

$x^{*(i)} = 0 (j=i+1, i+2, \dots, k)$ であるから、式(22)は下から並べ替えて、

$$\begin{aligned} dx^{*(i)}/dt &= -\alpha x^{*(i)} + x^{*(i+1)} - \beta \\ dx^{*(i-1)}/dt &= -\alpha x^{*(i-1)} + x^{*(i)} - 1 \\ dx^{*(j)}/dt &= -\alpha x^{*(j)} + x^{*(j+1)} \quad (j=1, 2, \dots, i-2) \end{aligned}$$

となる。この場合、強制項をもつのは、常に i 番めと $i-1$ 番めの式のみである。これを上から順に解いて

$$\begin{aligned} x^{*(i)} &= -\alpha^{-1} \beta (1 - e^{-\alpha t}) \\ x^{*(i-1)} &= -\alpha^{-1} (1 + \alpha^{-1} \beta) (1 - e^{-\alpha t}) + \alpha^{-1} \beta t e^{-\alpha t} \\ x^{*(i-j)} &= -\alpha^{-j} (1 + \alpha^{-1} \beta) (1 - e^{-\alpha t}) \\ &\quad + \alpha^{-1} \beta (t^j/j!) e^{-\alpha t} \\ &\quad + (1 + \alpha^{-1} \beta) \{ \alpha^{-1} t^{j-1}/(j-1)! + \alpha^{-2} t^{j-2}/(j-2)! \\ &\quad + \cdots + \alpha^{-(j-2)} t^2/2! + \alpha^{-(j-1)} t/1! \} e^{-\alpha t} \\ &\quad (j=2, 3, \dots, i-1) \end{aligned}$$

を得る。また、 $x^{*(i-1)}$ および $x^{*(i)}$ のラプラス変換を $X^{*(i-1)}$ および $X^{*(i)}$ とすると、対応する関係は、

$$\begin{aligned} X^{*(j)} &= 0 \quad (j=i+1, i+2, \dots, k) \\ X^{*(i)} &= -s^{-1} (s + \alpha)^{-1} \beta \\ X^{*(i-1)} &= -s^{-1} (s + \alpha)^{-2} (s + \alpha + \beta) \\ X^{*(i-j)} &= -s^{-1} (s + \alpha)^{-(j+1)} (s + \alpha + \beta) \end{aligned}$$

($j=2, 3, \dots, i-1$) となり、この $X^{*(i-1)}$ の分子の位相進み $s + \alpha + \beta$ が重根フィルタの安定化に役立っていることがわかる。例えば、 $dx/dt = -x + u$, $u = c(x_r - x)$, $c=1$ のとき、 $A=-1$, $B=1$, $C=1$, $G=1$, $\alpha = -(A - BCG) = 2$; $\beta = -BCG = -1$, であるから

$$(s + \alpha)^{-2} (s + \alpha + \beta) = (s + 2)^{-2} (s + 1)$$

となる。 $s = j\omega$ において、周波数応答を求めたとき、 $1 < \omega < 2$ で位相が進み、応答が改善されることが予想される。

3.3 固有ベクトル数値計算例 次に、 $n \geq 2$ の場合についての一般論は容易ではないので、数値計算で確かめることにする。以下に示す簡単な例について、行列 Γ , 変換行列 T , 逆行列 T^{-1} およびジョルダン標準形を求めた。

$$\begin{aligned} A &= \begin{bmatrix} -2 & 1 \\ 4 & -2 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, C = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \\ G &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, H = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

学習の回数は、代表例として $k=6$ の場合の計算結果を表1に示す。

表1の固有ベクトル入力に対する式(22)の応答、す

表 1 行列 Γ , 固有ベクトルを列とする変換行列, 逆行列およびジョルダン標準形

Matrix Gamma of Parallel Learning												
-4.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-2.0000	0.0000	-4.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	-2.0000	4.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-2.0000	0.0000	-2.0000	0.0000	-4.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	-2.0000	0.0000	-2.0000	4.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-4.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	4.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-4.0000	1.0000	0.0000	0.0000	0.0000
0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	4.0000	-4.0000	0.0000	0.0000	0.0000
-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-4.0000	1.0000	0.0000
0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	4.0000	-4.0000	0.0000
-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	1.0000
0.0000	-2.0000	0.0000	-2.0000	0.0000	0.0000	-2.0000	0.0000	-2.0000	0.0000	-2.0000	4.0000	-4.0000
Transfer Matrix T of Parallel Learning												
0.0000	0.0000	0.0000	0.0000	0.0000	-0.0156	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.0156
0.0000	0.0000	0.0000	0.0000	0.0000	-0.0313	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-0.0313
0.0000	0.0000	0.0000	0.0000	0.0000	0.0313	0.0938	0.0000	0.0000	0.0000	0.0000	-0.0313	-0.0938
0.0000	0.0000	0.0000	0.0000	0.0000	0.0625	0.1875	0.0000	0.0000	0.0000	0.0000	0.0625	0.1875
0.0000	0.0000	0.0000	-0.0625	-0.1563	-0.2500	0.0000	0.0000	0.0000	0.0000	0.0625	0.1563	0.2500
0.0000	0.0000	0.0000	-0.1250	-0.3125	-0.5000	0.0000	0.0000	0.0000	0.0000	-0.1250	-0.3125	-0.5000
0.0000	0.0000	0.1250	0.2500	0.3438	0.4063	0.0000	0.0000	-0.1250	-0.2500	-0.3438	-0.4063	-0.5000
0.0000	0.0000	0.2500	0.5000	0.6875	0.8125	0.0000	0.0000	0.2500	0.5000	0.6875	0.8125	0.9375
0.0000	-0.2500	-0.3750	-0.4375	-0.4688	-0.4844	0.0000	0.2500	0.3750	0.4375	0.4688	0.4844	0.5000
0.0000	-0.5000	-0.7500	-0.8750	-0.9375	-0.9688	0.0000	-0.5000	-0.7500	-0.8750	-0.9375	-0.9688	-1.0000
0.5000	0.5000	0.5000	0.5000	0.5000	0.5000	-0.5000	-0.5000	-0.5000	-0.5000	-0.5000	-0.5000	-0.5000
1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Inverse Transfer Matrix of T												
2.0000	1.0000	2.0000	1.0000	2.0000	1.0000	2.0000	1.0000	2.0000	1.0000	2.0000	1.0000	0.5000
-18.0000	-9.0000	-14.0000	-7.0000	-10.0000	-5.0000	-6.0000	-3.0000	-2.0000	-1.0000	0.0000	0.0000	0.0000
64.0000	32.0000	36.0000	18.0000	16.0000	8.0000	4.0000	2.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-111.9999	-56.0000	-40.0000	-20.0000	-8.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
96.0000	48.0000	16.0000	8.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-32.0000	-16.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-2.0000	1.0000	-2.0000	1.0000	-2.0000	1.0000	-2.0000	1.0000	-2.0000	1.0000	-1.0000	-0.5000	0.0000
18.0000	-9.0000	14.0000	-7.0000	10.0000	-5.0000	6.0000	-3.0000	2.0000	-1.0000	0.0000	0.0000	0.0000
-64.0000	32.0000	-36.0000	18.0000	-16.0000	8.0000	-4.0000	2.0000	0.0000	0.0000	0.0000	0.0000	0.0000
112.0000	-56.0000	40.0000	-20.0000	8.0000	-4.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
-96.0000	48.0000	-16.0000	8.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
32.0000	-16.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
Jordan Form of Parallel Learning												
-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	-2.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000	1.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	-6.0000

なわち, モードの計算結果を図 1 示す. 固有値 $\lambda = -2$ については $x^{*(6)}$, $\lambda = -6$ については $x^{*(12)}$ にユニットステップ入力を与えた. $x^{*(6)} = 1$ に対しては, $x^{*(6)}$ に基本のモードが, $x^{*(5)}$ に速いモードが現れる. また, $x^{*(12)} = 1$ に対しては, $x^{*(12)}$ に基本のモードが, $x^{*(11)}$ に速いモードが現れる. これらの結果から, 重根フィルタには位相進み補償機能のあることがわかる. $1 \leq i < 4$ および $7 \leq i \leq 10$ の遅いモードでは, 学習を繰返すたびに定常ゲインが増え続けるので, I 動作として働いて, 定常偏差の減少に寄与していることがわかる. 以上の検討から次の結果が得られる.

(1) 相異なる固有値をもつ系では, 任意の i 番め ($i=1, 2, \dots, k$) の固有ベクトル入力を加えると, その固有値に対応する i 番めのモードのみが現れるが, 学習のたびに固有値が不変で重複するだけの並列学習では, i 番め ($i=1, 2, \dots, k$) の固有ベクトル入力に対し $j=1, 2, \dots, i$ の各モードが現れる.

(2) $j > i$ となる, すなわち, $j=i+1, i+2, \dots, k$ のモードは現れない.

(3) 並列学習のモデルでは, i 番めの固有ベクトル入力に対し, i 番めのモードは基本のモードである.

図 1 並列学習のモードの計算結果

図2 並列学習のシミュレーション結果

(4) 一つ前の $i-1$ 番めのモードでは、基本のモードより位相の進んだ速い応答が現れる。

(5) $1, 2, \dots, i-2$ 番めのモードでは遅い応答が現れ、学習が $i-2, i-3, \dots$ とさかのぼるたびに減少していく。

実際の並列学習では、固有ベクトル入力ではなく、すべてのモデルに同じ目標値入力あるいは外乱入力が一斉に与えられるから、これらモードの線形結合として応答が現れる。参考に、同じ例題について、学習の効果を示すシミュレーション結果を図2に示す。入力は外乱 v_1 で、 $0 \leq t \leq 1$ のとき $v_1=0$ 、 $t > 1$ のとき $v_1=1$ とした。

4. ナイキストの安定判別

4.1 並列学習システムの伝達関数 線形ダイナミカルシステムを対象に並列学習を行って、重根フィルタの伝達関数行列を導く。 $Y=Y(s)$, $U=U(s)$, $V=V(s)$, $Y_T=Y_T(s)$ を $y=y(t)$, $u=u(t)$, $v=v(t)$, $y_T=y_T(t)$ のラプラス変換とすると、式(1)と(2)は

$$Y=PU+QV \quad (23)$$

となる。ここに、 P, Q はそれぞれ $m \times m$, $m \times 1$ の伝達関数行列で、

$$P=G(sI-A)^{-1}B \quad (24)$$

$$Q=G(sI-A)^{-1}H \quad (25)$$

である。同様に、式(3)と(4)もラプラス変換して、

$$U=W+Z \quad (26)$$

$$Z=C(Y_T-Y) \quad (27)$$

となる。式(23), (26), (27)から U, Z を消去すると、

$$Y=PC(Y_T-Y)+PW+QV \quad (28)$$

となり、もし $(I+PC)$ の逆行列 $(I+PC)^{-1}$ が存在すれば、

$$Y=(I+PC)^{-1}\{PCY_T+PW+QV\} \quad (29)$$

が得られる。式(23), (26), (27)で表される制御系を k 組コピーしたとき、 i 番めのモデルは、組の番号を変数の右肩に $(\)$ 付きで表すと、

$$Y^{(i)}=PU^{(i)}+QV^{(i)} \quad (30)$$

$$U^{(i)}=W^{(i)}+Z^{(i)} \quad (31)$$

$$Z^{(i)}=C(Y_T^{(i)}-Y^{(i)}) \quad (32)$$

となり、 i 番めのモデルの先行動作 $W^{(i)}$ に $i-1$ 番めのモデル操作量 $U^{(i-1)}$ を与えるとする、

$$W^{(i)}=U^{(i-1)} \quad (33)$$

$$U^{(i)}=W^{(i-1)}+Z^{(i-1)}+Z^{(i)} \quad (34)$$

を得る。これらの式から $U^{(i)}$, $W^{(i)}$, $Z^{(i)}$, $U^{(i-1)}$, $W^{(i-1)}$, $Z^{(i-1)}$ を消去すると、漸化式

$$Y^{(i)}-Y^{(i-1)}=PC(Y_T^{(i)}-Y^{(i)})+Q(V^{(i)}-V^{(i-1)}) \quad (i=1, 2, \dots, k) \quad (35)$$

が得られる。並列学習されたすべてのモデルに対し、同じ目標ベクトル Y_T , 同じ外乱ベクトル V が与えられるなら、 $Y^{(k)}$ について解き

$$Y^{(k)}=\{I-(I+PC)^{-k}\}Y_T+(I+PC)^{-k}QV \quad (36)$$

を得る。これはまた

$$Y^{(k)}=\{I+(I+PC)+(I+PC)^2+\dots+(I+PC)^{k-1}\}PC(Y_T-Y^{(k)})+QV \quad (37)$$

とも書ける。伝達関数行列

$$I+(I+PC)+(I+PC)^2+\dots+(I+PC)^{k-1}$$

を重根フィルタと名付けた。

4.2 ナイキストの安定判別

(1) 並列学習制御系のナイキスト軌跡 並列学習による重根フィルタでは、式(28)と(37)を比較して、 PC の代わりに重根フィルタの一巡伝達関数行列

$$[PC]_M=\{I+(I+PC)+(I+PC)^2+\dots+(I+PC)^{k-1}\}PC \quad (38)$$

のナイキスト軌跡を描けばよいことがわかる。前報の「制御系の並列学習(重根フィルタの提案)」の過渡応答計算で取り上げた次の例題

$$A=\begin{bmatrix} -3 & 1 & 0 \\ 2 & -3 & 1 \\ 0 & 2 & -3 \end{bmatrix}, B=\begin{bmatrix} -2 & 0 & 0 \\ 4 & -2 & 0 \\ 0 & 4 & -2 \end{bmatrix}, G=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, H=\begin{bmatrix} -2 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -2 \end{bmatrix}$$

について、操作ベクトル U に対する、出力ベクトル Y の伝達関数行列 P を求めると、

$$\begin{bmatrix} \frac{-2(s^2+4s+1)}{(s+1)(s+3)(s+5)} & \frac{-2}{(s+3)(s+5)} & \frac{-2}{(s+1)(s+3)(s+5)} \\ \frac{4(s+2)}{(s+1)(s+5)} & \frac{-2}{(s+5)} & \frac{-2}{(s+1)(s+5)} \\ \frac{8(s+2)}{(s+1)(s+3)(s+5)} & \frac{4(s+4)}{(s+3)(s+5)} & \frac{-2(s^2+6s+7)}{(s+1)(s+3)(s+5)} \end{bmatrix}$$

となる。

伝達関数行列 P の行列要素のなかで、 $P(2, 2)$ は 1 次遅れ、 $P(1, 2)$ 、 $P(2, 3)$ は 2 次遅れ、 $P(1, 3)$ は 3 次遅れ、 $P(1, 1)$ 、 $P(2, 1)$ 、 $P(3, 2)$ 、 $P(3, 3)$ は実質的には 1 次遅れ、 $P(3, 1)$ は実質的には 2 次遅れである。そこで、最も遅れの大きい 3 次遅れの $P(1, 3)$ を取り上げ、つまり Y_1 を U_3 で制御する問題で並列学習の安定性を調べる。したがって制御伝達関数 C は $C(3, 1)$ のみで、 $C(3, 1) = -c$ とすると、一巡伝達関数 PC は

$$PC = \begin{bmatrix} \frac{2c}{(s+1)(s+3)(s+5)} & 0 & 0 \\ \frac{2c}{(s+1)(s+5)} & 0 & 0 \\ \frac{2(s^2+6s+7)c}{(s+1)(s+3)(s+5)} & 0 & 0 \end{bmatrix} \quad \text{..... (39)}$$

となる。図 3 と 4 に、式 (38) で表された並列学習の一巡伝達関数行列 $[PC]_M$ のナイキスト線図を示す。学習の回数を 0, 1, 2, 3, ..., 14 回としたときのナイキスト軌跡をまとめて描いている。

図 3 とその拡大図 4 から、 $\omega=0$ すなわちナイキスト軌跡の出発点では、大きな半径の軌道を通るが、周波数 ω が大きくなって点 $(-1, 0)$ に近づくと急に右へ曲がり、点 $(-1, 0)$ の右側を通過することがわかる。この特徴をより明確に見るために、 c の値を $3/2$

図 3 並列学習のナイキスト線図

図 5 並列学習のナイキスト線図(拡大図-2)

図 4 並列学習のナイキスト線図(拡大図-1)

図 6 並列学習の過渡応答 (y_1 , u_3)

図 7 比例制御のナイキスト線図

図 8 比例制御のナイキスト線図(拡大図)

から 3 に変えて 14 回学習したときのナイキスト線図(拡大図-2)を図 5 に示した。参考のため、同じ伝達関数をもつ並列学習系の過渡応答シミュレーション結果を図 6 に示す。初期条件は $y_i=0$, $u_i=0$ ($i=1, 2, 3$)、入力は外乱 v_1 で、 $0 \leq t \leq 1$ で $v_1=0$, $t > 1$ で $v_1=1$ とした。

(2) フィードバック制御系のナイキスト軌跡

並列学習しないフィードバック制御系のナイキストのベクトル軌跡は、式(39)の一巡伝達関数 PC により、 c の値を変更して描いた。結果は図 7 およびその $(-1, 0)$ 近傍での拡大図 8 に示すようになる。

ゲイン c は、並列学習のときと同じ定常偏差が得られるように、並列学習の回数 $i=0, 1, 2, 3, 4, 5, \dots, 10, 11, 12, 13, 14$ に対応させ、 $c=1.50, 3.30, 5.46, 8.05, 11.2, 14.9, 19, \dots, 48, 59, 73, 89, 108$ と変えて計算している。拡大図 8 から、 $c > 89$ のとき不安定となることがわかる。対応する過渡応答シミュレーション結果を

図 9 比例制御の過渡応答 (y_1, u_3)

図 9 に示す。入力は、並列学習のときと同じ外乱 v_1 で、 $0 \leq t \leq 1$ のとき $v_1=0$, $t > 1$ のとき $v_1=1$ とした。

5. ま と め

本論文の結果は次のように要約される。

(1) 並列学習の系は、固有ベクトルによって作られる変換行列でジョルダン標準形になり、固有値が変わらず重複するだけなので、制御系は不安定とはならない。

(2) 固有値が変わらなくても、位相の進んだ速いモードが現れる。これが D 動作の機能として働き、安定化に貢献している。

(3) 学習を繰返すたびに遅いモードが現れ、定常ゲインが増え続ける I 動作の機能をもつ。

(4) フィードバック制御系の操作部を k 回並列学習することは、特性方程式の左辺 $(I+PC)$ を i 乗 ($i=0, 1, 2, \dots, k-1$) したものを合計して、制御対象伝達関数 P と制御伝達関数 C に直列に挿入したことと同等であるが、この直列補償によって、一巡伝達関数のナイキスト軌跡は常に点 $(-1, 0)$ を反時計方向に囲むように通過する。

(5) 点 $(-1, 0)$ の近傍に近づくと、並列学習では急速に、右側にカーブして、この点を回避する。

(6) 点 $(-1, 0)$ までの ω では、ナイキスト軌跡の半径は大きく、並列学習と比例制御でゲインを大きくしたのと差はない。このため制御偏差が減少する。

文 献

- (1) 松村・白坂・鈴木, 計装, 35-10(1992), 35.
- (2) 相楽・戸田, 火力原子力発電, 24-12(1973), 1405.
- (3) 相楽・山中・佐藤, 三菱重工技報, 17-2(1980), 268.
- (4) 有本, 機誌, 93-856(1990), 180
- (5) 黄・藤堂, 機論, 58-547, C(1992), 851.
- (6) 高橋・田島, 電学論, 112-3, C(1993), 201.