

状態推定理論による伝達関数の推定*

川 添 容 宏**

Estimation of Transfer Function Using State Estimation Theory

Yasuhiro KAWAZOE

Abstract

This paper considers estimation of transfer function in control systems. We will form discrete-type models of state equations and apply second order filter to its parameter estimation with the aid of the digital computer.

Transfer function will first be transformed into state differential equations, which are converted to the discrete forms. Parameters in discrete-type state equations and inner real states constitute a new combined vector.

We will adopt the second order filter to estimate the new vector involving unknown parameters because the resulting equation with the new vector is nonlinear. As numerical examples, estimation of the second order transfer function will be shown.

1. 序 論

ここ十数年の間制御理論の進歩は実に著しいものがあり、そのほとんどが緻密なる数学モデルによって展開されていることは周知のとおりである。したがって制御を行なおうとする場合、いかなる数学モデルを基にして考えるかが重要な問題となる。たとえば、現代制御理論においては最適制御を達成させるためには系の内部状態に至るまでの詳細な数学モデルが必要とされるが、古典制御理論の多くは入出力関係を表わす伝達関数さえわかっているならば古典制御理論の範囲で十分な制御がなされる場合がほとんどである。系の内部状態まで考慮した数学モデルは状態方程式で記述されるが、それに含まれるパラメータの推定問題はカルマンフィルタ、非線形フィルタの拡張応用によって状態値と同時に逐次推定する方法として多くの研究がなされている^{1)~4)}。

本稿では計算機の使用を前提として非線形フィルタ理論の応用による伝達関数の逐次推定アルゴリズムを導びく。伝達関数の推定法としては従来から解析法と試行法の2つの流れで研究がなされており⁵⁾、最近になって状態方程式中のパラメータ推定から伝達関数の分母、分子のパラメータを推定する方法が試みられて

いる⁶⁾。この方法は伝達関数モデルを状態方程式の標準形式に変換してパラメータを推定しようとするものである。本稿ではこれと違った2つの方法で伝達関数から連続形式の状態方程式を求め、さらにこれを離散値形式に変換し計算機の使用を前提とした未知パラメータの推定を状態値との同時推定によって非線形フィルタを適用することにより達成している。本稿の特長は状態方程式の連続形式から離散値形式への変換に適当な近似式を用いて実際面への非線形フィルタの適用を容易ならしめ、さらに伝達関数の係数パラメータだけでなく、制御対象の次元数などのないいわゆる形式が未知なる場合にも、伝達関数の分母、分子の次数をあらかじめ大きくとっておきパラメータの推定値の様子によってある程度の推定を可能としている点にある。

2. 問題の記述

一般に一入力一出力の n 次元制御対象の入出力関係は次のような伝達関数で表わすことができる。

$$G(s) = \frac{\beta_{n-1}S^{n-1} + \beta_{n-2}S^{n-2} + \dots + \beta_1S + \beta_0}{S^n + \alpha_{n-1}S^{n-1} + \dots + \alpha_1S + \alpha_0} \quad (1)$$

伝達関数の同定問題の場合、多くは系の次元数 n は与えられており未知パラメータ $\alpha_i, \beta_i, i=0, 1, 2, \dots, n-1$ の値を推定して伝達関数を同定してい

* 昭和45年11月21日電気四学会中国連大にて講演

** 工業短期大学部電気工学科

る。本稿では系の次元数 n は一般には与えられていなくてもよい同定法について考察する。ただし、 n の値の推定を解析的に行なうのではなく未知パラメータ $\alpha_i, \beta_i, i=0, 1, 2, \dots, n-1$ の推定値の収束の状況に応じて系の次元数 n を決めようとするのである。ここで、系への入力とは既知関数であるとし、系の出力を観測することによって伝達関数(1)式中に含まれている未知パラメータを推定するアルゴリズムを導出することが本稿の問題の設定である。

3. 伝達関数から状態方程式への変換

一般に状態方程式から伝達関数は一意的に決定されるが、逆に伝達関数から状態方程式を求めることは、内部状態の取り方いかんによって一意的には定まらない。ここでは、伝達関数のパラメータ推定を状態方程式中のパラメータ推定によって行なうことを考慮した、伝達関数から状態方程式への二つの変換法を示し、状態方程式を離散値形式にて得る。

3.1 直接プログラミング法による状態方程式への変換

制御対象への入力を $u(t)$ 、出力を $y(t)$ とし、そのラプラス変換表示をそれぞれ $U(s), Y(s)$ とすれば伝達関数 $G(s)$ は(1)式より、

$$G(s) = \frac{Y(s)}{U(s)} = \frac{\beta_{n-1}S^{-1} + \beta_{n-2}S^{-2} + \dots + \beta_1S^{-(n-1)} + \beta_0S^{-n}}{1 + \alpha_{n-1}S^{-1} + \dots + \alpha_1S^{-(n-1)} + \alpha_0S^{-n}} \quad (2)$$

となる。ここで、

$$E(s) = \frac{U(s)}{1 + \alpha_{n-1}S^{-1} + \dots + \alpha_1S^{-(n-1)} + \alpha_0S^{-n}} \quad (3)$$

とおけば、(2)式より

$$Y(s) = (\beta_{n-1}S^{-1} + \beta_{n-2}S^{-2} + \dots + \beta_1S^{-(n-1)} + \beta_0S^{-n})E(s) \quad (4)$$

をえ、また(3)式は

$$E(s) = U(s) - (\alpha_{n-1}S^{-1} + \alpha_{n-2}S^{-2} + \dots + \alpha_1S^{-(n-1)} + \alpha_0S^{-n})E(s) \quad (5)$$

となる。そこで、(4)、(5)両式から状態変数線図を描けばFig. 1 のようになる。

Fig. 1 で定義された状態値から次の式が成り立つ。

$$\dot{x} = A_D x + B_D u \quad (6)$$

$$y = C_D x \quad (7)$$

ここで、

$$A_D = \begin{pmatrix} \vdots & & & & \\ & \mathbf{0} & & & \\ & & \mathbf{I} & & \\ & & & \ddots & \\ & -\alpha_0 & -\alpha_1 & \dots & -\alpha_{n-2} & -\alpha_{n-1} \end{pmatrix} \quad (8)$$

$$B_D = [0, \dots, 0, 1]' \quad (9)$$

$$C_D = [\beta_0, \beta_1, \dots, \beta_{n-1}]' \quad (10)$$

であり \mathbf{I} は単位行列である。

連続形式の状態方程式から離散値形式の状態方程式を得るには厳密には文献7)の手法によればよいが、この方法は、たとえば駆動行列を得るのに行列 A_D の逆行列の計算が必要になるなどの煩わしさがあり、ここではサンプリング周期を小さくとることができるという仮定の下で次の近似離散化の手法をとる。

$$\dot{x} = \frac{x_{k+1} - x_k}{T} \quad (11)$$

ここで k は任意のサンプリング時点を表わし、 T はサンプリング周期とする。(11)式を用いて(6)、(7)式を離散値形式で表わせば

$$x_{k+1} = \Phi_D x_k + G_D u_k \quad (12)$$

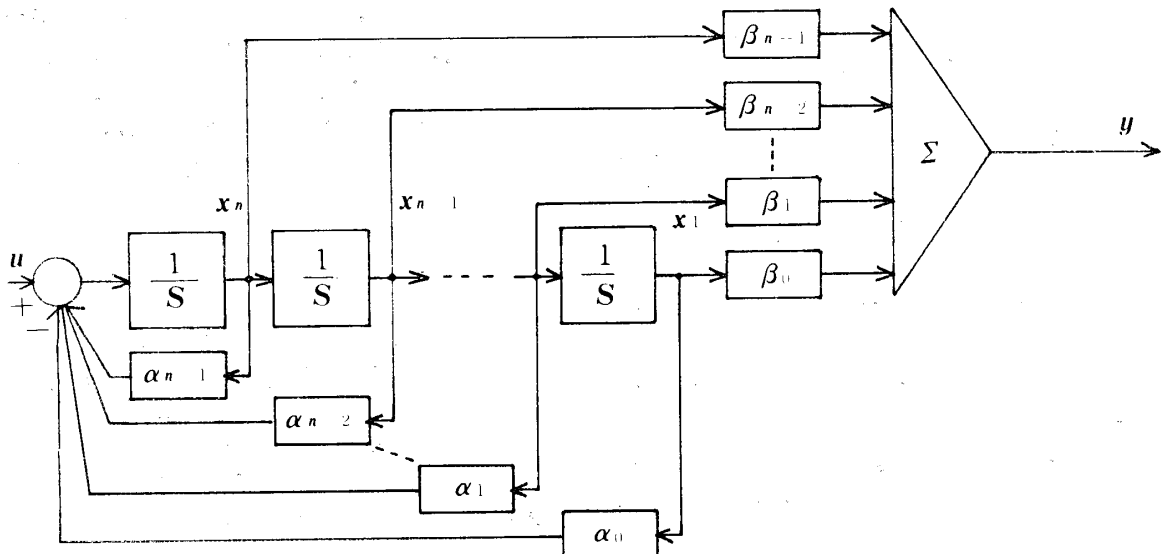


Fig.1 State-variable diagram for direct programming

$$y_k = C_D x_k \quad (13)$$

となる。ただし、

$$\Phi_D = \begin{bmatrix} 1 & T & & & 0 \\ & 1 & T & & \\ & & \ddots & \ddots & \\ & & & 1 & T \\ -\alpha_0 T & -\alpha_1 T & \dots & -\alpha_{n-2} T & 1 - \alpha_{n-1} T \end{bmatrix} \quad (14)$$

$$G_D = [0, \dots, 0, T] \quad (15)$$

とする。

3.2 繰り返しプログラミング法による状態方程式への変換

繰り返しプログラミング法は伝達関数の分母、分子を因数分解しその各項によってできる部分伝達関数を縦続に接続して状態変数線図を得る方法である。伝達関数に対する先験情報の程度により次の二つに分けて考える。

3.2.1 伝達関数の極、零点の数が未知のとき

伝達関数を次のような形に分解して考える。

$$G(s) = \frac{(b_1 s + c_1)(b_2 s + c_2) \dots (b_{n-1} s + c_{n-1})}{(s + a_1)(s + a_2) \dots (s + a_n)} \quad (16)$$

ここで、 $a_i, i = 1, 2, \dots, n, b_i, c_i, i = 1, 2, \dots, n-1$ はすべて実数であると仮定する。 $-a_i, -c_i/b_i$ はその推定値によって必ずしもそれぞれ極、零点を表わすとは限らないことに注意を要する。いま、

$$\left. \begin{aligned} G_1(s) &= \frac{b_1 s + c_1}{s + a_1} \\ &\dots \dots \dots \\ G_{n-1}(s) &= \frac{b_{n-1} s + c_{n-1}}{s + a_{n-1}} \\ G_n(s) &= \frac{1}{s + a_n} \end{aligned} \right\} \quad (17)$$

とすれば、

$$G(s) = \prod_{i=1}^n G_i(s) \quad (18)$$

となり、 $G_i(s), i = 1, 2, \dots, n$ の状態変数線図は 3.1 の手法によりただちに描かれるから、結局 $G(s)$ の状態変数線図はそれらの繰り返し使用によって Fig.2 のようになる。

ここで、状態値は次のように表わすことができる。

$$\left. \begin{aligned} \dot{x}_n &= u - a_n x_n \\ \dot{x}_{n-1} &= x_n - a_{n-1} x_{n-1} \\ \dot{x}_{n-i} &= \prod_{j=1}^{i-1} b_{n-j} x_n - \sum_{t=1}^{i-2} \prod_{j=t+1}^{i-1} b_{n-j} (b_{n-t} a_{n-t} - c_{n-t}) x_{n-t} - (b_{n-i-1} a_{n-i-1} - c_{n-i-1}) x_{n-i-1} - a_{n-i} x_{n-i} \\ & \quad i = 2, 3, \dots, n-1 \\ y &= b_1 \left\{ \prod_{j=1}^{n-2} b_{n-j} x_n - \sum_{t=1}^{n-3} \prod_{j=t+1}^{n-2} b_{n-j} (b_{n-t} a_{n-t} - c_{n-t}) x_{n-t} - (b_2 a_2 - c_2) x_2 \right\} - (a_1 b_1 + c_1) x_1 \end{aligned} \right\} \quad (19)$$

(19), (20) 式をベクトル表示すれば次のような形になる。

$$\dot{x} = A_I x + B_D u \quad (21)$$

$$y = C_I x \quad (22)$$

ここで、行列 A_I, C_I は(19), (20)両式からわかるように非常に複雑となるので詳細なる記述は省略する。(21), (22)式は(11)式によって次のように離散化される。

$$x_{k+1} = \Phi_I x_k + G_D u \quad (23)$$

$$y = C_I x \quad (24)$$

ここで、前と同様 Φ_I, C_I の詳細なる記述は省略する。

3.2.2 伝達関数の極、零点の数が既知のとき

前の 3.2.1 の方法は非常に複雑な状態方程式となり実際のパラメータ推定を非線形フィルタ適用の面から難しくしている。そこで次に先験情報の程度が幾分よくなった場合について考察する。すなわち極、零点の数がそれぞれ n, m として与えられた場合を考える。このとき伝達関数は次のように表わすことができる。

$$G(s) = \frac{K(S+b_1)(S+b_2) \dots (S+b_m)}{(S+a_1)(S+a_2) \dots (S+a_n)}, \quad m \leq n \quad (25)$$

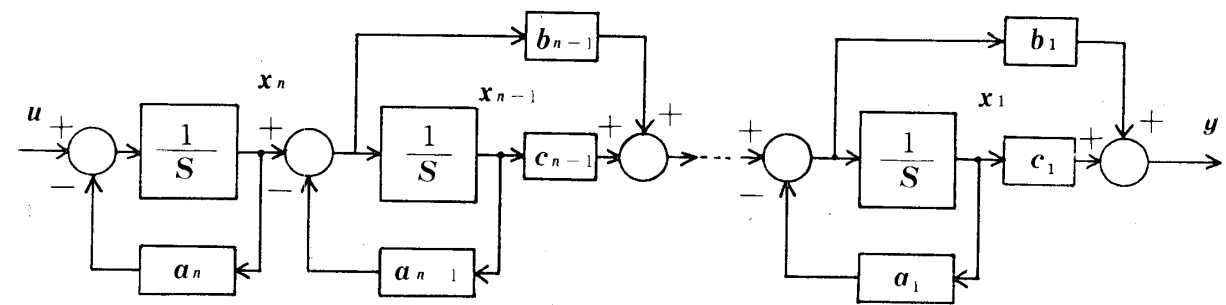


Fig.2 State-variable diagram for iterative programming in 3.2.1

ここで、 x は内部状態ベクトルである。結合ベクトル X で状態方程式、観測方程式を書き換えると次のような X についての非線形形式を得ることができる。

$$X_{k+1} = f(X_k) \tag{42}$$

$$y_k = h(X_k) \tag{43}$$

ここで、 f, h は X_k についての非線形関数である。なお、実際の制御系では状態付加雑音、観測誤差雑音の影響を無視できないから、このことを考慮すると上式は、

$$X_{k+1} = f(X_k) + v_k \tag{44}$$

$$y_k = h(X_k) + w_k \tag{45}$$

となる。ほとんどの場合、 v_k, w_k は近似的に正規性白色雑音とみなすことができ、(44)、(45)式に2次近似フィルタを適用することができ X_k の推定値を得ることができる。付録より推定値の逐次式は

$$\hat{X}_{k+1/k+1} = \hat{X}_{k+1/k} + K_{k+1} [y_{k+1} - h(\hat{X}_{k+1/k}) - \pi(X_{k+1/k})] \tag{46}$$

となる。

5. 例題 2次元伝達関数の推定

次式で表わされる2次元の伝達関数の推定を2次近似フィルタを用いて行なう。

$$G(s) = \frac{3s + 6}{s^2 + 5s + 4} = \frac{3(s + 2)}{(s + 1)(s + 4)} \tag{47}$$

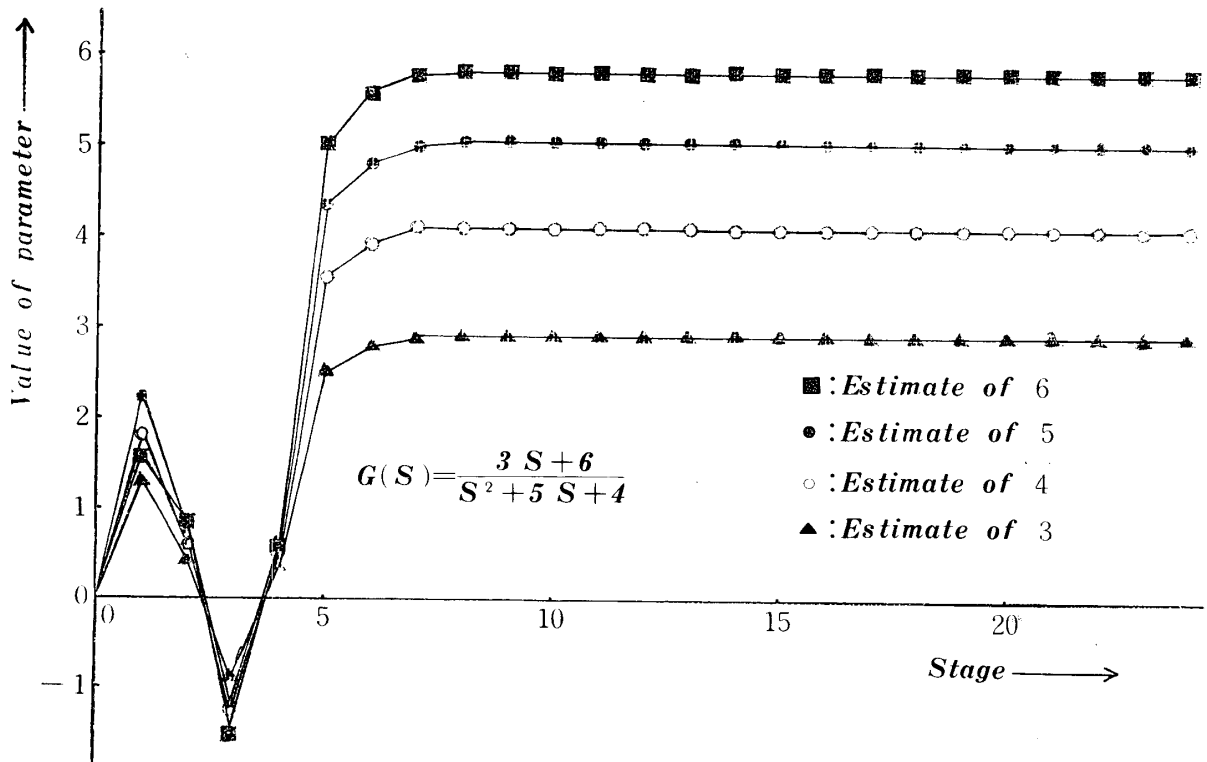


Fig.4 An example of transfer function (47) for direct programming

5.1 直接プログラミング法による場合

伝達関数の分母、分子の係数である 4, 5, 6, 3, 0 を未知係数であるとし、次のような条件下で推定計算を行なう。

初期条件 (真値) $X_0 = [1, 1, 4, 5, 6, 3, 0]^T$

先験予測値 $\hat{X}_{-1/0} = [0, 0, 0, 0, 0, 0, 0]^T$

先験予測誤差分散 (対角項)

$diag. C_{-1/0} = [0.5, 0.1, 2.0, 3.0, 4.0, 1.0, 0]$

先験予測誤差分散 (非対角項)

$$C_{-1/0}^{ij} = (C_{-1/0}^{ii} \cdot C_{-1/0}^{jj})^{1/2}$$

観測付加雑音 平均値 0, 分散 0.01 の正規性白色雑音

伝達関数のパラメータ推定の結果は Fig. 4 に示す。

5.2 繰り返しプログラミング法による場合

3.2.2 の手法による計算例を示す。極 -1, -4, 零点 -2 を未知パラメータとして、次のような条件下で推定計算を実施する。

初期条件 (真値) $X_0 = [1, 1, 1, 4, 2, 3]^T$

先験予測値 $\hat{X}_{-1/0} = [0, 0, 0, 0, 0, 0]^T$

先験予測誤差分散 (対角項)

$diag. C_{-1/0} = [0.5, 0.1, 0.1, 2.0, 0.5, 1.0]$

先験予測誤差分散 (非対角項)

$$C_{-1/0}^{ij} = (C_{-1/0}^{ii} \cdot C_{-1/0}^{jj})^{1/2}$$

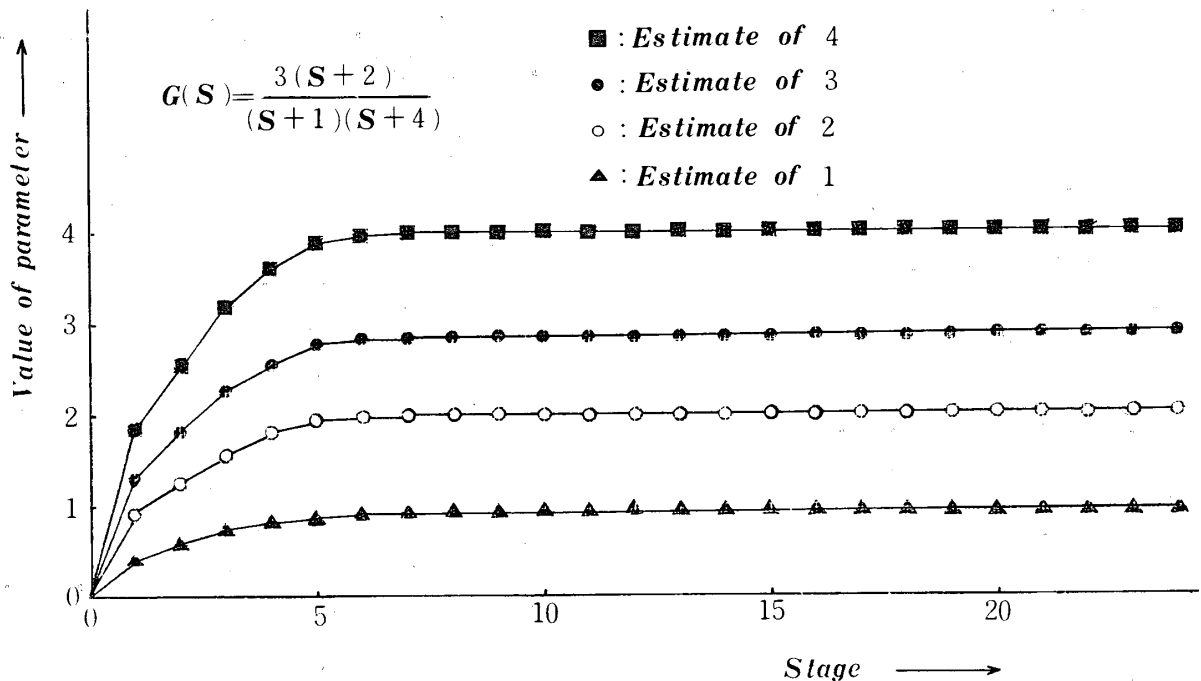


Fig.5 An example of parameter estimation of transfer function (47) for iterative programming.

観測付加雑音 平均値 0, 分散 0.01 の正規性白色雑音

Fig. 5 に推定結果を示す。

6. 検討ならびに結果

直接プログラミング法による伝達関数から状態方程式への変換ではパラメータを含めた結合系は 2 次の非線形性を持った非線形系となることから、非線形フィルタの適用は容易になる。ところが繰り返しプログラミング法によって求められた結合系は(19), (20)式からもわかるように非常に高次の非線形性を持っているため考える伝達関数の次元が大きくなるにつれ実際的な非線形フィルタの適用が困難となってくる。ただし、3.2.2 のような場合には高々 2 次の非線形性まで考慮すればよいことになる。しかしながらこの場合は先験情報としてより多くのもを期待しなければならない。いずれにしても繰り返しプログラミングによる方法は極、零点などが実数でなければならないという大きな制限が存在するため、直接プログラミングによる方法に比べて一般の伝達関数推定の場合実用的でない。

伝達関数の次元数が未知の場合、本稿の手法では最初、考えられ次元数よりも大きく見積っておいて未知係数の推定値の収束の様子によって、その次元数を決定すればよい。もっとも、次元数の高い伝達関数の場合でも実際面への適用に当っては計算上の都合などに

より、低次のそれで近似して用いるのが普通であるので最初から伝達関数の次元数を大きく見積っておくことは計算機利用の面からは好ましいことではない。パラメータ推定の良否は使用する非線形フィルタによる場所が大である。非線形フィルタについては現在非常に活発に研究がなされているが³⁾、高次の非線形系に至るまで良好な推定を得ることのできる実用的なフィルタはない。非線形フィルタの一般理論は連続系の場合、Kushner によって確率微分方程式論の立場より無限次元のモーメントに至るまで考慮しなければならないことが導びかれているが³⁾、実際面への適用については近似的に有限次元のモーメントで打切って計算を行わなければならない。計算容量などを考慮すれば 2 次のモーメント位で打切らなければならなくなってくるので、2 次近似フィルタとほとんど同じとなる。低次の非線形系については線形化カルマンフィルタなどが用いられるが推定値の収束性はよくない。このような中で、2 次近似フィルタは現段階ではかなりの実用性を有していると思われる。そこで、本稿では非線形フィルタに 2 次近似フィルタを適用している。5. の例題の数値例の結果は Fig. 4, Fig. 5 に示されるとおりである。先験予測値の誤差分散の良否によって推定値の収束はかなり変化し、はなはだしいときには推定値の収束が得られないことがあり、さらに非線形フィルタの改良が実用的なる面で望まれるのであ

る。いまのところ誤差分散の設定に注意しなければよい推定は得られない。例題の条件によると5.1の場合7段、5.2の場合6段にて各パラメータとも推定が良好に行なわれている。前者の結合系の次元が7、後者のそれが6であることを考えれば状態ベクトルの要素数に等しい段数にてほぼ推定がなされていることになる。

7. む す び

以上、伝達関数推定に状態推定のフィルタ理論を適用する方法について考察し、数値例によって本方法の妥当性を示した。直接プログラミングによる方法であればその使用に制限がなく、また結合系の非線形性も低次であるので実用的である。しかし、最終的には推定の精度は適用する非線形フィルタに左右されるのでより活用度の高い非線形フィルタの開発が望まれる。

参 考 文 献

- 1) T. Fukao: Bulletin of the Electrotechnical Laboratory, 29-5, 16 (1965)
- 2) 辻 節三, 裕百合雄: 第10回自動制御連合講演会前刷, 128 (1967)
- 3) 辻 節三, 川添容宏: 第12回自動制御連合講演会前刷, 151 (1969)
- 4) 辻 節三, 川添容宏: 九州大学工学集報, 43-4, 462 (1970)
- 5) 鈴木 胖: 制御工学, 14-10, 617 (1970)
- 6) R. C. K. Lee: Optimal Estimation, Identification, and Control, M. I. T. Press (1964) P.117
- 7) 高橋安人: システムと制御, 岩波 (1968) p.160
- 8) A. H. Jazwinski: Stochastic Processes and Filtering Theory, Academic Press (1970) p.162
- 9) H. J. Kushner: IEEE Trans. AC-12, 546 (1967)

付録 2次近似フィルタアルゴリズム

n 次元ベクトル状態方程式, m 次元ベクトル観測方程式がそれぞれ,

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k) + \mathbf{v}_k$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{w}_k$$

\mathbf{v}_k : n 次元正規性白色雑音ベクトル

平均値 0, 分散 V_k

\mathbf{w}_k : m 次元正規性白色雑音ベクトル

平均値 0, 分散 W_k

で表わされる場合の状態値 \mathbf{x} の2次近似フィルタによる逐次推定アルゴリズムは次式となる。

$$\hat{\mathbf{x}}_{k+1/k+1} = \hat{\mathbf{x}}_{k+1/k} + K_{k+1} [\mathbf{y}_{k+1} - \mathbf{h}(\hat{\mathbf{x}}_{k+1/k}) - \pi(\hat{\mathbf{x}}_{k+1/k})]$$

$$\hat{\mathbf{x}}_{k+1/k} = \mathbf{f}(\hat{\mathbf{x}}_{k/k}) + \boldsymbol{\mu}(\hat{\mathbf{x}}_{k/k})$$

$$\boldsymbol{\mu}(\hat{\mathbf{x}}_{k/k}) = \frac{1}{2} \sum_{i=1}^n \phi_i \text{tr} [f_{xx}^i(\hat{\mathbf{x}}_{k/k}) C_{k/k}]$$

$$\boldsymbol{\pi}(\hat{\mathbf{x}}_{k+1/k}) = \frac{1}{2} \sum_{i=1}^m \psi_i \text{tr} [h_{xx}^i(\hat{\mathbf{x}}_{k+1/k}) C_{k+1/k}]$$

$$K_{k+1} = C_{k+1/k} h_{x'}'(\hat{\mathbf{x}}_{k+1/k}) [h_x(\hat{\mathbf{x}}_{k+1/k}) C_{k+1/k} h_x(\hat{\mathbf{x}}_{k+1/k}) + W_{k+1} + L_{k+1}]^{-1}$$

$$C_{k+1/k} = f_x(\hat{\mathbf{x}}_{k/k}) C_{k/k} f_x'(\hat{\mathbf{x}}_{k/k}) + D_k + V_k$$

$$(D_k)^{ij} = \frac{1}{2} \text{tr} [f_{xx}^i(\hat{\mathbf{x}}_{k/k}) C_{k/k} f_{xx}^j(\hat{\mathbf{x}}_{k/k}) C_{k/k}]$$

$$C_{k+1/k+1} = C_{k+1/k} - K_{k+1} h_x(\hat{\mathbf{x}}_{k+1/k}) C_{k+1/k}$$

$$(L_{k+1})^{ij} = \frac{1}{2} \text{tr} [h^{i_{xx}}(\hat{\mathbf{x}}_{k+1/k}) C_{k+1/k} h^{j_{xx}}(\hat{\mathbf{x}}_{k+1/k}) C_{k+1/k}]$$

ここで,

$$\phi_i = [0, \dots, 0, 1, 0, \dots, 0]'$$

$$\begin{matrix} \vdots \\ \text{No. } i \\ \vdots \end{matrix}$$

$$\psi_i = [0, \dots, 0, 1, 0, \dots, 0]'$$

$$\begin{matrix} \vdots \\ \text{No. } i \\ \vdots \end{matrix}$$

$$\left. \begin{aligned} (f_x(\hat{\mathbf{x}}))^{ij} &= \frac{\partial f_i}{\partial x^j} \Big|_{\mathbf{x}=\hat{\mathbf{x}}} \\ (f^{i_{xx}}(\hat{\mathbf{x}}))^{jk} &= \frac{\partial^2 f_i}{\partial x^j \partial x^k} \Big|_{\mathbf{x}=\hat{\mathbf{x}}} \end{aligned} \right\} i, j, k=1, 2, \dots, n$$

$$\left. \begin{aligned} (h_x(\hat{\mathbf{x}}))^{ij} &= \frac{\partial h^i}{\partial x^j} \Big|_{\mathbf{x}=\hat{\mathbf{x}}} \\ (h^{i_{xx}}(\hat{\mathbf{x}}))^{jk} &= \frac{\partial^2 h^i}{\partial x^j \partial x^k} \Big|_{\mathbf{x}=\hat{\mathbf{x}}} \end{aligned} \right\} \begin{matrix} i=1, 2, \dots, m \\ j, k=1, 2, \dots, n \end{matrix}$$

である。また、上付添字はベクトルあるいは行列の要素を示し、 tr は行列の対角項の和を表わす。

(昭和46年6月3日受理)