広島大学大学院教育学研究科紀要　第二部　第53号　2004　61－67

シミュレーションを利用した数値計算法の理解と応用

向谷博明

(2004年9月16日受理)

Understanding of Numerical Computation via Simulation and Its Application

Hiroaki Mukaidani

The feature of the basic theorems that are related to the numerical computation and some examples of how to use them are introduced through the simulation. The new iterative techniques for solving the algebraic Riccati equation and algebraic Lyapunov equation are used for weakly-coupled systems as an application. It is shown that the free numerical application tool which is called Octave is very powerful tool and it is available for the understanding of the numerical computation for undergraduate students in the early learning stage.

Key Words : Algebraic equation, Numerical computation, Iterative·techniques, Octave
キーワード : 代数方程式, 数値計算, 再帰的法, オクターブ

1　はじめに

シミュレーションは, さまざまな数理工学の諸問題に対し, 解を得るための手段として必要不可欠なプロセスである. その課程でコンピュータを用いることが現在では必須の手法である. コンピュータは, アルゴリズムを論理的に誤りなく記述することによってはじめて動作する. しかしながら, 同じ結果を得るにも使用するアルゴリズムによって計算時間, メモリ領域, 計算精度に大きな違いをもたらす. したがって, シミュレーションで使用するアルゴリズムの決定には細心の注意と経験が必要となる.

通常, 数値計算法の講義では, 連立一次方程式, 数値積分, 微分方程式, 補間法等の項目が良く扱われており, シミュレーションに基づくプログラム演習を含んでいるのが一般的である. しかしながら, 初学者に対して, プログラム作成は困難を伴いがちである. さらには, シミュレーションで利用するプログラムから得られる結果の正当性を検証する方法は, 手計算によるものや教科書から得られる解答等を参考にすることが多く, 一般的な問題に対する解答を用意するには何らかのベンチマーク的存在のソフトウェアによる検証が効果的と考える.

本研究では, まず, 数値計算法の理解の題材として, 弱結合システムにおける最適制御問題を取り上げ, 行列代数方程式である Riccati 方程式及び Lyapunov 方程式 [1, 2, 3] を解くための反復アルゴリズムを新たに提案する. 提案されたアルゴリズムは, 陰関数定理, Newton-Kantorovich 定理, 不動点定理の3つの定理によって解の存在性や収束性が保証される. 続いて, 数値計算法を理解するために, シミュレーションを通して実際の問題に提案されたアルゴリズムを応用する. 最後に, Octave[18] と呼ばれる無料数値計算ソフトウェアが, 学部生に対して学習の初期段階で有用なツールであることが実践報告で示される.

本研究では, 以下の記号を利用する. $\mathcal{S}^T$ は行列 $\mathcal{S}$ の転置, **block diag** はブロック対角行列, $I_n \in \mathbf{R}^{n\times n}$ は n 次の単位行列, vec$\mathcal{S}$ は行列 $\mathcal{S}$ の列ベクトル化をそれぞれ表す [16]. また, U_{lm} は行列 $\mathcal{S} \in \mathbf{R}^{l\times m}$ に対して $U_{lm}\mathrm{vec}\mathcal{S} = \mathrm{vec}\mathcal{S}^T$ を満足する置換行列を表す [16].

2　弱結合システム

以下の弱結合システムを考える [7, 8, 9].

$$\dot{z}_1 = A_1 z_1 + \varepsilon A_{12} z_2 + B_1 u_1 + \varepsilon B_{12} u_2 \quad (1a)$$

$$\dot{z}_2 = \varepsilon A_{21} z_1 + A_2 z_2 + \varepsilon B_{21} u_1 + B_2 u_2 \quad (1b)$$

ここで, 初期値は $z_j(0) = z_j^0$, $(j = 1, 2)$ で与えられる. また, $z_j \in \mathbf{R}^{n_j}$ は状態ベクトル, $u_j \in \mathbf{R}^{m_j}$, $(j = 1, 2)$ は制御入力をそれぞれあらわす. 一方, ε は他のサブシステムの結合度合いを表現する摂動項に相当する十分小さな正

のパラメータである．弱結合システムにおける最適制御問題は，弱結合システム (1) の拘束のもと，評価関数 (2) を最小にする制御入力 $u = u^*$ を決定することである．

$$\mathcal{J} = \frac{1}{2}\int_0^\infty [z^T Q z + u^T R u]dt \tag{2}$$

ただし，

$$z := \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \in \mathbf{R}^n,\ u := \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \in \mathbf{R}^m$$
$$n := n_1 + n_2,\ m := m_1 + m_2$$
$$Q := \begin{bmatrix} Q_1 & \varepsilon Q_{12} \\ \varepsilon Q_{12}^T & Q_2 \end{bmatrix},\ Q_j = Q_j^T \geq 0,$$
$$(j = 1,\ 2),\ R = R^T > 0$$

このとき，弱結合システム (1) に対して，以下の仮定を導入する．

仮定 1 2つの行列対 $(A_j,\ B_j)$, $(A_j^T,\ \sqrt{Q_j}^T)$, $(j = 1,\ 2)$ はそれぞれ可安定かつ可検出である．

以下の補題が知られている [8, 9]．

補題 1 弱結合システム (1) の拘束のもとで評価関数 (2) を最小にする最適制御入力は (3) で与えられる．

$$u^* = -R^{-1}B^T P z \tag{3}$$

ここで，P は Riccati 方程式 (4) の準正定対称安定化解である．

$$A^T P + PA - PSP + Q = 0 \tag{4}$$

ただし，

$$S = BR^{-1}B^T$$
$$A := \begin{bmatrix} A_1 & \varepsilon A_{12} \\ \varepsilon A_{21} & A_2 \end{bmatrix},\ B := \begin{bmatrix} B_1 & \varepsilon B_{12} \\ \varepsilon B_{21} & B_2 \end{bmatrix}$$

Riccati 方程式 (4) の解法に関しては，現在までに多くの文献が報告されている．代表的なアルゴリズムとして，Kleinman による Newton 法に基づく方法 [4]，Arimoto–MacFarlane–Potter による固有ベクトル法に基づく方法 [11]，Laub による Schur 分解法に基づく方法 [5] 等が良く知られている．中でも Schur 分解法はアルゴリズムの安定性，精度の面から非常に優れた解法として利用されている．近年では，現在知られている Riccati 方程式の全てのタイプに対して，マトリクスペンシルに基づく解法が開発されている [6]．

準正定対称安定化解を求める場合，ソフトウェアである MATLAB[17] の使用が考えられる．MATLAB では Schur 分解法に基づくアルゴリズムが採用されており，命令コマンドも非常に簡単で `>> P=are(A,S,Q)` とすれば解を得ることが可能である．しかしながら，MATLAB は有償であるために通常多人数の数値解法の演習で使用することは現実的に難しい．そこで，本研究では，シミュレーションに無料数値計算ソフトウェアである Octave[18] の利用を考える．Octave は行列演算を中心とする数値計算用インタプリタ言語であり，行列やベクトルの演算，連立方程式，固有値，数値積分，常微分方程式，非線形方程式，統計計算，FFT 等理工系のほとんどの数値計算が可能である．さらには，制御工学で必要な数値計算を行うことが可能である．その他，MATLAB にあるコマンドの多くは Octave で実行可能であり，Octave 上でも `>> P=are(A,S,Q)` で解を求めることが可能である．

2.1 解の構造

この章では，Riccati 方程式 (4) の構造を明らかにする．まず，Octave(あるいは MATLAB) を利用して，様々な ε に対して解 P を求める．シミュレーションで用いる値は以下である．

$$A = \begin{bmatrix} 0 & \varepsilon \\ -2\varepsilon & -2 \end{bmatrix},\ S = \begin{bmatrix} 2 & \varepsilon \\ \varepsilon & 4 \end{bmatrix},\ Q = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & 1 \end{bmatrix}$$

$\varepsilon = 0.1$, $\varepsilon = 0.01$, $\varepsilon = 0.001$ に対する解 $P = P(\varepsilon)$ はそれぞれ以下であった．

$$P(0.1) = \begin{bmatrix} 7.00912596763799e-01 & 2.69234469630584e-02 \\ 2.69234469630582e-02 & 2.07604596355199e-01 \end{bmatrix}$$

$$P(0.01) = \begin{bmatrix} 7.07044729974853e-01 & 2.70210185642302e-03 \\ 2.70210185642277e-03 & 2.07111774507050e-01 \end{bmatrix}$$

$$P(0.001) = \begin{bmatrix} 7.07106160663470e-01 & 2.70219958541973e-04 \\ 2.70219958542208e-04 & 2.07106831121272e-01 \end{bmatrix}$$

したがって，

$$P = \begin{bmatrix} P_1 & \varepsilon P_{12} \\ \varepsilon P_{12}^T & P_2 \end{bmatrix} \tag{5}$$

となることが予想される．このように，Octave を利用すればどのような例でも瞬時に解を得ることができるので，プログラムの作成後のチェックや式の予想をたてること等に利用することができる．しかしながら，このような簡単なシミュレーションでも確認される重要な注意事項が存在する．それは，$P(0.1)$, $P(0.01)$, $P(0.001)$ のどの解も対称行列になっていないことである．したがって，弱結合システムに対する Riccati 方程式を解く場合，何らかの工夫が必要であることが分かる．以下の議論では，どのような ε でも対称解を得ることができる数値計算アルゴリズムについて焦点を絞る．

本研究では, 解 P を (5) のように仮定し, 陰関数定理によって,

$$P_i = \mathcal{F}_i(\varepsilon),\ (i=1,\ 2),\ P_{12} = \mathcal{F}_{12}(\varepsilon) \tag{6}$$

となる ε の関数 $\mathcal{F}_i = \mathcal{F}_i(\varepsilon)$, $\mathcal{F}_{12} = \mathcal{F}_{12}(\varepsilon)$ が存在する条件を導出する. 解 P(5) を Riccati 方程式 (4) に代入すれば以下の 3 組の行列非線形方程式 (7) が得られる.

$$\begin{aligned} F_1 &= A_1^T P_1 + P_1 A_1 + \varepsilon^2 (A_{21}^T P_{12}^T + P_{12} A_{21}) \\ &\quad - P_1 S_1 P_1 - \varepsilon^2 (P_1 S_{11} P_1 + P_{12} S_{112}^T P_1 \\ &\quad + P_{12} S_{122}^T P_1 + P_1 S_{112} P_{12}^T + P_1 S_{122} P_{12}^T \\ &\quad + P_{12} S_2 P_{12}^T + \varepsilon^2 P_{12} S_{22} P_{12}^T) + Q_1 = 0 \end{aligned} \tag{7a}$$

$$\begin{aligned} F_{12} &= A_1^T P_{12} + P_1 A_{12} + A_{21}^T P_2 + P_{12} A_2 \\ &\quad - P_1 S_1 P_{12} - \varepsilon^2 (P_1 S_{11} P_{12} + P_{12} S_{112}^T P_{12} \\ &\quad + P_{12} S_{122}^T P_{12} + P_1 S_{112} P_2 + P_1 S_{122} P_2 \\ &\quad + P_{12} S_2 P_2 + \varepsilon^2 P_{12} S_{22} P_2) + Q_{12} = 0 \end{aligned} \tag{7b}$$

$$\begin{aligned} F_2 &= A_2^T P_2 + P_2 A_2 + \varepsilon^2 (A_{12}^T P_{12} + P_{12}^T A_{12}) \\ &\quad - P_2 S_2 P_2 - \varepsilon^2 (P_2 S_{22} P_2 + P_{12}^T S_{112} P_2 \\ &\quad + P_{12}^T S_{122} P_2 + P_2 S_{112}^T P_{12} + P_2 S_{122}^T P_{12} \\ &\quad + P_{12}^T S_1 P_{12} + \varepsilon^2 P_{12}^T S_{11} P_{12}) + Q_2 = 0 \end{aligned} \tag{7c}$$

ただし, $S_i = B_i R^{-1} B_i^T$, $(i=1,\ 2)$, $S_{11} = B_{12} R^{-1} B_{12}^T$, $S_{22} = B_{21} R^{-1} B_{21}^T$, $S_{112} = B_1 R^{-1} B_{21}^T$, $S_{122} = B_{12} R^{-1} B_2^T$ である.

ここで, $\varepsilon \to +0$ としたときの行列非線形方程式 (7) の解を $\bar{P}_i$, $i=1,\ 2$, $\bar{P}_{12}$ と定義する. このとき, 解 $\bar{P}_i$ は以下の Ricatti 方程式 (8) を満足する.

$$A_i^T \bar{P}_i + \bar{P}_i A_i - \bar{P}_i S_i \bar{P}_i + Q_i = 0 \tag{8}$$

Riccati 方程式 (4) の解 P に対して, $\varepsilon \to +0$ としたときの解の振舞いは, 以下の定理で記述することが可能である.

定理 1 仮定 1 のもと $\varepsilon \in (0,\ \sigma^*)$ を満足する全ての ε に対して, Riccati 方程式 (4) の準正定対称安定化解 P が

$$\begin{aligned} P &= \bar{P} + O(\varepsilon) \\ &= \mathbf{block\ diag} \begin{pmatrix} \bar{P}_1 & \bar{P}_2 \end{pmatrix} + O(\varepsilon) \end{aligned} \tag{9}$$

と書けるような $\sigma^* > 0$ が存在する.

(証明) 陰関数定理によれば, $\varepsilon = 0$ での Jacobian が非特異であることを示せば十分である. $\varepsilon = 0$ での Jacobian 行列は (10) のように計算される.

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 & 0 & 0 \\ * & \mathbf{J}_{12} & * \\ 0 & 0 & \mathbf{J}_2 \end{bmatrix} \tag{10}$$

ただし,

$$\mathbf{J}_i = (A_i - S_i \bar{P}_i)^T \otimes I_{n_i} + I_{n_i} \otimes (A_i - S_i \bar{P}_i)^T,$$
$$\mathbf{J}_{12} = (A_2 - S_2 \bar{P}_2)^T \otimes I_{n_1} + I_{n_2} \otimes (A_1 - S_1 \bar{P}_1)^T$$

このとき, (10) の Jacobian は

$$\mathrm{det}\mathbf{J} = \mathrm{det}\mathbf{J}_1 \mathrm{det}\mathbf{J}_{12} \mathrm{det}\mathbf{J}_2 \tag{11}$$

となる. 明らかに, $\mathbf{J}_i$, $\mathbf{J}_{12}$ は Riccati 方程式 (8) が仮定 1 より準正定対称安定化解を持つので安定である. したがって, (10) の Jacobian は非特異であるので, (9) の構造を持つことが示された.

証明の残りは P が準正定対称安定化解である証明である. まず, 十分小さな ε に対して, 解の構造式 (9) を利用することによって

$$\begin{aligned} &A - SP \\ &= \mathbf{block\ diag} \begin{pmatrix} A_1 - S_1 \bar{P}_1 & A_2 - S_2 \bar{P}_2 \end{pmatrix} + O(\varepsilon) \end{aligned} \tag{12}$$

と変形される. 次に, 行列 $A_i - S_i \bar{P}_i$ の安定性を考慮すれば $A - SP$ が安定であるような十分小さな ε が存在することは容易に示される. ■

2.2 Newton 法

Riccati 方程式 (4) の解法に関しては, Schur 分解法がアルゴリズムの安定性, 精度の面から非常に優れた解法として利用されていることは先に述べた. しかしながら, Schur 分解法で必要な計算のワークスペースはもとの Riccati 方程式のサイズの 2 倍必要であることが知られている [5]. さらに, 弱結合システムのように摂動項 ε を含む場合, 得られた解が非対称行列になる場合が存在することも知られている [5]. 実際, 先の簡単なシミュレーションでも確認されている. 以上より, 本研究では, 前の章で得られた解の構造 (9) を考慮して Newton 法に基づくアルゴリズムの適用を考える.

定理 2 Newton 法 (13) を考える.

$$\begin{aligned} &P^{(k+1)}(A - SP^{(k)}) + (A - SP^{(k)})^T P^{(k+1)} \\ &\quad + P^{(k)} S P^{(k)} + Q = 0,\ k = 0,\ 1,\ \cdots \end{aligned} \tag{13a}$$

$$P^{(k)} = \begin{bmatrix} P_1^{(k)} & \varepsilon P_{12}^{(k)} \\ \varepsilon P_{12}^{(k)T} & P_2^{(k)} \end{bmatrix} \tag{13b}$$

ただし, 初期値を (14) のように設定する.

$$P^{(0)} = \mathbf{block\ diag} \begin{pmatrix} \bar{P}_1 & \bar{P}_2 \end{pmatrix} \tag{14}$$

仮定 1 のもと $\varepsilon \in (0,\ \bar{\sigma})$, $(0 < \bar{\sigma} \le \sigma^*)$ を満足する全ての ε に対して, アルゴリズム (13a) が 2 次収束であるような $\bar{\sigma}$ が存在する. このとき, (15) が成立する.

$$\|P^{(k)} - P\| = \frac{O(\varepsilon^{2^k})}{\beta \gamma 2^k},\ k = 0,\ 1,\ \cdots \tag{15}$$

ただし,

$$\gamma = 2\|S\| < \infty,\ \beta = \|[\nabla \mathcal{G}(P^{(0)})]^{-1}\|$$
$$\eta = \beta \cdot \|\mathcal{G}(P^{(0)})\|,\ \theta = \beta\eta\gamma,\ \nabla \mathcal{G}(P) = \frac{\partial \mathrm{vec}\mathcal{G}(P)}{\partial (\mathrm{vec}P)^T}$$
$$\mathcal{G}(P) = PA + A^T P - PSP + Q$$

(証明) 証明はNewton–Kantorovich定理 [10, 15] によって行われる. Riccati方程式 (4) は P に関して微分可能であるので容易に以下のように微分することができる.

$$\nabla\mathcal{G}(P) := \frac{\partial \mathrm{vec}\mathcal{G}(P)}{\partial(\mathrm{vec}P)^T}$$
$$= (A-SP)^T \otimes I_n + I_n \otimes (A-SP)^T \tag{16}$$

したがって, 任意の P_a, P_b に対して

$$\|\nabla\mathcal{G}(P_a) - \nabla\mathcal{G}(P_b)\| \le \gamma\|P_a - P_b\| \tag{17}$$

が成立する. ただし, $\gamma = 2\|S\|$ である. さらに, (12) の安定性の結果を利用すれば, 十分小さな $\varepsilon \in (0,\ \bar{\sigma})$, $\bar{\sigma} \le \sigma^*$ に対して $\nabla\mathcal{G}(P)$ が非特異であるような $\bar{\sigma}$ が存在することが分かる. したがって, $\|[\nabla\mathcal{G}(P)]^{-1}\| \equiv \beta$ を満たす β が存在する. 一方, 定理1より $\|\mathcal{G}(P)\| = O(\varepsilon)$ が示されるので $\|[\nabla\mathcal{G}(P)]^{-1}\| \cdot \|\mathcal{G}(P)\| \equiv \eta = O(\varepsilon)$. となるような η が存在することが分かる. したがって, $\eta = O(\varepsilon)$ を考慮すれば $\theta \equiv \beta\gamma\eta < 2^{-1}$ となるような θ の存在が示される. 後はNewton–Kantorovich定理によって2次収束であることが容易に示される. ■

2.3 不動点アルゴリズム

アルゴリズム (13a) は, 非線形行列方程式ではなく, Lyapunov方程式で表現される線形方程式であるので通常の連立1次方程式に変換して解くことが可能である. しかしながら, P_1, P_{12}, P_2 が行列であり, これらの行列のサイズが大きくなった場合, 以下のように連立1次方程式に変換すれば数値計算のワークスペースの増加を引き起こす.

$$\begin{aligned}
&P^{(k+1)}(A-SP^{(k)}) + (A-SP^{(k)})^T P^{(k+1)} \\
&\quad + P^{(k)}SP^{(k)} + Q = 0 \\
&P^{(k)} = \begin{bmatrix} P_1^{(k)} & \varepsilon P_{12}^{(k)} \\ \varepsilon P_{12}^{(k)T} & P_2^{(k)} \end{bmatrix} \\
&\Leftrightarrow \begin{bmatrix} \bar{A}_1^T \otimes I_{n_1} + I_{n_1} \otimes \bar{A}_1^T & \varepsilon^2(\bar{A}_{21}^T \otimes I_{n_1} + I_{n_1} \otimes \bar{A}_{21}^T) & 0 \\ \bar{A}_{12}^T \otimes I_{n_2} & \bar{A}_2^T \otimes I_{n_1} + I_{n_2} \otimes \bar{A}_1^T & I_{n_1} \otimes \bar{A}_{21}^T \\ 0 & \varepsilon^2(\bar{A}_{21}^T \otimes I_{n_1} + (I_{n_1} \otimes \bar{A}_{21}^T)U_{n_1 n_2}) & (\bar{A}_{12} \otimes I_{n_2})U_{n_1 n_2} + I_{n_2} \otimes \bar{A}_{12} \end{bmatrix} \\
&\quad \cdot \begin{bmatrix} \mathrm{vec}P_1^{(k+1)} \\ \mathrm{vec}P_{12}^{(k+1)} \\ \mathrm{vec}P_2^{(k+1)} \end{bmatrix} = - \begin{bmatrix} \mathrm{vec}\bar{Q}_1 \\ \mathrm{vec}\bar{Q}_{12} \\ \mathrm{vec}\bar{Q}_2 \end{bmatrix} \qquad (18) \\
&A - SP^{(k)} := \begin{bmatrix} \bar{A}_1 & \varepsilon\bar{A}_{12} \\ \varepsilon\bar{A}_{21} & \bar{A}_2 \end{bmatrix} \\
&P^{(k)}SP^{(k)} + Q := \begin{bmatrix} \bar{Q}_1 & \varepsilon\bar{Q}_{12} \\ \varepsilon\bar{Q}_{12}^T & \bar{Q}_2 \end{bmatrix}
\end{aligned}$$

したがって, 数値計算に必要なワークスペースの減少を実現させるために不動点定理に基づくアルゴリズムを提案する. アルゴリズム (13a) を一般化した以下のLyapunov方程式を考える.

$$E^T X + XE + H = 0 \tag{19}$$

ただし, $E_i \in \mathbf{R}^{n_i}$ は安定行列かつ $X_i = X_i^T \ge 0 \in \mathbf{R}^{n_i}$, $H_i = H_i^T \ge 0 \in \mathbf{R}^{n_i}$, $(i = 1,\ 2)$ である.

$$E := \begin{bmatrix} E_1 & \varepsilon E_{12} \\ \varepsilon E_{21} & E_2 \end{bmatrix},\ H := \begin{bmatrix} H_1 & \varepsilon H_{12} \\ \varepsilon H_{12}^T & H_2 \end{bmatrix}$$
$$X = \begin{bmatrix} X_1 & \varepsilon X_{12} \\ \varepsilon X_{12}^T & X_2 \end{bmatrix}$$

解 X をLyapunov方程式 (19) に代入すれば以下の3組の行列線形方程式 (20) が得られる.

$$E_1^T X_1 + X_1 E_1 + \varepsilon^2(E_{21}^T X_{12}^T + X_{12}E_{21}) + H_1 = 0 \tag{20a}$$
$$E_1^T X_{12} + X_1 E_{12} + E_{21}^T X_2 + X_{12}E_2 + H_{12} = 0 \tag{20b}$$
$$E_2^T X_2 + X_2 E_2 + \varepsilon^2(E_{12}^T X_{12} + X_{12}^T E_{12}) + H_2 = 0 \tag{20c}$$

行列線形方程式 (20) を解くためのアルゴリズムは (21) で与えられる.

$$\begin{aligned}
&E_1^T X_1^{(k+1)} + X_1^{(k+1)} E_1 \\
&\quad + \varepsilon^2(E_{21}^T X_{12}^{(k)T} + X_{12}^{(k)} E_{21}) + H_1 = 0 \qquad (21a) \\
&E_2^T X_2^{(k+1)} + X_2^{(k+1)} E_2 \\
&\quad + \varepsilon^2(E_{12}^T X_{12}^{(k)} + X_{12}^{(k)T} E_{12}) + H_2 = 0 \qquad (21b) \\
&E_1^T X_{12}^{(k+1)} + X_{12}^{(k+1)} E_2 \\
&\quad + X_1^{(k+1)} E_{12} + E_{21}^T X_2^{(k+1)} + H_{12} = 0 \qquad (21c) \\
&X_i^{(0)} = 0,\ (i = 1,\ 2), X_{12} = 0,\ (k = 0,\ 1,\ \cdots)
\end{aligned}$$

アルゴリズム (21) の収束に関して以下の定理を得ることが出来る.

定理 3 E_i, $(i = 1,\ 2)$ が安定行列であれば, $\varepsilon \in (0,\ \hat{\sigma})$, $(0 < \hat{\sigma})$ を満足する全ての ε に対して, アルゴリズム (21) が1次収束であるような $\hat{\sigma}$ が存在する. すなわち, (22) が成立する.

$$\begin{aligned}
&\|X_i^{(k)} - X_i\| = O(\varepsilon^{2k}),\ (i = 1,\ 2) \\
&\|X_{12}^{(k)} - X_{12}\| = O(\varepsilon^{2k}),\ k = 1,\ 2,\ \cdots
\end{aligned} \tag{22}$$

(証明) 収束に関する証明は不動点定理 [13, 14] によって行われる. まず, アルゴリズム (21) はそれぞれ (23) と等価である.

$$\begin{aligned}
&X_1^{(k+1)} := \mathcal{Z}_1(X_{12}^{(k)}) \\
&= \varepsilon^2 \int_0^\infty \exp(E_1^T s)(E_{21}^T X_{12}^{(k)T} + X_{12}^{(k)} E_{21}) \exp(E_1 s) ds \\
&\quad + \int_0^\infty \exp(E_1^T s) H_1 \exp(E_1 s) ds \qquad (23a) \\
&X_2^{(k+1)} := \mathcal{Z}_2(X_{12}^{(k)})
\end{aligned}$$

$$= \varepsilon^2 \int_0^\infty \exp(E_2^T s)(E_{12}^T X_{12}^{(k)} + X_{12}^{(k)T} E_{12}) \exp(E_2 s) ds$$
$$+ \int_0^\infty \exp(E_2^T s) H_2 \exp(E_2 s) ds \quad (23b)$$
$$X_{12}^{(k+1)} := \mathcal{Z}_3(X_1^{(k+1)},\ X_2^{(k+1)}) = \mathcal{Z}_3(X_{12}^{(k)}) \quad (23c)$$

したがって, 関係式 (23a) より E_1 の安定性を考慮して $\|\exp(E_1^T s)\| \le m_1 \exp(-\phi_1 s)$ となる $m_1 > 0$, $\phi_1 > 0$ が存在する [12] ので任意の X_{12}^a 及び X_{12}^b に対して

$$\|\mathcal{Z}_1(X_{12}^a) - \mathcal{Z}_1(X_{12}^b)\|$$
$$= \varepsilon^2 \|E_{21}\| \cdot \|X_{12}^a - X_{12}^b\| \int_0^\infty m_1^2 \exp(-2\phi_1 s) ds$$
$$= \varepsilon^2 \mathcal{M}_1 \|X_{12}^a - X_{12}^b\|$$

となる $\mathcal{M}_1$ 及び $\varepsilon^2 \mathcal{M}_1 < 1$ を満足する $\varepsilon = \varepsilon_1$ が存在する. 同様にして

$$\|\mathcal{Z}_2(X_{12}^a) - \mathcal{Z}_2(X_{12}^b)\| = \varepsilon^2 \mathcal{M}_2 \|X_{12}^a - X_{12}^b\|$$
$$\|\mathcal{Z}_3(X_{12}^a) - \mathcal{Z}_2(X_{12}^b)\| = \varepsilon^2 \mathcal{M}_3 \|X_{12}^a - X_{12}^b\|$$

となる $\mathcal{M}_i$ 及び $\varepsilon^2 \mathcal{M}_i < 1$ を満足する $\varepsilon = \varepsilon_i$, $(i = 2,\ 3)$ が存在する. 以上より, 不動点定理によりアルゴリズム (21) が収束することが分かる. アルゴリズム (21) が 1 次収束である証明, すなわち (22) の証明は数学的帰納法により示される. (21) から (20) を引けば (24) を得る.

$$E_1^T(X_1^{(k+1)} - X_1) + (X_1^{(k+1)} - X_1)E_1$$
$$+\varepsilon^2\{E_{21}^T(X_{12}^{(k)} - X_{12})^T + (X_{12}^{(k)} - X_{12})E_{21}\} = 0 \quad (24a)$$
$$E_2^T(X_2^{(k+1)} - X_2) + (X_2^{(k+1)} - X_2)E_2$$
$$+\varepsilon^2\{E_{12}^T(X_{12}^{(k)} - X_{12}) + (X_{12}^{(k)} - X_{12}^T E_{12}\} = 0 \quad (24b)$$
$$E_1^T(X_{12}^{(k+1)} - X_{12}) + (X_{12}^{(k+1)} - X_{12})E_2$$
$$+(X_1^{(k+1)} - X_1)E_{12} + E_{21}^T(X_2^{(k+1)} - X_2) = 0 \quad (24c)$$

(24) で $k = 0$ とすれば (25) を得る.

$$E_1^T(X_1^{(1)} - X_1) + (X_1^{(1)} - X_1)E_1 + O(\varepsilon^2) = 0 \quad (25a)$$
$$E_2^T(X_2^{(1)} - X_2) + (X_2^{(1)} - X_2)E_2 + O(\varepsilon^2) = 0 \quad (25b)$$

したがって, E_i の安定性を考慮すれば Lyapuno 方程式の性質 [2] より $X_i^{(1)} - X_i = O(\varepsilon^2)$ を得る. さらに (24c) に代入して $X_{12}^{(1)} - X_{12} = O(\varepsilon^2)$ を得る. 以上より $k = 1$ のとき成立する. $k = l$ のとき

$$\|X_i^{(l)} - X_i\| = O(\varepsilon^{2l}),\ (i = 1,\ 2)$$
$$\|X_{12}^{(l)} - X_{12}\| = O(\varepsilon^{2l}),\ l \ge 2 \quad (26)$$

が成立すると仮定すれば, (24) で $k = l$ として (27) を得る.

$$E_1^T(X_1^{(l+1)} - X_1) + (X_1^{(l+1)} - X_1)E_1 + O(\varepsilon^{2l+2}) = 0 \quad (27a)$$
$$E_2^T(X_2^{(1)} - X_2) + (X_2^{(1)} - X_2)E_2 + O(\varepsilon^{2l+2}) = 0 \quad (27b)$$

したがって, $X_i^{(l+1)} - X_i = O(\varepsilon^{2l+2})$ を得る. 同様に (24c) に代入して $X_{12}^{(l+1)} - X_{12} = O(\varepsilon^{2l+2})$ を得る. 以上より (22) と比較して $k = l + 1$ のときも成立する. 最終的に数学的帰納法により (22) が成立することが示された. ■

3 シミュレーション

提案されたアルゴリズム (13), (21) の有用性を確認するためにシミュレーションを行う. 文献 [8] を参考に弱結合システム (1) の行列を以下に与える.

$$A_1 = \begin{bmatrix} 0 & 1 & -0.266 & -0.009 \\ -2.75 & -2.78 & -1.36 & -0.037 \\ 0 & 0 & 0 & 1 \\ -4.95 & 0 & -55.5 & -0.039 \end{bmatrix}$$

$$\varepsilon A_{12} = \begin{bmatrix} 0.0024 & 0 & -0.087 & 0.002 \\ -0.185 & 0 & 1.11 & -0.011 \\ 0 & 0 & 0 & 0 \\ 0.222 & 0 & 8.17 & 0.004 \end{bmatrix}$$

$$\varepsilon A_{21} = \begin{bmatrix} 0.021 & 0 & 0.121 & 0.003 \\ -1.1 & 0 & -1.62 & -0.015 \\ 0 & 0 & 0 & 0 \\ -2.43 & 0 & 1.37 & -0.034 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} -0.21 & 1 & -1.6 & -0.005 \\ -1.9 & -1.8 & 9.3 & -0.12 \\ 0 & 0 & 0 & 1 \\ -3.1 & 0 & -56 & 0.032 \end{bmatrix}$$

$$B_1 = \begin{bmatrix} 0 \\ 36.1 \\ 0 \\ 0 \end{bmatrix},\ B_2 = \begin{bmatrix} 0 \\ 78.9 \\ 0 \\ 0 \end{bmatrix},\ B_{12} = B_{21} = 0$$

Riccati 方程式 (4) の R, Q をそれぞれ $R = 1$, $Q = 0.5 I_8$ に設定する. 提案された手法を利用して, 初期値 $P^{(0)}$ および $\varepsilon = 0.1$ のときの解 $P(0.1)$ は Table 1 に与えられる.

収束の様子を確認するために誤差のノルムを Table 2 に与える. ただし, 収束判定を $\|\mathcal{G}(P^{(k)})\| < e - 10$ とした.

Table 2.

k	$\|\mathcal{G}(P^{(k)})\|$
0	$6.3881e-01$
1	$4.6192e-02$
2	$3.1037e-04$
3	$8.4569e-10$
4	$1.1457e-11$

Table 2 より, 提案されたアルゴリズムは 2 次収束であることが確認される. さらに, 異なる ε に対する収束の様子を Table 3 に与える. Table 3 より, 全ての ε に対して提案されたアルゴリズムは 2 次収束であることが確認される.

最後に, アルゴリズム (21) の収束性を確認する. $\varepsilon = 0.1$, Newton 法の 1 回目の繰り返し計算内での収束の様子を Table 4 に与える. ただし, 収束判定を $\|\mathcal{H}(X^{(k)})\| = \|E^T X^{(k)} + X^{(k)} E + H\| < e - 10$ とした.

Table 1.

$P^{(0)} = 1.0e+01 \times$

$$\begin{bmatrix} 8.74e-02 & 3.00e-03 & 3.71e-01 & -8.46e-03 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 \\ 3.00e-03 & 1.87e-03 & 1.32e-02 & 1.70e-04 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 \\ 3.71e-01 & 1.32e-02 & 4.10e+00 & -2.20e-02 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 \\ -8.46e-03 & 1.70e-04 & -2.20e-02 & 7.33e-02 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 \\ 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 6.39e-02 & 1.09e-03 & 3.36e-01 & -8.86e-03 \\ 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 1.09e-03 & 8.87e-04 & 6.05e-03 & -5.13e-05 \\ 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 & 3.36e-01 & 6.05e-03 & 6.12e+00 & -2.85e-02 \\ 0.00e+00 & 0.00e+00 & 0.00e+00 & 0.00e+00 & -8.86e-03 & -5.13e-05 & -2.85e-02 & 1.10e-01 \end{bmatrix}$$

$P(0.1) = 1.0e+01 \times$

$$\begin{bmatrix} 8.74e-02 & 3.00e-03 & 3.69e-01 & -8.23e-03 & 9.23e-04 & 1.06e-05 & 5.75e-03 & -4.39e-03 \\ 3.00e-03 & 1.87e-03 & 1.31e-02 & 1.76e-04 & 4.71e-05 & 6.85e-07 & 5.07e-04 & -1.43e-04 \\ 3.69e-01 & 1.31e-02 & 4.09e+00 & -2.20e-02 & -7.60e-03 & -1.78e-04 & -2.11e-01 & -4.44e-02 \\ -8.23e-03 & 1.76e-04 & -2.20e-02 & 7.32e-02 & 2.56e-03 & 4.17e-05 & 4.43e-02 & -3.10e-03 \\ 9.23e-04 & 4.71e-05 & -7.60e-03 & 2.56e-03 & 6.41e-02 & 1.09e-03 & 3.39e-01 & -8.94e-03 \\ 1.06e-05 & 6.85e-07 & -1.78e-04 & 4.17e-05 & 1.09e-03 & 8.87e-04 & 6.11e-03 & -5.15e-05 \\ 5.75e-03 & 5.07e-04 & -2.11e-01 & 4.43e-02 & 3.39e-01 & 6.11e-03 & 6.18e+00 & -2.84e-02 \\ -4.39e-03 & -1.43e-04 & -4.44e-02 & -3.10e-03 & -8.94e-03 & -5.15e-05 & -2.84e-02 & 1.11e-01 \end{bmatrix}$$

Table 3.

k	$\varepsilon = e-01$	$\varepsilon = e-02$	$\varepsilon = e-03$	$\varepsilon = e-04$	$\varepsilon = e-05$
0	$6.3881e-01$	$6.3881e-02$	$6.3881e-03$	$6.3881e-04$	$6.3881e-05$
1	$4.6192e-02$	$3.5599e-04$	$3.5457e-06$	$3.5457e-08$	$3.5630e-10$
2	$3.1037e-04$	$2.9105e-08$	$6.8918e-12$	$4.9959e-12$	$2.4082e-12$
3	$8.4569e-10$	$4.4492e-12$	-	-	-
4	$1.1457e-11$	-	-	-	-

[Q1] コンピュータ使用歴についてお尋ねします．以下のうち最も適当なものを選んでください．

1 年以内	6	1 年から 2 年以内	3	2 年から 3 年以内	2	3 年以上	4

[Q2] プログラミング歴についてお尋ねします．以下のうち最も適当なものを選んでください．

今まで経験がない	10	半年	1	半年から 1 年以内	0	1 年以上	4

[Q3] C 言語を利用したプログラミングは簡単でしたか．

簡単だった	1	普通	2	少し難しかった	6	難しかった	6

[Q4] GNU octave を利用したプログラミングは簡単でしたか．

簡単だった	2	普通	2	少し難しかった	5	難しかった	6

[Q5] 数値計算法を習う上で GNU octave のようなアプリケーションツールは必要であると考えますか．

非常に必要である	2	必要である	8	どちらともいえない	3	必要でない	2

[Q6] GNU octave は作成したプログラムによる結果の確認に役立ちましたか．

非常に役にたった	2	役にたった	7	どちらともいえない	5	役にたたなかった	1

Table 4.

k	$\|\mathcal{H}(X^{(k)})\|$
0	$6.39e-01$
1	$7.31e-01$
2	$3.14e-03$
3	$5.58e-05$
4	$5.75e-07$
5	$6.93e-09$
6	$8.21e-11$

Table 4 より，提案されたアルゴリズムは関係式 (22) を満足していることが確認される．

以上より，Newton 法の初期値 (14) が，十分小さな ε に対して 2 次収束を保証すること．及び，全システムの次元 8 ではなく，サブシステムの次元 4 ですべての繰返し計算が実行できることを考慮すれば，新たに提案されたアルゴリズム (13), (21) は大変有効であると考えられる．

4　実践報告

シミュレーションを利用した数値計算法の講義を実践し，アンケートを実施した．アンケートの対象は入学してまだ半年しかたっていない学生 16 人で，有効な回答として 15 人分を回収した．なお，学生は教育学部に所属し，中学校技術科教員もしくは高等学校情報教員を目指している．アンケートを実施した学生は，90 分の講義を 1 コマとして 4 コマ分の数値計算法の基本を学習した．内容は C 言語の基本を中心に配列を利用した行列の掛算までで，プログラムの検証には Octave を利用した．さらに，応用として Octave を利用してアルゴリズム (21) のプログラミングを行った．

このページの上段に関係するアンケート個目の結果を与える．[Q1] より，平成 15 年度入学の 60%の学生が，大学入学以前にコンピュータを何らかの形で使用していることが伺える．[Q2] より，学生の 66%がプログラミングの経験がないと回答している．その結果，4 コマ分の数値計算法の講義では，[Q3], [Q4] より 70%以上の学生がどの言語でもプログラミングは難しいと感じている．その反面，[Q5],

[Q6] からは, 60%以上の学生が Octave の必要性を感じている.

最後に, [Q7] として調査した GNU octave を利用した感想のうち, 有効な回答を以下に紹介する.

- 難しいということは否めない事実であるが, 使用を鑑みるとわからないことではないと感じられた.
- 単に数値のみを求めるのなら, GNU octave のほうがいいと思います. アルゴリズムを考えさせる場合でも Octave のほうがいいのではないでしょうか. C でやるとなると if,for などを勉強しないといけないので時間がかかる. (C の勉強しているのなら別)
- プログラム自体が初めてだったので, かなり戸惑いました. でも利用することでかなりの利点があると思いました. でもやっぱり難しかったです.
- 便利なのは, なんとなく分かったけれど, 難しい. ただただ, 難解だった.
- プログラミングをしたことがなかったのでついていくのがやっとでした. まだ入力してないのに先へ進むと, そこから遅れを取り戻すのが大変でした. でも数学をする上で便利だということなのでうまく利用できればいいなと思います.
- 数値計算の結果を見るときには非常に便利だと思いました. もっと詳しく使い方を勉強すればもっと利用価値の高いものになるだろうと思った.
- いちいち Octave のためだけに使い方を覚えるくらいなら C を覚えて C で組むほうが楽なような気がします.
- 難しくて正直, よく意味がわからなかった. けど, 行列の計算などが非常に楽で役に立つとは思った.
- よくわからなかったけど, C 言語よりは簡単だと思いました.

上記の感想より, プログラミングが得意な一部の学生には, Octave の必要性を感じていないようだが, プログラミングの初学者の多くは, Octave が何らかの形でプログラミングの理解に役立っていると感じているようである. 以上から, Octave が数値計算法の初期学習段階での理解の助けになることが分かる.

5 まとめ

本研究では, 弱結合システムに対して, Riccati 方程式及び Lyapunov 方程式を解くための反復アルゴリズムを新たに提案した. 証明は, 数値計算で基本的な 3 つの定理を利用することによって行われた. その結果, 解の存在性, アルゴリズムの収束性が示された. さらに, 提案されたアルゴリズムに基づき, シミュレーションの実習を通して数値計算の基礎を確認した. 最後に, Octave と呼ばれる無料数値計算ソフトウェアが学部生に対して初期の学習段階で数値計算法の理解の助けになることをアンケートによって示した.

References

[1] Lancaster, P. and Rodman, L.: *Algebraic Riccati Equations*, Clarendon Press, Oxford (1995).

[2] Zhou, K.: *Essentials of robust control*, Prentice–Hall, New Jersey (1998).

[3] 西村敏充, 狩野弘之: 制御のためのマトリクス・リカッチ方程式, 朝倉書店 (1996).

[4] Kleinman, D. L.: On the Iterative Technique for Riccati Equation Computations, *IEEE Trans. Automatic Control*, Vol. 13, No. 2, pp. 114–115 (1968).

[5] Laub, A. J.: A Schur Method for Solving Algebraic Riccati Equations, *IEEE Trans. Automatic Control*, Vol. 24, No. 6, pp. 913–921 (1979).

[6] Ionescu, V., Oară C. and Weiss, M.: *Generalized Riccati Theory and Robust Control*, John Wiley and Sons, New York (1999).

[7] Gajic, Z., Petkovski, D. and Shen, X.: *Singularly Perturbed and Weakly Coupled Linear System–a Recursive Approach*, Lecture Notes in Control and Information Sciences, 140, Springer-Verlag, Berlin (1990).

[8] Delacour, J. D., Darwish, M. and Fantin, J.: Control Strategies for Large–Scale Power Systems, *Int. J. Control*, Vol. 27, No. 5, pp. 753–767 (1978).

[9] Shen, X., Gourishankar, V. G., Xia, Q. and Rao, M.: Optimal Control for Large–Ccale Systems: a Recursive Approach, *Int. J. Systems Sciences*, Vol. 25, No. 12, pp. 2235–2244 (1994).

[10] Ortega, J. M.: *Numerical analysis, A second course*, SIAM, Philadelphia (1990).

[11] 安藤和昭, 田沼正也ほか: 数値解析手法による制御系設計, 計測自動制御学会 (1986).

[12] 児玉 真三, 須田 信英: システム制御理論のためのマトリクス理論, 計測自動制御学会 (1978).

[13] 広中平祐編: 現代数理科学辞典, 大阪書籍 (1991).

[14] 堀内和夫, 大石進一: 不動点論をめぐって, 情報処理 Vol. 33, No. 4, pp. 308–317 (1992).

[15] Yamamoto, T.: A Method for Finding Sharp Error Bounds for Newton's Method Under the Kantorvich Assumptions, *Numerische Mathematik*, Vol. 49, pp. 203–220 (1986).

[16] Magnus, J. R. and Neudecker, H.: *Matrix Differential Calculus with Applications in Statistics and Econometrics*, John Wiley and Sons, New York (1999).

[17] MATLAB 公式ホームページ
`http://www.cybernet.co.jp/matlab/`

[18] Octave 公式ホームページ
`http://www.octave.org/`