数理解析研究所講究録
第1814巻 2012年 1-8

行列の最小多項式計算について

田島慎一
Shinichi Tajima
筑波大学大学院数理物質科学研究科
Graduate School of Pure and Applied Sciences, University of Tsukuba*

奈良洸平
Kohei Nara
新潟大学大学院自然科学研究科
Graduate School of Sciece and Technology, Niigata University†

小原功任
Katsuyoshi Ohara
金沢大学理工研究域
Department of Computational Science, Kanazawa University‡

1 はじめに

整数（もしくは有理数）を成分にもつ n 次正方行列 A とその特性多項式 $\chi_A(\lambda)$ が与えられたとする．このとき，第18回数式処理学会大会（2009年6月）において報告した方法を用いると，特性多項式の因数分解を利用し行列の最小多項式 $\pi_A(\lambda)$ を求めることができる．最小多項式の各因子の重複度が低い場合には，この方法により最小多項式を効率良く求めることができる．しかしその反面，求めるべき最小多項式が重複度の高い因子を持つ場合，この計算法では，最小多項式候補を求める際の計算コストがかなり高くなってしまうことが分かっている．そのため，特性多項式のみならず求めるべき最小多項式も重複度の高い因子を含んでいるような場合に最小多項式候補を効率良く計算するような新たな計算方法が望まれていた．

さて，n 次単位行列を E とおき，行列 $\lambda E - A$ の全ての $n-1$ 次小行列式の λ の多項式としての最大公約数を $d_A(\lambda)$ とおくと，行列 A の最小多項式 $\pi_A(\lambda)$ は

$$\pi_A(\lambda) = \frac{\chi_A(\lambda)}{d_A(\lambda)}$$

により与えられることが知られている ([1]). このことは即ち，行列の特性多項式が与えられている時，$d_A(\lambda)$ を求めることで最小多項式を与えることができることを意味している．ここで，多項式 $d_A(\lambda)$ の次数が低い場合は最小多項式 $\pi_A(\lambda)$ の各因子の重複度と特性多項式 $\chi_A(\lambda)$ のそれとの差が少ないことになる．この事

*tajima@ie.niigata-u.ac.jp
†f09e058g@mail.cc.niigata-u.ac.jp
‡ohara@air.s.kanazawa-u.ac.jp

は，求めるべき最小多項式の因子の重複度が特性多項式のそれとあまり差がないような場合には，上述した公式に基づくことで最小多項式をより効率良く求めるアルゴリズムを導出できる可能性があることを示唆する．

本研究では，この事実に注目し，最小多項式を求める新たな計算法を導出した．本稿では，まず，アルゴリズム導出の基礎となる定理を述べ，導出したアルゴリズムの概略を与える．次に，数式処理システム Risa/Asir に実装したアルゴリズムの特性を調べる．従来の計算法と本研究で得たアルゴリズムとの計算所要時間の比較を行い，最小多項式が重複度が高い因子を含むような場合には，従来の計算法に比べ本提案法の方が計算効率が良いことを示す．

2 最小多項式の性質

この節では，アルゴリズム導出の基礎となる定理を与える．以下，A は整数を成分に持つ n 次正方行列であるとする．行列 A の特性多項式を $\chi_A(\lambda)$ ，最小多項式を $\pi_A(\lambda)$ で表す．特性多項式の因数分解が次で与えられたとする．

$$\chi_A(\lambda) = f_1(\lambda)^{m_1} \cdot f_2(\lambda)^{m_2} \cdots f_s(\lambda)^{m_s}$$

このとき，各 $i = 1, 2, ..., s$ に対し $b(x, y)$ を

$$f_i(x)^{m_i} - f_i(y)^{m_i} = b_i(x, y)(x - y)$$

を満たす 2 変数多項式として定め，さらに $B_i(A, \lambda) = b_i(A, \lambda E)$ とおく．次が成り立つ．

定理 1

行列 $f_1(A)^{m_1} \cdots f_{i-1}(A)^{m_{i-1}} \cdot B_i(A, \lambda) \cdot f_{i+1}(A)^{m_{i+1}} \cdots f_s(A)^{m_s}$ のすべての成分と，$f_i(\lambda)^{m_i}$ との λ の多項式としての最大公約数を $d_i(\lambda)$ とおく．このとき，行列 A の最小多項式 $\pi_A(\lambda)$ は

$$\pi_A(\lambda) = \frac{f_1(\lambda)^{m_1}}{d_1(\lambda)} \cdot \frac{f_2(\lambda)^{m_2}}{d_2(\lambda)} \cdots \frac{f_s(\lambda)^{m_s}}{d_s(\lambda)}$$

を満たす．

各因子毎に，最小多項式の重複度を決定できることを注意しておく．

この定理は，スペクトル分解に関し，論文 [3] において与えた議論と同じ議論を行うことで容易に証明することができる．本稿では証明を省略する．

3 アルゴリズム

前の節で与えた定理をそのままの形で用いて，最小多項式を求めるアルゴリズムを構成することができる．しかし，行列のサイズが大きい場合は定理をそのままの形で用いたような計算法では計算量が膨大なものとなり，最小多項式の計算法としては実用性に乏しいものとなる．そこで本研究では，この定理とランダムに生成させた 2 つのベクトルを組み合わせて利用することで最小多項式候補を構成するアルゴリズムを導出した．

アルゴリズム導出の際の基本的アイデアを以下に述べる．まず，整数を成分に持つ 2 つの n 次元ベクトル $\boldsymbol{v}, \boldsymbol{w}$ を用意する．ただし $\boldsymbol{v}$ は横ベクトル，$\boldsymbol{w}$ は縦ベクトルとする．多項式 $g_i(\lambda)$ を

$$g_i(\lambda) = \boldsymbol{v} f_1(A)^{m_1} \cdots f_{i-1}(A)^{m_{i-1}} \cdot B_i(A, \lambda) \cdot f_{i+1}(A)^{m_{i+1}} \cdots f_s(A)^{m_s} \boldsymbol{w}$$

で定める．このとき，多項式 $g_i(\lambda)$ は，(定理の中で定義された) 多項式 $d_i(\lambda)$ により割り切れることは明らかである．従って，多項式 $g_i(\lambda)$ の因子 $f_i(\lambda)$ に関する重複度を t_i とおくと，行列 A の最小多項式 $\pi_A(\lambda)$ の因子 $f_i(\lambda)$ の重複度は $m_i - t_i$ 以上となる．ベクトル $\boldsymbol{v}, \boldsymbol{w}$ としてランダムに生成させた整数を成分にもつようなものを用いれば，ほとんどの場合，$m_i - t_i$ は最小多項式の因子 $f_i(\lambda)$ の重複度と一致することになる．

さて，$g_i(\lambda)$ は，横ベクトル $\boldsymbol{v}\, f_1(A)^{m_1} \cdots f_{i-1}(A)^{m_{i-1}}$ に行列 $B_i(A,\lambda)$ を右から施した結果と，縦ベクトル $f_{i+1}(A)^{m_{i+1}} \cdots f_s(A)^{m_s}\, \boldsymbol{v}$ との内積をとることで求めることができる．いま，$\boldsymbol{v}_i = \boldsymbol{v} f_1(A)^{m_1} \cdots f_{i-1}(A)^{m_{i-1}}$ とおき，多項式 $f_i(\lambda)^{m_i}$ の展開式を

$$f_i(\lambda)^{m_i} = \lambda^{m_i \deg(f_i)} + c_1 \lambda^{m_i \deg(f_i)-1} + c_2 \lambda^{m_i \deg(f_i)-2} + \cdots + c_{m_i \deg(f_i)-1}\lambda + c_{m_i \deg(f_i)}$$

とおくと，横ベクトル $\boldsymbol{v}_i B_i(A,\lambda)$ は，ベクトル $\boldsymbol{v}_i$ に行列 A を右から $f_i(\lambda)^{m_i}$ の展開係数を用いてホーナー法に従って施していくことで求めることができる． $\boldsymbol{v}_i B_i(A,\lambda)$ の計算では多項式 $f_i(\lambda)^{m_i}$ の $c_1, c_2, \ldots, c_{m_i \deg(f_i)-1}$ までの展開係数を用いることになる．この方法で最後に構成されるベクトルに右から A を施して得られるベクトルにさらに， $\boldsymbol{v}_i$ を $c_{m_i \deg(f_i)}$ 倍したものを加えれば，$\boldsymbol{v}_i f_i(A)^{m_i}$ 即ち，ベクトル

$$\boldsymbol{v}_{i+1} = \boldsymbol{v} f_1(A)^{m_1} \cdots f_{i-1}(A)^{m_{i-1}} f_i(A)^{m_i}$$

が構成できる点に注意する．このことに着目すると，最小多項式候補を求める際の計算量をかなり軽減させることができる．以下にアルゴリズムの概略を与える．

3.1 行列全体の最小多項式の計算アルゴリズム（アルゴリズム１）

定理１を利用して行列全体の最小多項式を求めるアルゴリズムを説明する．

STEP 1 ベクトルの生成

- ランダムな整数を成分とする n 次元横ベクトル $\boldsymbol{v}$ を生成する．
- ランダムな整数を成分とする n 次元縦ベクトル $\boldsymbol{w}$ を生成する．

STEP 2 $\boldsymbol{w_2}$, $\boldsymbol{w_3}$, ..., $\boldsymbol{w_s}$ の計算

- $\boldsymbol{w_s} \leftarrow \boldsymbol{w}$
- for $i \leftarrow s-1$ to 2

  $\boldsymbol{w_i} \leftarrow f_i(A)^{m_i}\, \boldsymbol{w_{i+1}}$ とする

STEP 3 最小多項式の因子のべきの候補の計算

- for $i \leftarrow 1$ to s

  $C \leftarrow [\,]$ (空リスト)

  $deg \leftarrow f_i(\lambda)^{m_i}$ の次数

  $g(\lambda) \leftarrow (\boldsymbol{v} \cdot \boldsymbol{w_i})\, \lambda^{deg-1}$

  $f_i(\lambda)^{m_i}$ の各項の係数を次数の小さい方からリスト C に格納（リストの先頭に $deg-1$ 次の項の係数 c_1 が入る）

  $\boldsymbol{p} \leftarrow \boldsymbol{v}$

  for $j \leftarrow 1$ to $deg - 1$

  $\boldsymbol{q} \leftarrow \boldsymbol{p}\, A + \boldsymbol{v}\, c_j$

$g(\lambda) \leftarrow g(\lambda) + (\boldsymbol{q} \cdot \boldsymbol{w_i})\lambda^{(deg-j-1)}$

$\boldsymbol{p} \leftarrow \boldsymbol{q}$

$t_i \leftarrow$ 多項式 $g(\lambda)$ の因子 $f_i(\lambda)$ に関する重複度

$k_i = m_i - t_i$ とする.

if $i < s$

$\boldsymbol{v} \leftarrow \boldsymbol{p}\, A + \boldsymbol{v}\, c_{deg}$

STEP 4 最小多項式の候補が正しいかどうかチェック

・$f_1(A)^{k_1} \cdot f_2(A)^{k_2} \cdots f_s(A)^{k_s}$ を計算し，零行列かどうかチェックする.

・零行列でなければ零行列になるまで足りない因子を増やしていくことにより，最小多項式の各因子の重複度 $r_1, r_2, \cdots, r_s$ を求める.

ここで，STEP 4 の操作はランダムに生成したベクトルが特殊なものになって正しい最小多項式の因子のべきが求められなかった場合のための措置である.

3.2 行列の一つの列に対する最小多項式（最小消去多項式）の計算アルゴリズム（アルゴリズム２）

行列の一つの列に対する最小多項式を求めるアルゴリズムを説明する．こちらも第２節で与えた定理が利用できる．ここでは求めたい行列の列を l 列目とする.

STEP 1 ベクトルの生成

・ランダムな整数を成分とする n 次元横ベクトル $\boldsymbol{v}$ を生成する.

・第 l 成分のみ 1，他は全て零なる基本単位ベクトル $\boldsymbol{e_l}$ を生成する.

STEP 2 $\boldsymbol{w_2}$, $\boldsymbol{w_3}$, ..., $\boldsymbol{w_s}$ の計算

・$\boldsymbol{w_s} \leftarrow \boldsymbol{e_l}$

・for $i \leftarrow s-1$ to 2

$\boldsymbol{w_i} \leftarrow f_i(A)^{m_i}\, \boldsymbol{w_{i+1}}$ とする

STEP 3 最小多項式の因子のべきの候補の計算

・for $i \leftarrow 1$ to s

$C \leftarrow [\,]$（空リスト）

$deg \leftarrow f_i(\lambda)^{m_i}$ の次数

$g(\lambda) \leftarrow (\boldsymbol{v} \cdot \boldsymbol{w_i})\ \lambda^{deg-1}$

$f_i(\lambda)^{m_i}$ の各項の係数を次数の小さい方からリスト C に格納（リストの先頭に $deg-1$ 次の項の係数 c_1 が入る）

$\boldsymbol{p} \leftarrow \boldsymbol{v}$

for $j \leftarrow 1$ to $deg - 1$

$\boldsymbol{q} \leftarrow \boldsymbol{p}\, A + \boldsymbol{v}\, c_j$

$g(\lambda) \leftarrow g(\lambda) + (\boldsymbol{q} \cdot \boldsymbol{w_i})\lambda^{(deg-j-1)}$

$\boldsymbol{p} \leftarrow \boldsymbol{q}$
$t_i \leftarrow$ 多項式 $g(\lambda)$ の因子 $f_i(\lambda)$ に関する重複度
$k_i = m_i - t_i$ とする.
if $i < s$
$\boldsymbol{v} \leftarrow \boldsymbol{p}\ A + \boldsymbol{v}\ c_{deg}$

STEP 4 最小多項式の候補が正しいかどうかチェック
・$f_1(A)^{k_1} \cdot f_2(A)^{k_2} \cdots f_s(A)^{k_s} \boldsymbol{e}_l$ を計算し，零ベクトルかどうかチェックする.
・零ベクトルでなければ零ベクトルになるまで足りない因子を増やしていくことにより，最小多項式の各因子の重複度 $r_{l,1}, r_{l,2}, \cdots, r_{l,s}$ を求める.

アルゴリズム 1 と 2 の違いは，STEP 1 で生成する縦ベクトルの成分と STEP 4 でのチェックを行列に対して行うかベクトルに対して行うかという 2 点のみである.

蛇足 このアルゴリズムはアルゴリズム 1 の計算効率を調べる時に比較の為に作成したものである．そのため，一つの列に対する最小多項式を求めるアルゴリズムとしては余計な計算を行う可能性のある箇所がある．ここでは，プログラムの改良は行っていない.

4 プログラム

これら 2 つのアルゴリズムを，数式処理システム Risa/Asir 用のプログラムとして実装した．仕様は以下の通りである.

- 機能：
  - **(a)** 行列全体の最小多項式を計算
  - **(b)** 行列の一つの列に対する最小多項式を計算
- 呼出し形式：
  - **(a)** …関数名（Mat, Poly)
  - **(b)** …関数名（Mat, Poly, Col)
- 入力：
  - Mat $\leftarrow n$ 次正方行列
  - Poly ←Mat の特性多項式
  - Col ← 列番号（(b) の場合）
- 出力：
  - **(a)** …行列全体の最小多項式の [因子，因子の重複度]
    $([[f_1(\lambda), r_1], \cdots, [f_s(\lambda), r_s]])$
  - **(b)** …行列の第 l 列に対する最小多項式の [因子，因子の重複度]
    $([[f_1(\lambda), r_{l,1}], \cdots, [f_s(\lambda), r_{l,s}]])$

5 時間計測実験

論文 [3] で発表した行列の最小多項式を求めるプログラムと今回の定理 1 を利用したアルゴリズム 1 のプログラムの CPU 時間を測定，比較した．またアルゴリズム 1 と 2 のプログラムも比較した．

5.1 測定環境

- 使用ソフト…数式処理システム Risa/Asir
- 実験で使用したコンピュータ
  - OS…Windows XP Professional
  - CPU…Intel Pentium(R)4 CPU 3.20GHz
  - メモリ…2.00GB

5.2 実験 1 （以前のプログラムとアルゴリズム 1 のプログラム比較）

5.2.1 測定方法

- アルゴリズム STEP4 の最小多項式の候補のチェックは共に行わずにそれぞれ測定
- n 次正方行列に対する最小多項式の各因子 $f_i(\lambda)$ の係数は，-1024～1024 の範囲の整数（最高次を除く）
- 測定に使用する行列は，要素が-1024～1024 の範囲の行列を 10 回作成してビット長の平均を取り，それとの誤差が ±10 %となるように作成
- 特性多項式の因子の次数は 4 で固定，行列のサイズは 320，特性多項式の重複度は全て 20 で固定
- 最小多項式の重複度を 1 ～ 4 の間でランダム，９～１２の間でランダム，１７～２０の間でランダムと変えてそれぞれ測定
- CPU 時間は Asir の time() で取得
- Asir の設定はデフォルト

5.2.2 測定結果

表 1 ：実験結果

因子の重複度	CPU 時間 (sec)	
	以前の求め方	今回の求め方
1 ～ 4 でランダム	15.58	40.66
９～１２でランダム	32.69	34.02
１７～２０でランダム	69.63	34.83

5.3 実験２（アルゴリズム１とアルゴリズム２のプログラム比較）

5.3.1 測定方法

- アルゴリズム STEP5 の最小多項式の候補のチェックを行った場合と行わない場合をそれぞれ測定
- n 次正方行列に対する最小多項式の各因子 $f_i(\lambda)$ の係数は，-1024～1024 の範囲の整数（最高次を除く）
- 測定に使用する行列は，要素が-1024～1024 の範囲の行列を 10 回作成してビット長の平均を取り，それとの誤差が ±10 %となるように作成
- 特性多項式の因子の次数は 4 で固定
- 行列のサイズを 160，240，320（特性多項式の重複度はそれぞれ 10,15,20）と変えてそれぞれ測定
- 最小多項式の重複度は特性多項式の重複度に近い範囲でランダム
- 行列全体の最小多項式計算では 2 回，行列の一つの列に対する最小多項式計算ではランダムに列を 5 つ選び一回ずつ測定し，平均を算出
- CPU 時間は Asir の time() で取得
- Asir の設定はデフォルト

5.3.2 測定結果

表２：最小多項式の候補のチェックを行わない場合

サイズ	CPU 時間 (sec)	
	行列全体	任意の一列
160	3.73	3.34
240	13.91	13.43
320	33.84	27.41

表３：最小多項式の候補のチェックを行った場合

サイズ	CPU 時間 (sec)	
	行列全体	任意の一列
160	80.69	5.08
240	541.08	20.39
320	1589.95	38.31

5.4 結論

実験１ 以前のアルゴリズムでは最小多項式の重複度が大きくなると最小消去多項式候補を求めるための計算時間が相当増えるが今回のアルゴリズムでは重複度に影響されないことが分かる．また，最小多項式の重複度が小さい行列に対しては以前のアルゴリズムの方が効率的に最小多項式候補を求めているが，重複度が大きい場合は今回導出したアルゴリズムの方が計算所要時間が短いことがわかる．したがって，最小多項式の重複度が大きいと予想されるような行列に対しては本稿で提案した方法を適用することがより効率的となる．

実験2 最小多項式の候補のチェックを行わない場合はアルゴリズム1とアルゴリズム2のCPU時間に大きな差はないことが確認できた. このことは, 行列の最小多項式候補を求める際の計算量が行列の一つの列に対する最小多項式候補（最小消去多項式候補）を求める際のそれとほとんど変わらないことになる. つまり, 行列の最小多項式候補を求める計算法としてこの手法が優れたものであることを意味する. チェックを行った場合はCPU時間に大きな差が現れる. これは勿論, 与えられた行列の最小多項式候補が真の最小多項式であるか否かをチェックするためには行列計算が必要となることによる. 最小多項式を確実に求める必要がある場合は, 最小多項式候補を求めた後, チェックのため行列計算を並列化することで計算効率の向上を図る必要があると結論できる.

6 まとめ・今後の課題

今回の研究では本稿の第2節に与えた定理に基づくことで, 行列の最小多項式を求める新たなアルゴリズムを導出した. このアルゴリズムを数式処理システムRisa/Asirに実装し, 最小多項式が重複度の高い因子からなるような場合は今回提案した計算方法の方が以前に導出した計算方法に比べ実際に計算効率が良いことを示した.

本稿で与えたアルゴリズムは, 特性多項式のすべての因子に対し, 最小多項式としての重複度を求めるものであるが, 導出の基礎をなった定理の主張からも明らかなように, 注目した幾つかの因子に対してのみ最小多項式の因子としての重複度の下からの評価を与えるようなアルゴリズムを作ることも容易である. 従って, 与えられた行列の特性多項式が重複度の高い因子を幾つか持ち, 最小多項式の因子としても重複度が高い可能性があるような場合に, 本稿で提案した計算方法をそれらの因子に対してのみ適用し, 他の因子に対しては別の計算方法を適用する等の対策をとることで最小多項式の計算法を改良することが可能と思われる. 最小多項式計算の改良に関しては今後の課題としたい.

参 考 文 献

[1] 笠原晧司：線形代数と固有値問題ースペクトル分解を中心にー, 現代数学社, 1972.

[2] F. シャトラン（訳：伊理正夫・伊理由実）：行列の固有値ー最新の解法と応用ー, シュプリンガー・フェアラーク東京, 1997.

[3] 田島慎一：微分作用素を用いたレゾルベントの留数解析と行列のスペクトル分解, 数理解析研究所講究録掲載予定.

[4] 奈良洸平, 田島慎一, 小原功任：行列の最小多項式の計算について, 数式処理 第16巻 第2号 (2009).