Vol.2018-MPS-119 No.11
2018/7/30

DQDS法とOQDS法を用いた列空間の計算法

荒木 翔[1,a)]　田中 博基[2,b)]　髙田 雅美[3,c)]　木村 欣司[4,d)]　中村 佳正[1,e)]

概要：本稿では, Sakurai-Sugiura 法による行列の一般化固有値分解において用いられる長方行列の列空間を計算するために, DQDS（differential qd with shift）法と OQDS（orthogonal qd with shift）法を組み合わせた計算法を提案する. 列空間の計算においては, まず Householder 変換を用いて, 与えらえれた長方行列を 2 つの直交行列と 2 重対角行列に分解したのち, 2 重対角行列の列空間を求める. 提案法では, 2 重対角行列の列空間の計算において, 最初に DQDS 法を用いて行列の数値ランクを求める. 次に DQDS 法で得られた数値ランクを用いて, OQDS 法によって下 2 重対角行列の行空間を求めることができる. この下 2 重対角行列の行空間は, 上 2 重対角行列の列空間に等しい. 数値実験では, 2 重対角行列の全ての右特異ベクトルを求める従来法と, 提案法との性能比較を行う. 比較の指標として, 行空間の計算に要した時間と, 得られた行空間の直交性を用いる.

Computational Method of Column Space using the DQDS and the OQDS Methods

Sho Araki[1,a)]　Hiroki Tanaka[2,b)]　Masami Takata[3,c)]　Kinji Kimura[4,d)]
Yoshimasa Nakamura[1,e)]

1. はじめに

対称行列と正定値対称行列で構成される一般化固有値問題は, 分子軌道法やデータクラスタリングなどのアプリケーションで利用されている. これらのアプリケーションは, 与えられた行列の全ての固有値と固有ベクトルではなく, 一部の固有値とそれに対応する固有ベクトルのみが必要となる. 部分的な固有値分解を行う手法として, Sakurai-Sugiura 法 [1] が提案されている. Sakurai-Sugiura 法は, 長方行列の左零空間を除いた左特異ベクトルによって張られる列空間を用いて, 特定の円形領域内の固有値とそれに対応する固有ベクトルを求める手法である. 列空間を特異値分解によって求める場合, 特異値分解は, 多くの計算コストを必要とするため, 計算時間が長いことや丸め誤差の蓄積による精度の悪化という問題点が存在する. そこで, Sakurai-Sugiura 法が必要とするのは, 左特異ベクトルではなく列空間であるという事実に注目し, 特異値分解を行うことなく列空間を求めることで, 計算時間と精度の改善を行う.

本稿では, DQDS (differential qd with shift) 法 [2], [3] と OQDS (orthogonal qd with shift) 法 [4] を組み合わせることによって, 上 2 重対角行列の列空間を計算する方法を提案する. このとき, 前述の長方行列はあらかじめ Householder 変換 [5] によって 2 つの直交行列と上 2 重対角行列との積の形に変形されているものとする. 提案法では, まず全ての特異値の分布を調べるために, DQDS 法を適用し, 得られた特異値の分布から, 上 2 重対角行列の数値ランクを確定する. さらに OQDS 法を用いて, 転置された下 2 重対角行列の行空間を取得する. 得られた行空間は, 元の上 2 重対角行列の列空間に等しいことを注意する.

[1] 京都大学
Kyoto University, Kyoto, Kyoto 606–8501, JAPAN
[2] 立命館大学
Ritsumeikan University, Kusatsu, Shiga 525–8577, JAPAN
[3] 奈良女子大学
Nara Women's University, Nara, Nara 630–8506, JAPAN
[4] サレジオ工業高等専門学校
Salesian Polytechnic, Machida, Tokyo 194–0215, JAPAN
[a)] araki@amp.i.kyoto-u.ac.jp
[b)] rokitaniplus1@gmail.com
[c)] takata@ics.nara-wu.ac.jp
[d)] kimura.kinji.7z@kyoto-u.ac.jp
[e)] ynaka@i.kyoto-u.ac.jp

Algorithm 1 DQDS 法

Set $d_1 := q_1 - s$;
for $k := 1, 2, \cdots, n-1$ **do**
  Set $\hat{q}_k := d_k + e_k$;
  if $SM \times \hat{q}_k < q_{k+1}$ and $SM \times q_{k+1} < \hat{q}_k$ **then**
    Set $\hat{e}_k := (q_{k+1}/\hat{q}_k)\, e_k$;
    Set $d_{k+1} := (q_{k+1}/\hat{q}_k)\, d_k - s$;
  else
    Set $\hat{e}_k := (e_k/\hat{q}_k)\, q_{k+1}$;
    Set $d_{k+1} := (d_k/\hat{q}_k)\, q_{k+1} - s$;
  end if
end for
Set $\hat{q}_n := d_n$;

数値実験では, 計算時間と計算された行空間の直交性という基準を用いて, 従来法と提案法を比較する. なお, 従来法では, すべての右特異ベクトルを計算した後, 行空間を取得するものとする.

本稿の第 2 章では, qd 法の改良である DQDS 法と OQDS 法について紹介する. 第 3 章では, 列空間を求めるための計算法を提案し, 第 4 章で数値実験により提案法の計算時間と計算された行空間の直交性を評価する.

2. qd 法

本章では提案手法で用いる DQDS 法と OQDS 法について概説する.

2.1 DQDS 法

$n \times n$ 次 2 重対角行列 B を,

$$B = \begin{pmatrix} \sqrt{q_1} & \sqrt{e_1} & & & \\ & \sqrt{q_2} & \ddots & & \\ & & \ddots & \sqrt{e_{n-1}} & \\ & & & \sqrt{q_n} \end{pmatrix}, \qquad (1)$$

とする. DQDS 法を用いて, 2 重対角行列 B を, $n \times n$ 上 2 重対角行列 $\hat{B}$ に変換する,

$$\hat{B} = \begin{pmatrix} \sqrt{\hat{q}_1} & \sqrt{\hat{e}_1} & & & \\ & \sqrt{\hat{q}_2} & \ddots & & \\ & & \ddots & \sqrt{\hat{e}_{n-1}} & \\ & & & \sqrt{\hat{q}_n} \end{pmatrix}. \qquad (2)$$

Algorithm 1 は, DQDS 法の 1 反復の操作を表す. ここで, 変数 s は, シフト量を表し, SM は, $1/SM$ がオーバーフローを起こさない最小の正数である. この操作の反復により, q_i は, 行列 B の特異値 σ_i に, e_i は 0 に収束することが示されており, これによって行列 B の特異値を得る.

なお, DQDS 法の変数 d_k の計算は, $f \times g + h$ の形であらわされるが, 現行の多くの汎用 CPU においては, この形式の融合積和演算を高精度に行うことが可能となっている. そのため, DQDS 法は実装面でも, 高精度性を期待できる.

DQDS 法では, シフトを導入することによって, 収束までの反復回数を減らすことができる. この場合, Σ を特異値 $\sigma_1, \ldots, \sigma_n$ を成分にもつ対角行列とすると, Algorithm 1 の反復によって, $\hat{B}$ は, 対角行列 D に収束する. この際, D は, $\Sigma_{kk} = \sqrt{D_{kk} + S}$ $(k = 1, \cdots, n)$ を満たす. ここで, S は, シフト量 s の総和である.

DQDS 法では, $q_i \geq 0 (i = 1, \ldots, n)$, $e_i \geq 0 (i = 1, \ldots, n-1)$ であり, また, 変数 d_k も $d_k (k = 1, \cdots, n-1) > 0$ を満たす必要がある. もし, $d_k = 0$ ならば, $s > 0$ であるため, $d_{k+1} < 0$ となり, DQDS 法の計算は, 値の非負性を保てないという意味で, 失敗とみなす. 逆に, $\hat{q}_n = d_n = 0$ は, 問題ない. ゆえに, シフト量 s は, $B^\top B$ の最小固有値 $\lambda_{\min}\left(B^\top B\right)$ 以下の値に設定する必要がある.

また前述の通り, シフトが導入された特異値計算では, 得られた対角行列にシフト量の総和を加えなければならない. i 番目の反復において計算されたシフト量 s を $s^{(i)}$ とすると, シフト量の総和 S は,

$$S = \sum_{i=1}^{i_0} s^{(i)}, \qquad (3)$$

と表される. 式 (3) の計算において, 既に計算されたシフトの量の総和に対し, 相対的に小さい数 $s^{(i)}$ が足しこまれることがある. この和算の際に情報落ちが生じる. この問題を解決するために, 1 つの数を 2 つの倍精度変数で表すことができる double-double 計算 [6] を適用すべきである.

DQDS 法が終了する際, 対角要素は, その特異値の昇順に並べられている. 実装上は, $q_1 < q_n$ の場合, B の要素を次のように置き換える,

$$B_r = \begin{pmatrix} \sqrt{q_n} & \sqrt{e_{n-1}} & & & \\ & \sqrt{q_{n-1}} & \ddots & & \\ & & \ddots & \sqrt{e_1} & \\ & & & \sqrt{q_1} \end{pmatrix}. \qquad (4)$$

この置き換えを用いた場合でも, B_r の特異値は B の特異値に等しい.

2.2 OQDS 法

特異値を計算する別の方法として, OQDS 法がある. OQDS 法は数学的には DQDS 法と等価な手法であるが, 相対誤差の面で, より高精度に特異値を計算するのに適した手法である. 一方で, OQDS 法はその定式化において平方根演算を含むという欠点がある. これに対して DQDS 法は, 変数変換によって平方根演算の除去を行っており, OQDS 法に比べて高速に実行することができる. しかし, DQDS 法は, OQDS 法と違い, 特異ベクトルを計算することができない. 以上より, 本稿では, DQDS 法を特異値の分布の取

Algorithm 2 $u^{(i)} > 0$ の場合の LU ステップ

Set $\rho_1^{(i)} := \sqrt{\alpha_1^{(i)} - u^{(i)}}\sqrt{\alpha_1^{(i)} + u^{(i)}}$;
for $k := 1, 2, \cdots, n-1$ **do**
  Set $\gamma_k^{(i)} := \sqrt{\left(\rho_k^{(i)}\right)^2 + \left(\beta_k^{(i)}\right)^2}$, $\zeta_k^{(i)} := \frac{\beta_k^{(i)}}{\gamma_k^{(i)}}\alpha_{k+1}^{(i)}$;
  Set $\rho_{k+1}^{(i)} := \sqrt{\frac{\rho_k^{(i)}}{\gamma_k^{(i)}}\alpha_{k+1}^{(i)} - u^{(i)}}\sqrt{\frac{\rho_k^{(i)}}{\gamma_k^{(i)}}\alpha_{k+1}^{(i)} + u^{(i)}}$;
end for
Set $\gamma_n^{(i)} := \rho_n^{(i)}$;

Algorithm 3 $u^{(i)} = 0$ の場合の LU ステップ

Set $\rho_1^{(i)} := \alpha_1^{(i)}$;
for $k := 1, 2, \cdots, n-1$ **do**
  Set $\gamma_k^{(i)} := \sqrt{\left(\rho_k^{(i)}\right)^2 + \left(\beta_k^{(i)}\right)^2}$;
  if $\gamma_k^{(i)} = 0$ **then**
    Set $\zeta_k^{(i)} := 0$, $\rho_{k+1}^{(i)} := \alpha_{k+1}^{(i)}$;
  else
    Set $\zeta_k^{(i)} := \left(\beta_k^{(i)}/\gamma_k^{(i)}\right)\alpha_{k+1}^{(i)}$;
    Set $\rho_{k+1}^{(i)} := \left(\rho_k^{(i)}/\gamma_k^{(i)}\right)\alpha_{k+1}^{(i)}$;
  end if
end for
Set $\gamma_n^{(i)} := \rho_n^{(i)}$;

得のために, OQDS 法を特異ベクトル計算のために, それぞれ用いている.

$L^{(i)}$ と $U^{(i)}$ を, それぞれ, $n \times n$ 下 2 重対角行列と $n \times n$ 上 2 重対角行列とする,

$$L^{(i)} = \begin{pmatrix} \alpha_1^{(i)} & & & \\ \beta_1^{(i)} & \alpha_2^{(i)} & & \\ & \ddots & \ddots & \\ & & \beta_{n-1}^{(i)} & \alpha_n^{(i)} \end{pmatrix}, \tag{5}$$

$$U^{(i)} = \begin{pmatrix} \gamma_1^{(i)} & \zeta_1^{(i)} & & \\ & \gamma_2^{(i)} & \ddots & \\ & & \ddots & \zeta_{n-1}^{(i)} \\ & & & \gamma_n^{(i)} \end{pmatrix}. \tag{6}$$

上記の $L^{(i)}$ と $U^{(i)}$ を用いると, OQDS 法の 1 反復は, 次の 3 つの操作によって表すことができる.

(1) シフト量 $u^{(i)}$ の計算

$$0 \le u^{(i)} \le \sigma_{\min}\left(L^{(i)}\right). \tag{7}$$

(2) LU ステップ

$$P^{(i)} \begin{pmatrix} L^{(i)} \\ t^{(i)} I_n \end{pmatrix} = \begin{pmatrix} U^{(i)} \\ t^{(i+1)} I_n \end{pmatrix}, \tag{8}$$

$$t^{(i+1)} = \sqrt{\left(t^{(i)}\right)^2 + \left(u^{(i)}\right)^2} \tag{9}$$

$u^{(i)} > 0$ の場合の LU ステップを, Algorithm 2 に, $u^{(i)} = 0$ の場合の LU ステップを, Algorithm 3 に示す.

(3) UL ステップ

$$\begin{pmatrix} I_n & O \\ O & \left(Q^{(i)}\right)^\top \end{pmatrix} \begin{pmatrix} U^{(i)} \\ t^{(i+1)} I_n \end{pmatrix} Q^{(i)} = \begin{pmatrix} L^{(i+1)} \\ t^{(i+1)} I_n \end{pmatrix}. \tag{10}$$

行列の下半分は, 自明である. ゆえに, $U^{(i)}Q^{(i)} = L^{(i+1)}$ のみ考慮すればよい. UL ステップを, Algorithm 4 に示す.

Algorithm 4 UL ステップ

Set $\eta_1^{(i)} := \gamma_1^{(i)}$;
for $k := 1, 2, \cdots, n-1$ **do**
  Set $\alpha_k^{(i+1)} := \sqrt{\left(\eta_k^{(i)}\right)^2 + \left(\zeta_k^{(i)}\right)^2}$;
  if $\alpha_k^{(i+1)} = 0$ **then**
    Set $\beta_k^{(i+1)} := 0$, $\eta_{k+1}^{(i)} := \gamma_{k+1}^{(i)}$;
  else
    Set $\beta_k^{(i+1)} := \left(\zeta_k^{(i)}/\alpha_k^{(i+1)}\right)\gamma_{k+1}^{(i)}$;
    Set $\eta_{k+1}^{(i)} := \left(\eta_k^{(i)}/\alpha_k^{(i+1)}\right)\gamma_{k+1}^{(i)}$;
  end if
end for
Set $\alpha_n^{(i)} := \eta_n^{(i)}$;

$P^{(i)}$ と $Q^{(i)}$ は, $2n \times 2n$ 直交行列と $n \times n$ 直交行列である. $P^{(i)}$ は, Givens 回転と一般化 Givens 回転を用いることによって計算できる [4]. $Q^{(i)}$ は, Givens 回転で構成されている. $t^{(i)}I_n$ は対角行列であり, すべての要素は同じ値である. Σ は, 特異値が大きい順に並んでいる対角行列である. SPLIT [3] が生じていない場合, $L^{(i_0)}$ と $t^{(i_0)}I_n$ が, それぞれ, 対角行列 E と tI_n に収束するならば, $\Sigma_{kk} = \sqrt{E_{kk}^2 + t^2}$ $(k = 1, \cdots, n)$ となる. 右特異ベクトルを得るために, 直交行列 V は, $V = Q^{(0)} \cdots Q^{(i_0-1)}$ と表される. 最後に, Givens 回転を用いることで, $E = L^{(i_0)}$ に tI_n を加える. これにより, $L^{(i_0)}$ と tI_n は, Σ と 0 になる. なお, 左特異ベクトルからなる直交行列 U は, Sakurai-Sugiura 法では用いられない. ゆえに, 本稿では, U を計算する際に必要となる $P^{(i)}$ の説明を省略する.

シフト量の総和は,

$$t^{(i+1)} = \sqrt{\left(t^{(i)}\right)^2 + \left(u^{(i)}\right)^2}, \tag{11}$$

によって計算される. この際, それまでの総和の値に対して, 非常に小さい値が足しこまれるために, 情報落ちが生じる. この問題を回避するために, double-double 計算を適用すべきである.

OQDS 法において, SPLIT 操作が発生せずに終了した場合, E の対角成分は降順に並んでいる. そのため, $\alpha_1^{(i)} < \alpha_n^{(i)}$ となる場合,

$$\begin{pmatrix} U^{(i)} \\ t^{(i)} I_n \end{pmatrix} \leftarrow \begin{pmatrix} Y & O \\ O & Y \end{pmatrix} \begin{pmatrix} L^{(i)} \\ t^{(i)} I_n \end{pmatrix} Y, \tag{12}$$

を用いて, すべての要素を逆順に入れ替える. ここで, 行列 Y は,

$$Y = \begin{pmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{pmatrix}, \tag{13}$$

である.

3. 提案法

本章では, 上 2 重対角行列の列空間を計算するための方法を提案する. 前章で説明した通り, DQDS 法は OQDS 法に比べ特異値の高速計算に適しているため, 全特異値の分布情報の取得に用いる. この特異値の分布情報から, 与えられた行列の数値ランクを決定することができる. 与えられた $n \times n$ 行列 の K 個の特異値が 0 近傍にある場合, 数値ランクは, $n - K$ となる. OQDS 法の実行中に SPLIT が生じなかったと仮定した場合, OQDS 法で計算された E の対角成分は降順に並べられる. すなわち, OQDS 法において, 非対角成分が完全に 0 にならない場合, 小さい方から K 番目までの特異値に対応する右特異ベクトルを計算できた可能性が高い. OQDS 法は, K 個の 0 近傍特異値に対応する右特異ベクトルとして, 下 2 重対角行列 $L^{(0)}$ の 零空間を計算することができる. このとき, OQDS 法を用いた零空間の計算コストは, すべての右特異ベクトルを計算するためのコストと比べて非常に小さい. また, 零空間に対応する特異ベクトルを計算すると同時に, その補空間として行空間を得ることができる. ここで, 下 2 重対角行列 $L^{(0)}$ の行空間は, 上 2 重対角行列 $\left(L^{(0)}\right)^\top$ の列空間に等しい. ゆえに, 列空間は, 入力行列 $L^{(0)}$ に対する OQDS 法で計算される $n \times n$ 行列 $V = Q^{(0)} \cdots Q^{(i_0-1)}$ の最初の $n - K$ 個のベクトルから得ることができる. 最後に, OQDS 法によって計算された特異値と DQDS 法によって計算された特異値を比較することで, 零空間が正しく計算できているかを確認する.

4. 数値実験

提案法の性能を確認するために, 計算された行空間の直交性と計算時間を比較する. 比較対象は, OQDS 法を用いてすべての右特異ベクトルを計算する従来法とする. 実験環境は **表 1** の通りである. 実験のための行列の行列は, 次の性質を満たす.

$$\sigma_i := \varepsilon^{\frac{i-1}{n-1}}, \quad i = 1, \cdots, n - t_0 \tag{14}$$

$$\sigma_i := \varepsilon^{2\frac{i-1}{n-1}}, \quad i = n - t_0 + 1, \cdots, n \tag{15}$$

ここで, σ_i は, 特異値を意味する. ε は倍精度浮動小数点数のマシンイプシロン $2.22044604925031 \times 10^{-16}$ に, t_0 は 20 に設定する. 行列サイズ n は, 128 とする. この行列の 0 に近い特異値の数 K は, DQDS 法によって取得する. 相対ギャップは, DQDS 法によって計算された特異値 $\hat{\sigma}_i$ を用いて, $\hat{\sigma}_{i+1}/\hat{\sigma}_i$ と定義する. この相対ギャップから, K は t_0 とおくことができる. **表 2** は, 計算された行空間の直交性と計算時間を表す. 直交性は, $||V^\top V - I||_F$ の計算によって得られたものである.

表 1 実験環境

CPU	Intel(R) Core(TM) CPU i3-7100 @ 3.90GHz
RAM	4 GB
OS	Ubuntu 16.04.3 LTS
Compiler	gcc version 5.4.0, gfortran version 5.4.0
Options	-O3 -mtune=native -march=native
	-Wall -fopenmp
Software	Intel Math Kernel Library 2018

表 2 直交性と計算時間

	従来法	提案法
直交性	1.23×10^{-14}	4.76×10^{-15}
計算時間	5.27×10^{-4}	1.03×10^{-4}

5. まとめ

本稿では, 長方行列の列空間を計算するために, DQDS 法と OQDS 法を組み合わせた計算法を提案した. この提案法では, すべての特異値の分布を確認するために, DQDS 法を適用した後, OQDS 法を用いて, 下 2 重対角行列の行空間を得る. また, 数値実験の結果から, 提案法は, 計算時間と得られた列空間の直交性について, 従来法に対する改善がみられることを確認した.

今後の課題として, 提案法を実際に Sakurai-Sugiura 法に適用することが挙げられる.

謝辞 本研究は JSPS 科研費 JP17H02858 と JP17H00167 の助成を受けたものです.

参考文献

[1] Sakurai, T., and Tadano, H.: *CIRR: a Rayleigh-Ritz type method with counter integral for generalized eigenvalue problems*, Hokkaido Math. J., Vol. 36, pp. 745–757 (2007).
[2] Fernando, K. V., and Parlett, B. N.: *Accurate singular values and differential qd algorithms*, Numer. Math., Vol.67, pp.191–229 (1994).
[3] Parlett, B. N., and Marques, O. A.: *An Implementation of the dqds Algorithm (Positive Case)*, Lin. Alg. Appl, Vol. 309, No. 1-3, pp. 217–259 (2000).
[4] Matt, U. von: *The orthogonal qd-algorithm*, SIAM J. Sci. Comput., Vol. 18, pp. 1163–1186 (1997).
[5] Demmel, J.: *Applied Numerical Linear Algebra*, SIAM, Philadelphia (1997).
[6] Hida, Y., Li, X. S., and Bailey, D. H.: *Library for Double-Double and Quad-Double Arithmetic*, Proc. 15th Symposium on Computer Algorithmetic, pp.155–162 (2007).