日本応用数理学会論文誌
Vol. 16, No. 4, 2006, pp. 507～534

密行列固有値解法の最近の発展（II）
- マルチシフトQR法 -

山本有作
名古屋大学大学院工学研究科計算理工学専攻

Recent Developments in Algorithms for Solving Dense Eigenproblems (II)
- Multishift QR Algorithms -

Yusaku Yamamoto
Department of Computational Science & Engineering, Nagoya University

Abstract. The QR algorithm is one of the most reliable and widely used methods to compute the eigenvalues of symmetric and nonsymmetric matrices. However, it is not straightforward to execute the QR algorithm efficiently on modern architectures such as processors with hierarchical memory or parallel computers because of its inherent sequential nature and low data reference locality. To overcome this difficulty, Bai & Demmel proposed the multishift QR algorithm in 1989 and this idea has been greatly expanded since then. In this paper, we introduce the basic theory of the multishift QR algorithm and review recent developments to improve its efficiency, such as the two-tone QR algorithm, aggressive early deflation and the fully-pipelined multishift QR algorithm. Directions for future research are also discussed.

1 はじめに

固有値の計算は科学技術計算の様々な分野で現れる重要な線形計算の一つである [2]．本論文では，$n \times n$の行列Aに対する標準固有値問題

$$A\mathbf{x} = \lambda\mathbf{x} \tag{1.1}$$

において，特にAが対称あるいは非対称の密行列である場合を考える．近年では，シミュレーションの大規模化・複雑化に伴って解くべき行列も大型化しつつあり，たとえば分子軌道法によるたんぱく質の解析 [31][38] では$n = 10^5$以上の対称密行列の固有値・固有ベクトル計算が必要とされつつある．また，流体力学の分野では，乱流における相関関数の計算 [52] のため，$n = 10^5$程度の非対称密行列の固有値計算が必要となる場面が生じている．このような問題を扱うには，大規模行列に対して高速・高精度・安定に解を求められるアルゴリズムが重要である．特に，最近のプロセッサで性能を出すにはキャッシュの有効利用が必須であり，この点を十分考慮したアルゴリズムが求められる．また，大規模計算では並列計算機の利用が不可欠であり，アルゴリズムが並列計算向きであることも重要である．

密行列の固有値計算は，通常，次のようなステップに従って行われる [19][32][1].

(1) 相似変換により，密行列 A をヘッセンベルグ行列 H（非対称行列の場合）あるいは対称三重対角行列 T（対称行列の場合）に変形する.

(2) H または T の固有値 $\{\lambda_i\}$，固有ベクトル $\{\mathbf{u}_i\}$ を求める.

(3) 逆変換により，$\{\mathbf{u}_i\}$ から A の固有ベクトル $\{\mathbf{v}_i\}$ を求める.

このうち，ステップ (1) は通常ハウスホルダー法により行われる．(2) の固有値・固有ベクトルの計算は，非対称行列の場合，QR 法 [29][30][45] が最も信頼性の高いアルゴリズムとされており，標準的な解法となっている[2]．一方，対称行列の場合には様々な解法があり，標準的な行列計算ライブラリである LAPACK[1] だけを見ても，QR 法，二分法・逆反復法，分割統治法，MR^3 アルゴリズムなどが実装されている．このうち QR 法は，二分法・逆反復法などと並んで，必ずしも最高速ではないが最も信頼性の高い解法として広く使われている．特に，縮重に近い固有値を持つ対称三重対角行列の場合[3]，二分法・逆反復法 [61]，分割統治法 [34][35]，MR^3 アルゴリズム [22][21] では固有ベクトルの直交性を確保するために様々な工夫を要するが，QR 法では固有ベクトルの行列が直交行列の積として計算されるため，直交性は自動的に保証される．この点は他の解法にない大きな長所である.

全体の演算量の中での QR 法の割合を考えると，非対称行列の場合，ヘッセンベルグ行列への変換は $\frac{10}{3}n^3$ であるのに対し，ダブルシフト QR 法による固有値計算には約 $10n^3$ の演算量が必要である．したがって QR 法は演算量の大部分を占め，その高速化は重要である．一方，対称行列の場合は，三重対角化の演算量が $\frac{4}{3}n^3$ であるのに対し，QR 法による固有値計算の演算量は $O(n^2)$ と小さい．しかし，三重対角化のアルゴリズムに関しては，キャッシュ向けの最適化手法 [25][7][63]，および分散メモリ型並列計算機向けの効率の良い並列化手法 [13][24] が開発されており，多数のプロセッサを使うことにより大幅な時間短縮が可能である．そのため，QR 法による固有値計算の部分についても，より高速化ができることが望ましい.

しかし，オリジナルの QR 法を最近の高性能計算機上で効率よく実行することは容易ではない．最近のプロセッサでは，動作周波数の向上に伴ってピーク性能は数 GFLOPS にまで向上しているが，主記憶の速度向上はそれに比べて遅く，両者の速度差は開きつつある．このような状況でプロセッサの性能を引き出すには，データ参照の局所性が高く，キャッシュメモリを有効に利用できるアルゴリズムが必須である [32][55]．また，最近では PC 用のプロセッサもマルチコア化が進み，共有メモリ型並列計算機が一段と身近になりつつあるが，その性能を引き出すには，大きな並列粒度を持ち，プロセッサ間の同期が少なくて済むアルゴリズムが不可欠である [60]．ところが，オリジナルの非対称行列向け QR 法は，1 反復ごとに行列全体へのアクセスが必要なためにデータ参照の局所性が低く，並列粒度も高々 $O(n)$ と小さいためにプロセッサ間の同期が頻発する．また，対称行列向け QR 法は本質的に逐次型の

[1]ただし，対称行列向けには三重対角行列を経由しないヤコビ法もあり，通常の方法に比べて精度が上回るという報告もある [18].

[2]非対称行列向けの分割統治法も提案されているが [5]，安定性に問題があり，広く使われるには至っていない.

[3]既約（すべての副対角要素が非ゼロ）な対称三重対角行列は，縮重固有値を持つことはない [48].

アルゴリズムである．このように，オリジナルのQR法の特徴は，高性能計算のための要求と相反する．

そこで，QR法を高性能計算機上で効率良く実行するため，様々な改良が行われてきた．そのうちもっとも有望な方法がマルチシフトQR法である．マルチシフトQR法とは，QR法において複数回の反復をまとめて1回の反復として行い，複数個の固有値を同時に求める方法の総称である．これは，2個のシフトを用いて2個の固有値を同時に求めるダブルシフトQR法の自然な拡張になっており，複数回の反復をまとめることにより，データ参照の局所性と並列粒度を向上させることを可能とする．マルチシフトQR法は1989年にBai & Demmel[3]により（非対称行列向けに）提案され，その後LAPACKにも取り入れられている．Bai & Demmelのアルゴリズムはシフトを多数用いると数値的安定性が損なわれるため，データ参照の局所性を十分高められないという問題があったが，最近になってこの問題点を解決したTwo-Tone QR法[10]と呼ばれる新しいマルチシフトQR法がBramanらによって提案された．さらにBramanらは，収束途上の行列から固有値を取り出せるaggressive early deflation[11]と呼ばれる新しいデフレーションの手法を提案し，従来のQR法の演算量を大きく削減できることを示した．一方，従来困難と思われてきた対称行列向けQR法の並列化についても，マルチシフトQR法の様々な改良により，効率の良いアルゴリズムを構築できる可能性が出てきた[42][54][46].

本論文では，このマルチシフトQR法を取り上げ，非対称，対称両方のアルゴリズムについて，最近までの発展をサーベイする．また，収束性，安定性についての理論も紹介する．なお，以下ではすべて実行列に対して議論を行うが，複素非対称行列についても，マルチシフトQR法に関する技法の多くはそのまま適用できる．ただし，複素非対称行列の中でも，ユニタリ行列などの特別な行列に対しては，専用の解法が提案されている[17]．また，複素エルミート行列に対しては，ユニタリ変換によりエルミート三重対角行列に変換でき，さらに対角行列によるユニタリ変換で実三重対角行列に変換できる．そのため，実三重対角行列に対するQR法のみを考えても一般性を失わないことに注意する．

本論文の構成は以下の通りである．まず，第2章では非対称QR法とその高性能計算における問題点を説明し，それを解決する手法としてマルチシフトQR法を導入する．また，最近の発展であるTwo-Tone QR法とaggressive early deflationについても紹介する．次に第3章で対称行列向けのQR法とその並列化における困難について説明し，マルチシフトQR法により並列化が可能となることを示す．また，並列化効率を向上させる手法である遅延シフト，最近の発展であるfully pipelinedマルチシフトQR法について紹介する．最後に第4章で，まとめと今後の課題について述べる．

2 非対称行列向けのマルチシフトQR法

2.1 従来の非対称行列向けQR法

2.1.1 ダブルシフトQR法

非対称行列に対するQR法では，行列A_0から出発して，次のようにQR分解と行列Qによる相似変換とを繰り返してゆく[19][32].

$$\begin{aligned} A_0 &= Q_0R_0 \\ A_1 &= R_0Q_0 \quad (= Q_0^{-1}A_0Q_0) \\ A_1 &= Q_1R_1 \\ A_2 &= R_1Q_1 \quad (= Q_1^{-1}A_1Q_1 = Q_1^{-1}Q_0^{-1}A_0Q_0Q_1) \end{aligned} \tag{2.1}$$

このとき，適当な条件の下で，行列A_kは（ブロック）上三角行列に収束する．たとえば，A_0のn個の固有値$\lambda_1, \lambda_2, \ldots, \lambda_n$が

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0 \tag{2.2}$$

を満たす場合，A_kは上三角行列に収束し，その対角要素$a_{ii}^{(k)}$は固有値λ_iに1次収束する．また，非対角要素a_{ij}（$j < i$）は収束率$\rho_{ij} \equiv |\lambda_i|/|\lambda_j|$で0に1次収束する．

いま，ある固有値λ_iの近似値をsとして$A_k - sI$（Iは単位行列）に対して次のようにQR法を適用する．

$$\begin{aligned} A_k - sI &= Q_kR_k, \\ A_{k+1} &= R_kQ_k + sI \quad (= Q_k^{-1}A_kQ_k). \end{aligned} \tag{2.3}$$

すると，この反復の収束率は$|\lambda_i - s|/|\lambda_j - s| \simeq 0$となり，収束を大幅に加速できる．これをシフト付きQR法と呼ぶ．

ただし，A_0が実非対称行列でも，その固有値は一般に複素数となりうる．この場合，シフト付きQR法をそのまま適用すると，複素数の演算を行う必要が生じてしまう．そこで，実行列の複素固有値は共役複素数として現れることを利用し，その近似値$s, \bar{s}$をそれぞれシフトとする2回の反復を同時に行う．これをダブルシフトQR法と呼ぶ．ダブルシフトQR法の計算式を整理した形で書くと次のようになる．

$$\begin{aligned} (A_k - sI)(A_k - \bar{s}I) &= Q_kR_k \\ A_{k+2} &= Q_k^{-1}A_kQ_k. \end{aligned} \tag{2.4}$$

これにより，演算は実数のみで済む．シフト$s, \bar{s}$は，通常，A_kの右下隅の2×2行列の固有値に選ぶ（Francisシフト）．この場合，ダブルシフトQR法は局所的に2次収束する．以下では，右下隅の2×2行列の固有値が実数の場合も同様に扱うこととし，2個のシフトをs_1, s_2と書く．

2.1.2 Implicit 型ダブルシフト QR 法

QR 法を実行するに当たっては，通常，演算量削減のため，予め行列をヘッセンベルグ行列（$i > j+1$ のとき $a_{ij} = 0$ の行列）に変換しておく．A_0 がヘッセンベルグ行列とすると，QR 法を適用して得られる行列 $A_1, A_2, \ldots$ もすべてヘッセンベルグ行列となる．さらに，次の定理が成り立つ [19][32].

定理 1（Implicit Q 定理） U，V が直交行列で，$G = U^t AU$，$H = V^t AV$ が共に既約な（$1 \leq i \leq n-1$ に対して $a_{i+1,i} \neq 0$ となる）ヘッセンベルグ 行列であるとする．このとき，もし U と V の第 1 列が等しいならば，± 1 を要素に持つ対角行列 D が存在して，$V = UD$，$H = DGD$ が成り立つ.

この定理は，行列 A を既約な ヘッセンベルグ 行列に相似変換する直交行列は，第 1 列目だけが与えられれば（実質的に）一意に定まることを示す．これを用いると，QR 分解と $Q^{-1}AQ$ の計算を陽に行うことなく，ダブルシフト QR 法を実行できる．実際，次のように計算を行えばよい.

(1) $(A_k - s_2 I)(A_k - s_1 I)$ の第 1 列を $\mathbf{e}_1$（単位行列の第 1 列ベクトル）の定数倍にするハウスホルダー変換 H_0 を求める.

(2) $A'_k \equiv H_0^t A_k H_0$

(3) 直交行列による相似変換を繰り返すことにより，A'_k を再びヘッセンベルグ 行列に変形する.

これにより得られる行列を A'_{k+2} とすると，A'_{k+2} は A_k をある直交行列 Q によって変換した行列となっており，かつヘッセンベルグ行列である．また，Q の第 1 列は Q_k の第 1 列に等しい（H_0 の第 1 列は Q_k の第 1 列に等しく，かつステップ (3) は第 2 行・第 2 列目以降のみに影響するから）．したがって，Implicit Q 定理より，もし A_{k+2}，A'_{k+2} が共に既約ならば，それらは行・列の ± 1 倍を除いて等しくなる．上記 (1)，(2)，(3) により計算を行う方法を Implicit QR 法と呼ぶ.

上記 (2) で得られる A'_k は，ヘッセンベルグ形に第 (3,1)，(4,1)，(4,2) 要素の 3 つの非ゼロ要素からなる出っ張り（bulge）が加わった行列である．これをヘッセンベルグ形に戻すには，まず左から適当なハウスホルダー変換 H_1^t を掛けることにより，第 (3,1)，(4,1) 要素を消去する．ただし，相似変換としなくてはならないため，右からも H_1 を掛ける．これにより，今度は (4,2)，(5,2)，(5,3) の 3 つの要素が bulge となる．このようにして相似変換 1 回ごとに bulge を 1 行・列ずつ右下に移動させていき，$n-2$ 回の相似変換を行うことで行列を再びヘッセンベルグ行列に変形する．この操作を bulge chasing と呼ぶ．Implicit QR 法の演算量のほとんどは bulge chasing で占められ，1 回の bulge chasing の演算量は $O(n^2)$ である.

2.1.3 Implicit 型ダブルシフト QR 法の演算パターンとその特徴

Implicit QR 法の bulge-chasing における相似変換では，bulge の最初の列の 2 個の要素を消去するため，左右からそれぞれ 3×3 のハウスホルダー変換を作用させる．この様子を Fig. 1 に示す．ハウスホルダー変換を左から掛けることにより 3 本の行が更新され，右から掛けることにより 3 本の列が更新される．

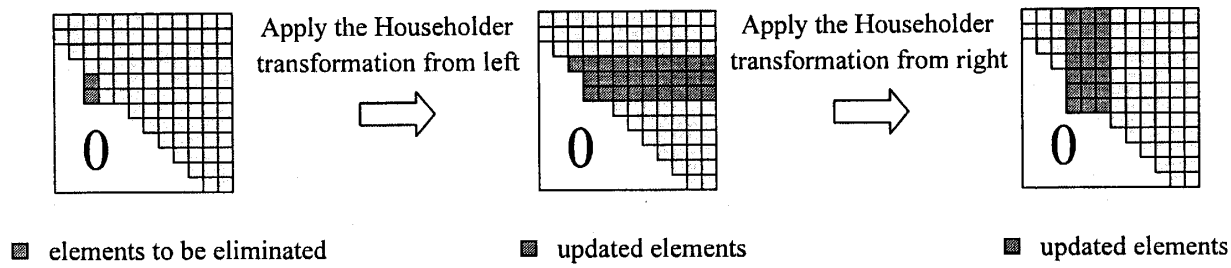

図 1: Matrix elements updated by the application of the Householder transformation from the left and from the right.

この演算の並列性は $O(n)$ であり，これを利用して並列化を行うことは可能である．実際，Henry らは，行列データを反対角線に沿って帯状に分割し，各帯をプロセッサに割り当てることにより，分散メモリ型並列計算機上での並列化が可能なことを示した [36]．また，須田らはこのデータ分割方式を更に改良し，ソフトウェアパイプライン化の手法を用いてプロセッサ待ち時間の削減と通信の隠蔽が可能なことを示した [53]．しかし，最近の分散メモリ型並列計算機では，プロセッサ間通信のコストがますます大きくなってきており，$O(n)$ の並列性では演算に比べて通信の回数が多すぎて，十分な性能を出すことは困難になりつつある．共有メモリ型並列計算機においても，プロセッサ間同期のコストが大きくなっており，事情は同様である．そのため，より大きな並列粒度を持つアルゴリズムが必要である．

一方，データ参照の局所性について考えると，A_k から A_{k+2} の計算においては，A_k の全要素に対するアクセスが必要であり，1 個の要素に対する更新演算の回数は左からと右からを合わせて 6 回，すなわち $O(1)$ である．このため，データ参照の局所性は低く，キャッシュの有効利用が期待できない．これは，単体プロセッサの性能向上の上で問題となるだけでなく，共有メモリ型並列計算機においても，プロセッサ間のアクセス競合により性能が制約されるという問題を引き起こす．したがって，データ参照の局所性を高めるという面でもアルゴリズムの改良が必要となる．

2.2 マルチシフト QR 法

2.2.1 原理

これらの問題を解決するために Bai & Demmel により提案された解法がマルチシフト QR 法である [3]．これはダブルシフト QR 法の素直な拡張であり，m 個のシフトを使って QR 法

の m ステップを同時に行う．シフトを $s_1, s_2, \ldots, s_m$ とし，マルチシフト QR 法の計算式を整理した形で書くと次のようになる．

$$\begin{aligned}(A_k - s_m I)\cdots(A_k - s_2 I)(A_k - s_1 I) &= Q_k R_k \\ A_{k+m} &= Q_k^{-1} A_k Q_k.\end{aligned} \tag{2.5}$$

ここで，シフトは A_k の右下隅の $m \times m$ 行列の固有値に選ぶ．これを一般化レイリー商シフトと呼ぶ．このとき，次の収束定理が成り立つ [58].

定理 2 A_0 の固有値がすべて異なるならば，上記のようにシフトを取ったとき，マルチシフト QR 法は局所的に 2 次収束する．特に，$Q_0, Q_m, Q_{2m}, \ldots, Q_k$ の積を $Q^{(k)}$ とするとき，$Q^{(k)}$ の最後の m 列の張る部分空間は，A_0 のある不変部分空間に 2 次収束する．

2.2.2 Implicit 型マルチシフト QR 法

Implicit Q 定理を使うことにより，マルチシフト QR 法についても Implicit 型のアルゴリズムを考えることができる．以下では，マルチシフト QR 法と言えば Implicit 型を指すとする．A_k から A_{k+m} を求める手順は次のようになる．

(1) $(A_k - s_m I)\cdots(A_k - s_2 I)(A_k - s_1 I)$ の第 1 列を $\mathbf{e}_1$ の定数倍にするハウスホルダー変換 H_0 を求める．

(2) $A'_k \equiv H_0^t A_k H_0$

(3) 直交行列による相似変換を繰り返すことにより，A'_k を再びヘッセンベルグ 行列に変形する．

$(A_k - s_m I)\cdots(A_k - s_2 I)(A_k - s_1 I)$ の第 1 列は最初の $m+1$ 個の要素のみが非ゼロのベクトルであるから，H_0 は最初の $m+1$ 行（列）のみに作用するハウスホルダー変換である．これを左右から掛けることにより，A'_k は，ヘッセンベルグ形に $3 \le i \le m+2$，$1 \le j \le m$ の範囲の非ゼロ要素からなる bulge が加わった行列となる．これをヘッセンベルグ形に戻すには，まず第 2 行から第 $m+2$ 行に作用するハウスホルダー変換 H_1 を左から掛け，bulge のうち第 1 列に属する部分を消去する．ただし，相似変換としなくてはならないため，右からも H_1 を掛ける．これにより，第 $m+3$ 行の第 2 要素から第 $m+1$ 要素が新たに非ゼロとなり，bulge は 1 行・列だけ右下に移動する．このようにして相似変換 1 回ごとに bulge を 1 行・列ずつ右下に移動させていき，最終的に行列を再びヘッセンベルグ行列に変形する．ダブルシフト QR 法の場合と同様，この操作を bulge chasing と呼ぶ．なお，H_0 を求めるに当たっては，右下隅の $m \times m$ 行列の固有値を計算せず，その特性多項式の係数のみを用いて求める方法もあり [27]，計算量と精度の面でメリットがある．

2.2.3 Implicit 型マルチシフト QR 法の演算パターンとその特徴

Implicit 型のマルチシフト QR 法の bulge chasing では，相似変換に $(m+1) \times (m+1)$ のハウスホルダー変換を用いる．ハウスホルダー変換を左から掛けることにより $m+1$ 本の行が更新され，右から掛けることにより $m+1$ 本の列が更新される．

この演算の並列性は $O(mn)$ であり，ダブルシフト QR 法の m 倍となっている．また，A_k から A_{k+m} の計算においては，1 個の行列要素に対する更新演算の回数は $O(m)$ となる．1 個の要素に対するアクセス回数は，各々のハウスホルダー変換をそのまま適用したのでは $O(m)$ となってしまうが，WY representation[8][51] を利用して複数のハウスホルダー変換をまとめて適用すれば，$O(1)$ に削減できる．したがって，行列要素に対する演算回数とアクセス回数の比もダブルシフト QR 法の m 倍に向上できる．さらに，WY representation では，計算を level-3 BLAS（行列乗算）[23] の形で行うことが可能であり，高性能な行列乗算ルーチン [59][33] の利用による性能向上を行う余地がある．

以上より，マルチシフト QR 法は，ダブルシフト QR 法に比べて並列性，データ参照の局所性の両面で優位性があり，高性能計算機での実行に原理的に適したアルゴリズムであると考えられる．なお，本節で述べた形のマルチシフト QR 法は，LAPACK の DHSEQR として実装されている[4]．

2.3 マルチシフト QR 法の収束特性

2.3.1 マルチシフト QR 法の実際の性能

マルチシフト QR 法で高性能計算機の性能を引き出すには，m を十分大きく取る必要がある．たとえば，2 次キャッシュが 256KB のプロセッサの場合，WY representation における行列乗算の性能を最適化するには，m を数十程度に取ることが必要である．

ところが，前節で述べた形のマルチシフト QR 法では，m を大きくすると，収束までの反復回数が大幅に増加し，演算量が増大するという傾向が見られる [26][56][10]．たとえば Braman らは，典型的な例として，2000 元の行列に対し，$m = 50$ のとき演算量がダブルシフト QR 法の 2 倍程度，$m = 80$ のとき 3.3 倍程度になるというデータを挙げている [10]．このように演算量が増大したのでは，キャッシュ利用効率が向上したとしても，その効果は相殺され，性能向上は実現できない．そのため，DHSEQR では $m = 6$ という値を採用している．しかし，この値は最近の高性能計算機で必要とされる m の値とは大きな開きがあり，十分な性能を引き出すには至っていない．

2.3.2 Watkins の定理

この収束性悪化の原因を解明するため，Watkins はマルチシフト QR 法におけるシフトの伝播について解析を行い，次の定理を得た [57]．

[4]ただし DHSEQR では level-3 BLAS の利用は行っていない [10]．

定理3 マルチシフトQR法において，A'_k に l 回目の相似変換を行って得られる行列を $A_k^{(l)}$ とし，$A_k^{(l)}$ の $l+2 \leq i \leq m+l+2$，$l+1 \leq j \leq m+l+1$ の範囲の要素からなる $(m+1)\times(m+1)$ の部分行列（bulgeの部分）を B_l とする．また，固有値0に属する $m+1$ 次のジョルダン細胞を N とする．このとき，シフト $s_1, s_2, \ldots, s_m$ は行列束 (B_l, N) の m 個の有限固有値である．

Watkinsはこの定理に基づき，マルチシフトQR法におけるシフトの情報は，bulge chasingの過程において，行列束 (B_l, N) の m 個の有限固有値として左上隅から右下隅まで伝播すると主張した．また，m が大きくなると，これらの固有値は摂動に対して極めて敏感になることを数値実験によって示し，丸め誤差による摂動のためにシフトが劣化して，シフト情報が右下隅まで正しく伝わらなくなることが収束性悪化の原因だとした．また，Kressnerは，m が大きくなるとき初期の行列束 (B_0, N) の有限固有値が摂動に敏感になる理由について，制御理論の問題からの類推に基づく直感的な説明を与えた[44][43].

上記の定理に基づく説明は，m が大きいときのマルチシフトQR法の収束性悪化の説明として，必ずしも満足できるものではない．なぜなら，この説明はシフト情報の伝播経路のうち1つだけを同定し，その経路に沿った伝播において情報が劣化することを述べているに過ぎないからである．しかし，直感的にわかりやすく，また，これ以上の解析がなされていないこともあり，上記の説明は広く受け入れられている．

2.4 Small-bulgeマルチシフトQR法

2.4.1 原理

前節では，m 個のシフトを用いて A_k から A_{m+k} を直接計算するマルチシフトQR法について述べた．一方，m が偶数の場合は，同じ計算を次のようにダブルシフトQR法の計算を $m/2$ 回繰り返すことによっても実現できる．

$$
\begin{aligned}
(A_k - s_2 I)(A_k - s_1 I) &= Q_k R_k \\
A_{k+2} &= Q_k^{-1} A_k Q_k \\
&\vdots \\
(A_{k+m-2} - s_m I)(A_{k+m-2} - s_{m-1} I) &= Q_{k+m-2} R_{k+m-2} \\
A_{k+m} &= Q_{k+m-2}^{-1} A_{k+m-2} Q_{k+m-2}.
\end{aligned} \tag{2.6}
$$

ただし，$s_1, s_2, \ldots, s_m$ は A_k の右下隅の $m\times m$ 行列の固有値である．実際には，計算はImplicit型で行うので，ダブルシフトQR法と同様のbulge chasingを $m/2$ 回行うことになる．この方法をsmall-bulgeマルチシフトQR法と呼ぶ．この方法で計算した A_{k+m} が無限精度演算では前節のマルチシフトQR法の結果と等しくなることは，簡単な式変形とQR分解の一意性により示せる．

しかし，有限精度計算では両者の結果は大きく異なる．Dubrulleは，small-bulgeマルチシフトQR法では m を大きくしても収束回数の増加が起こらず，ダブルシフトQR法と同程度の演算量で計算が完了することを報告している[26]．また，Bramanらは1000元から3000元

までの行列に対して数値実験を行い，m を 80 まで大きくしても収束までの演算量がダブルシフト QR 法とほとんど変わらないことを報告している [10]．前節で紹介した Watkins の考え方に基づくと，この収束特性の違いは，従来のマルチシフト QR 法では bulge が大きいため摂動によりシフト情報が劣化しやすいのに対し，small-bulge マルチシフト法では多数の小さい bulge に分けて chasing を行うためシフト情報の劣化が起こりにくく，定理 2 で述べた 2 次収束が実現していることによると解釈できる．

2.4.2 Small-bulge マルチシフト QR 法の並列性

Small-bulge マルチシフト QR 法の bulge-chasing では，ダブルシフト QR 法と同様，1 個の bulge を 1 行・列だけ右下に移動するため，3×3 のハウスホルダー変換を左右から掛ける．これにより更新されるのは，行列の 3 本の行・列のみである．したがって，更新範囲が重ならないようにすれば，複数の bulge に対する chasing を同時に行うことが可能である．実際，Fig. 2 に示すように $m/2$ 個の bulge を 3 行ずつ離して chasing すれば，対角要素付近の一部の要素を除いては更新範囲は重ならず，$3 * (m/2)$ 本の行・列を同時に更新できる．そして，複数の bulge による更新範囲が重なる対角付近の要素に対してのみ，逐次的な更新を行えばよい．これにより，並列性は従来のマルチシフト QR 法と同様 $O(mn)$ となる．

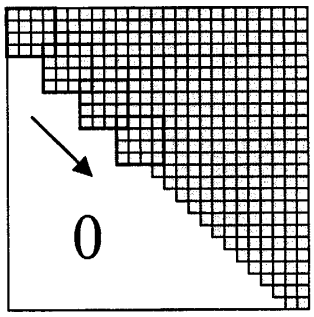

図 2: Parallel bulge chasing in the small-bulge multishift QR algorithm. Four bulges that are three lines apart from each other can be chased simultaneously.

Henry らはこの並列性を利用し，small-bulge マルチシフト QR 法に基づく分散メモリ型並列計算機上向けのアルゴリズムを提案した [37]．このアルゴリズムでは，2.1.3 項で述べたダブルシフト QR 法に基づく並列化に比べ，並列粒度を m 倍にできるという利点がある．また，データ分割方式はブロックサイクリック分割で良いため，他の線形計算ルーチンとの相性が良いという利点もある．このアルゴリズムは分散メモリ型並列計算機向けのライブラリ ScaLAPACK [9] に実装されている．

2.4.3 Level-3 BLAS の利用

Small-bulge マルチシフト QR 法では，A_k から A_{k+m} の計算において各行列要素に対して $O(m)$ 回の更新を行うため，原理的には従来のマルチシフト QR 法と同様にデータ参照の局

所性を高めることが可能である．しかし，更新は 3×3 の小さなハウスホルダー変換を用いて行われるため，WY representation を用いてこれらを効率良くまとめることは困難である．そこで Braman らは，複数のハウスホルダー変換の積を 1 個の行列 U に蓄積し，これを更新対象の行・列に掛けることにより，更新演算を行列乗算を用いて行う方法を提案した [10]．

いま，行列の対角線上に $m/2$ 個の bulge が 3 行ずつ離れて並んでいるとし，これらをそれぞれ p 行・列ずつ右下に chasing する処理を一まとまりとして考える．この処理に関係する行・列の数はそれぞれ $3*(m/2)+p$ 本である．この処理を次のように分割して行う．

(1) 大きさ $(3*(m/2)+p) \times (3*(m/2)+p)$ の対角ブロック内部で，$m/2$ 個の bulge をそれぞれ p 行・列 chasing する．また，chasing に使ったハウスホルダー変換の積を，$(3*(m/2)+p) \times (3*(m/2)+p)$ の行列 U に蓄積する．

(2) 更新対象の $3*(m/2)+p$ 本の行の対角ブロック以外の部分に，左から U を掛けて更新する．

(3) 更新対象の $3*(m/2)+p$ 本の列の対角ブロック以外の部分に，右から U^t を掛けて更新する．

これを Two-Tone QR 法（TTQR 法）と呼ぶ．Two-tone とは，処理を対角ブロックの更新と非対角ブロックの更新の 2 つに分けて行うという意味である．この処理の様子を Fig. 3 に示す．

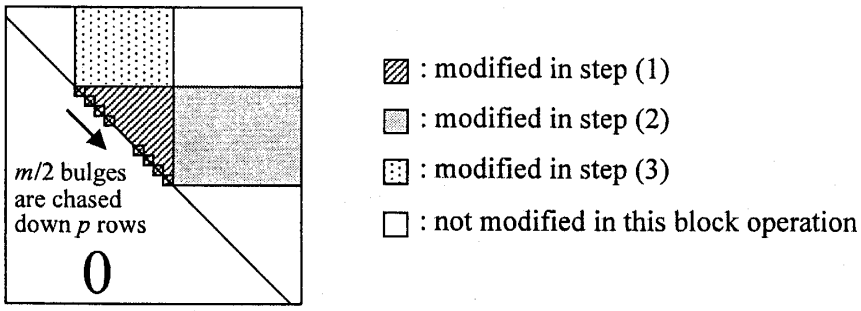

図 3: Work in one block operation of the TTQR method. The work is divided into (1) bulge-chasing in the diagonal block, (2) update of the off-diagonal rows and (3) update of the off-diagonal columns.

この方法では，演算量の大部分を占める (2)，(3) の部分を level-3 BLAS を用いて実行することが可能となる．ただし，ハウスホルダー変換の積を陽的に作ってそれを作用させるため，演算量が増加する．Braman らは演算量の詳細な解析を行うことにより，(i) $p \simeq 3*(m/2)$ としたとき A_k から A_{k+m} を求める演算量が最小になること，(ii) この p の値を用い，かつ (2)，(3) で U の非ゼロ構造を考慮して計算を行うことで，A_k から A_{k+m} を求める演算量はダブルシフト QR 法の 1.6 倍程度に抑えられること，を明らかにした [10]．一般に，演算量増加がこの程度であればデータ参照の局所性向上による効果のほうがずっと大きいため，TTQR 法は

キャッシュ向けに有効なアルゴリズムだと考えられる．実際，Bramanらは，Origin2000の1プロセッサ上での性能評価において，TTQR法が従来のダブルシフトQR法の最大3倍の性能を達成できることを報告している．

2.4.4 デフレーション

ヘッセンベルグ行列に対するQR法の計算では，反復の途中で下側副対角要素$a_{i+1,i}$（$1 \leq i \leq n-1$）の値を調べる．そして，これらのどれかが十分0に近い値になった場合，それを0で置き換えて固有値問題を2つの小さな固有値問題に分離する．標準的な方法では，$|a_{i+1,i}| \leq \epsilon(|a_{i,i}| + |a_{i+1,i+1}|)$（ただし$\epsilon$はマシンイプシロン）が成り立つとき$|a_{i+1,i}|$が十分0に近いと見なす．この操作をデフレーションと呼ぶ．Small-bulgeマルチシフトQR法では，次の2つのデフレーション方式が考えられる[56]．

(a) A_kからA_{k+m}の計算が終了した時点で下側副対角要素の値を調べる．

(b) 1個のbulgeによるchasingを行うたびに下側副対角要素の値を調べる．したがって，A_kからA_{k+m}の計算で，各要素は$m/2$回調べられることになる．

(b)の方式をvigilant deflationと呼ぶ．1個のbulgeが通過した後に$a_{i+1,i}$が0に近くなり，次のbulgeの通過後に再び大きい値になる場合，(a)の方式ではデフレーションの機会を見逃してしまうが，vigilant deflationではこの機会を捉え，行列を分離できる[5]．そのため，非対称行列向けのsmall-bulgeマルチシフトQR法ではvigilant deflationを採用するほうが良いとされる[56][10]．

2.4.5 従来のマルチシフトQR法との組み合わせ

以上で紹介したsmall-bulgeマルチシフトQR法では，A_kからA_{k+m}の計算を，2個のシフトを含むbulgeを$m/2$個chasingすることにより行っていた．これを一般化し，q個のシフトを含むbulgeをm/q個chasingすることにより計算を行うことも可能である[44][43]．これは，従来のマルチシフトQR法とsmall-bulgeマルチシフトQR法との組み合わせと考えられる．

TTQR法では，非対角ブロックとUの乗算が演算量の大部分を占める．したがって，行列Uに詰め込まれた情報は多いほうが良い．いま，たとえば$m=8$の場合を考えると，2.4.3項で述べたTTQR法（$q=2$）ではUのサイズは$3*(8/2)+3*(8/2)=24$となり，ここに12行分のbulge chasingの情報が詰め込まれる．一方，1個のbulgeに含まれるシフト数を$q=4$とすると，2.4.2項と同様の考察により，bulgeどうしの間隔は5行となり，最初の状態で2個のbulgeが10行を占める．これを$24-10=14$行chasingできるから，Uには14行分のbulge chasingの情報が詰め込める．このように，Uに入る情報の量は一般にqにつれて増える[6]．ただし，qを大きくすると従来のマルチシフトQR法と同様，収束性が悪化する．

[5]ただし，途中でのデフレーションにより複数のbulge chasingからなるパイプライン処理が乱れてしまうので，それに対処するためプログラムは少し複雑になる．

[6]実際にはUの非ゼロ要素数もqにつれて増え，それによりUによる乗算の演算量も増加するが，その影響は小さい．

Kressnerは様々な行列に対して数値実験を行い，多くの場合に$q = 6$とするのが最適であると結論している[44][43].

2.4.6 共有メモリ型計算機向けの並列化

TTQR法では演算の大部分がlevel-3 BLASで行われるため，共有メモリ型並列計算機では，並列化BLASを使うことで容易に並列化を行うことが可能である．実際，Bramanらはこの方法により，Origin2000上で8プロセッサの場合に最大60%程度の並列化効率が得られることを報告している[10].

並列化効率を更に向上させるには，逐次的な処理である対角ブロック内部でのbulge chasingを，非対角ブロックとUの乗算とオーバーラップさせればよい[10]．これは，外積形式のガウス消去法において，ピボット列の作成と消去演算とをオーバーラップさせることにより並列化効率を上げるのと同じ技法である．具体的には，Uの乗算による非対角ブロック行・列の更新において，次のp行のbulge chasingに必要な部分のみを先に更新し，それから対角ブロック内部での次のp行のbulge chasing（1プロセッサが担当）と残りの部分の更新（残りのプロセッサが担当）とを並行して行えばよい．

2.5 Aggressive Early Deflation

2.5.1 デフレーションの一般化

QR法では，デフレーションを行うため，ヘッセンベルグ行列A_kの下側副対角要素で0に近い値を探す．これは，$A_k + E$の固有値問題が2つの固有値問題に分離できるような，微小な摂動行列Eを探していると解釈できる．ただし，Eは下側副対角部分にのみ非ゼロ要素を持つ行列である．ここで，Eとしてより広い範囲に非ゼロ要素を持つ行列を許容すれば，デフレーションを行える機会はより増加し，結果としてQR法の計算を高速化できる可能性がある．Bramanらは，この考え方に基づき，aggressive early deflationと呼ばれる新しいデフレーションの方法を提案した[11].

Aggressive early deflationでは，ある整数wを決めて行列の右下の$w \times (w+1)$の小行列をdeflation windowと定義し，Eとしてこの範囲に（複数の）非ゼロ要素を持つ行列を許容する．ただし，Eをこのように取ると，一般に$A_k + E$はヘッセンベルグ行列ではなくなる．そのため，デフレーションの後に相似変換を行ってヘッセンベルグ行列に戻す必要がある．これに関わるのは$A_k + E$の最後のw列のみであり，その演算量は$O(nw^2)$である[43].

2.5.2 摂動行列 E の決め方

摂動行列 E の決め方には多くの可能性がありうるが，Bramanらは次のような方法を提案している [11][43]．まず，A_k，E を

$$A_k = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ 0 & A_{32} & A_{33} \end{bmatrix}, \quad E = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & E_{32} & E_{33} \end{bmatrix} \tag{2.7}$$

のように分割する．ここで，第1，2，3ブロックに属する行および列の本数は，それぞれ $n-w-1$ 本，1本，w 本である．次に，直交行列 Q_1 により A_{33} を $T_{33} = Q_1^t A_{33} Q_1$ とブロック上三角化し，

$$T_{33} = \begin{bmatrix} \tilde{T}_{11} & \tilde{T}_{12} \\ 0 & \tilde{T}_{22} \end{bmatrix} \tag{2.8}$$

と分割する．ここで，$\tilde{T}_{22}$ は 1×1（$\tilde{T}_{22}$ の固有値が実数の場合）または 2×2（固有値が共役複素数の場合）の行列である．また，

$$Q_1^t A_{32} = \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \end{bmatrix} \tag{2.9}$$

とおく．ここで，右辺のベクトルの分割は式 (2.8) の分割と同じとする．いま，もし $\| \mathbf{s}_2 \|_2$ が十分小さければ，

$$E_{32} = -Q \begin{bmatrix} 0 \\ \mathbf{s}_2 \end{bmatrix}, \quad E_{33} = O \tag{2.10}$$

とおくと，E のノルムも十分小さい．さらに，

$$(I_{n-w} \otimes Q)^t (A_k + E)(I_{n-w} \otimes Q) = \begin{bmatrix} A_{11} & A_{12} & \tilde{A}_{13} & \tilde{A}_{14} \\ A_{21} & A_{22} & \tilde{A}_{23} & \tilde{A}_{24} \\ 0 & \mathbf{s}_1 & \tilde{T}_{11} & \tilde{T}_{12} \\ 0 & 0 & 0 & \tilde{T}_{22} \end{bmatrix} \tag{2.11}$$

（ただし $[\tilde{A}_{13}\ \tilde{A}_{14}] = A_{13}Q$，$[\tilde{A}_{23}\ \tilde{A}_{24}] = A_{23}Q$）であるから，$A_k + E$ は直交行列 $I_{n-w} + Q$ による相似変換を施すことにより，右下隅の小行列 $\tilde{T}_{22}$ が分離可能となる．これによりデフレーションが実行できる．

もし $\| \mathbf{s}_2 \|_2$ が十分小さくない場合は，T_{33} において $\tilde{T}_{22}$ を他の対角ブロックと入れ替え [4]，再び $\| \mathbf{s}_2 \|_2$ の値を調べる．これをすべての対角ブロックを候補として，十分小さい $\| \mathbf{s}_2 \|_2$ が見つかるまで繰り返す．もし見つからなければ，A_k に対しては aggressive early deflation はできないと判定し，QR法の反復に戻る．

一方，式 (2.11) の形に変形できた場合は，$\tilde{T}_{11}$ を更に式 (2.8) のように分割し，同様の操作を繰り返す．これにより，式 (2.11) において第3ブロックの行・列の本数が減少し，その分だけ第4ブロックの行・列の本数が増加する．この操作がそれ以上できない段階に来たら，$\tilde{T}_{22}$ を分離し，デフレーション後の行列を，前項で述べたように相似変換により再びヘッセンベルグ形に戻す．

なお，aggressive early deflationではdeflation windowの中だけを見てデフレーションを行うため，それ以外の部分で下側副対角要素が0に近くなる効果は考慮できない．そのため，通常のデフレーションも併せて行う必要がある．

2.5.3 Aggressive early deflationの効果

BramanらはTTQR法にaggressive early deflationを組み合わせて乱数行列から実問題の行列まで様々な行列に対して評価を行い，vigilant deflationのみのTTQR法に比べて多くの場合に3～5倍の大きな性能向上が得られることを示した[11]．従来のマルチシフトQR法を用いたLAPACKのDHSEQRと比べると，最大10倍程度の性能向上となる．このようにaggressive early deflationが有効に働くのは，$\| s_2 \|_2$が0に近くなるという状況が極めて頻繁に起こることによる．[11]では，この理由についての解析も行っている．

[11]の著者のBraman，Byers，Mathiasは，この論文により2003年のSIAM線形代数分科会の賞を受賞している．また，aggressive early deflation付きのTTQR法は，2007年に公開予定のLAPACK 4.0に組み込まれる予定になっている[20]．

2.6 今後の発展方向

2.6.1 他の固有値計算アルゴリズムへの適用

本章では非対称の標準固有値問題向けのマルチシフトQR法について述べたが，マルチシフトQR法の考え方は，LR法[49]，QZ法[47]，SR法[12]など他の固有値計算アルゴリズムにも適用できる．実際，[58]では，QR法，LR法，SR法などを一般化したGR法というアルゴリズムに対して，マルチシフト法の収束理論を展開している．また，Kågströmらは，TTQR法とaggressive early deflationとの組み合わせが一般固有値問題向けのQZ法にも拡張できることを示した[41]．このアルゴリズムはLAPACK 4.0に組み込まれる予定である．

2.6.2 自動チューニングの適用

TTQR法の性能を引き出すには，シフトの数mを適切に決めることが重要である．最適なmの値は対象とするマシン，行列のサイズ，プロセッサ数などによって大きく異なる．実際，[10]では，Origin2000（1プロセッサ）上での数値実験において，$1000 \leq n \leq 1999$では$m = 60$，$2000 \leq n \leq 2499$では$m = 116$，$2500 \leq n \leq 3999$では$m = 150$という値を設定している．

最適なmの値を自動的に決定するため，[62]ではTTQR法の実行時間をモデル化し，最短実行時間を与えるmの値をモデル上で求めている．ここで，性能モデリングの手法としては，アルゴリズムをBLASなどの構成部品に分割し，各部品の実行時間を精密にモデル化してその積み上げとして全実行時間を求める階層的なモデリング手法[15][14][16]を用いている．この結果，Opteron，PowerPC G5などをプロセッサとする共有メモリ型並列計算機上で，最

適または最適に近い m の値を実行前に決定できることを示している．

TTQR 法の性能に影響するパラメータは，m の他にも存在する．たとえば，2.4.6 項で述べた並列化方式では，対角ブロック内部での bulge chasing を担当するプロセッサとしないプロセッサとの間で，非対角ブロックの更新演算をどんな比率で配分するかを決める必要がある．これらも含めて自動チューニングを行うことは今後の課題である．また，aggressive early deflation では，deflation window のサイズ w を決める必要があるが，これは行列の数値的性質により最適値が異なる可能性もあるため，より難しい問題となる．この最適決定も今後の課題である．

2.6.3 分散メモリ型計算機向けの並列化

TTQR 法の考え方を利用して，並列化効率が高く，かつ単体プロセッサの性能も十分引き出せる分散メモリ向けのアルゴリズムを作ることは今後の課題である．特に，[53] で提案されたデータ分割方式と組み合わせることにより，対角ブロック内部での bulge chasing の時間と通信時間とを共に効率よく隠蔽できるアルゴリズムが作れれば興味深い．

3 対称行列向けのマルチシフト QR 法

3.1 従来の対称行列向け QR 法

3.1.1 Implicit shift QR 法

対称行列に対する QR 法は，非対称行列に対する QR 法の特別な場合として理解できる．以下では，最初の行列 A_0 が既約な対称三重対角行列の場合を考える．このとき，反復式 (2.1) によって計算される $A_1, A_2, \ldots$ はすべて対称三重対角行列となる [32][48]．また，$k \to \infty$ のとき A_k は A_0 の固有値を対角要素とする対角行列に収束する．収束の次数は 1 次である．

シフト付きの QR 法は，非対称行列の場合と同様，式 (2.3) で定義される．シフト s は，A_k の (n,n) 要素に選ぶ方法（レイリー商シフト），右下隅の 2×2 小行列の固有値のうち，(n,n) 要素に近い方を選ぶ方法（Wilkinson シフト）などがある．Wilkinson シフトを用いた場合，大域的収束が保証され，収束の次数はほとんどの場合 3 次となる [48]．

三重対角行列はヘッセンベルグ行列の一種であるから，定理 1 の Implicit Q 定理が成り立ち，次のように Implicit shift 型のアルゴリズムを構成できる．

(1) $(A_k - sI)$ の第 1 列を $\mathbf{e}_1$ の定数倍にするハウスホルダー変換 H_0 を求める．

(2) $A'_k \equiv H_0^t A_k H_0$

(3) 直交行列による相似変換を繰り返すことにより，A'_k を再び三重対角行列に変形する．

上記 (2) で得られる A'_k は，三重対角行列の $(3,1)$ 要素が非ゼロとなった行列である．そこで (3) では，相似変換によってこの非ゼロ要素（bulge）を 1 行・列ずつ右下に移動させ，右

下隅から追い出す．具体的には，bulgeが第$(l+2,l)$要素の位置にある場合，第$l+1$行と第$l+2$行に作用するギブンス回転G^tを左から掛け，bulgeを消去する．その後，Gを右から掛けることにより，第$(l+3,l+1)$要素の位置に新たなbulgeが生じる．これを$n-2$回繰り返すことにより，行列を三重対角形に復元できる．この計算が対称三重対角行列用QR法におけるbulge chasingである．1回のbulge chasingの演算量は$O(n)$である．

3.1.2 Implicit shift QR法の演算パターンとその特徴

上記のbulge chasingにおける相似変換では，2本の行あるいは列が更新される．各行あるいは列の非ゼロ要素数は高々3個か4個であるため，各相似変換の演算量は$O(1)$である．また，1個の相似変換が終わらないと次のbulgeが定まらず，次の相似変換が開始できないという依存関係がある．したがって，対称三重対角行列用QR法におけるbulge chasingは，極めて逐次性の強いアルゴリズムであると言える．

一方，データ参照の局所性について考えると，bulge chasingでは三重対角行列の前要素に対するアクセスが必要であり，1個の要素に対する更新演算の回数は$O(1)$である．したがって，データ参照の局所性は低い．ただし，三重対角行列の場合には元々データ量が$O(n)$と少なく，行列全体がキャッシュに収まる場合も多いため，この点についてはそれほど問題にならないと考えられる．

3.1.3 Implicit shift QR法の並列化の試み

前項で述べたことより，対称三重対角行列用QR法をそのまま並列化するのは困難である．そのため，ScaLAPACKのQR法では，固有値計算の部分は全プロセッサで同じ計算を実行し，固有ベクトル計算のみを並列化している．

アルゴリズムの書き換えにより並列化する試みとしては，Samehらによる研究[50]とBaronらによる研究[6]がある．このうち[50]は，bulge chasingに現れる非線形の漸化式を変数の置き換えで線形の漸化式に変換することにより，$O(n)$個のプロセッサを使って$O(\log n)$時間で計算するアルゴリズムを提案している．しかし，このアルゴリズムでは誤差がnに関して指数的に増加する可能性がある．一方，[6]では，三重対角行列をブロックに分割して各ブロック内部でそれぞれbulge chasingを行い，その後に境界部で整合を行うアルゴリズムを提案している．このアルゴリズムは数値的安定性が証明されており，非常に興味深いが，並列計算機上での実装と性能評価は行われていない．また，演算量がオーダーでしか議論されておらず，プロセッサ数を何個以上用いれば従来のQR法より速くなるかなど，より詳細な評価が必要である．

次節以降では，マルチシフトQR法に基づく並列化手法について紹介する．

3.2 マルチシフト QR 法

3.2.1 原理

対称三重対角行列向けのマルチシフト QR 法では，非対称向けの small-bulge マルチシフト QR 法と同様，m 個のシフトを使って次の式により A_k から A_{k+m} を計算する．

$$\begin{aligned} A_k - s_1^{(k)} I &= Q_k R_k \\ A_{k+1} &= Q_k^{-1} A_k Q_k \\ &\vdots \\ A_{k+m-1} - s_m^{(k)} I &= Q_{k+m-1} R_{k+m-1} \\ A_{k+m} &= Q_{k+m-1}^{-1} A_{k+m-1} Q_{k+m-1}. \end{aligned} \tag{3.1}$$

ただし，シフト $s_1^{(k)}, s_2^{(k)}, \ldots, s_m^{(k)}$ は A_k の右下隅の $m \times m$ 行列の固有値に選ぶ．このとき，次の収束定理が成り立つ [58].

定理 4 上記のようにシフトを取ったとき，マルチシフト QR 法は局所的に 3 次収束する．特に，$Q_0, Q_m, Q_{2m}, \ldots, Q_k$ の積を $Q^{(k)}$ とするとき，$Q^{(k)}$ の最後の m 列の張る部分空間は，A_0 のある不変部分空間に 2 次収束する．

3.2.2 共有メモリ型計算機向けの並列化手法とその問題点 (I)

上記のアルゴリズムにおいて，各シフトに対する bulge chasing の相似変換は，2 本の行・列のみに対して作用する．したがって，m 個の bulge を 2 行ずつ離して chasing すれば，互いに干渉なく同時に計算を進めることが可能である．

Kaufman はこのことを利用し，CRAY YMP 上で対称三重対角行列向けマルチシフト QR 法のベクトル化を行った [42]．この実装では，$m = 30 \sim 100$ として同時に bulge chasing を行い，m に関するループをベクトル化している．

Kaufman の手法を用いて共有メモリ型並列計算機上で並列化を行うには，m をプロセッサ数に等しく取り，各プロセッサが同期を取りながら 2 行ずつ離れた bulge を同時に chasing するようにすればよい．SIMD 型の共有メモリ型並列計算機であれば，命令ごとにプロセッサ間で自動的に同期が取られるため，この手法で効率良く並列化することが可能である．

しかしこの方法は，最近主流となっている MIMD 型の共有メモリ型並列計算機には適さない．最近の共有メモリ型並列計算機ではプロセッサ間同期のコストが極めて大きく，数百サイクル以上にも及ぶ．そのため，同期と同期との間に少なくとも数百以上の演算がなければ，並列化の効果は全く得られない．Kaufman の手法では，同期と同期の間の演算回数は $O(1)$ であるから，全く性能が出ないことになる．

3.2.3 共有メモリ型計算機向けの並列化手法とその問題点 (II)

一方，別な並列化手法として，領域分割型の並列化が考えられる．この方法では，三重対角行列をプロセッサ数pと等しい数の行ブロックに分割し，各ブロックを1個のプロセッサに担当させる．そして，まずシフト$s_1^{(k)}$に対するbulge chasingを開始し，このbulgeが第1のブロックを抜けたら，シフト$s_2^{(k)}$に対するbulge chasingを開始する．こうして，各ブロック内には常に1個のbulgeしか存在しないようにして計算を進めれば，ブロック内部のbulge chasingでは，プロセッサ間同期は不要である．同期が必要なのは，bulgeがブロックの境界を横切るときだけであり，同期の回数は1回のbulge chasingに対して$O(m)$回で済む．この手法による並列実行の様子をFig. 4に示す．ここでは，プロセッサ数（=シフト数）は4個であり，点線が各プロセッサの担当する領域の境界を示す．

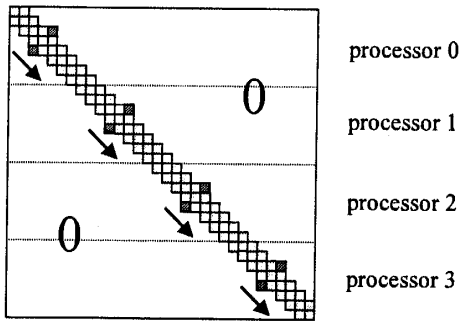

図 4: Bulge chasing in the parallel tridiagonal multishift QR method ($m = 4$). The 1st, 2nd, 3rd and 4th bulges are chased within the 1st, 2nd, 3rd and 4th row blocks, respectively.

しかし，この手法ではプロセッサの利用効率に問題がある．実際，最初のbulgeが第1ブロック内部を移動している間は，第1ブロックを担当するプロセッサしか使われない．その後，2番目，3番目のbulgeを導入するにつれ，使用されるプロセッサは2個，3個と増えていくが，最後のシフトが第1ブロックを通過すると，今度は第1ブロックを担当するプロセッサが空いてしまう．こうして，平均すると半分のプロセッサしか使われず，利用効率が悪い．なお，次のm個のシフト$s_1^{(k+m)}, s_2^{(k+m)}, \ldots, s_m^{(k+m)}$は$A_{k+m}$の右下隅の$m \times m$行列の固有値として計算されるから，シフト$s_m^{(k)}$に対するbulgeが行列の右下隅から追い出されるまでは計算できないことに注意する．

この手法の一般化として，ブロックの数を増やして1ブロック当たりの行数を少なくすれば，プロセッサの利用効率を向上させることが可能である．しかし，同期回数はブロックの数に比例するため，プロセッサの利用効率と同期オーバーヘッド削減との間にトレードオフがある．

3.3 遅延シフトを用いる方法

3.3.1 原理

以上の問題を解決するため，van de Geijnは遅延シフトを用いるマルチシフトQR法を提案した[54]．なお，[54]では非対称の場合も扱っているが，ここでは対称の場合のみについて紹介する．3.2.3項で述べた領域分割分割型の並列化では，最後のbulgeが通過した後，そのブロックを担当するプロセッサは空いてしまう．これは，次のシフト $s_1^{(k+m)}, s_2^{(k+m)}, \ldots, s_m^{(k+m)}$ がまだ計算できておらず，次のbulge chasingを開始できないからである．そこでvan de Geijnは，A_k から A_{k+m} の計算において，A_{k-m} の右下隅の $m \times m$ 行列の固有値 $s_1^{(k-m)}, s_2^{(k-m)}, \ldots, s_m^{(k-m)}$ をシフトとして使う方式を提案した．これを遅延型シフトと呼ぶ．

この方式によれば，A_{k+m} から A_{k+2m} の計算では，A_k から計算したシフトが使われる．そのため，A_{k+m} の計算が終了しないうちに（すなわち最後のbulgeが追い出される前に）A_{k+2m} の計算のためのbulge chasingを開始できる．したがってプロセッサの空きは生じず，プロセッサ利用効率をほぼ100%にできる．

3.3.2 遅延シフトを用いたマルチシフトQR法の収束性

ただし，この方式は古いシフトを使っているため，収束性が悪化する可能性がある．これについて，van de Geijnは次の定理を示した[54]．

定理5 マルチシフトQR法において h 回だけ古いシフトを使った場合の収束次数 τ_h は，非対称マルチシフトQR法の場合，非線形方程式

$$t^{h+1} - t^t - 1 = 0 \tag{3.2}$$

の唯一の正の解として与えられる．また，対称マルチシフトQR法の場合は，非線形方程式

$$t^{h+1} - t^t - 2 = 0 \tag{3.3}$$

の唯一の正の解として与えられる．特に，非対称の場合は $\tau_1 = (1+\sqrt{5})/2$，対称の場合は $\tau_1 = 2$ である．

ここで，シフトの遅れ h は，A_k の部分行列の固有値の代わりに A_{k-m} の部分行列の固有値を使った場合に $h = 1$ と数える．この定理より，遅延型シフトを用いたマルチシフトQR法は，同期回数を $O(p)$ で抑えつつプロセッサ利用効率を100%に上げる代償として，収束次数を3次から2次に落としていることがわかる．

3.4 Fully Pipelinedマルチシフト QR法

3.4.1 基本的なアイディア

収束性を落とさずに $O(p)$ の同期回数と100%に近いプロセッサ利用効率とを同時に達成するため，宮田らはfully pipelinedマルチシフトQR法と呼ぶ新しいアルゴリズムを提案した

[46]. このアルゴリズムは，遅延シフトを用いるアルゴリズムと同様，領域分割型の並列化に基づく．しかし，このアルゴリズムでは，m 回の bulge chasing を終えた後ではなく，1回の bulge chasing が終わるたびに，右下隅の $m \times m$ 行列の固有値を計算する．そして，m 個の固有値のうち1個を新しいシフトとして直ちに新しい bulge chasing を開始する．

より具体的には，A_k から始めて，m 個のシフト $s_1^{(k-m+1)}, s_2^{(k-m+2)}, \ldots, s_m^{(k)}$ を使って式(3.1)の計算を行うことを考える[7]．計算では，三重対角行列を m 個の行ブロックに分けて領域分割型の並列化を行う．計算の過程で，最初のシフト $s_1^{(k-m+1)}$ による bulge chasing が終わると，A_{k+1} の計算ができたことになる．そこで，その右下隅の $m \times m$ 行列の m 個の固有値を計算する．そして，そのうちでいま bulge chasing が終わったばかりのシフトにもっとも近い固有値を選び，これを次のシフト $s_1^{(k+1)}$ として bulge chasing の計算を開始する．このとき，シフト $s_m^{(k)}$ による bulge chasing はちょうど第1ブロックを通過し終えたところであるので，第1ブロックを担当するプロセッサは休むことなく計算を続行できる．また，この計算は，A_{k+m} から A_{k+m+1} の計算に相当する．

この方法では，一般に，A_{k+l} から A_{k+l+1} の計算 $(0 \leq l \leq m-1)$ において，$A_{k-m+l+1}$ から計算したシフト $s_{l+1}^{(k-m+l+1)}$ を使う．3.2項で紹介した従来のマルチシフトQR法では A_k から計算したシフトを使うので，それよりは古いが，遅延シフトでは A_{k-m} から計算したシフトを使うので，それよりは新しいことになる．ここで重要なのは，$A_{k-m+l+1}$ では，シフト $s_{l+1}^{(k-2m+l+1)}$ による bulge chasing の効果が既に入っているということである．これは，直感的に言えば，$s_{l+1}^{(k-m+l+1)}$ が近似しようとしている固有値に関しては，最新の情報までが入っているということである．残りのシフト $s_{l+2}^{(k-2m+l+2)}, \ldots, s_m^{(k-m)}$ の効果はまだ入っていないが，これらは A_0 の異なる固有値を近似するシフトであり，$s_{l+1}^{(k-m+l+1)}$ が近似する固有値の収束には影響を及ぼさないのではないかと考えられる．そのため，遅延シフトの場合のような収束性悪化は生じないのではないかと期待される．

この方法のデメリットは，1回の bulge chasing が終わるたびに $m \times m$ の行列の固有値計算が必要なことである．このため，従来のマルチシフトQR法や遅延シフトを用いる方法に比べて，この部分の計算時間は m 倍に増える．しかし，$m \ll n$ である限り，その影響は小さい．

3.4.2 共有メモリ型並列計算機上での性能

宮田らは，従来のマルチシフトQR法，遅延シフトを用いる方法，前項で述べた fully pipelined マルチシフトQR法の3種を，共有メモリ型並列計算機である PrimePower HPC 2500 の1ノード上で実装し，32個までのプロセッサを用いて実行時間と収束特性の予備的な評価を行った[46]．その結果，fully pipelined マルチシフトQR法は最も高速であり，2番目に速い遅延シフトQR法に比べ，最大1.5倍の高速化を達成できることを示した．10万元の三重対角行列の固有値を16プロセッサで計算する場合，fully pipelined マルチシフトQR法は逐次型の(3.1.1項で述べた)QR法を1プロセッサで実行した場合と比べて10倍の加速を得ている．また，収束特性については，従来のマルチシフトQR法，遅延シフトを用いた方

[7]以下で述べるように，シフトの上付き添字は，その計算に用いた行列の番号を表している．

法は m が増えると平均反復回数が増加するのに対し，fully pipelined マルチシフト QR 法では一定に留まるという結果を得ている．

Fully pipelined マルチシフト QR 法は，シフトの計算のための演算量が従来の m 倍となるため，プロセッサ数（=シフト数）が増えると効率が低下する可能性がある．これについては，シフト計算を行う最下段のブロック担当のプロセッサに対し，ブロックの行数を他より少なく設定して負荷の均衡化を行うなどの手段により，ある程度改善可能である．この点に関しても [46] で報告されている．

3.4.3 理論的解析の必要性

Fully pipelined マルチシフト QR 法は，マルチシフト QR 法でありながら，シフトの更新を bulge chasing が 1 回終わるたびに行っている．また，計算した m 個の固有値のうち 1 個のみをシフトとして採用している．このため，定理 4 で示した Watkins の理論の枠組みでは収束性を解析できない．また，定理 5 の van de Geijn の枠組みでも解析できない．そのため，収束性を解析する新たな理論が必要であり，今後の課題である．

3.5 今後の発展方向

3.5.1 分散メモリ型計算機向けの並列化

Fully pipelined マルチシフト QR 法は，分散メモリ型並列計算機に対しても適用できる．ハウスホルダー法による三重対角化まで含めて考えると，超大規模の固有値計算は分散メモリ型で行う必要があるため，この拡張は重要である．この場合，各行ブロックを 1 個のプロセッサに担当させ，bulge がブロックの境界を通過する時点でプロセッサ間通信を行うことになる．また，行列は固有値が求まるたびにデフレーションにより小さくなっていくので，プロセッサ間の負荷の均衡を保つため，データ分割を動的に更新していくことが必要となる．

3.5.2 Aggressive early deflation の適用

2.5 節で紹介した aggressive early deflation はもともと非対称行列用に提案されたが，これを対称行列向けのマルチシフト QR 法に適用することも可能である．これにより，前節で述べたアルゴリズムの性能を更に向上できる可能性がある．

3.5.3 qd 法と mdLVs 法への適用

本章で述べたマルチシフトの技法は，形式的には特異値計算のための qd 法 [28]，mdLVs 法 [39][40] にも適用できる．これにより，これらの手法でも並列化を行うことが可能となる．ただし，qd 法，mdLVs 法では，精度と安定性を保証するため，シフトの値を最小特異値より小さく取ることが必要であるとされる [28][40]．ところが，マルチシフトでは，複数の特異値に対する近似値を同時にシフトとして使うため，必然的にこの制限を破ることになる．この

ことが精度・安定性に及ぼす影響を調べる必要がある.

4 おわりに

本論文では,QR法を高性能計算機上で効率良く実行するためのアルゴリズムであるマルチシフトQR法を取り上げ,非対称,対称両方の場合について,最近の発展を紹介した.ここで取り上げた様々なアルゴリズムは,今後,ライブラリとして整備されるに従い,固有値計算の有力な方法になると期待される.

密行列の固有値計算の分野では,この他に,分割統治法に基づく解法についても近年様々な発展がある.また,条件数の極めて大きな行列の固有値を相対精度の意味で高精度に計算しようとする超高精度アルゴリズムの研究も進展している.これらについても,機会があれば別のサーベイ論文で報告したい.

謝辞 日頃からご指導頂いている東京大学大学院情報理工学系研究科の杉原正顯教授と名古屋大学大学院工学研究科の張紹良教授に感謝いたします.また,本論文の内容を2006年1月のLA研究会で発表した際にご討論下さった皆様と,本論文に対して有益なご指摘を下さった査読者の皆様に感謝いたします.なお,本研究は名古屋大学21世紀COEプログラム「計算科学フロンティア」,科学研究費補助金基盤研究(C)(課題番号18560058),および科学研究費補助金特定領域研究「i-explosion」(課題番号18049014)の補助を受けている.

参考文献

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide.* SIAM, 1992.

[2] Z. Bai, D. Day, J. W. Demmel, and J. J. Dongarra. A test matrix collection for non-Hermitian eigenvalue problems (release 1.0). Technical Report CS-97-355, Department of Computer Science, University of Tennessee, Knoxville, 1997.

[3] Z. Bai and J. Demmel. On a block implementation of Hessenberg QR iteration. *Int. J. of High Speed Computing*, 1:97–112, 1989.

[4] Z. Bai and J. Demmel. On swapping diagonal blocks in real Schur form. *Linear Algebra Appl.*, 186:73–95, 1993.

[5] Z. Bai, J. Demmel, and M. Gu. An inverse free parallel spectral divide and conquer algorithm for nonsymmetric eigenproblems. *Numer. Math.*, 76:279–308, 1997.

[6] I. Bar-On and B. Codenotti. A fast and stable parallel QR algorithm for symmetric tridiagonal matrices. *Linear Algebra Appl.*, 220:63–95, 1995.

[7] C. Bischof, B. Lang, and X. Sun. Parallel tridiagonalization through two-step band reduction. Technical Report 17, PRISM Working Note, 1994.

[8] C. Bischof and C. F. van Loan. The WY representation for products of householder matrices. *SIAM J. Sci. Stat. Comput.*, 8(1):s2–s13, 1987.

[9] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley. *ScaLAPACK User's Guide*. SIAM, 1997.

[10] K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. part I: Maintaining well-focused shifts and level 3 performance. *SIAM J. Matrix Anal. Appl.*, 23(4):929–947, 2002.

[11] K. Braman, R. Byers, and R. Mathias. The multishift QR algorithm. part II: Aggressive early deflation. *SIAM J. Matrix Anal. Appl.*, 23(4):948–973, 2002.

[12] A. Bunse-Gerstner and V. Mehrmann. A symplectic QR like algorithm for the solution of a real algebraic Riccati equation. *IEEE Trans. Auto. Control*, AC-31:1104–1113, 1986.

[13] H. Chang, S. Utku, M. Sakama, and D. Rapp. A parallel Householder tridiagonalization using scattered square decomposition. *Parallel Computing*, 6(3):297–311, 1988.

[14] J. Cuenca, L-P. Garcia, and D. Gonzalez Gimenez. Empirical modelling of parallel linear algebra routines. In *Proceedings of the 5th International Conference on Parallel Processing and Applied Mathematics (PPAM2003)*, number 3019 in Lecture Notes in Computer Science, pages 169–174. Springer-Verlag, 2004.

[15] J. Cuenca, D. Gimenez, and J. Gonzalez. Architecture of an automatically tuned linear algebra library. *Parallel Computing*, 30:187–210, 2004.

[16] K. Dackland and B. Kågström. A hierarchical approach for performance analysis of ScaLAPACK-based routines using the distributed linear algebra machine. In *Proceedings of Workshop on Applied Parallel Computing in Industrial Computation and Optimization (PARA96)*, number 1184 in Lecture Notes in Computer Science, pages 187–195. Springer-Verlag, 1996.

[17] Roden J. A. David and D. S. Watkins. Efficient implementation of the multishift QR algorithm for the unitary eigenvalue problem. *SIAM J. Matrix Anal. Appl.*, 28:623–633, 2006.

[18] J. Demmel and K. Veselic. Jacobi's method is more accurate than QR. *SIAM J. Matrix Anal. Appl.*, 13:1204–1246, 1992.

[19] J. W. Demmel. *Applied Numerical Linear Algebra.* SIAM, 1997.

[20] J. W. Demmel and J. J. Dongarra. LAPACK 2005 prospectus: Reliable and scalable software for linear algebra computations on high end computers. LAPACK Working Notes 164, 2005.

[21] I. S. Dhillon and B. N. Parlett. Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices. *Linear Algebra Appl.*, 387(1):1–28, 2004.

[22] I. S. Dhillon and B. N. Parlett. Orthogonal eigenvectors and relative gaps. *SIAM J. Matrix Anal. Appl.*, 25(3):858–899, 2004.

[23] J. Dongarra, J. D. Croz, S. Hammarling, and R. J. Hanson. An extended set of FORTRAN basic linear algebra subprograms. *ACM Trans. Math. Software*, 14(1):1–17, 1988.

[24] J. J. Dongarra and R. A. van de Geijn. Reduction to condensed form for the eigenvalue problem on distributed architectures. *Parallel Computing*, 18(9):973–982, 1992.

[25] Jack Dongarra, S. J. Hammarling, and D. C. Sorensen. Block reduction of matrices to condensed forms for eigenvalue computations. *J. Comput. Appl. Math.*, 27:215–227, 1989.

[26] A. Dubrulle. The multishift QR algorithm: Is it worth the trouble? Palo Alto Scientific Center Report G320-3558x, IBM Corp., 1991.

[27] A. Dubrulle and G. H. Golub. A multishift QR iteration without computation of the shifts. *Numer. Algorithms*, 7:173–181, 1994.

[28] K. Fernando and B. N. Parlett. Accurate singular values and differential qd algorithms. *Numer. Math.*, 67(2):191–229, 1994.

[29] J. G. F. Francis. The QR transformation. a unitary analogue to the LR transformation. I. *Comput. J.*, 4:265–271, 1961/1962.

[30] J. G. F. Francis. The QR transformation. II. *Comput. J.*, 4:332–345, 1961/1962.

[31] K. Fukuzawa, K. Kitaura, K. Nakata, T. Kaminuma, and T. Nakano. Fragment molecular orbital study of the binding energy of ligands to the estrogen recepter. *Pure and Applied Chemistry*, 75:2405–2410, 2003.

[32] G. H. Golub and C. F. van Loan. *Matrix Computations.* Johns Hopkins University Press, third edition, 1996.

[33] K. Goto and R. A. van de Geijn. On reducing TLB misses in matrix multiplication. Technical Report TR-2002-55, The University of Texas at Austin, Department of Computer Sciences, 2002.

[34] M. Gu and S. C. Eisenstat. A stable and efficient algorithm for the rank-1 modification of the symmetric eigenproblem. *SIAM J. Matrix Anal. Appl.*, 15(4):1266–1276, 1994.

[35] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. *SIAM J. Matrix Anal. Appl.*, 16(1):172–191, 1995.

[36] G. Henry and R. van de Geijn. Parallelizing the QR algorithm for the unsymmetric algebraic eigenvalue problem: Myths and reality. *SIAM J. Sci. Comput.*, 17(4):870–883, 1996.

[37] G. Henry, D. S. Watkins, and J. Dongarra. A parallel implementation of the nonsymmetric QR algorithm for distributed memory architectures. *SIAM J. Sci. Comput.*, 24(1):284–311, 2002.

[38] Y. Inadomi, T. Nakano, K. Kitaura, and U. Nagashima. Definition of molecular orbitals in fragment molecular orbital method. *Chemical Physics Letters*, 364:139–143, 2002.

[39] M. Iwasaki and Y. Nakamura. An application of the discrete lotka-volterra system with variable step-size to singular value decomposition. *Inverse Problems*, 20:553–563, 2004.

[40] M. Iwasaki and Y. Nakamura. Accurate computation of singular values in terms of shifted integrable schemes. *Japan J. Indust. Appl. Math.*, 23, 2006.

[41] B. Kågström and D. Kressner. Multishift variants of the QZ algorithm with aggressive early deflation. LAPACK Working Notes 173, 2006.

[42] L. Kaufman. A parallel QR algorithm for the symmetric tridiagonal eigenvalue problem. *J. Parallel and Distributed Comput.*, 3:429–434, 1994.

[43] D. Kressner. *Numerical Methods for General and Structured Eigenvalue Problems.* Springer-Verlag, 2005.

[44] D. Kressner. On the use of larger bulges in the QR algorithm. *Electronic Trans. on Numer. Anal.*, 20:50–63, 2005.

[45] V. N. Kublanovskaya. On some algorithms for the solution of the complete eigenvalue problem. *U.S.S.R. Comput. Math. and Math. Phys.*, 3:637–657, 1961.

[46] T. Miyata and Y. Yamamoto. A fully pipelined multishift QR algorithm for the parallel solution of symmetric tridiagonal eigenproblem. in preparation.

[47] C. B. Moler and G. W. Stewart. An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.*, 10:241–256, 1973.

[48] B. N. Parlett. *The Symmetric Eigenvalue Problem.* SIAM, 1997.

[49] H. Rutishauser. Solution of eigenvalue problems with the LR transformation. *National Bureau of Standards Applied Mathematics Series*, 49:47–81, 1958.

[50] A. H. Sameh and D. J. Kuck. A parallel QR algorithm for symmetric tridiagonal matrices. *IEEE Trans. Comput.*, C-26:147–153, 1977.

[51] R. Schreiber and C. F. van Loan. A storage-efficient WY representation for products of householder transformations. *SIAM J. Sci. Stat. Comput.*, 10(1):53–57, 1989.

[52] B. I. Shraiman and E. D. Siggia. Scalar turbulance. *Nature*, 405:639–646, 2000.

[53] R. Suda, A Nishida, and Y Oyanagi. A high performance parallelization scheme for the Hessenberg double shift QR algorithm. *Parallel Computing*, 25(6):729–744, 1999.

[54] R. A. Van de Geijn. Deferred shifting schemes for parallel QR methods. *SIAM J. Matrix Anal. Appl.*, 14(1):180–194, 1993.

[55] K. R. Wadleigh and I. L. Crawford. *Software Optimization for High Performance Computing: Creating Faster Applications.* Prentice Hall, 2000.

[56] D. S. Watkins. Shifting strategies for the parallel QR algorithm. *SIAM J. Sci. Comput.*, 15:953–958, 1994.

[57] D. S. Watkins. The transmission of shifts and shift blurring in the QR algorithm. *Linear Algebra Appl.*, 241/243:877–896, 1996.

[58] D. S. Watkins and L. Elsner. Convergence of algorithms of decomposition type for the eigenvalue problem. *Linear Algebra Appl.*, 143:19–47, 1991.

[59] R. Whaley, A. Petitet, and Jack Dongarra. Automated empirical optimizations of software and the ATLAS project. *Parallel Computing*, 27:3–35, 2001.

[60] B. Wilkinson and M. Allen. *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers.* Prentice Hall, 1999.

[61] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, 1965.

[62] Y. Yamamoto. Performance modeling and optimal block size selection for the small-bulge multishift QR algorithm. to be presented at ISPA2006.

[63] Y. Yamamoto. Performance of fully BLAS-3 based tridiagonalization algorithms on modern SMP machines. in preparation.

山本有作（正会員）〒464-8603 愛知県名古屋市千種区不老町

1992年 東京大学大学院工学系研究科 物理工学専攻 修士課程終了．同年（株）日立製作所入社．現在，名古屋大学大学院工学研究科 計算理工学専攻 助教授．並列数値計算アルゴリズムとその応用の研究に従事．博士（工学）．

（2006年3月31日 受付）
（2006年10月4日 最終稿受付）