

## ハウスホルダーQR分解のための AllReduceアルゴリズムの性能と精度

森 大 介<sup>†1</sup> 山 本 有 作<sup>†1</sup> 張 紹 良<sup>†1</sup>

従来、QR分解を計算するに当たって、直交性の高い $Q$ が得られることからハウスホルダー変換が用いられてきた。しかし、並列計算によって高速化を図る際に、アルゴリズムの強い逐次性から十分な並列性能が発揮できたとはいえない。そこへ近年、並列粒度の高いAllReduceアルゴリズムが提案された。このアルゴリズムは行列を上下に分割してそれぞれに対してQR分解を行うことにより、並列粒度の増大を可能としている。AllReduceアルゴリズムは様々なQR分解手法に適用できるが、本稿ではハウスホルダー変換による手法に言及し、従来の手法とAllReduceアルゴリズムを用いた手法の性能の比較ならびに精度の比較を行う。

### Performance and accuracy of the AllReduce algorithm for the Householder QR factorization

DAISUKE MORI,<sup>†1</sup> YUSAKU YAMAMOTO<sup>†1</sup> and SHAO-LIANG ZHANG<sup>†1</sup>

Up to now, the Householder QR algorithm has been one of preferred methods for QR factorization because of its high orthogonality of  $Q$ . However, due to its strong sequential nature, it is not straightforward to accelerate the algorithm by parallel computation. Recently the AllReduce algorithm, which has large grain size, was proposed. This algorithm makes it possible to increase the grain size by dividing the target matrix into the upper and the lower submatrices and performing the QR factorization of each submatrix independently. While the AllReduce algorithm is applicable to various QR factorization methods, we focus on application to the Householder QR algorithm in this report. We will compare the performance and the accuracy of the conventional method and the AllReduce algorithm.

#### 1. はじめに

$M \times N$  行列  $A$  を  $M \times N$  列直交行列  $Q$  と  $N \times N$  上三角行列  $R$  に分解する QR 分解は様々な応用分野で利用されている基本的な線形計算である。特に  $M \gg N$  の長方形行列に対する QR 分解は、特異値分解<sup>2)</sup>、最小自乗法、電子状態計算<sup>7)</sup> などの分野で利用され、大規模な行列に対する高速かつ高精度な手法が求められている。QR 分解を行うアルゴリズムとして、グラムシュミットの直交化やハウスホルダー変換が用いられている。 $Q$  の直交性の点ではハウスホルダー変換が優れているとされ、これまでブロック化手法など様々な高速化手法が提案されている<sup>1)3)</sup>。これらの手法は主にキャッシュの有効活用を目指す方法である。

また、別の高速化手法として、並列計算が挙げられる。この観点からみるとハウスホルダー変換に比べて古典的グラムシュミット直交化が良いとされる。ハウ

スホルダー変換が並列計算に向いていない最大の要因は計算の逐次性によるものである。計算手順が逐次的であり、並列化は AXPY 演算、内積演算などの BLAS レベル 1 に対してしか行えないため、並列粒度が小さく、プロセッサ間の通信コストが大きい。

そこで近年、Langou(ら)により AllReduce アルゴリズムが提案された。このアルゴリズムは従来のハウスホルダー変換に比べて演算量が増加するものの、非常に並列粒度が大きく並列計算による大幅な高速化が見込める。

本稿では、AllReduce アルゴリズムの性能および精度の評価について述べる。以下、第 2 章ではハウスホルダー変換による QR 分解について、第 3 章では数値実験結果、第 4 章では実験に対する考察を述べることとする。最後に第 5 章でまとめと今後の課題を述べる。

#### 2. ハウスホルダー変換による QR 分解

##### 2.1 従来のハウスホルダー変換

ハウスホルダー変換は直交変換の一種で、与えられたベクトルの第一成分以外の成分を 0 にする変換である。この変換を行列の各列に一行ずつずらして作用さ

<sup>†1</sup> 名古屋大学大学院工学研究科計算理工学専攻  
Department of Computational Science and Engineering,  
Graduate School of Engineering, Nagoya University

せることにより行列を三角化することで、その行列の QR 分解を得る。変換を作用させるベクトルを  $a_i$  で表し、以下に簡略にアルゴリズムを示す。

```

algorithm Householder QR:
for i = 1 to N
  compute  $\|a_i\|_2$  (□)
  generate  $u_i$  from  $\|a_i\|_2$ 
   $H_i = I - 2u_i u_i^T$ 
   $A_i = H_i A_i$  (◇)
end for

```

ハウスホルダー変換で得られる列直交行列  $Q$  は高い直交性を持つが、列  $i$  に対する変換  $H_i$  の作成は  $H_1, H_2, \dots, H_{i-1}$  を列  $i$  に作用させた後でないと行えないため、計算の逐次性が非常に強い。いま、並列計算機上で行列  $A$  を上下に分割して各プロセッサに格納してあるとすると、並列化が考えられる部分は (□) と (◇) に現れる計算である。これらの計算は各  $i$  に対して順々に行われ、2 並列の場合を考えても、総通信回数は  $2N$  回必要となる。つまり、並列計算でハウスホルダー変換の高速化を図ると並列粒度の小さい非効率な並列計算が行われることとなる。

### 2.2 AllReduce アルゴリズム

従来のハウスホルダー変換での QR 分解は並列計算による非効率性は上で述べたとおりである。それに対し大粒度の並列化が可能な AllReduce アルゴリズムが提案された<sup>5)</sup>。図 1 のように行列を上 ( $A_1$ ) 下 ( $A_2$ ) に分割し、それぞれを QR 分解する。アルゴリズムは以下のものである。なお、アルゴリズム中の *Householder QR*( $A, Q, R$ ) は入力行列  $A$  に対し、出力  $Q, R$  が得られるハウスホルダー QR 分解とする。

```

algorithm AllReduce:
Householder QR( $A_1, Q_1, R_1$ ) (*)
Householder QR( $A_2, Q_2, R_2$ ) (**)
 $\tilde{R} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix}$  (***)
Householder QR( $\tilde{R}, \tilde{Q}, R$ )

```

このアルゴリズムから、

$$A = \begin{pmatrix} Q_1 & O \\ O & Q_2 \end{pmatrix} \tilde{Q} R \quad (1)$$

という分解を得る。ここで、 $Q_1, Q_2$  は  $\frac{M}{2} \times N$  行列、 $\tilde{Q}$  は  $2N \times N$  行列、 $R$  は  $N \times N$  行列である。

式 (1) で得られた  $Q_1, Q_2, \tilde{Q}$  はすべて列直交行列であるため、第一、二項の積

$$\begin{pmatrix} Q_1 & O \\ O & Q_2 \end{pmatrix} \tilde{Q}$$

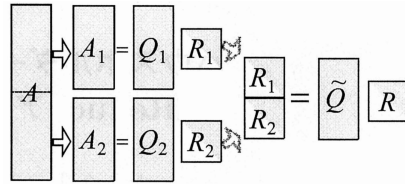


図 1 AllReduce アルゴリズム：イメージ図

CPU	Intel Xeon 2.8GHz
メモリ	4GB
OS	Red Hat Enterprise Linux
コンパイラ	gcc version 3.2.3

も列直交行列で、 $R$  が上三角行列であることから、もとの行列  $A$  の QR 分解に対する  $Q$  と  $R$  と等価な  $\tilde{Q}$  と  $R$  を得られたことになる。

このアルゴリズムの利点は並列計算を考えたときに (\*) と (\*\*) の計算の際に通信コストがかからず各々独立に計算できることである。実際の並列計算の際、通信コストがかかるのは (\*\*\*) のときの  $R_2$  の受け渡し 1 回の通信のみになる。また、AllReduce アルゴリズムは再帰的に呼び出すことにより、行列の分割数を増やすことができる。しかし、後に示すが分割数が増えるにつれ演算量も増加するという問題点も持つ。

### 2.3 演算量の比較と評価

ハウスホルダー変換にもとづく QR 分解の演算量は以下のようになる。

$$2MN^2 - \frac{2}{3}N^3 + O(MN). \quad (2)$$

これに対し、AllReduce アルゴリズムは再帰の段数を  $k$  とすると演算量は

$$2MN^2 + \frac{8 \cdot 2^k - 10}{3}N^3 + O(MN) \quad (3)$$

となる。従来のハウスホルダー変換は再帰段数  $k = 0$  のときとみなせる。この評価式をみると、再帰段数が増えるにつれ、演算量は指数関数的に増大してゆくことがわかる。

## 3. 数値実験

### 3.1 計算時間の比較

2.3 節の演算量評価式を確かめるべく、 $4000 \times 100$  の乱数行列を QR 分解するのに逐次演算で要する時間を計測した。パラメータとして再帰段数  $k$  を 0 から 5 まで変化させた。プログラム言語は C で、その他の計算機環境は表 1 のようである。図 2 をみると、 $k$  が増えるにつれ計算時間が増加してゆく様子がわかる。

### 3.2 精度の比較

まず、従来のハウスホルダー変換と再帰段数  $k = 1$  の AllReduce アルゴリズムの精度の比較を行った。両方のアルゴリズムに対して、表 2 で示す様々な行列サイズで実験を行った。行列の各要素は  $[0, 1)$  の乱数で

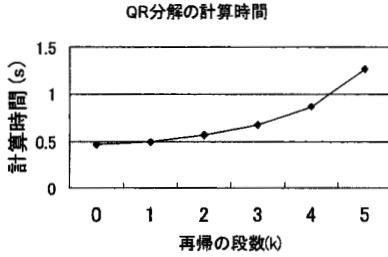


図2 従来のハウスホルダー変換と AllReduce アルゴリズムの計算時間 (行列サイズ  $4000 \times 100$ )

	M	N
I	4000	100~500
II	1000~5000	100

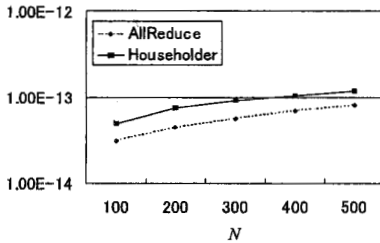


図3  $\|Q^T Q - I\|_F$   
 $M = 4000$

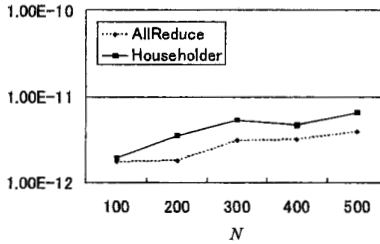


図4  $\|QR - A\|_F$   
 $M = 4000$

ある。比較項目は  $Q$  の直交性  $\|Q^T Q - I\|_F$  および残差  $\|QR - A\|_F$  である。以後、この二つのフロベニウスノルムを精度と呼ぶこととする。

$M = 4000$  と固定し、 $N$  を変えた場合の両アルゴリズムの精度を図3, 4に、 $N = 100$  と固定し、 $M$  を変えた場合の精度を図5, 6に示す。図3~6をみると両方の手法において行列サイズが大きくなるほど精

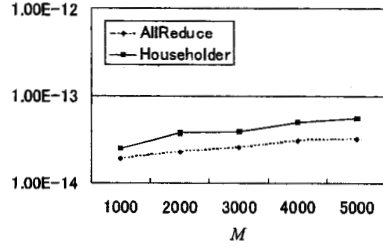


図5  $\|Q^T Q - I\|_F$   
 $N = 100$

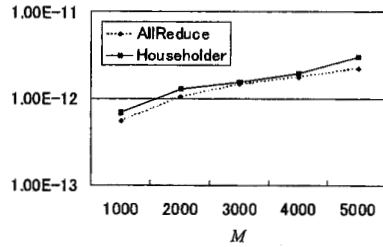


図6  $\|QR - A\|_F$   
 $N = 100$

度が悪化していることがわかる。また、同一サイズで手法による精度を比較すると、従来のハウスホルダー変換の精度よりも AllReduce アルゴリズムのほうが良い結果を示した。

次に、分解する行列を

$$A = Q'(\alpha E + I)$$

とした。 $Q'$  は任意の  $M \times N$  列直交行列、 $E$  はすべての行列成分が1の  $N \times N$  行列、 $\alpha$  は任意の実数とする。この行列は、 $\alpha$  を大きくすると、列ベクトル間の線形従属性が強くなる。実際、

$$\begin{aligned} A^T A &= (\alpha E + I) Q'^T Q' (\alpha E + I) \\ &= (\alpha E + I)^2 \\ &= (N\alpha^2 + 2\alpha) E + I \end{aligned}$$

より、 $A^T A$  の最小固有値は1, 最大固有値は  $(N\alpha + 1)^2$  となる。したがって、行列  $A$  の条件数は  $N\alpha + 1$  であり、 $\alpha$  を大きくすると、 $A$  の条件数も大きくなる。数値実験では1から  $10^6$  まで  $\alpha$  を10倍ずつ増やして条件数による精度の変化をみた。

図7をみると、条件数による直交性の悪化はみられなかった。しかし、残差は  $\alpha$  が大きくなると共に増大する(図8)。これは、 $\|A\|_F$  が  $\alpha$  にほぼ比例して増大するためである。これらの結果は通常のハウスホルダー変換に対する誤差解析の結果<sup>4)6)</sup> とよく一致している。

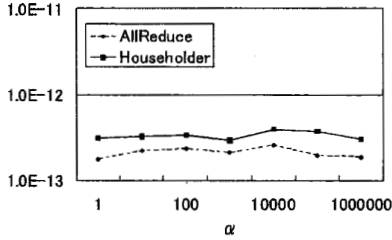


図7  $\|Q^T Q - I\|_F$  (行列サイズ 4000 × 100)

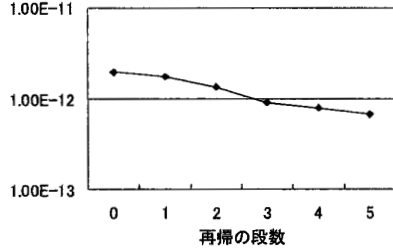


図10  $\|QR - A\|_F$  (行列サイズ 4000 × 100)

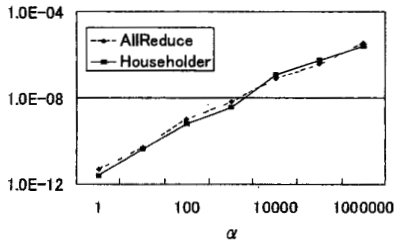


図8  $\|QR - A\|_F$  (行列サイズ 4000 × 100)

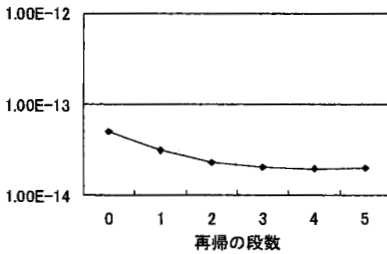


図9  $\|Q^T Q - I\|_F$  (行列サイズ 4000 × 100)

最後に再び乱数行列でサイズを 4000 × 100 に固定して、再帰段数を変化させて精度を評価した。その結果を図9, 10に示す。再帰の段数0は従来のハウスホルダー変換を意味する。

演算量が増えるにもかかわらず、再帰段数が深くなればなるほど良い精度を示す結果となった(図9, 10)。

## 4. 考 察

### 4.1 計算時間および計算量

計算時間の比較においてはおおむね評価式どおりの結果が得られた。再帰段数  $k$  が増えるにつれ、計算時間は指数関数的に増大していく。

ここで、並列化による高速化の効果を考察する。従

来のハウスホルダー変換を  $K$  並列で計算を行うとすると、計算時間は

$$\frac{1}{K} \left( 2MN^2 - \frac{2}{3}N^3 \right) + O(MN) \quad (4)$$

に従う時間へと高速化される。

この演算量を見ると、並列化すればするほど高速になるようみえるが、実際はこの計算時間に加え、 $2(K-1)N$  回の通信が必要になる。

一方で、AllReduce アルゴリズムを  $K = 2^k$  並列で計算を行うことを考える。すると、再帰の各段における QR 分解はそれぞれ独立に計算できるため、

$$\frac{M}{2^{k-1}} N^2 + \frac{10k-2}{3} N^3 + O(MN) \quad (5)$$

に相当する計算時間がかかる。

また、通信回数は、 $2^k - 1 = K - 1$  回であり、従来のハウスホルダー変換に比べてほぼ  $1/N$  の少ない通信回数で計算が実行できる。

### 4.2 精度について

前章で得られた結果を見ると、従来のハウスホルダー変換よりも演算量の多い AllReduce アルゴリズムのほうが総じて良い精度が得られている。また、AllReduce アルゴリズム同士の比較でも、演算量の多い多段再帰となるほど精度が良くなっている。

ここで、再び AllReduce アルゴリズムを細かくみていくことにする。AllReduce アルゴリズムでは行列を分割して、それを各々 QR 分解し、その結果得られる  $R$  を上下に二つ並べた行列を再び QR 分解するという手続きを繰り返す。ゆえに、QR 分解は常にもとの行列サイズに比べて小さい行列に対して行うことになる。例えば、4000 × 100 の行列を再帰段数3で8分割した時は、実際に計算される行列サイズは 500 × 100 と 200 × 100 のものとなり、もとの 4000 × 100 に比べてとても小さいサイズとなる。このことを踏まえると、前節で得られた結果は、QR 分解を複数回繰り返すことの影響よりも、小さいサイズですべての計算を行えたという利点が勝ったと考えられる。

これをもう少し理論的にみることにする。従来のハウスホルダー変換を用いて数値的に求められた  $Q$  と  $R$  を、それぞれ  $\hat{Q}, \hat{R}$  と書く。このとき、後退誤差解析の結果によれば、 $\hat{R}$  は次の式を満たすことが知られ

ている<sup>4)</sup>。

$$A + \Delta A = \bar{Q}\hat{R}. \quad (6)$$

ただし、 $\Delta A$  は、

$$\begin{aligned} \|\Delta A\|_F &\leq \frac{cMN\varepsilon}{1-cM\varepsilon}\|A\|_F \\ &\sim cMN\varepsilon\|A\|_F \end{aligned} \quad (7)$$

を満たす  $M \times N$  行列であり、 $\bar{Q}$  は QR 分解の各ステップにおけるハウスホルダー変換の積を無限精度で計算して得られる列直交行列である。また、 $\varepsilon$  は計算機イブシロン、 $c$  は小さな整数定数である。

これより、

$$\begin{aligned} \|\bar{Q}\hat{R} - A\|_F &\leq \|\Delta A\|_F \\ &\leq cMN\varepsilon\|A\|_F. \end{aligned} \quad (8)$$

$\bar{Q}$  を計算値  $\tilde{Q}$  で近似すると、

$$\|\tilde{Q}\hat{R} - A\|_F \leq cMN\varepsilon\|A\|_F \quad (9)$$

が得られる。

一方、直交性については、

$$\|\hat{Q}^T\tilde{Q} - I\|_F \leq 2\sqrt{N}\varphi(M, N)\varepsilon \quad (10)$$

が成り立つ<sup>6)</sup>。ただし、 $\varphi(M, N)$  は  $M, N$  について緩やかに増加する関数である。

以上の結果を使って、AllReduce アルゴリズムの残差について考察する。A の要素が一様な大きさの場合、 $\|A\|_F$  は  $\sqrt{M}$  に比例するから、式 (9) より、残差の上限は  $M\sqrt{M}$  に比例する。したがって、 $K$  分割した時の上限は  $1/K\sqrt{K}$  となる。また、QR 分解の繰り返しは  $\log_2 K + 1 = k + 1$  回必要となる。これは残差を大きくする方向に働くと考えられる。AllReduce アルゴリズムでは、後者の影響よりも、前者の効果が強く、残差が減少したと考えられる。しかし、この点を検証するにはより詳しい誤差解析が必要である。また、直交性が向上している点についても解析が必要である。

## 5. おわりに

今回は、従来のハウスホルダー変換に比べて AllReduce アルゴリズムが良い精度を示すことを数値実験的に示した。また、AllReduce アルゴリズムは再帰を繰り返すことにより更なる精度の改善がみられた。今後の課題としては、これらの振る舞いの理由を誤差解析を通して明らかにすること、AllReduce アルゴリズムの並列化を行いその性能を定量的にみる事が挙げられる。

## 参 考 文 献

- 1) Demmel, J. W.: *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
- 2) 深谷猛, 山本有作, 畝山多加志, 堀玄, 梅野健: 長方形行列向け特異値分解の浮動小数点コプロセッサによる高速化, 情報処理学会論文誌 ACS, Vol. 48, No. SIG8 (ACS18), pp. 31-43, 2007.

- 3) Golub, G. and Loan C.: *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, USA, Third Ed., 1996.

- 4) Higham, N. J.: *Accuracy and Stability of Numerical Algorithms*, SIAM, Philadelphia, 1996.

- 5) Langou, J.: AllReduce algorithms: application to Householder QR factorization, *Proceedings of the 2007 International Conference on Preconditioning Techniques for Large Sparse Matrix Problems in Scientific and Industrial Applications*, pp. 103-106, 2007.

- 6) Stewart, G. W.: *Matrix Algorithms. Vol. 1: Basic Decompositions*, SIAM, Philadelphia, 1998.

- 7) Toledo, S. and Van Rabani, E.: Very Large Electronic Stridtire Calculations using an Out-of-Core Filter Diagonalization Method, *J. Compute. Phys.* Vol. 180, No. 1, pp. 256-269, 2002.