ハイパフォーマンス
コンピューティング 76－7
（1999．5．14）

CGSS：ソートを用いた新しいGram-Schmidt直交化法

片桐孝洋† 金田康正††

本論文では Classical Gram-Schimidt 直交化法において，ソートを利用することにより精度や数値安定性の改善を計る新しい直交化方式を提案する．さらに加えて，この手法を日立の分散メモリ型並列計算機 SR2201 に実装し，試験行列と応用プログラムを用いて数値実験を行なった．その結果，本手法は (1) ソート後の演算方式が精度に大きく影響すること; (2) 従来の方式に比べて精度の改善が得られる場合があること; がわかった．

CGSS : A New Gram-Schmidt Orthogonalization Method with Sorting

TAKAHIRO KATAGIRI† and YASUMASA KANADA††

In this paper, we propose a new Gram-Schmidt orthogonalization method. This new method is based on Classical Gram-Schmidt (CGS) method using sorting. We expect that this new method can improve accuracy and unstability of the CGS method. In addition, we implemented the new method on the Hitachi SR2201 which is a distributed memory parallel machine, and checked accuracy for the new method with test matrices and an application program. From the experiments, we could found that (1) Orthogonal accuracy depends on the order of calculation after sorting; (2) High accuracy to conventional methods is observed; for the new method.

1. はじめに

直交化処理は固有値計算や QR 分解などの線形計算に必要とされる重要な処理である．この直交化処理は，必要とされる直交化済のベクトルの個数により，全く処理の性質が異なるという事実がある．そこで本論文では，直交化処理を以下に示す二つの種類に分類することにする．

- QR 分解 (QR factorization)
  正規ベクトル $a_1, a_2, \cdots, a_n$ から，正規直交ベクトル $q_1, q_2, \cdots, q_n$ を求める処理．
- 再直交化 (re-orthogonalization)
  すでに互いに正規直交化済みであるベクトル $q_1, q_2, \cdots, q_{i-1}$ から，正規ベクトル a_i を用いて正規直交ベクトル q_i を求める処理．

本論文では，この 2 種の直交化処理を Gram-Schmidt (GS) 直交化法を用いて並列化する場合について議論する．特に，並列性が高いが数値的に不安定で精度が低い手法である Classical GS (CGS) 法の処理の途中でソートを用いることで，不安定性と精度を改良できる可能性がある．本論文では，この新しい直交化方式を CGSS (Classical GS with Sorting) 法と呼び，これを提案することを目的とする．

精度を改善する目的で演算順序などを変更するというアイデアは，本研究に限らずいくつかの報告がなされている．浮動小数点演算の加算の順序を変える幾つかの手法とその解析は Higham[1] によってなされている．また多倍長計算の精度を向上させる目的で，加算と減算を交互に行なうようにアルゴリズムを再構築した例 [2),3)] が報告されている．

まず 2 章で GS 法の逐次アルゴリズムを紹介する．3 章において，その並列アルゴリズムについて検討する．4 章では，本報告で提案する CGSS 法の詳細を説明する．5 章において，CGSS 法と従来法の精度，および並列実行速度の観点から実験的な評価を行なう．最後に 6 章でまとめを行なう．

2. 逐次アルゴリズム

2.1 QR 分解

QR 分解を GS 直交化法を用いて行なう場合，以下に述べる 2 つの実現方法が知られている．GS 直交化方式をそのまま実現した方法である Classical GS (CGS) 法と，安定性に改良を加えた Modified GS (MGS) 法である．

QR 分解における MGS 法を **図 1** に示す．なお，図中の $(\cdot,\cdot)$ の表記は内積を示している．図 1 によって

† 東京大学大学院理学系研究科情報科学専攻
Department of Information Science, Graduate School of Science, The University of Tokyo
†† 東京大学情報基盤センタースーパーコンピューティング研究部門
Computer Centre, The University of Tokyo

```
⟨1⟩  do i = 1, n
⟨2⟩    a_i^(0) = a_i
⟨3⟩    do j = 1, i − 1
⟨4⟩      a_i^(j) = a_i^(j−1) − (q_j^T, a_i^(j−1)) q_j
⟨5⟩    enddo
⟨6⟩    q_i = a_i^(i−1) を正規化
⟨7⟩  enddo
```

図 1　QR 分解における MGS 法

得られる直交ベクトル q_i を列ベクトルに持つ行列を Q とし，計算途中で得られる内積値 $(q_j^T, a_i^{(j-1)})$ を j 行 i 列に並べて形成される上三角行列を R とする．このとき，この分解は行列 A を直交行列 Q と 上三角行列 R に分解する 処理であるので QR 分解と呼ばれる．また，QR 分解における MGS 法のデータ依存図を図 **2** に示す．図 2 からわかるように，この処

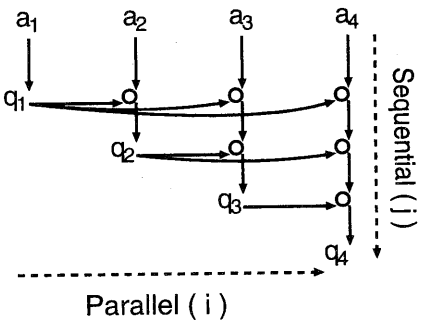

図 2　QR 分解における MGS 法のデータ依存図．○は最内核の処理 ⟨4⟩ を示している．

理では互いに直交化すべきベクトル間 (i-ループの方向) は，必要な直交ベクトル q_i が得られた時に並列性が存在する．一方で，直交化しているベクトルの方向 (j-ループの方向) にフロー依存があり，並列性がないことがわかる．

QR 分解における CGS 法を 図 **3** に示す．QR 分

```
⟨1⟩  do i = 1, n
⟨2⟩    q_i = a_i
⟨3⟩    do j = 1, i − 1
⟨4⟩      q_i = q_i − (q_j^T, a_i) q_j
⟨5⟩    enddo
⟨6⟩    q_i を正規化
⟨7⟩  enddo
```

図 3　QR 分解における CGS 法

解における CGS 法のデータ依存図を 図 **4** に示す．図 4 からわかるように，この処理では，互いに直交化すべきベクトル間 (i-ループ) の方向と直交化している

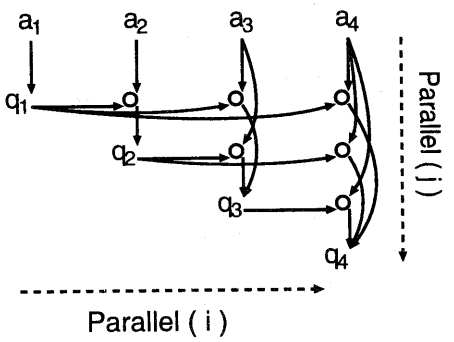

図 4　QR 分解における CGS 法のデータ依存図

ベクトルの方向 (j-ループ) の方向の両方に並列性が存在するのが大きな特徴である．

2.2 再直交化

再直交化では MGS 法，CGS 法ともに，QR 分解における最外の i-ループを除去することで実現される．よってこれらのデータ依存図は，例えば 3 つのベクトルと再直交化する場合は，図 2，4 において，$a_1 - a_3$ のデータ依存を取り除いたものに等しい．

3. GS 直交化法の並列化

3.1 QR 分解

3.1.1 行方向に分割する場合

この場合の並列化方針は，正規ベクトル a_i，正規直交ベクトル q_i，$i = 1, 2, ..., n$ の各ベクトルの要素で分割する．いま行列 A を $A = (a_1, a_2, \cdots, a_n)$，行列 Q を $Q = (q_1, q_2, \cdots, q_n)$ とする．このとき，この分割は，行列 A, Q について (Block, *) 分割や (Cyclic, *) 分割などをすることである．

MGS 法，CGS 法ともに，この分割を前提とする並列処理を行なう場合，内積演算 $(q_j^T, a_i^{(j-1)})$ および (q_j^T, a_i) を計算するために，スカラリダクション演算が必要になる．なぜなら，内積に必要なベクトルの要素を全て所有していないからである．

図 **5** に MGS 法のアルゴリズムを示す．ここで，図中の表記 Global sum とは，集団通信で総和をとり，その結果を全ての Processing Element (PE) で所有する処理を意味している．MPI (Message Passing Interface) では，`MPI_ALLREDUCE` で実現できる処理である．図 5 ではループの最も内側に Global sum があることから，毎回 スカラに対する PE 間総和を行なう必要があることがわかる．この理由は MGS 法では計算で求めたベクトルを，すぐ次の演算の内積に用いなければならないことによる．このことは 図 2 のデータ依存図で，直交化しているベクトルの方向 (j-ループ) にフロー依存があることからも明らかである．

次に，CGS 法のアルゴリズム図 **6** を示す．図 6 か

```
⟨1⟩  do i = 1, n
⟨2⟩    a_i^(0) = a_i
⟨3⟩    do j = 1, i − 1
⟨4⟩      Local (q_j^T, a_i^(j−1))
⟨5⟩      Global sum of η = (q_j^T, a_i^(j−1))
⟨6⟩      Local a_i^(j) = a_i^(j−1) − η q_j
⟨7⟩    enddo
⟨8⟩    q_i = a_i^(i−1) を正規化
⟨9⟩  enddo
```

図 5　行方向分割における MGS 法

```
⟨1⟩  do i = 1, n
⟨2⟩    q_i = a_i
⟨3⟩    do j = 1, i − 1
⟨4⟩      Local (q_j^T, a_i)   enddo
⟨5⟩    Global sum of η_j = (q_j^T, a_i)
                        (j = 1, ···, i − 1)
⟨6⟩    do j = 1, i − 1
⟨7⟩      Local q_i = q_i − η_j q_j
⟨8⟩    enddo
⟨9⟩    q_i を正規化
⟨10⟩ enddo
```

図 6　行方向分割における CGS 法

らわかるように，CGS 法では毎回 Global sum を行なう必要はない．この理由は，CGS 法では計算で求めたベクトルを，すぐ次の演算の内積に用いる必要がないことによる．このことは 図 4 のデータ依存図で，直交化しているベクトルの方向（j-ループ）に依存がなく，正規化の直前で加算するだけで良いことからも明らかである．さらに加えて，CGS 法では Global sum はベクトルに対して行なっている．その長さは 0, 1, 2, ···, $n-1$ まで順に大きくなっている．

最後に MGS 法と CGS 法との Global Sum の実行回数を比較すると，$(n^2-n)/2$ 回 と $n-1$ 回 なので，CGS 法のほうが実行速度の観点では有効であるといえる．しかしながら精度の観点では，理論上は MGS 法の精度 は CGS 法の精度よりも良い．これが CGS 法の問題点である．

3.1.2　列方向に分割する場合

この場合の分割法では，正規ベクトル a_i，正規直交ベクトル q_i の組 (a_i, q_i) を同一プロセッサに割り当てる方法である．同様に行列 A, Q を定義するとき，(*, Block) 分割，(*, Cyclic) 分割などを行なうことに相当する．またデータ依存図から，(*, Block) 分割は負荷バランスは (*, Cyclic) 分割よりも悪いことが明らかなので，(*, Cyclic) 分割の方が良い．しかし，(*, Cyclic) 分割をすると，正規直交ベクトルが求まるごとに，全プロセッサに放送しなくてはならなくなる．よって，この放送回数を減らす目的から，ブロックサイクリック分割 (*, Cyclic(m)) が有効となる可能性がある．またこの分割方式では，MGS 法，CGS 法とも同じ回数の放送処理が必要となる．

以上のように列方向にデータを分散する方法も並列性能の観点では有効であると考えられる．しかし本稿では議論を簡単にするために，これら列方向アルゴリズムに関してはこれ以上触れないことにする．

3.2　再 直 交 化

3.2.1　行方向に分割する場合

MGS 法，CGS 法はともに 図 5，6 の最外ループである i-ループを取り除き，i に $i-1$ 個の直交化済みベクトルと直交化するベクトル番号をを設定することで実現される．処理の振舞いは QR 分解と類似しているので省略する．

3.2.2　列方向に分割する場合

再直交化において列方向に分割する場合，MGS 法は完全に逐次化されて並列化はできない．この理由は 図 2 における直交化している方向 (j-ループの方向) にフロー依存があることによる．一方，CGS 法は並列化が可能である．我々は既に，この方式を用いた並列 CGS 法を並列固有値ソルバ [4),5)] に実装している．

4.　CGS 法における精度改良手法の提案

まず CGS 法の精度悪化の要因を考える．この要因として

要因 (i)　理論上の直交化の崩れ

要因 (ii)　浮動小数点の加減算による情報の埋没

の 2 つが考えられる．

要因 (i) は QR 分解では避けられない問題である．ところが再直交化においては，直交させるべきベクトルを MGS 法などを用いて高精度で計算しておけば，たとえ CGS 法を用いたとしても 要因 (i) による精度悪化は理論上は無くなるものと考えられる．

また CGS 法では，精度の低下要因として要因 (ii) の影響も大きいと考えられる．要因 (ii) を改善するため，まず Å. Björck による浮動小数演算における誤差解析の結果 [6)] を示す．Å. Björck は，GS 法の主演算である

$$a_i^{(j)} = a_i^{(j-1)} - \eta_{ji} q_j \qquad (1)$$

において，

$$||a_i^{(j)}||_2 \ll ||a_i^{(j-1)}||_2 \qquad (2)$$

となる場合に，深刻な直交化の崩れが生じることを示した [6)]．この結果はデータの依存関係は異なるが，主演算の観点で MGS 法，CGS 法ともに同じであることから，どちらの方法についてもいえるという点に注意する．

以上の議論から，CGS 法における精度悪化を解決する方法として以下の手法が考えられる．すなわち直

交ベクトル計算の際の

$$a_i^{(j)} = a_i - \eta_{1i}q_1 - \eta_{2i}q_2 - \cdots - \eta_{i-1,i}q_{i-1}$$

の演算に関して，演算順序を変更し各値 η_{ji} の差が極端に違わないベクトルについて演算を行なうようにする．このことにより，式 (2) の条件が生じ難くなることが期待できる．よって，各値 η_{ji} をキーとしてソートした後に演算をする手法が考えられる．このアイデアを適用した CGS 法 による直交化処理の方法を **CGSS (CGS with Sorting)** 法と呼ぶことにする．CGSS 法のアルゴリズムを図 **7** に示す．

```
⟨1⟩  q_i = a_i
⟨2⟩  do j = 1, i-1
⟨3⟩    η_j = (q_j^T, a_i)
⟨4⟩  enddo
⟨5⟩  (ζ_1, ζ_2,..., ζ_{i-1}) = Sort(η_1, η_2,..., η_{i-1})
⟨6⟩  do j = 1, i-1
⟨7⟩    idx = Index(ζ_j)
⟨8⟩    q_i = q_i - ζ_j · q_idx
⟨9⟩  enddo
```

図 7　再直交化に対するアルゴリズム CGSS. 図中の関数 Sort() はソートする関数を示している．また 関数 Index() は並べ換えたデータに対して，並べ変える前の添字を返す関数である．

5. 性能評価

ここでは，日立 SR2201 に本稿で示した 各 GS 直交化法を実現し，その性能について評価する．SR2201 の各 PE の理論ピーク性能は 300MFlops, PE 間は三次元クロスバ網で結合されており，その最大転送性能は 300Mbyte/秒である☆．

5.1 実験条件

5.1.1 直交性の定義

まずはじめに，MGS 法と CGS 法の直交性を定義する．QR 分解に対する直交性は，各直交化済ベクトル q_i からなる行列 $Q = (q_1, q_2, ..., q_n)$ に関するユークリッド・ノルム

$$||I - Q^TQ||_E \tag{3}$$

で定義する．また再直交化に対しては直交性を

$$\text{if } i \neq j,\ \max\sqrt{|q_i^Tq_j|}\ (i,j = 1,2,\cdots,k)$$
$$\text{otherwise, } \max\left|1 - \sqrt{|q_i^Tq_i|}\right|\ (i = 1,2,\cdots,k) \tag{4}$$

と定義して議論を行なうことにする．直交化には倍精度演算を用い，その計算されたベクトルに対する直交性の計算には四倍精度演算を用いて検証した．

5.1.2 テスト行列

テスト行列 A として，以下に示す 2 種の行列を用いた．

- 乱数行列

行列の各要素 x が $0 \leq x < 1$ の区間に一様分布する疑似乱数により構成される行列．

- Läuchli 行列 [6]

$$A = \begin{pmatrix} 1 & \cdots & 1 \\ \epsilon & & \\ & \ddots & \\ & & \epsilon \end{pmatrix} \tag{5}$$

という行列．この行列は，ϵ が小さくなるにつれ CGS 法による QR 分解が破綻するという特長をもつ．

5.1.3 CGSS 法の実装詳細

次に，CGSS 法ではソート後におけるデータの演算方法が精度に影響することが予想される．すなわち

(i) 正，負の順を考慮しない
(ii) 正，負の順に演算する
(iii) 正，負の各数値ごとに演算後，両者を加算
(1) 符号付き実数としてソート
(2) 絶対値実数としてソート
(a) 小さい数から演算する
(b) 大きい数から演算する

などの要因を考慮する必要がある．本実験では，以下に示す計 9 通りの方法で CGSS 法を実装して評価した．

図中の表記	方法	図中の表記	方法
CGSSsp	(i)(1)(a)	CGSSdsp	(ii)(1)(a)
CGSSsm	(i)(1)(b)	CGSSdsm	(ii)(1)(b)
CGSSap	(i)(2)(a)	CGSSdap	(ii)(2)(a)
CGSSam	(i)(2)(b)	CGSSdam	(ii)(2)(b)
		CGSSdsam	(iii)(2)(b)

なお，本実験で用いたソートには，再帰を除去し，さらに挿入法と組みあわせたクイックソートを用いている．このクイックソートにおける挿入法との切替え幅は 10 とした．

5.2 QR 分解

5.2.1 行方向に分割する場合

図 **8** に，$n = 1500$，ベクトル次元 1500 における QR 分解の各直交化方式の実行時間 (乱数行列) を示す．図 8 からわかるように，MGS 法は並列化の効果がない．この理由は，スカラリダクションが毎回必要なので通信時間が PE 数が増加するにつれて増大することによると考えている．

一方，CGS 法，CGSS 法では並列化の効果が観測

☆ 東京大学情報基盤センターが所有している 1024PE の SR2201 のうち 128PE を使用した．また，コンパイラとして 日立の最適化 FORTRAN90 V02-06-/B，オプションとして-rdma -W0,'PVEC(PVFUNC(1), VERCHK(0), DIAG(1)), opt(o(s), fold(2), prefetch(1), rapidcall(1), ischedule(3), reroll(1), scope(1), split(2), uinline(2)) を指定した．測定日は 1999 年 1 月 27 日 から 4 月 19 日である．

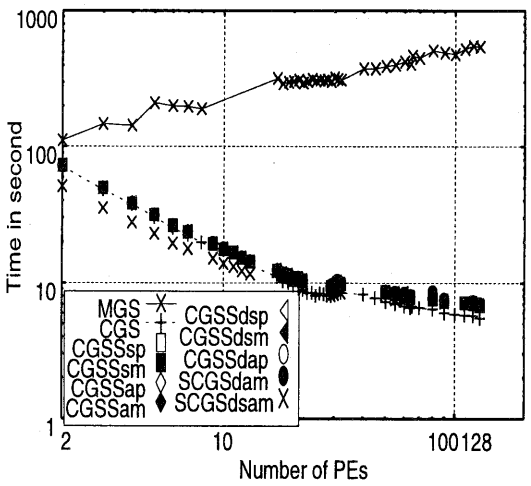

図 8 QR 分解における実行時間 (乱数行列, $n = 1500$, ベクトル次元 1500)

された．このことから，スカラリダクションの削減が並列性能の劇的な向上に寄与することがわかる．図 8 から，CGSS 法の実行時間は CGS 法とあまり差がない．また，CGSSdsam 法が CGS 法よりも実行時間が短くなっているが，この原因は解析中である．

図 **9** に QR 分解における直交性を計算した結果を示す．図 9 から，

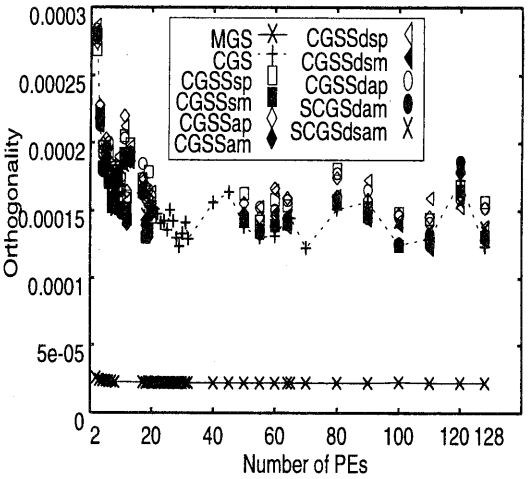

図 **9** QR 分解における直交性 (乱数行列, $n = 1500$, ベクトル次元 1500)

i) MGS 法は CGS 法よりも 1/2 桁ほど精度が良い
ii) CGSS 法は CGS 法に対して，精度が改良される場合もあれば，改悪される場合もある
iii) CGS 法，CGSS 法は PE 台数が増えると精度が改良される傾向がある

ことがわかる．i), ii) から乱数行列を用いたこの例では，CGSS 法は MGS 法に対して良い精度を得られるわけではなく，また必ずしも CGS 法を越える精度が得られる方法ではないことがいえる．一方 iii) は，並列処理により精度改善が望めることを意味しているように思える．しかしながら本稿で示した並列 GS 法は，計算を分割して一時的な記憶領域を PE 台数分用意することで逐次計算機で完全に模倣できる．このことを考慮すると，iii) から「計算を分割したこと」により精度の改善が期待できる可能性があると結論づけられる．また並列処理により，(1PE 当たり) 少ない記憶領域と速い実行時間で同様の結果を期待できる．これらの観点から，並列 GS 法は逐次 GS 法に対して記憶領域と実行時間の利点があるといえる．

5.3 再直交化

5.3.1 行方向に分割する場合

ここでは与えられた行列 A に対する再直交化による CGSS 法の精度を検討する．特に直交化済のベクトルの精度が，再直交化したベクトルの精度に影響することから，既に直交化済のベクトルは MGS 法を用いて直交化を行なう場合に限って評価する．

図 10 は Läuchli 行列において $\epsilon = 1$ とした場合の直交性を示している．図 10 から

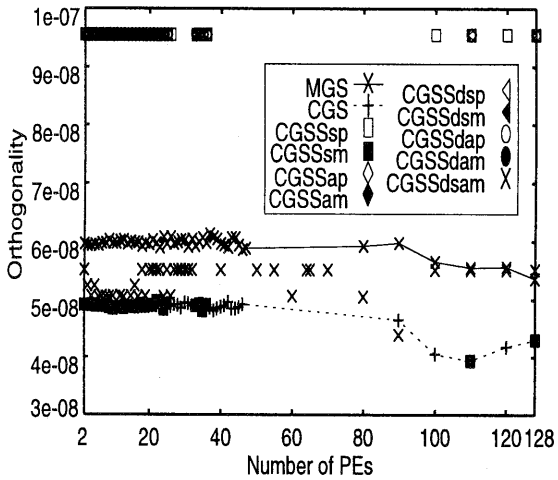

図 **10** 再直交化における MGS で直交済ベクトルを用意した場合の直交性．(Läuchili 行列, $n = 1500$, $k = 1500$, $\epsilon = 1$)

i) MGS 法よりも CGS 法や CGSS 法の精度が良い場合がある
ii) 「計算を分割する」と直交性が向上する傾向があまりない
iii) CGS 法と CGSS 法の直交性の差が生じている
iv) CGSS 法において，ソート後に小さな数から演算する場合の直交性が極端に悪化している

ことがわかる．特に i), iv) の結果が重要であると考

える．すなわち，(1)MGS 法と CGS 法の直交性は入力行列の性質に依存する; (2)CGSS 法の演算方式も入力データに依存し，特にソート後に小さな数から演算すると大きな精度悪化を引き起こすことがある; ことがいえる．

5.4 応用プログラムを用いた評価

ここでは，応用プログラムにおける CGSS 法の性能を評価する．そこで CGSS 法を，21 次元の Wilkinson 行列で構成された glued Wilkinson 行列 W_g^+ の全固有値，全固有ベクトルを 二分法 + 逆反復法 で求める並列固有値ソルバ [4),5)] の再直交化部分に実装した．この並列固有値ソルバ [4),5)] では，固有ベクトルの分割方式が (*, Block) 方式を用いて設計されている．この理由から 図 7 の CGSS 法を実装すると，ソート後の直交化処理が逐次化されるので高速化が期待できない．よってここでは CGSS 法における，直交性の評価のみを行なうことを目的とする．

本実験では問題サイズを 1260 次元とした．この場合，−1.1–10.8 の間に 60 個 ごとに重複した固有値群をなす分布になっている．

5.4.1 直 交 性

図 11 は，各直交化方式を再直交化に用いた場合の全固有ベクトルの直交性をユークリッド・ノルムで評価した結果である．なお，図 11 中の distance とは，固有値が重複しているとみなす距離を制御する定数 [4)] である．この定数が大きいほどその距離が長くなるので，多くの固有値が重複しているものとみなされる．

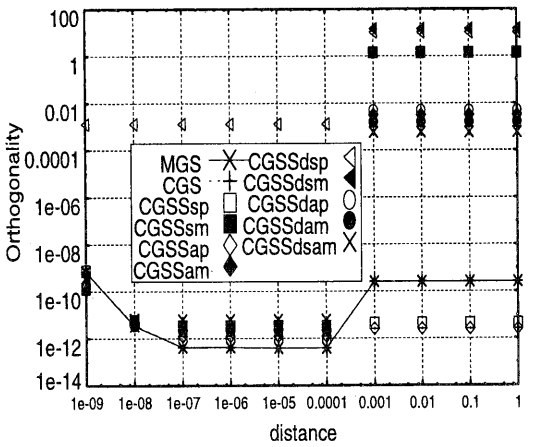

図 11 各直交化方式を用いた逆反復法によって計算された固有ベクトルの直交性 ($n = 1260$ の場合)

図 11 によると，distance が大きい場合 (重複固有値が多い場合) 以下のことが指摘できる．

- CGSS 法 (CGSSsp，CGSSap) が従来手法より 1 桁ほど直交性が良い．
- CGSS 法が極めて精度が悪くなる例が観測された．

以上から，CGSS 法を用いることで直交性を従来手法に対して改善できる例があるといえる．しかしながらソート後の処理の選択が不適切であれば，直交性が極端に悪化する要因になる．

6. おわりに

本稿では，精度改善の為に CGS 法にソートを付けた新しい直交化方式である CGSS 法を提案した．数値実験の結果から，CGSS 法 は CGS 法 に対して精度改善がほとんどない場合もあるが，1 桁程度の精度改善が得られる例も確認された．一方で我々は，GS 法において「計算を分割すること」により，分割しない場合よりも精度が改善できることを数値実験より発見した．並列処理によりこの効果が記憶容量と実行速度の観点で容易に得られると期待できることから，GS 法の並列処理は逐次処理に対して「2 重に」有利である．

また CGSS 法のソート後における計算順序が，精度の悪化を導くことがある．このことから CGSS 法では，ソート後における計算順序の決定が極めて重要であるといえる．また我々は，ソートされた内積値の分布等を解析することで，適した演算順序を決定できると考えている．今後の課題として，内積値の分布等と精度の関係を解析することがあげられる．この内積値の分布等と精度の関係が明らかになれば，CGSS 法において自動的に演算順序を決定する手法が確立できる．このことで CGSS 法は，精度と並列性の観点で有効な手法となりうるはずである．

参 考 文 献

1) Higham, N. J.: The Accuracy of Floating Points Summation, *SIAM J. Sci. Comput.*, Vol. 14, No. 4, pp. 783–799 (1993).
2) Crandall, R. and Fagin, B.: Discrete Weighted Transforms and Large-Integer Arithmetic, *Mathmatics of Computation*, Vol. 62, No. 205, pp. 305–324 (1994).
3) 高橋大介, 金田康正: 分散メモリ型並列計算機による円周率の 515 億桁計算, 情報処理学会論文誌, Vol. 39, No. 7, pp. 2074–2083 (1998).
4) 片桐孝洋, 金田康正: 並列固有値ソルバーの実現とその並列性の改良, JSPP'98 論文集, pp. 223–230 (1998).
5) Katagiri, T. and Kanada, Y.: A Parallel Implementation of Eigensolver and Its Performance, *Ninth SIAM Conference on Parallel Processing for Scientific Computing* (1999). As a poster.
6) Björck, Å.: Numerics of Gram-Schmidt Orthogonalization, *Linear Argebra and its Applications*, Vol. 197–198, pp. 297–316 (1994).