数理解析研究所講究録
1040巻 1998年 180-187

ブロック5重対角行列群に対するベクトル計算機向けの効率的な解法について

筑波大学大学院理工学研究科　伊藤　祥司 (Shoji ITOH)
筑波大学電子情報工学系　張　紹良 (Shao-Liang ZHANG)
筑波大学電子情報工学系　名取　亮 (Makoto NATORI)

1 はじめに

3次元圧縮性流体を記述するナビエ・ストークス方程式を解く際，4次精度 AF 法 (Approximate Factorization method) により離散化すると，計算空間の3方向 (i,j,k) においてブロック5重対角行列群が現れる．この時，物理方程式の求解は，その行列群 $\hat{A}_{j,k}$ （i は行方向をなす）を係数行列とする複数の連立一次方程式

$$\hat{A}_{j,k}\boldsymbol{x}_{j,k} = \boldsymbol{b}_{j,k} \tag{1}$$

を解くことに帰着される [7].

ベクトル計算機で解く場合，式 (1) のような複数の方程式を同時に解くには，反復法を用いると収束判定が難しいため，直接法である LU 分解法（ガウス消去法）が用いられることが多い [6]．この時，行列群 $\hat{A}_{j,k}$ を LU 分解し，j または k 方向をベクトル化して複数の方程式を同時に解くことで計算の効率化を図ることができる．しかし，計算機の性能を十分に発揮させる観点から，従来の LU 分解法には改善の余地がある．

本研究では，従来の LU 分解法の計算過程に着目し，行列群 $\hat{A}_{j,k}$ に対し対角方向の両端から LU 分解を同時に行うアイデアに基づいた，ベクトル計算機での計算効率を向上させる解法を提案する．また，従来の LU 分解法との性能比較実験の結果により，本解法の優れた計算効率を示す．

本稿の内容は次の通りである．第2章で，係数行列 $\hat{F}_{j,k}$ がブロック5重対角行列群である連立一次方程式に対し，ベクトル計算機での計算効率を向上させる解法を提案する．第3章では，係数行列 $\hat{G}_{j,k}$ が周期境界要素（偏微分方程式の離散化で周期境界条件から生ずる行列要素）も持つブロック5重対角行列群である連立一次方程式に対して，第2章で提案する解法を適用するための効率的な前処理方法を提案する．

2 係数行列がブロック5重対角行列群のとき

このとき，係数行列 $\hat{F}_{j,k}$ は以下のとおりである．

$$\hat{F}_{j,k} = \begin{bmatrix} C^1_{j,k} & D^1_{j,k} & E^1_{j,k} & & & & 0 \\ B^2_{j,k} & C^2_{j,k} & D^2_{j,k} & E^2_{j,k} & & & \\ A^3_{j,k} & B^3_{j,k} & C^3_{j,k} & D^3_{j,k} & E^3_{j,k} & & \\ & \cdots & \cdots & \cdots & \cdots & \cdots & \\ & & A^{l-2}_{j,k} & B^{l-2}_{j,k} & C^{l-2}_{j,k} & D^{l-2}_{j,k} & E^{l-2}_{j,k} \\ & & & A^{l-1}_{j,k} & B^{l-1}_{j,k} & C^{l-1}_{j,k} & D^{l-1}_{j,k} \\ 0 & & & & A^{l}_{j,k} & B^{l}_{j,k} & C^{l}_{j,k} \end{bmatrix} \tag{2}$$

式 (2) において，$A^i_{j,k}, B^i_{j,k}, C^i_{j,k}, D^i_{j,k}, E^i_{j,k} (i = 1,...,l; j = 1,...,m; k = 1,...,n)$ は，5×5 ブロック小行列である．式 (2) の第 i 行は，4次精度 AF 法を用いた時の差分情報を表わす．

2.1 従来の LU 分解法

従来の LU 分解法では，$\hat{F}_{j,k}$ は式 (3) のように分解される．

$$\hat{F}_{j,k} = \hat{L}_{j,k}\hat{U}_{j,k} = \begin{bmatrix} C^1_{j,k} & & & & & & 0 \\ B^2_{j,k} & \tilde{C}^2_{j,k} & & & & & \\ A^3_{j,k} & \tilde{B}^3_{j,k} & \tilde{C}^3_{j,k} & & & & \\ & \cdots & \cdots & \cdots & & & \\ & & A^{l-2}_{j,k} & \tilde{B}^{l-2}_{j,k} & \tilde{C}^{l-2}_{j,k} & & \\ & & & A^{l-1}_{j,k} & \tilde{B}^{l-1}_{j,k} & \tilde{C}^{l-1}_{j,k} & \\ 0 & & & & A^{l}_{j,k} & \tilde{B}^{l}_{j,k} & \tilde{C}^{l}_{j,k} \end{bmatrix} \times \begin{bmatrix} I & \tilde{D}^1_{j,k} & \tilde{E}^1_{j,k} & & & & 0 \\ & I & \tilde{D}^2_{j,k} & \tilde{E}^2_{j,k} & & & \\ & & I & \tilde{D}^3_{j,k} & \tilde{E}^3_{j,k} & & \\ & & & \cdots & \cdots & \cdots & \\ & & & & I & \tilde{D}^{l-2}_{j,k} & \tilde{E}^{l-2}_{j,k} \\ & & & & & I & \tilde{D}^{l-1}_{j,k} \\ 0 & & & & & & I \end{bmatrix} \tag{3}$$

ここで，Iは単位行列を表し，$\hat{L}_{j,k}$は前進消去の行列群，$\hat{U}_{j,k}$は後退代入の行列群をそれぞれ表す．ベクトル計算機を用いる場合，ベクトル長（同時にLU分解できる要素数）は，j方向をベクトル化するならばm，k方向をベクトル化するならばnである．式(3)の上波線付きブロック小行列は，分解により更新されたブロック小行列を表す．

すなわち，このようなブロック5重対角行列群を上下ブロック3重対角行列群の積の形に分解し，前進消去・後退代入を行い解を求める[6].

前進消去のプログラミングを略記したのがプログラム1[1]である．数学上の演算イメージを表現するために，*do*ループ内では式(2)以後用いられているブロック小行列の演算を記述した．ここでは，j方向をベクトル化している．

```
   parameter (ix=l, iy=m, iz=n)
   dimension (A(iy,iz,ix,5,5), B(iy,iz,ix,5,5),
  c    C(iy,iz,ix,5,5), D(iy,iz,ix,5,5), E(iy,iz,ix,5,5))

   do 1 i=1,ix
   do 1 k=1,iz
   do 1 j=1,iy
      B̃^i_{j,k} = B^i_{j,k} − A^i_{j,k} × D̃^{i-2}_{j,k}
      C̃^i_{j,k} = C^i_{j,k} − A^i_{j,k} × Ẽ^{i-2}_{j,k} − B̃^i_{j,k} × D̃^{i-1}_{j,k}
      D̃^i_{j,k} = (C̃^i_{j,k})^{-1} × (D^i_{j,k} − B̃^i_{j,k} × Ẽ^{i-1}_{j,k})
      Ẽ^i_{j,k} = (C̃^i_{j,k})^{-1} × E^i_{j,k}
      b̃^i_{j,k} = (C̃^i_{j,k})^{-1}
            ×(b^i_{j,k} − A^i_{j,k} × b̃^{i-2}_{j,k} − B^i_{j,k} × b̃^{i-1}_{j,k})
1  continue
```

プログラム1　従来のLU分解法

2.2 Rotated Alternative LU 分解法

一方で，行列群の対角方向の両端から中心の要素まで同時に分解し，そこで折り返して後退代入することができる[8]．すなわち，$\hat{F}_{j,k}$は式(4)のように分解でき，$\hat{P}_{j,k}$は，前進消去の行列群，$\hat{Q}_{j,k}$は後退代入の行列群をそれぞれ表す．ここでは，lが奇数のときの分解を表わしている．lが偶数のときは，$i=1,...,\frac{l}{2}$，$i=(\frac{l}{2}+1),...,l$で分解される．式(3)と同様，上波線付きブロック小行列は，分解により更新されたブロック小行列を表す．基本的な原理は，iの偶奇によらないため，以下，iが奇数の場合について説明する．

[1]実際のプログラミングでは，$i=1,2,(l-1),l$に対しては場合分けが生じる．また，ブロック小行列の演算は，話の本題からそれるため明記してないが，最外ループとして演算を行う．

$$\hat{F}_{j,k}=\hat{P}_{j,k}\hat{Q}_{j,k}=$$

$$\begin{bmatrix}
C^1_{j,k} & & & & & & & 0 \\
B^2_{j,k} & \tilde{C}^2_{j,k} & & & & & & \\
A^3_{j,k} & \tilde{B}^3_{j,k} & \tilde{C}^3_{j,k} & & & & & \\
 & \cdots & \cdots & \cdots & & & & \\
 & A^{\frac{l+1}{2}}_{j,k} & \tilde{B}^{\frac{l+1}{2}}_{j,k} & \tilde{C}^{\frac{l+1}{2}}_{j,k} & \tilde{D}^{\frac{l+1}{2}}_{j,k} & E^{\frac{l+1}{2}}_{j,k} & & \\
 & & & \cdots & \cdots & \cdots & & \\
 & & & & \tilde{C}^{l-2}_{j,k} & \tilde{D}^{l-2}_{j,k} & E^{l-2}_{j,k} & \\
 & & & & & \tilde{C}^{l-1}_{j,k} & D^{l-1}_{j,k} & \\
0 & & & & & & C^l_{j,k}
\end{bmatrix}$$

$$\times\begin{bmatrix}
I & \tilde{D}^1_{j,k} & \tilde{E}^1_{j,k} & & & & & 0 \\
 & I & \tilde{D}^2_{j,k} & \tilde{E}^2_{j,k} & & & & \\
 & & I & \tilde{D}^3_{j,k} & \tilde{E}^3_{j,k} & & & \\
 & & & \cdots & \cdots & \cdots & & \\
 & 0 & 0 & I & 0 & 0 & & \\
 & \cdots & \cdots & \cdots & & & & \\
 & & \tilde{A}^{l-2}_{j,k} & \tilde{B}^{l-2}_{j,k} & I & & & \\
 & & & \tilde{A}^{l-1}_{j,k} & \tilde{B}^{l-1}_{j,k} & I & & \\
0 & & & & \tilde{A}^l_{j,k} & \tilde{B}^l_{j,k} & I &
\end{bmatrix} \quad (4)$$

この分解法では，分解過程から名付けられた，"Alternative decomposition(双方向分解)[8]，別名：Twisted factorization(ツイスト分解)[9]"が行われる．双方向分解は，単一の連立方程式に対してCG法など反復解法の前処理の並列化に使用されている[8] [9].

本研究では，この分解法を複数の連立一次方程式を解くための直接法に応用し，ベクトル計算機の計算効率を改善するようなアルゴリズムへと更に改良することを考える．

一般に，ベクトル計算機を用いて問題を解く場合，ベクトル長を十分に長くすると計算機性能を引き出すことができる[2]．計算機性能が飽和するに至らない程度のベクトル長においては，その長さを2倍にすることで計算効率の向上が期待できる．そこで本研究では，このベクトル計算機の一般的なアーキテクチャ特性を念頭に，式(4)の双方向分解の$\hat{P}_{j,k}\hat{Q}_{j,k}$に対して，中央から上下に二分割し，下半分の要素を180度回転してベクトル化方向の要素へ組み込むことでベクトル長を倍増させる．本研究の中では，この解法をRotated Alternative LU分解法（以下，Rotated ALU分解法と略記）と呼ぶことにする．詳細は以下の通りである．

図1は，$\hat{P}_{1,1}$に対する双方向分解(左)とRotated ALU分解(右)をイメージで表現したものである．

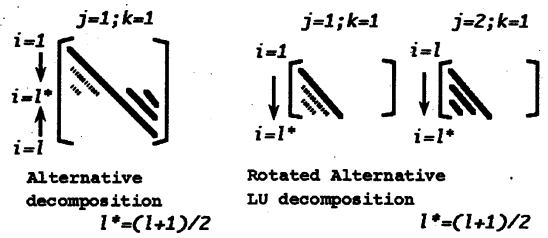

図 1: 双方向分解 (左) と Rotated ALU 分解 (右) のイメージ

図中の実線，点線は左右の図で各々対応しており，矢印は LU 分解の順序を表している．Rotated ALU 分解では，$\hat{P}_{1,1}$ を上下に分割し[2]，下半分要素を図 1の通り 180 度回転して $j = 1$，$j = 2$ とする．$\hat{Q}_{1,1}$ についても同様である．

行列群 $\hat{P}_{j,k}$ （ここで，$j = 1, ..., m; k = 1, ..., n$) に対する Rotated ALU 分解では，上半分要素だけを並べて $j = 1, ..., m$ を形成し，下半分要素を 180 度回転して $j = (m+1), ..., 2m$ を形成し，j 方向をベクトル化する．これを $k = 1, ..., n$ について行う．行列群 $\hat{Q}_{j,k}$ についても同様である．

プログラム 1 と同様に前進消去の様子を次に示す．

```
parameter (ix=(l+1)/2, iy=2m, iz=n)
dimension (A(iy,iz,ix,5,5), B(iy,iz,ix,5,5),
c    C(iy,iz,ix,5,5), D(iy,iz,ix,5,5), E(iy,iz,ix,5,5))

do 2 i=1,ix
do 2 k=1,iz
do 2 j=1,iy
  B~(i)(j,k) = B(i)(j,k) - A(i)(j,k) × D~(i-2)(j,k)
  C~(i)(j,k) = C(i)(j,k) - A(i)(j,k) × E~(i-2)(j,k) - B~(i)(j,k) × D~(i-1)(j,k)
  D~(i)(j,k) = (C~(i)(j,k))^-1 × (D(i)(j,k) - B~(i)(j,k) × E~(i-1)(j,k))
  E~(i)(j,k) = (C~(i)(j,k))^-1 × E(i)(j,k)
  b~(i)(j,k) = (C~(i)(j,k))^-1
        ×(b(i)(j,k) - A(i)(j,k) × b~(i-2)(j,k) - B(i)(j,k) × b~(i-1)(j,k))
2  continue
```

プログラム 2 Rotated ALU 分解法

2.3 数値実験 1

ここでは，従来の LU 分解法と Rotated ALU 分解法とを比較した結果を示す．実験では，流体問題で扱うサイズに基づいた，様々なベクトル長に対する計算時間を測定した．

[2]プログラミングでは，テクニックとして，l の偶奇や LU 分解でのデータ参照に応じて二分割後の i 要素ににダミー領域を設ける．

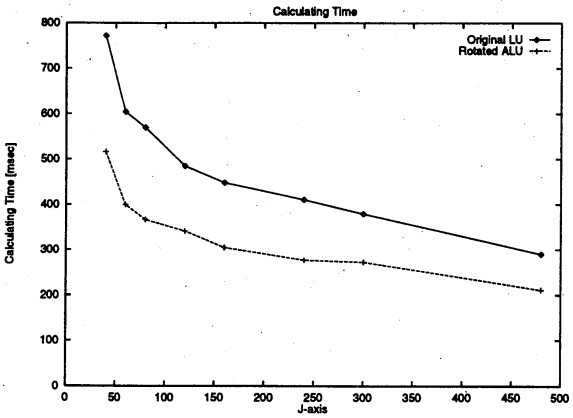

図 2: 従来の LU 分解法と Rotated ALU 分解法との計算時間の比較

2.3.1 実験内容

係数行列 $\hat{F}_{j,k}$ を構成するブロック小行列は，Frank 行列を基に LU 分解できるよう作成した．真の解 $\boldsymbol{x}_{j,k}$ は，乱数を用いて作成した．右辺項 $\boldsymbol{b}_{j,k}$ は，$\hat{F}_{j,k}$，$\boldsymbol{x}_{j,k}$ と式 (1) を用いて作成した．また，様々なベクトル長に対しても，全要素数 $(l \times m \times n)$ は一定となるように，LU 分解する i 方向の要素数は固定し ($l = 63$)，それ以外の方向は $m \times n = 2400$ (一定) となるように m，n を変化させた (表 1)．本実験では，j 方向をベクトル化した．

時間計測は，従来の LU 分解と Rotated ALU 分解による連立一次方程式の求解部分に対して実施した．計算機は，富士通 VPP500 の 1PE を用いた．

2.3.2 実験結果

計算時間を，図 2に示す．横軸は，j 方向の要素数を表し，縦軸は，計算時間 [msec] を表しており，表 1のサイズにおける計算時間をプロットした．この時間を基に計算時間比をグラフにしたのが，図 3である．

表 1: 実験モデル

l	63	63	63	63	63	63	63	63
m	40	60	80	120	160	240	300	480
n	60	40	30	20	15	10	8	5

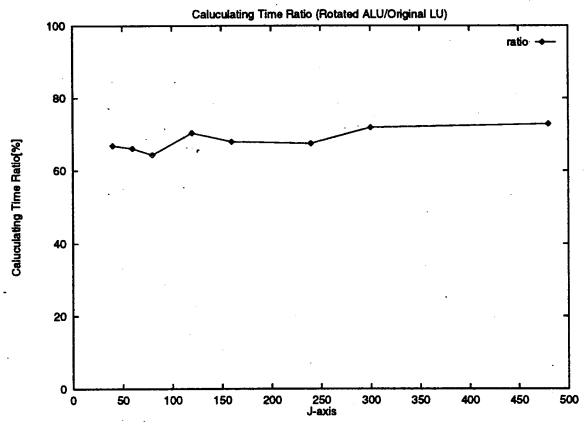

図 3: 従来の LU 分解法と Rotated ALU 分解法との計算時間比

これらから，従来の LU 分解法に比べて Rotated ALU 分解法では，最高 30% 程度まで計算時間が短縮したことがわかる．

解の精度に関して，単精度で計算を行ったところ，表 1の全モデルに対して，両アルゴリズムの誤差のオーダーは式 (5) に示す通りであり，このテスト問題において同等の精度であることが分かった．

$$\frac{\Sigma|\, \boldsymbol{x}_{j,k} - \bar{\boldsymbol{x}}_{j,k} \,|^2}{\Sigma|\, \boldsymbol{x}_{j,k} \,|^2} \sim O(10^{-6}) \qquad (5)$$

$\boldsymbol{x}_{j,k}$: 真の解, $\bar{\boldsymbol{x}}_{j,k}$: 数値解

3 係数行列に周期境界要素を持つとき

ここで，係数行列 $\hat{G}_{j,k}$ は式 (6) のとおりである．式 (6) において周期境界要素は，$A^1_{j,k}$, $B^1_{j,k}$, $A^2_{j,k}$, $E^{l-1}_{j,k}$, $D^l_{j,k}$, $E^l_{j,k}$ である．

$\hat{G}_{j,k} =$

$$\begin{bmatrix}
C^1_{j,k} & D^1_{j,k} & E^1_{j,k} & & & & A^1_{j,k} & B^1_{j,k} \\
B^2_{j,k} & C^2_{j,k} & D^2_{j,k} & E^2_{j,k} & & & & A^2_{j,k} \\
A^3_{j,k} & B^3_{j,k} & C^3_{j,k} & D^3_{j,k} & E^3_{j,k} & & 0 & \\
\cdots & \cdots & \cdots & \cdots & \cdots & & & \\
 & \cdots & \cdots & \cdots & \cdots & \cdots & & \\
 & & \cdots & \cdots & \cdots & \cdots & \cdots & \\
 & 0 & A^{l-2}_{j,k} & B^{l-2}_{j,k} & C^{l-2}_{j,k} & D^{l-2}_{j,k} & E^{l-2}_{j,k} & \\
E^{l-1}_{j,k} & & & A^{l-1}_{j,k} & B^{l-1}_{j,k} & C^{l-1}_{j,k} & D^{l-1}_{j,k} & \\
D^l_{j,k} & E^l_{j,k} & & & & A^l_{j,k} & B^l_{j,k} & C^l_{j,k}
\end{bmatrix} \qquad (6)$$

3.1 Rotated ALU 分解法の適用上の問題点

式 (1) において係数行列 $\hat{G}_{j,k}$ が式 (6) で表されるとき，従来の LU 分解法を用いると次式 (7) のように分解される．このとき，行列群 $\hat{G}_{j,k}$ の周期境界要素により，式 (7) 中の $\star$ 印で示した fill-in を生じる [1]. fill-in とは，分解前の係数行列ではゼロであった要素が，厳密な分解により非ゼロになることである．

$\hat{G}_{j,k} =$

$$\begin{bmatrix}
C^1_{j,k} & & & & & & & \\
B^2_{j,k} & \tilde{C}^2_{j,k} & & & & & & \\
A^3_{j,k} & \tilde{B}^3_{j,k} & \tilde{C}^3_{j,k} & & & 0 & & \\
 & \cdots & \cdots & \cdots & & & & \\
 & & \cdots & \cdots & \cdots & & & \\
 & & & \cdots & \cdots & \cdots & & \\
 & 0 & & & A^{l-2}_{j,k} & \tilde{B}^{l-2}_{j,k} & \tilde{C}^{l-2}_{j,k} & \\
\tilde{E}^{l-1}_{j,k} & \star & \star & \cdots & \star & A^{l-1}_{j,k} & \tilde{B}^{l-1}_{j,k} & \tilde{C}^{l-1}_{j,k} \\
D^l_{j,k} & \tilde{E}^l_{j,k} & \star & \cdots & \star & \star & A^l_{j,k} & \tilde{B}^l_{j,k} & \tilde{C}^l_{j,k}
\end{bmatrix}$$

$$\times \begin{bmatrix}
I & \tilde{D}^1_{j,k} & \tilde{E}^1_{j,k} & & & 0 & & \tilde{A}^1_{j,k} & \tilde{B}^1_{j,k} \\
 & I & \tilde{D}^2_{j,k} & \tilde{E}^2_{j,k} & & & & \star & \tilde{A}^2_{j,k} \\
 & & I & \tilde{D}^3_{j,k} & \tilde{E}^3_{j,k} & & & \star & \star \\
 & & & \cdots & \cdots & \cdots & & \vdots & \vdots \\
 & & & & \cdots & \cdots & & \star & \star \\
 & & & & & \cdots & & \star & \star \\
 & & & & & & I & \tilde{D}^{l-2}_{j,k} & \tilde{E}^{l-2}_{j,k} \\
 & & 0 & & & & & I & \tilde{D}^{l-1}_{j,k} \\
 & & & & & & & & I
\end{bmatrix} \qquad (7)$$

一方，求解時間を短縮する観点から，従来の LU 分解法の代わりに前章で提案した Rotated ALU 分解法を適用すると有効である．この分解法は，Twisted factorization （ツイスト分解） [9] を基に，ベクトル計算機の性能を引き出すように改良を加えた解法であり，ツイスト分解が不可能な問題に対しては，Rotated ALU 分解法の適用も不可能である．

ところが，$\hat{G}_{j,k}$ を係数行列とする問題に対してツイスト分解法を適用するとき，周期境界要素のために，前進消去の過程において式 (8) 中の $\star$ 印で示す fill-in を生じる．前進消去が対角方向の両端から始まり中央で完了した時，fill-in による要素のために，解くべき方程式の未知数の数が方程式数を上回る．このため，後退代入の段階では中央におい

て方程式は解けない．すなわち，式(8)の方法ではRotated ALU分解法を用いた求解は不可能である．

$\hat{G}_{j,k} =$

$$
\begin{bmatrix}
C^1_{j,k} & & & & & & & & \\
B^2_{j,k} & \tilde{C}^2_{j,k} & & & & & & & \\
A^3_{j,k} & \tilde{B}^3_{j,k} & \tilde{C}^3_{j,k} & & & & & & \\
\cdots & \cdots & \cdots & & & & & & \\
 & & A^{\frac{l+1}{2}}_{j,k} & \tilde{B}^{\frac{l+1}{2}}_{j,k} & \tilde{C}^{\frac{l+1}{2}}_{j,k} & \tilde{D}^{\frac{l+1}{2}}_{j,k} & E^{\frac{l+1}{2}}_{j,k} & & \\
 & & & & & \cdots & \cdots & \cdots & \\
 & & & & & & \tilde{C}^{l-2}_{j,k} & \tilde{D}^{l-2}_{j,k} & E^{l-2}_{j,k} \\
 & & 0 & & & & & \tilde{C}^{l-1}_{j,k} & \tilde{D}^{l-1}_{j,k} \\
 & & & & & & & & \tilde{C}^{l}_{j,k}
\end{bmatrix}
$$

$$
\times
\begin{bmatrix}
I & \tilde{D}^1_{j,k} & \tilde{E}^1_{j,k} & & & & 0 & & \tilde{A}^1_{j,k} & \tilde{B}^1_{j,k} \\
 & I & \tilde{D}^2_{j,k} & \tilde{E}^2_{j,k} & & & & & \star & \tilde{A}^2_{j,k} \\
 & & I & \tilde{D}^3_{j,k} & \tilde{E}^3_{j,k} & & & & \star & \star \\
 & & & \cdots & \cdots & \cdots & & & \star & \star \\
\star & \star & 0 & 0 & I & 0 & 0 & \star & \star & \\
\star & \star & & \cdots & \cdots & \cdots & & & & \\
\star & \star & & & \tilde{A}^{l-2}_{j,k} & \tilde{B}^{l-2}_{j,k} & I & & & \\
\tilde{E}^{l-1}_{j,k} & \star & & & & \tilde{A}^{l-1}_{j,k} & \tilde{B}^{l-1}_{j,k} & I & & \\
\tilde{D}^{l}_{j,k} & \tilde{E}^{l}_{j,k} & & 0 & & & \tilde{A}^{l}_{j,k} & \tilde{B}^{l}_{j,k} & I &
\end{bmatrix}
\tag{8}
$$

3.2 Sherman-Morrison-Woodbury公式の適用について

本研究では，Rotated ALU分解を用いる上でのこの問題点を解決するために，$\hat{G}_{j,k}$への前処理として Sherman-Morrison-Woodbury（SMW）公式 [3]

$$
\begin{aligned}
&(P+UV^t)^{-1} \\
&\quad = \quad P^{-1} - P^{-1}U(I+V^tP^{-1}U)^{-1}V^tP^{-1}
\end{aligned}
\tag{9}
$$

を利用する方法（Split/SMW）[10]を考える．

以下にその方法を示す．ここで，

$$
\begin{aligned}
\check{C}^1_{j,k} &= C^1_{j,k} - A^1_{j,k}, \\
\check{D}^1_{j,k} &= D^1_{j,k} - B^1_{j,k}, \\
\check{C}^2_{j,k} &= C^2_{j,k} - A^2_{j,k}, \\
\check{C}^{l-1}_{j,k} &= C^{l-1}_{j,k} - E^{l-1}_{j,k}, \\
\check{B}^{l}_{j,k} &= B^{l}_{j,k} - D^{l}_{j,k}, \\
\check{C}^{l}_{j,k} &= C^{l}_{j,k} - E^{l}_{j,k}
\end{aligned}
$$

と置き，ブロック5重対角行列群と周期境界要素とに分離して後者を2つのブロックベクトルの積として表すことにする．この時，係数行列$\hat{G}_{j,k}$は次のように変形される．

$\hat{G}_{j,k} =$

$$
\begin{bmatrix}
\check{C}^1_{j,k} & \check{D}^1_{j,k} & E^1_{j,k} & & & & & & & \\
B^2_{j,k} & \check{C}^2_{j,k} & D^2_{j,k} & E^2_{j,k} & & & & & & \\
A^3_{j,k} & B^3_{j,k} & C^3_{j,k} & D^3_{j,k} & E^3_{j,k} & & & & & \\
 & \cdots & \cdots & \cdots & \cdots & \cdots & & & & \\
 & & \cdots & \cdots & \cdots & \cdots & \cdots & & & \\
 & & & \cdots & \cdots & \cdots & \cdots & \cdots & & \\
 & & & & A^{l-2}_{j,k} & B^{l-2}_{j,k} & C^{l-2}_{j,k} & D^{l-2}_{j,k} & E^{l-2}_{j,k} & \\
 & 0 & & & & A^{l-1}_{j,k} & B^{l-1}_{j,k} & \check{C}^{l-1}_{j,k} & D^{l-1}_{j,k} & \\
 & & & & & & A^{l}_{j,k} & \check{B}^{l}_{j,k} & \check{C}^{l}_{j,k} &
\end{bmatrix}
$$

$$
+
\begin{bmatrix}
A^1_{j,k} & B^1_{j,k} & 0 & \cdots & 0 & A^1_{j,k} & B^1_{j,k} \\
0 & A^2_{j,k} & 0 & \cdots & 0 & 0 & A^2_{j,k} \\
0 & 0 & \cdots & \cdots & 0 & 0 & \\
\vdots & \vdots & & & & \vdots & \vdots \\
\vdots & \vdots & & & & \vdots & \vdots \\
\vdots & \vdots & & & & \vdots & \vdots \\
0 & 0 & \cdots & \cdots & 0 & 0 & \\
E^{l-1}_{j,k} & 0 & 0 & \cdots & 0 & E^{l-1}_{j,k} & 0 \\
D^{l}_{j,k} & E^{l}_{j,k} & 0 & \cdots & 0 & D^{l}_{j,k} & E^{l}_{j,k}
\end{bmatrix}
$$

$$
\begin{aligned}
&= \quad P + \begin{bmatrix}
A^1_{j,k} & B^1_{j,k} \\
0 & A^2_{j,k} \\
0 & 0 \\
\vdots & \vdots \\
\vdots & \vdots \\
\vdots & \vdots \\
0 & 0 \\
E^{l-1}_{j,k} & 0 \\
D^{l}_{j,k} & E^{l}_{j,k}
\end{bmatrix}
\begin{bmatrix}
I\,0\,0\cdots\cdots\cdots 0\,I\,0 \\
0\,I\,0\cdots\cdots\cdots 0\,0\,I
\end{bmatrix} \\
&= \quad P + [U_1\ U_2]\begin{bmatrix} V_1^t \\ V_2^t \end{bmatrix} \;=\; P + UV^t.
\end{aligned}
\tag{10}
$$

したがって，式(1)は，

$$
\hat{G}_{j,k}\boldsymbol{x}_{j,k} \;=\; (P+UV^t)\boldsymbol{x}_{j,k} = \boldsymbol{b}_{j,k}, \tag{11}
$$

$$
\boldsymbol{x}_{j,k} \;=\; (P+UV^t)^{-1}\boldsymbol{b}_{j,k} \tag{12}
$$

と表わされる．

式(12)に式(9)を適用すると，式(1)の解は，

$$
\begin{aligned}
\boldsymbol{x}_{j,k} &= \{P^{-1} - P^{-1}U(I + V^tP^{-1}U)^{-1}V^tP^{-1}\}\,\boldsymbol{b}_{j,k} \\
&= \left\{I - Z\left(I + V^tZ\right)^{-1}V^t\right\}P^{-1}\,\boldsymbol{b}_{j,k} \\
&= \left\{I - [Z_1 Z_2]\left(I + \begin{bmatrix} V_1^t \\ V_2^t \end{bmatrix}[Z_1 Z_2]\right)^{-1}\begin{bmatrix} V_1^t \\ V_2^t \end{bmatrix}\right\}P^{-1}\,\boldsymbol{b}_{j,k} \\
&= \left\{I - [Z_1 Z_2]\begin{bmatrix} W_1^t \\ W_2^t \end{bmatrix}\right\}\boldsymbol{y}_{j,k}
\end{aligned}
\tag{13}
$$

から求められる．ここで，

$$\boldsymbol{y}_{j,k} = P^{-1}\boldsymbol{b}_{j,k}, \tag{14}$$

$$Z = P^{-1}U, \tag{15}$$

$$Z = [Z_1 Z_2], \tag{16}$$

$$\begin{bmatrix} W_1^t \\ W_2^t \end{bmatrix} = \left(I + \begin{bmatrix} V_1^t \\ V_2^t \end{bmatrix}[Z_1 Z_2]\right)^{-1}\begin{bmatrix} V_1^t \\ V_2^t \end{bmatrix} \tag{17}$$

である．

以上から，式(13)の通り前処理を行ったことにより，式(1)はPを係数行列とする連立一次方程式(14)，(15)を解くことに帰着されたことがわかる[3]．また，Pは周期境界要素を持たないブロック5重対角行列群であるため，これらの式に対してRotated ALU分解法を適用することが可能となった[5]．本研究では，この解法をSplit/SMW+ Rotated ALU分解法と呼ぶ．

3.3 数値実験2

ここでは，式(6)を係数行列にもつ連立一次方程式に対して，従来のLU分解法とSplit/SMW+ Rotated ALU分解法とを適用した結果を示す．実験方法および使用計算機は，数値実験1と同様であるが，数値実験2に特有の箇所のみ説明する．

3.3.1 実験内容

係数行列$\hat{G}_{j,k}$を構成するブロック小行列は，Frank行列を基にしてLU分解できるよう，かつ，式(9)におけるPと$(I + V^tP^{-1}U)$が正則となるよう作成した．真の解$\boldsymbol{x}_{j,k}$は，乱数を用いて作成した．右辺項$\boldsymbol{b}_{j,k}$は，$\hat{G}_{j,k}$，$\boldsymbol{x}_{j,k}$と式(1)を用いて作成した．実験モデルは，表1の通りであり，j方向をベクトル化した．

時間計測は，式(7)に示した従来のLU分解法と，Split/SMW+ Rotated ALU分解法を用いた解法による連立一次方程式の求解部分に対して実施した．

[3]式(17)は2ブロック×2ブロックの連立一次方程式であり，ブロックLU分解を用いて計算する．

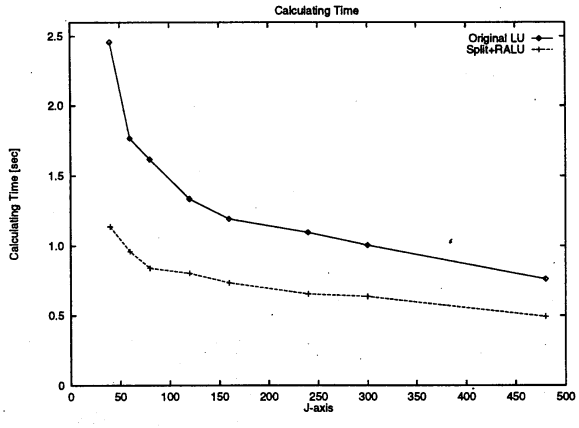

図4: 従来のLU分解法とSplit/SMW+ Rotated ALU分解法 との計算時間の比較

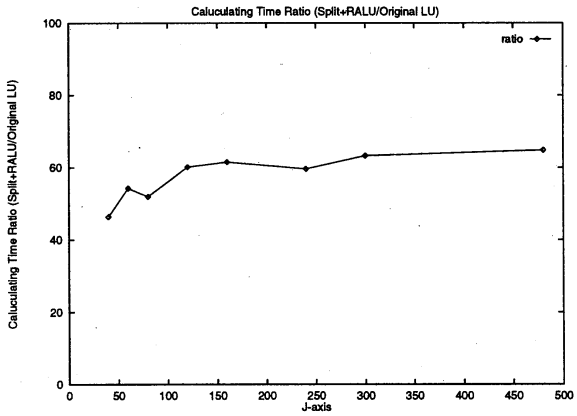

図5: 従来のLU分解法とSplit/SMW+ Rotated ALU分解法 との計算時間比

3.3.2 実験結果

計算時間を，図4に示す．横軸は，j方向の要素数を表し，縦軸は，計算時間[sec]を表しており，表1のサイズにおける計算時間をプロットした．この時間を基に計算時間比をグラフにしたのが，図5である．（図中では，Split/SMW+ Rotated ALU分解法をSplit+RALUと略記している．）

以上から，従来のLU分解法に比べてRotated ALU分解法では，35～40%程度まで計算時間が短縮したことが分かった．

解の精度に関して，単精度で計算を行ったところ，表1の全モデルに対して，両アルゴリズムの誤差のオーダーは式(18)に示すとおりであり，このテスト問題において同等の精度であることが分かった．

$$\frac{\Sigma|\, x_{j,k} - \bar{x}_{j,k}\,|^2}{\Sigma|\, x_{j,k}\,|^2} \sim O(10^{-6}). \quad (18)$$

$x_{j,k}$：真の解，$\bar{x}_{j,k}$：数値解

4 まとめ

本研究では，流体計算に現れるブロック5重対角行列群を係数行列とする大規模な複数の連立一次方程式に対して，ベクトル計算機を用いて効率的に求解するためのアルゴリズムを提案し，数値実験によりその効果を示した．

第2章では，Rotated ALU 分解法についてアルゴリズムの詳細を説明し，数値実験により従来の LU 分解法と比較を行った．その結果から，Rotated ALU 分解法は，ベクトル計算機の性能を更に引き出すアルゴリズムであり，計算精度も保持していることが確認された．流体の問題に限らず他の問題についても，同様な対角要素に対して対称構造を持つ行列群に対してベクトル計算機で解く場合は，Rotated ALU 分解法を用いることを提案する．

第3章では，係数行列が周期境界要素を伴う場合の大規模な複数の連立一次方程式を解く問題を扱った．この場合，Rotated ALU 分解法を適用するために，周期境界要素とブロック5重対角行列群とを分離し，前処理として SMW を用いる方法を提案した．また，数値実験により，従来の LU 分解のみの求解方法と比較して，Split/SMW+ Rotated ALU 分解法は計算時間を短縮することができ，計算精度も保持していることが数値例で確認された．

参考文献

[1] Anderson,D.A., Tannehill,J.C. and Pletcher, R.H.: Computataional Fluid Mechanics and Heat Transfer, Hemisphere Publishing Corporation(1984).

[2] Dongarra,J.J., Duff,I.S., Sorensen,D.C. and Van der Vorst,H.A.: Solving Linear Systems on Vector and Shared Memory Computers, SIAM(1991). 小国 力（訳）：コンピュータによる連立一次方程式の解法－ベクトル計算機と並列計算機－，丸善 (1993)

[3] Golub,G.H. and Van Loan,C.F.: Matrix Computations ,Johns Hopkins University Press (1996).

[4] 伊藤祥司，張 紹良，名取 亮：ブロック5重対角行列群に対する Rotated Alternative LU 分解法について，情報処理学会論文誌, Vol.38, No.11, pp.2402–2405 (1997).

[5] 伊藤祥司，張 紹良，名取 亮：周期境界要素を持つブロック5重対角行列群への Rotated Alternative LU 分解法の適用について，情報処理学会論文誌, Vol.39,No.1,pp.153–156(1998).

[6] Pulliam,T.H.: Euler and Thin Layer Navier-Stokes Codes:ARC2D,ARC3D, *Notes for Computational Fluid Dynamics User's Workshop*, pp. 15.1–15.84 (1984).

[7] 標 宣男，鈴木正昭，石黒美佐子，寺坂晴夫：数値流体力学，朝倉書店 (1994).

[8] Van der Vorst,H.A.: Analysis of a parallel solution method for tridiagonal linear systems,*Parallel Computing* , Vol.5,pp. 303–311(1987).

[9] Van der Vorst,H.A.: Large Tridiagonal and Block Tridiagonal Linear Systems on Vector and Parallel Computers, *Parallel Computing*,Vol.5, pp. 45–54 (1987).

[10] Yarrow,M.: Solving Periodic Block Tridiagonal Systems Using the Sherman-Morrison-Woodbury Formula,*AIAA 9th Computational Fluid Dynamics Conf.*, Jun.13-15, pp. 188–196 (1989).

A ベクトル計算機の特性について

本研究では，ベクトル計算機の一般的な特性に着目した解法を提案した．そのベクトル計算機の特性について説明する．

```
      dimension (A(n), B(n), C(n))
      do 3 i=1,n
        A(i) = A(i) + B(i) * C(i)
    3 continue
```

プログラム3 ベクトルどうしの演算

プログラム3を実行する場合，スカラー計算機では，*do* ループに対し $A(i)$，$B(i)$，$C(i)(i = 1, ..., n)$ の1要素ずつをメモリからレジスタへロードし，$i = 1$ から $i = n$ まで逐次的に演算する．

一方，ベクトル計算機では，do ループに対してデータ列 $A(i)$，$B(i)$，$C(i)(i = 1, ..., n)$ を，各配列の n 要素分を一度にメモリからベクトルレジスタへロードし，データ列のまま演算する．この時，配列の要素数 n をベクトル長と呼び，1つのベクトル命令でベクトル演算を実行する要素数を表す．ただし，要素数 n がベクトルレジスタの容量を超過する場合，このロードやベクトル演算は，数回（m 回とする）に分割されて繰り返される．この時，分割された後の要素数（n/m 回）をベクトル長と呼ぶ．

行列積など，配列どうしの演算のために多重ループとなっている場合は，最内ループがベクトル演算の対象となる．プログラム4では，$i-$ ループに対してベクトル演算されている．

```
      parameter (ix=l, iy=m, iz=n)
      dimension (A(ix,iy), B(ix,iz), C(iz,iy))
      do 4 j=1,iy
      do 4 k=1,iz
      do 4 i=1,ix
        A(i,j) = A(i,j) + B(i,k) * C(k,j)
    4 continue
```

プログラム4 行列積

プログラム4を取り上げ，ベクトル長に対する計算性能を測定した結果が，図6である．ここで，横軸の値がベクトル長である．使用した計算機は，本研究の数値実験で使用した，富士通 VPP500/1PE（ピーク性能 1600[MFlops/PE]）である．

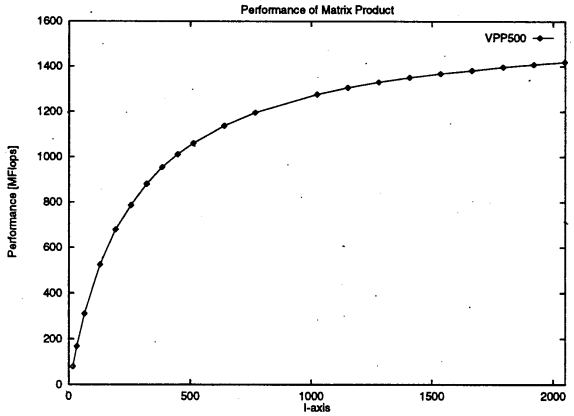

図 6: ベクトル長に対する計算性能の様子

この結果から，ベクトル長が長くなるにつれて計算効率が向上していることがわかる．