ハイパフォーマンス
コンピューティング　90－1
（２００２．５．２７）

シフト方程式に有効な前処理について

平野友貴† 野寺 隆††

現在，大型で疎な連立１次方程式を解く算法が数多く提案されている．しかし，時間依存型の偏微分方程式や量子色力学の格子ゲージ計算で現れるシフト方程式を解く算法はあまり提案されていない．本稿では，このシフト方程式を解く算法として，前処理を加えて近似解の収束性を向上させたGMRES法を提案する．また，前処理によって計算時間が大きく変わることもあるので，いくつかのシフト方程式を解く算法を並列計算機Origin 2400に実装し，数値実験を行ない，各算法を比較検討する．

Effective Preconditioning for Shifted Linear Systems

Yuki Hirano† and Takashi Nodera††

There are a lot of numerical algorithms for solving large sparse linear systems of equations. However, there are few algorithms for the solution of shifted linear systems which are generally used in time-dependent partial differential equations or lattice gauge computations in quantum chromodynamics. In this paper, we propose the GMRES procedure with preconditioning for the shifted linear systems. Since the computation time changes by the preconditioners, we compare several algorithms by the numerical experiments on the parallel machine Origin 2400 and carry out the comparison of numerical results.

1. はじめに

並列計算機の性能が向上したことで大規模な線形計算が可能になり，理工学の様々な分野において良いアルゴリズムの開発が行なわれている．その代表的な解法としてGMRES(m)法がある[1]．このGMRES(m)法は，大型で疎な非対称行列を係数にした連立１次方程式

$$A\boldsymbol{x}=\boldsymbol{b},\quad A\in\boldsymbol{C}^{n\times n}\quad \boldsymbol{x},\boldsymbol{b}\in\boldsymbol{C}^{n} \qquad (1)$$

を解くための反復法の１つである．ただし，式(1)の係数行列Aは正則とする．

本稿では上の式(1)とは他に，以下のような特殊な方程式を扱うことにする．

$$A'\boldsymbol{x}'=\boldsymbol{b},\quad A'\in\boldsymbol{C}^{n\times n}\quad \boldsymbol{x}',\boldsymbol{b}\in\boldsymbol{C}^{n} \qquad (2)$$

ただし，$A'=A+cI(c\in\boldsymbol{C},I:n\times n$の単位行列)とする．この方程式は式(1)の係数行列の対角項が異なる係数行列をもつ方程式であり，シフト方程式(shifted linear systems)と呼ばれる[2]．また，この方程式は主に時間依存の偏微分方程式[3]や量子色力学(QCD)の格子ゲージ計算[4]，制御理論[5]などで必要となる場合がある．

これらの方程式を解くようなアルゴリズムとして，すでにGMRES(m)法を用いたアルゴリズムがFrommerら[2]によって提案されている．その算法がShifted-GMRES(m)法である．だが，この算法には前処理行列が使われておらず，改善の余地が残されている．本稿では，Shifted-GMRES(m)法の代替法として，前処理行列を適用して，収束性を向上させるようなGMRES(m)法を提案する．前処理行列としては，特にシフト方程式の係数行列に対して有効な近似逆行列を用いる．その算法をConcurrent-GMRES(m)法と呼ぶことにする．

第2節では，従来のGMRES(m)法について簡単に述べる．第3節では，Shifted-GMRES(m)法について記述する．第4節では，Concurrent-GMRES(m)法について述べ，その算法で用いる2種類の前処理を紹介する．第5節では，Concurrent-GMRES(m)法とShifted-GMRES(m)法を比較した数値実験の結果を示す．最後に第6節において，結論を述べることにする．

2. GMRES(m)法

GMRES法は，1986年にSaadら[1]によって提案された，クリロフ部分空間法の１つである．この解法は，初期近似解$\boldsymbol{x}_0$で，初期残差ベクトル$\boldsymbol{r}_0=\boldsymbol{b}-A\boldsymbol{x}_0$と

† 慶應義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University
†† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

したとき，i番目の反復における近似解は$\boldsymbol{x}_i = \boldsymbol{x}_0 + \boldsymbol{z}_i$となる．この$\boldsymbol{z}_i$は，以下の式で決定することになる．

$$\min_{\boldsymbol{z}\in K_i} \|\boldsymbol{b} - A(\boldsymbol{x}_0 + \boldsymbol{z})\|_2 = \min_{\boldsymbol{z}\in K_i} \|\boldsymbol{r}_0 - A\boldsymbol{z}\|_2$$

また，クリロフ部分空間

$K_i(A, \boldsymbol{r}_0) = \mathrm{span}\{\boldsymbol{r}_0, A\boldsymbol{r}_0, A^2\boldsymbol{r}_0, \ldots, A^{i-1}\boldsymbol{r}_0\}$

の正規直交ベクトルを各列にもつ基底行列 $V_i = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_i)$ により，$\boldsymbol{z} = V_i\boldsymbol{y}$と記述できる．よって，$i$番目の近似解$\boldsymbol{x}_i$は，$\boldsymbol{x}_i = \boldsymbol{x}_0 + V_i\boldsymbol{y}$で求められることになる．

この$\boldsymbol{y}$は，1 つめの正規直交ベクトルを$\boldsymbol{v}_1 = \boldsymbol{r}_0/\|\boldsymbol{r}_0\|_2$として以下の最小 2 乗問題に帰着できる．

$$\begin{aligned}\min_{\boldsymbol{z}\in K_i} \|\boldsymbol{r}_0 - A\boldsymbol{z}\|_2 &= \min_{\boldsymbol{y}} \|\gamma\boldsymbol{v}_1 - AV_i\boldsymbol{y}\|_2 \\ &= \min_{\boldsymbol{y}} \|V_{i+1}(\gamma\boldsymbol{e}_1 - \bar{H}_i\boldsymbol{y})\|_2 \\ &= \min_{\boldsymbol{y}} \|\gamma\boldsymbol{e}_1 - \bar{H}_i\boldsymbol{y}\|_2 \quad (3)\end{aligned}$$

ただし，$\gamma = \|\boldsymbol{r}_0\|_2$，$\boldsymbol{e}_1 = (1, 0, \ldots, 0)^T$である．また，$\bar{H}_i$はアーノルディ過程で得られる上ヘッセンベルグ行列である．式 (3) は，通常$\bar{H}_i$をギブンス回転行列で QR 分解することによって解くことができる．

通常，GMRES 法は記憶容量と計算量の軽減のため，リスタートを用いることが多い．これはもし反復回数 i がリスタート周期 m を越えるときに，新しい初期近似解$\boldsymbol{x}_0$を m 回反復したときの近似解$\boldsymbol{x}_m$で置き換えて残差ベクトルを再計算し，これを m 回毎に繰り返す方法である．このリスタート版の GMRES 法を GMRES(m) 法と呼び，この算法を基にして次節以降の算法を議論する．

3. Shifted-GMRES(m) 法

この算法は Frommer ら[2]がシフト方程式に対しても利用できるように GMRES(m) 法を改良した方法である．ただし，以下のような制約がかかる．

$$\boldsymbol{r}_0' = \beta_0\boldsymbol{r}_0, \quad \beta_0 \in \boldsymbol{C} \qquad (4)$$

この条件は共線形条件 (colinear condition) と呼ばれるもので，式 (1) の方程式の初期残差$\boldsymbol{r}_0$と式 (2) のシフト方程式の初期残差$\boldsymbol{r}_0'$を強制的に関連づけることになる．この条件によって，GMRES(m) 法で用いる 2 つの方程式のクリロフ部分空間が一致することになり，アルゴリズム中のアーノルディ過程で求める正規直交基底行列が同じものになる．それに伴い，行列とベクトルの積の回数を減らし，計算コストの削減になる．

この共線形条件のために，前処理行列を用いることができないが，その代わりにシュールコンプルメント (Schur complement) を用いたブロック処理によって計算を短縮している．例えば，RB オーダリング (Red-Black ordering) で式 (1) の方程式を以下のようにブロック分割する．

```
choose x_0, x_0'
r_0 := b - A x_0;
r_0' := b - A' x_0';   β_0 := r_0'/r_0;
γ := ||r_0||_2;   v_1 := r_0/γ;
start
for n := 1 to m do
begin
  w := A v_n;
  for i := 1 to n do
  begin
    h_{i,n} := w^T v_i;
    w := w - h_{i,n} v_i;
  end
  h_{n+1,n} := ||w||_2;
  v_{n+1} := w/h_{n+1,n};
  compute y_n = min_y ||γ e_1 - H̄_n y||;
end
x_m := x_0 + V_m y_m;
z_{m+1} := γ e_1 - H̄_m y_m;   b' := β_0 γ e_1;
H̄'_m := H̄_m + c ( I_m ; 0 );
( y'_m ; β_m ) := ( H̄'_m, z_{m+1} )^{-1} b';
x'_m := x'_0 + V_m y'_m;
if ||b - A x_m|| ≤ ε then
  stop iteration
endif
x_0 := x_m;   r_0 := b - A x_0;
γ := ||r_0||_2;   v_1 := r_0/γ;
x'_0 := x'_m;   β_0 := β_m;
goto start
```

図 1　Shifted-GMRES(m) 法

$$\begin{pmatrix} A_{rr} & A_{rb} \\ A_{br} & A_{bb} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_r \\ \boldsymbol{x}_b \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_r \\ \boldsymbol{b}_b \end{pmatrix} \qquad (5)$$

ただし，RB オーダリングを用いた方程式の特徴として，2 つの小行列 A_{rr} と A_{bb}は単位行列の定数倍，つまりαIとなる．もし，シフトを行なった時でもこれらの行列が $(\alpha + c)I$になり，単位行列の定数倍であることに変わらない．式 (5) においてこれらの小行列を入れ換えたブロック形式の方程式を次のような 2 つの式に置き換える．

$$(\alpha^2 I - A_{br}A_{rb})\boldsymbol{x}_b = \alpha\boldsymbol{b}_b + A_{br}\boldsymbol{b}_r \qquad (6)$$

$$\boldsymbol{x}_r = \frac{1}{\alpha}(\boldsymbol{b}_r + A_{rb}\boldsymbol{x}_b) \qquad (7)$$

式 (6) は左辺の係数行列 $(\alpha^2 I - A_{br}A_{rb})$ の次元がほぼ半分に下がっているので，計算時間が短縮できることになる．そして，式 (6) の解$\boldsymbol{x}_b$によって，順に式 (7) の解$\boldsymbol{x}_r$が計算でき，最終的な解$\boldsymbol{x} = (\boldsymbol{x}_r, \boldsymbol{x}_b)^T$が求まる．この算法のアルゴリズムを図 1に示す．

4. Concurrent-GMRES(m) 法

本節では，Shifted-GMRES(m) 法に代わる算法と

各プロセッサを求めたいシフト方程式に
均等に割り当て, 以下の計算を行なう.
choose x_0, c
$r_0 := b - (A + cI)x_0$;
$\gamma := \|r_0\|_2$;　$v_1 := r_0/\gamma$;
start
for $n := 1$ **to** m **do**
begin
　$w := (A + cI)M_c v_n$;
　for $i := 1$ **to** n **do**
　begin
　　$h_{i,n} := w^T v_i$;
　　$w := w - h_{i,n} v_i$;
　end
　$h_{n+1,n} := \|w\|_2$;
　$v_{n+1} := w/h_{n+1,n}$;
　compute $y_n = \min_y \|\gamma e_1 - \bar{H}_n y\|$;
end
$x_m := x_0 + M_c V_m y_m$;
if $\|b - (A + cI)x_m\| \leq \varepsilon$ **then**
　stop iteration
endif
$x_0 := x_m$;　$r_0 := b - (A + cI)x_0$;
$\gamma := \|r_0\|_2$;　$v_1 := r_0/\gamma$;
goto start
最後に同期をとり, 各シフト方程式の
解 x を表示する.

図 2　Concurrent-GMRES(m) 法

して，シフト方程式に対して効果的な近似逆行列を前処理行列とする GMRES(m) 法を提案する．

ここでの前処理行列というのは，前処理を用いて式 (1) の連立 1 方程式を次のような式に置き換えたときの行列 Mのことである．ここでは右前処理を用いることにすると

$$AM\boldsymbol{y} = \boldsymbol{b}, \quad M\boldsymbol{y} = \boldsymbol{x} \tag{8}$$

というような連立 1 次方程式になる．この連立 1 次方程式は左の式の解$\boldsymbol{y}$を求めた後，右の式によって最終的な解$\boldsymbol{x}$を求めることになる．この前処理によって，1 回の反復における計算量は増えることになるが，収束までの反復回数を減らすことができる場合がある．そして，この前処理をシフト方程式にも適用しようと試みるのが，この Concurrent-GMRES(m) 法である．

また，Concurrent-GMRES(m) 法における原理は次のような並列化である．式 (1) と式 (2) の方程式は互いに干渉せずに計算可能であるので，これらの 2 つの式は並列化可能である．並列計算機のようなプロセッサ数がいくつかある環境ならば，シフト方程式を含めたそれぞれの方程式を各プロセッサに割り当てることができ，各方程式を並列実装した GMRES(m) 法で解くことができる．そのときの計算時間は最も時間のかかる方程式に依存することになるが，シフト方程式の解も効率良く求めることができ，効果的な前処理行列を用いることによって計算コストの削減につながる．この算法のアルゴリズムを図 2に示す.

4.1　代数的マルチレベル前処理

シフト方程式の前処理行列の実装に関して，まず 1 つめの前処理として，代数的マルチレベル前処理 (Algebraic Multilevel preconditioning)[6] について述べる.

式 (5) の方程式のようにブロック分割したものを用いると，係数行列 A に対する逆行列は以下で与えられる.

$$A^{-1} = \begin{pmatrix} I & -A_{rr}^{-1}A_{rb} \\ 0 & I \end{pmatrix} \begin{pmatrix} A_{rr}^{-1} & 0 \\ 0 & S_A^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -A_{br}A_{rr}^{-1} & I \end{pmatrix} \tag{9}$$

ただし，S_A^{-1}は $S_A = A_{bb} - A_{br}A_{rr}^{-1}A_{rb}$ の逆行列である．一方，A_{rr}^{-1}は，$A_{rr} = \alpha I$より，$A_{rr}^{-1} = (1/\alpha)I$ となる．そこで，S_A^{-1}に対しては便宜的に近似逆行列 $\tilde{S}_A^{-1}$を考える．この近似逆行列を用いて，式 (9) を置き換えると

$$M = \begin{pmatrix} I & -\frac{1}{\alpha}A_{rb} \\ 0 & I \end{pmatrix} \begin{pmatrix} \frac{1}{\alpha}I & 0 \\ 0 & \tilde{S}_A^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ -\frac{1}{\alpha}A_{br} & I \end{pmatrix} \tag{10}$$

となり，これが前処理行列 Mとなる．このような前処理行列を用いれば，係数行列 A の対角要素αにシフト値 c を加えるだけで各シフト方程式に対するそれぞれの前処理行列が求まり，簡単な実装で効果的な方法になる.

4.2　シュールコンプルメント前処理

次に，もう 1 つのシフト方程式に効果的な前処理としてシュールコンプルメント前処理について述べる.

式 (8) における係数行列 A と右辺項 b は，Shifted-GMRES(m) 法で用いたシュールコンプルメントによるブロック処理のものと同じものを与えるものとする．つまり，式 (6) の係数行列を $S \equiv \alpha^2 I - A_{br}A_{rb}$ とし，$\hat{\boldsymbol{b}} \equiv \alpha\boldsymbol{b}_b + A_{br}\boldsymbol{b}_r$ とすれば

$$SM\hat{\boldsymbol{y}} = \hat{\boldsymbol{b}}, \quad M\hat{\boldsymbol{y}} = \boldsymbol{x}_b \tag{11}$$

となり，シフト方程式に対しても非常に有効な前処理行列を構成できる.

このシュールコンプルメント前処理の構成方法としては近似逆行列を用いたものを扱う．まず，近似逆行列として以下の最適化問題を解くような前処理行列を考える.

$$\min_M \|I - SM\|_F^2 \tag{12}$$

ただし，行列 Mは $n \times n$ の疎行列とし，$\|\cdot\|_F$はフロベニウスノルム (Frobenius norm) である．この最適化問題は以下の n 個の最小化問題に帰着できる.

$$\min_{m_j} \|\boldsymbol{e}_j - S\boldsymbol{m}_j\|_2^2 \quad j = 1, 2, \ldots, n \qquad (13)$$

ただし，$\boldsymbol{e}_j$と$\boldsymbol{m}_j$はそれぞれ単位行列Iと前処理行列Mのj番目の列ベクトルである．これらn個の最小化問題は互いに独立で並列化可能である．よって，それぞれの問題についてQR分解で$\boldsymbol{m}_j$を求めることで，最終的に近似前処理行列$M = (\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_n)$が求まる．また，近似逆行列を計算する際にいくつかのフィルイン(fill-in)が生じるが，そのフィルインは行列Sと同じ構造をもつように制限し，行列Sと同じように近似逆行列Mを疎行列にする．

このような前処理行列を用いればシフト方程式に対して更に効率的なGMRES(m)法を得ることができ，Shifted-GMRES(m)法よりも計算コストが削減可能であると期待できる．

5. 数値実験

本稿で述べてきたShifted-GMRES(m)法と2種類の前処理を適用したConcurrent-GMRES(m)法の計3つの解法を比較した数値実験を行なった．Concurrent-GMRES(m)法の前処理においては，第4節で紹介した代数的マルチレベル前処理を用いたものをCM-GMRES(m)法，シュールコンプルメント前処理を用いたものをCS-GMRES(m)法として，それぞれの算法を比較した．

使用する計算機はSGI社の並列計算機Origin2400とする．この並列計算機Origin2400については全16CPUで，各CPUがMIPS社のR12000 300MHzであり，メモリが8GB，OSはIRAX 6.5.12である．また，数値実験は倍精度のC言語を用い，収束判定条件は$\|\boldsymbol{r}_m\|_2/\|\boldsymbol{r}_0\|_2 \leq 1.0 \times 10^{-12}$とする．また，初期近似解は$\boldsymbol{x}_0 = \boldsymbol{x}_0' = (0, 0, \ldots, 0)^T$とした．

反復回数についてはアーノルディ過程の1反復につき1回と数える．そのため，Shifted-GMRES(m)法とConcurrent-GMRES(m)法は，最も反復のかかる元の方程式のみを反復回数として数える．また，計算時間は初期値代入から解の算出までの時間とする．clock関数を用いて3回計測した平均値を用いることにする．

並列化に関しては各解法ともにプロセッサの台数を8つとする．Concurrent-GMRES(m)法では前述のとおり各方程式にプロセッサを均等に割り当てた上で，ベクトルとベクトルの和，ベクトルのスカラー倍，ベクトルの内積，行列とベクトルの積の各計算部分を並列化した．Shifted-GMRES(m)法でも同様に上記の計算部分を並列化した．

5.1 数値例1

シフト方程式が現れるような時間依存型偏微分方程式を扱う．数値例として，領域$\Omega = [0, 1] \times [0, 1]$の2次元熱伝導方程式の問題を考える[3]．

$$\frac{\partial u(x,y,t)}{\partial t} = \nabla^2 u(x,y,t) + D\frac{\partial u(x,y,t)}{\partial x} + g(x,y,t) \quad \text{in } \Omega$$
$$u(x,y,t) = 0 \quad \text{on } \partial\Omega$$

ただし，式(4)の共線形条件を満たすような行列をもつ方程式を扱うため，$D = 10.0$と定める．この方程式をx, yに関しては中心差分近似を，そしてtに関しては前進差分近似を用いて離散化する．$h = 1/257$とし，そこで得られた行列の次元は$n = 256^2 = 65536$である．真の解を

$$u(x,y,t) = \frac{x(1-x)y(1-y)}{1+t}$$

と設定して右辺を決定する．時間に関しては表1のように各4つの$0.0 \leq t_i \leq 1.0 \quad (i = 1, 2, 3, 4)$を任意に選び，異なった時間に対する解を同時に求めることにする．

各算法に対する反復回数と計算時間(秒単位)で表した結果を表1に示す．この表によると，各算法ともリスタート周期が10のときが最も短い計算時間になっている．中には，リスタート周期が20や30のときの反復回数が最小になることもあるが，やはり計算時間に関してはリスタート周期が10のときが最も速い結果となった．

代数的マルチレベル前処理を用いたCM-GMRES(m)法よりシュールコンプルメント前処理を用いたCS-GMRES(m)法のほうが多少速くなっていることがわかる．他の算法に比べたCS-GMRES(m)法の計算時間の削減値は，Shifted-GMRES(m)法に対して最大で46%，CM-GMRES(m)法に対して最大で25%となった．

$t_1 = 0.1$における各解法のリスタート周期mを10とし，縦軸に相対残差ノルム，横軸に反復回数をとったグラフを図3に，横軸に計算時間をとったグラフを図4に示す．このグラフにおいては，残差ノルムが10^{-12}のときに解が収束して，求まったと見なしているので，CS-GMRES(m)法が最も速く収束したととれる．どちらの前処理を用いたとしてもConcurrent-GMRES(m)法ほうがShifted-GMRES(m)法よりも速い結果になった．

5.2 数値例2

量子色力学の格子ゲージ計算として単純なモデル問題を扱う[4]．512×512の2次元格子上での近傍点結合を考える．すると，この格子ゲージ計算としての方程式は

$$\boldsymbol{x}_p - \kappa \sum_{\mu=1}^{2} \{U_\mu(p)\boldsymbol{x}_{p+e_\mu} + U_\mu^H(p-e_\mu)\boldsymbol{x}_{p-e_\mu}\} = \boldsymbol{b}_p$$

のように与えられる．ただし，pは格子点の2次元ポインタ(lattice pointer)であり，$U(\cdot)$については$U_\mu(p) = I$, $U_\mu^H(p - e_\mu) = -I$ (I：単位行列)とす

表 1　数値例 1 の結果 (N：反復回数, T：計算時間 (秒))

t_1	1.0		0.5		0.1		0.05		0.01	
t_2	0.5		0.1		0.05		0.01		0.005	
t_3	0.1		0.05		0.01		0.005		0.001	
t_4	0.05		0.01		0.005		0.001		0.0005	
Algorithm	N	T	N	T	N	T	N	T	N	T
Shifted-GMRES(10)	593	46.71	624	51.25	619	54.48	619	49.25	519	38.64
Shifted-GMRES(20)	674	64.96	693	72.36	530	54.73	541	57.77	403	40.48
Shifted-GMRES(30)	732	100.08	642	77.71	641	92.33	630	83.73	400	54.51
Shifted-GMRES(40)	679	100.41	723	103.47	802	127.76	619	107.44	423	65.21
Shifted-GMRES(50)	857	144.87	903	148.07	784	169.40	692	122.82	497	93.64
CM-GMRES(10)	470	42.66	471	42.19	438	39.19	423	32.18	302	25.33
CM-GMRES(20)	480	47.74	549	58.18	455	47.64	481	47.85	308	32.85
CM-GMRES(30)	588	66.70	600	70.87	605	71.06	555	65.16	298	36.65
CM-GMRES(40)	640	89.22	680	95.90	659	94.13	576	81.49	368	55.11
CM-GMRES(50)	686	116.74	613	119.96	621	105.82	601	108.62	316	60.09
CS-GMRES(10)	380	32.35	412	34.16	360	29.90	369	30.58	307	26.41
CS-GMRES(20)	455	50.63	447	45.28	468	46.26	398	37.81	281	29.60
CS-GMRES(30)	705	79.01	540	60.96	570	71.08	540	62.30	301	43.03
CS-GMRES(40)	543	79.06	698	83.17	671	88.98	562	87.71	320	50.78
CS-GMRES(50)	721	123.56	813	122.97	720	109.17	549	90.46	378	66.22

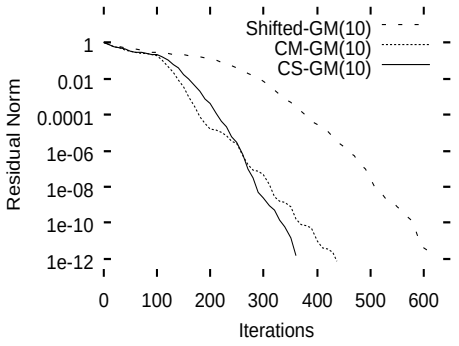

図 3　数値例 $1(t_1=0.1)$ の残差と反復回数

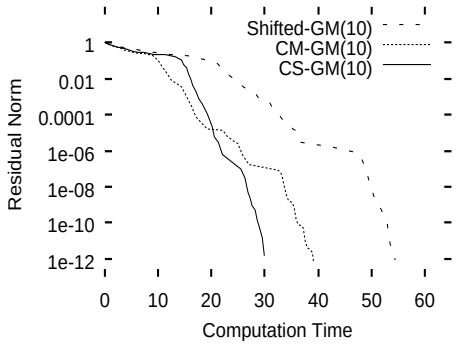

図 4　数値例 $1(t_1=0.1)$ の残差と計算時間

る．また，κは跳躍パラメータ (hopping parameter) で，$0 \leq \kappa < \kappa_c$ (κ_c : critical value) となるようにとる．そこで数値例として表 2のように各 4 つの $0.0 \leq \kappa_i \leq 5.4 \quad (i = 1,2,3,4)$ を任意に選び，異なった跳躍パラメータに対する解を同時に求めることにする．右辺項$\boldsymbol{b}$については，RB オーダリングを用いたときに$\boldsymbol{b}_b = 0$, $\boldsymbol{b}_r = 1$ ($\boldsymbol{b}$：点源 (point source)) となるようにする．

各算法に対する反復回数と計算時間 (秒単位) で表した結果を表 2に示す．この数値例においては，各算法ともにリスタート周期が 30 前後が最も計算時間の少ない結果となっている．よって，最小時間のリスタート周期は経験でしかわからないが，この例での計算時間はそれほどリスタート周期に依存していないため，どのリスタート周期でもあまり計算時間は変わらないことがわかる．

CS-GMRES(m)法の計算時間はShifted-GMRES(m)法に対して最大で 46%の削減，CM-GMRES(m)法に対して最大で 25%の削減となり，他の算法に比べ最も速くなっていることがわかる．この例では Concurrent-GMRES(m) 法の前処理は代数的マルチレベル前処理よりシュールコンプルメント前処理を用いたほうがよりよい結果となった．

$\kappa_1 = 5.4$ における各解法のリスタート周期 m を 30 とし，縦軸に相対残差ノルム，横軸に反復回数をとったグラフを図 5に，横軸に計算時間をとったグラフを図 6に示す．数値例 1 のグラフに比べ，やや直線的なものになっているが，このグラフなら前の例よりも明らかに CS-GMRES(m) 法が最も速く収束している．

6. ま と め

シフト方程式に対して効果的な前処理を加えた Concurrent-GMRES(m) 法とシフト方程式を解くために提案されていた Shifted-GMRES(m) 法を数値実験により比較した．上記の数値例では前処理を加えた Concurrent-GMRES(m) 法のほうが良い結果が得られた．

表 2　数値例 2 の結果 (N：反復回数, T：計算時間 (秒))

κ_1	5.4		5.0		4.6		4.2		3.8	
κ_2	5.3		4.9		4.5		4.1		3.7	
κ_3	5.2		4.8		4.4		4.0		3.6	
κ_4	5.1		4.7		4.3		3.9		3.5	
Method	N	T	N	T	N	T	N	T	N	T
Shifted-GMRES(10)	648	195.76	576	191.11	502	211.42	433	167.10	368	157.31
Shifted-GMRES(20)	432	199.50	388	158.36	347	141.97	307	128.93	269	142.88
Shifted-GMRES(30)	372	171.73	340	152.96	306	138.34	275	124.74	243	126.01
Shifted-GMRES(40)	346	226.69	317	173.30	289	167.26	259	140.66	231	146.51
Shifted-GMRES(50)	332	208.43	306	173.70	278	171.57	252	187.71	225	150.60
CM-GMRES(10)	340	91.31	302	76.83	266	76.21	234	63.60	203	62.46
CM-GMRES(20)	251	82.69	229	73.77	207	62.53	186	62.99	165	51.60
CM-GMRES(30)	229	74.74	209	64.03	191	69.04	172	71.32	156	60.66
CM-GMRES(40)	220	82.84	202	78.61	184	61.95	168	54.80	150	54.75
CM-GMRES(50)	215	85.27	197	102.23	180	84.18	165	73.24	147	60.22
CS-GMRES(10)	265	83.34	231	58.52	210	59.70	181	46.87	160	40.98
CS-GMRES(20)	207	64.33	189	56.99	171	59.74	153	50.81	136	43.34
CS-GMRES(30)	193	58.67	176	73.08	161	57.11	144	46.77	130	36.35
CS-GMRES(40)	186	80.68	171	65.03	155	50.45	141	51.95	128	50.91
CS-GMRES(50)	182	78.82	168	74.67	154	85.00	138	51.29	125	56.00

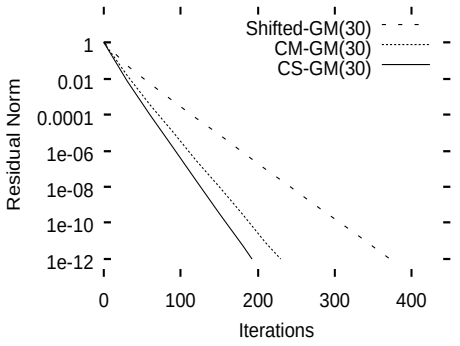

図 5　数値例 $2(\kappa_1=5.4)$ の残差と反復回数

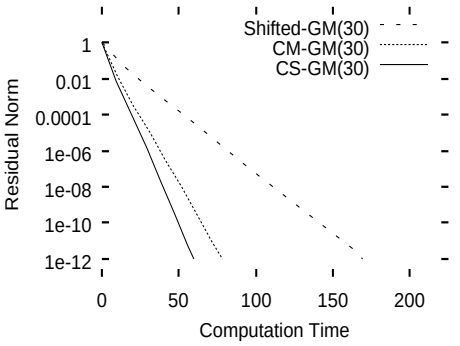

図 6　数値例 $2(\kappa_1=5.4)$ の残差と計算時間

2 つの前処理の比較に関しては，代数的マルチレベル前処理よりシュールコンプルメント前処理のほうが速く解が求まったが，問題のパラメータの値やリスタート周期によっては結果が異なることもあることがわかった．結局，どちらの前処理でもシフト方程式に対しては有効だといえる．

シフト方程式をはじめとする連立 1 次方程式を解く際には前処理によってかなりの時間短縮が見られたことから，前処理が非常に重要であると考えられる．

参 考 文 献

1) Saad Y. and Schultz M. H.: GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.*, Vol. 7, pp. 856–869 (1986).
2) Frommer A. and Glässner U.: Restarted GMRES for shifted linear systems, *SIAM J. Sci. Comput.*, Vol. 19, pp. 15–26 (1998).
3) Gallopoulos E. and Saad Y.: Efficient solution of parabolic equations by Krylov approximation methods, *SIAM J. Sci. Stat. Comput.*, Vol. 13, pp. 1236–1264 (1992).
4) Wilson K.: Quarks and strings on a lattice, *in New Phenomena in Subnuclear Physics, Plenum Press*, pp. 69–142 (1975).
5) Laub A.: Numerical linear algebra aspects of control design computations, *IEEE Trans. Automat. Control*, Vol. 30, pp. 97–108 (1985).
6) Medeke B.: On algebraic multilevel preconditioners in lattice gauge theory, *in Numerical Challenges in Lattice Quantum Chromodynamics, LNCSE, Springer Verlag, Heidelberg*, pp. 99–114 (1999).
7) Saad Y. and Sosonkina M.: Distributed schur complement techiques for general sparse linear systems, *SIAM J. Sci. Comput.*, Vol. 21, pp. 1337–1356 (1999).