閾値を用いたE-SSOR前処理の改良

岩里 洸介[1]　藤野 清次[2]

概要：反復法を用いて連立一次方程式 $A\boldsymbol{x} = \boldsymbol{b}$ の数値解を効率よく求めることを考える．そのためには，前処理の適切な選択，計算量の削減そして並列化による高速化が重要であるとされる．本論文で取り扱うEisenstat 型-SSOR 前処理は，行列の不完全分解ではなく，行列の分離に基づく前処理であるため，計算量が低く抑えられ，収束性も比較的良好であり，ある種理想に近い前処理と思われる．そこで，既報において，残余行列 R の導入によって，E-SSOR 前処理の並列化を我々は成し終えた．さらに，本論文では，残余行列の生成に閾値 (Threshold) を導入して，E-SSOR 前処理のさらなる効率化を目指す．数値実験を通じて，他の前処理の収束性と比較し，提案する閾値つき E-SSOR(tol) 前処理のそれが優れていることを明らかにする．

キーワード：Eisenstat-SSOR，前処理，閾値，残余行列

Improvement of E-SSOR preconditioning by adoption of tolerance value

Abstract: We consider an efficient solution of linear systems by iterative methods. In this case, it is crucial for us to select preconditioning technique, to reduce computation cost and to complete parallelism. In this paper, we treat with Eisenstat- S(Symmetric)SOR preconditioning because of low computation cost owing to adoption of matrix splitting in place of incomplete factorization of matrix. In the previous paper, we completed parallelism of E-SSOR preconditioning by means of introduction of remainder matrix. Moreover, we introduce a concept of tolerance value in order to improve the performance of E-SSOR preconditioning. Through numerical experiments, we will make clear that the proposed Eisenstat-SSOR preconditioning with tolerance value works well as compared with the conventional preconditionings.

Keywords: Eisenstat-SSOR, Preconditioning, Tolerance value, Remainder matrix

1. はじめに

解くべき連立一次方程式を

$$A\boldsymbol{x} = \boldsymbol{b} \tag{1}$$

とする．ただし，$A \in \mathbb{R}^{N\times N}$ を実非対称行列，$\boldsymbol{x} \in \mathbb{R}^N$ を解ベクトル，$\boldsymbol{b} \in \mathbb{R}^N$ を右辺ベクトルとする．この方程式に対する求解法として反復法が用いられる．反復法は前処理と呼ばれる処理によって大幅に収束性を改善することが出来る [5]．一方，前処理は反復 1 回当たりの計算量が増加するため，解く問題によっては収束までの合計時間が増える場合がある．そこで，Eisenstat は E-SSOR 前処理を考案した [2]．E-SSOR 前処理は計算量をほとんど増加させずに，反復法の収束性を改善する有用な前処理であることがよく知られている．

さらに，反復法を高速化するために並列化が重要である．E-SSOR 前処理はアルゴリズム中に逐次性の強い前進 (後退) 代入演算を含むため，並列計算を行うことは困難である．そこで，E-SSOR 前処理に残余行列 R(Remainder の略) を導入し，前処理の並列性を向上させた残余行列を用いた Eisenstat-SSOR 前処理 [4] を提案した．この前処理は E-SSOR 前処理の逐次性の影響を軽減し，並列計算を可能にした．しかし，反復法自体の収束性が悪化するため，行列によっては並列計算を行っても高速化されないケースがあった．

そこで，本論文では，残余行列の生成に閾値 (tolerance)

[1] 九州大学大学院システム情報学府情報学専攻
Graduate School of Information Science and Electrical Engineering, Kyushu University
[2] 九州大学情報基盤研究開発センター
Research Institute for Information Technology, Kyushu University

を導入することにより，E-SSOR 前処理を高速化する閾値つき E-SSOR 前処理 (*tol*) を提案する．ここで，パラメータ “*tol*” は許容値 (tolerance) を意味する．さらに，数値実験において，3 種類の E-SSOR 前処理と ILU(0) 前処理の収束性を比較し，閾値つき E-SSOR 前処理 (*tol*) が他の前処理と比較して収束性が優れていることを明らかにする．

本論文の構成は以下の通りである．第 2 節で，2 種類の E-SSOR 前処理について記述する．そして，第 3 節で，閾値つき E-SSOR 前処理 (*tol*) を提案する．第 4 節では，数値実験を通して，閾値つき E-SSOR 前処理 (*tol*) の性能を評価する．最後に，第 5 節で，まとめを行う．

2. E-SSOR 前処理の概観

2.1 従来型 E-SSOR 前処理

(1) 式において，行列 A を

$$A = L + D + U \tag{2}$$

と分離する．ここで，L，D，U は，行列 A の狭義下三角行列，対角行列，狭義上三角行列を各々意味する．ω は緩和係数を意味する．SSOR 前処理では，前処理行列 M を次式で与える．

$$M = (L + \frac{D}{\omega})(\frac{D}{\omega})^{-1}(U + \frac{D}{\omega}). \tag{3}$$

このとき，(1) 式に前処理行列を両側からかけると，(1) 式は以下のように変換できる．

$$\begin{aligned}&(\frac{D}{\omega})(L + \frac{D}{\omega})^{-1}A(U + \frac{D}{\omega})^{-1}(U + \frac{D}{\omega})\boldsymbol{x}\\ &= (\frac{D}{\omega})(L + \frac{D}{\omega})^{-1}\boldsymbol{b}\end{aligned} \tag{4}$$

前処理後の係数行列 $\tilde{A}$，解ベクトル $\tilde{\boldsymbol{x}}$，残差ベクトル $\tilde{\boldsymbol{r}}$ は各々

$$\tilde{A} = (\frac{D}{\omega})(L + \frac{D}{\omega})^{-1}A(U + \frac{D}{\omega})^{-1}, \tag{5}$$

$$\tilde{\boldsymbol{x}} = (U + \frac{D}{\omega})\boldsymbol{x}, \tag{6}$$

$$\tilde{\boldsymbol{r}} = (\frac{D}{\omega})(L + \frac{D}{\omega})^{-1}\boldsymbol{r} \tag{7}$$

と表される．このとき，前処理後の係数行列 $\tilde{A}$ を

$$\begin{aligned}\tilde{A} &= (\frac{D}{\omega})(L + \frac{D}{\omega})^{-1}A(U + \frac{D}{\omega})^{-1}\\ &= (\frac{D}{\omega})((U + \frac{D}{\omega})^{-1}\\ &\quad +(L + \frac{D}{\omega})^{-1}(I + (1 - \frac{2}{\omega})D(U + \frac{D}{\omega})^{-1}))\end{aligned} \tag{8}$$

と式変形すると，$\tilde{A}$ とベクトル $\boldsymbol{v}$ の積を次の手順で計算することで計算量を削減できる [1][2].

1. $\boldsymbol{y} = (U + \frac{D}{\omega})^{-1}\boldsymbol{v}$ (9)
2. $\boldsymbol{z} = \boldsymbol{v} + (1 - \frac{2}{\omega})D\boldsymbol{y}$ (10)
3. $\boldsymbol{w} = (L + \frac{D}{\omega})^{-1}\boldsymbol{z}$ (11)
4. $\tilde{A}\boldsymbol{v} = (\frac{D}{\omega})(\boldsymbol{y} + \boldsymbol{w})$ (12)

本論文では，この前処理を，従来型 E-SSOR 前処理と呼ぶ．

2.2 残余行列 R を用いた E-SSOR 前処理

次に，残余行列 R を用いた E-SSOR 前処理 [4] を導出する．この前処理は残余行列 R (Remainder の頭文字) の適用によって，E-SSOR 前処理の並列化を可能にする前処理である．**図 1** に 4 個の対角ブロックに分割した場合の残余行列を用いた E-SSOR 前処理における領域分割図を示す．ここで，行列 $\hat{L}$ と $\hat{U}$ は，係数行列の狭義下三角部分，狭義上三角部分の内，並列時に前処理行列として用いられる行列を表す．残余行列 R は，$A = L + D + U$ と分離した場合の狭義下三角行列 L，狭義上三角行列 U の要素の内，上記の $\hat{L}$，$\hat{U}$ に含まれない要素の集まり (行列) を表す．また，残余行列 R の要素うち，狭義下三角行列 L に該当する要素を R_{L}，狭義上三角行列 U に該当する要素を R_{U} と表す．つまり，残余行列は $R = R_{\mathrm{L}} + R_{\mathrm{U}}$ で表される．前処理行列 M_{R} は次のように定義する．

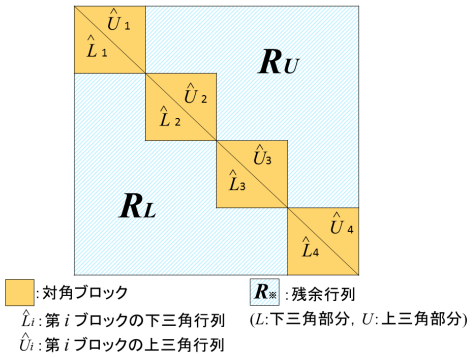


図 1 残余行列を用いた E-SSOR 前処理における領域分割図
Fig. 1 A diagram of division of coefficient in E-SSOR preconditioning with remainder matrix.

$$M_{\mathrm{R}} = (\hat{L} + \frac{D}{\omega})(\frac{D}{\omega})^{-1}(\hat{U} + \frac{D}{\omega}). \tag{13}$$

このとき，(1) 式は以下のように変換される．

$$\begin{aligned}&(\frac{D}{\omega})(\hat{L} + \frac{D}{\omega})^{-1}A(\hat{U} + \frac{D}{\omega})^{-1}(\hat{U} + \frac{D}{\omega})\boldsymbol{x}\\ &= (\frac{D}{\omega})(\hat{L} + \frac{D}{\omega})^{-1}\boldsymbol{b}\end{aligned} \tag{14}$$

前処理後の係数行列 $\tilde{A}$，$\tilde{\boldsymbol{x}}$，$\tilde{\tilde{r}}$ は各々

$$\tilde{A} = (\frac{D}{\omega})(\hat{L}+\frac{D}{\omega})^{-1}A(\hat{U}+\frac{D}{\omega})^{-1}, \quad (15)$$

$$\tilde{\boldsymbol{x}} = (\hat{U}+\frac{D}{\omega})\boldsymbol{x}, \quad (16)$$

$$\tilde{\boldsymbol{r}} = (\frac{D}{\omega})(\hat{L}+\frac{D}{\omega})^{-1}\boldsymbol{r} \quad (17)$$

と表される．このとき，前処理後の係数行列 $\tilde{A}$ は

$$\begin{aligned}\tilde{A} &= (\frac{D}{\omega})(\hat{L}+\frac{D}{\omega})^{-1}A(\hat{U}+\frac{D}{\omega})^{-1}\\ &= (\frac{D}{\omega})(\hat{L}+\frac{D}{\omega})^{-1}\\ &\quad\times(\hat{L}+\frac{D}{\omega}+\hat{U}+\frac{D}{\omega}+(R_{\mathrm{L}}+R_{\mathrm{U}})+(1-\frac{2}{\omega})D)(\hat{U}+\frac{D}{\omega})^{-1}\\ &= (\frac{D}{\omega})((\hat{U}+\frac{D}{\omega})^{-1}+(\hat{L}+\frac{D}{\omega})^{-1}\\ &\quad\times(I+(1-\frac{2}{\omega})D(\hat{U}+\frac{D}{\omega})^{-1}+R(\hat{U}+\frac{D}{\omega})^{-1})) \quad (18)\end{aligned}$$

と式変形できる．以上から，行列 $\tilde{A}$ とベクトル $\boldsymbol{v}$ の積 $\tilde{A}\boldsymbol{v}$ の計算は次の 5 段階の手順で行える．ここで，下線部は，従来の E-SSOR 前処理と異なるところである．

1. $\boldsymbol{y} = (\hat{U}+\frac{D}{\omega})^{-1}\boldsymbol{v}$
2. $\boldsymbol{z} = \boldsymbol{v} + (1-\frac{2}{\omega})D\boldsymbol{y}$
3. $\underline{\boldsymbol{u} = R\boldsymbol{y}}$
4. $\boldsymbol{w} = (\hat{L}+\frac{D}{\omega})^{-1}\underline{(\boldsymbol{z}+\boldsymbol{u})}$
5. $\tilde{A}\boldsymbol{v} = (\frac{D}{\omega})(\boldsymbol{y}+\boldsymbol{w})$

この前処理を残余行列 R を適用した前処理として，R 型 E-SSOR 前処理と呼ぶ．

3. 閾値つき E-SSOR(tol) 前処理の提案

本節では閾値を用いて残余行列 R を生成する閾値つき E-SSOR(tol) 前処理を導出する．まず，本前処理では前処理行列を構成する三角行列を $\bar{L}$，$\bar{U}$ と定義する．三角行列 L，U の要素のうち，要素の絶対値が閾値よりも大きいものは $\bar{L}$，$\bar{U}$ に格納し，閾値よりも小さいものは残余行列 R に格納する．**図 2** に閾値つき E-SSOR(tol) 前処理の概略を示す．ここで，a_{ij} は元の係数行列の i 行 j 列の要素，τ は閾値を意味する．以下に三角行列 $\bar{L}$，$\bar{U}$ と残余行列 R の生成手順を示す．ここで，n は次元数を示す．

1. $i = 1, n$
2. $\quad j = 1, n$
3. $\quad\quad$ if $(i = j)$ then
4. $\quad\quad\quad D \leftarrow a_{ij}$
5. $\quad\quad$ else if $(|a_{ij}| < \tau)$ then
6. $\quad\quad\quad R \leftarrow a_{ij}$
7. $\quad\quad$ else if $(|a_{ij}| \geq \tau)$ then
8. $\quad\quad\quad$ if $(i > j)$ then
9. $\quad\quad\quad\quad \bar{L} \leftarrow a_{ij}$
10. $\quad\quad\quad$ else if $(i < j)$ then
11. $\quad\quad\quad\quad \bar{U} \leftarrow a_{ij}$
12. $\quad\quad\quad$ end if
13. $\quad\quad$ end if
14. $\quad$ end do
15. end do

このとき，前処理行列 M_{T} を $\bar{L}$，D，$\bar{U}$ を用いて次のように構成する．

$$M_{\mathrm{T}} = (\bar{L}+\frac{D}{\omega})(\frac{D}{\omega})^{-1}(\bar{U}+\frac{D}{\omega}). \quad (19)$$

前処理後の係数行列 $\tilde{A}$，$\tilde{\boldsymbol{x}}$，$\tilde{\boldsymbol{r}}$ は各々

$$\tilde{A} = (\frac{D}{\omega})(\bar{L}+\frac{D}{\omega})^{-1}A(\bar{U}+\frac{D}{\omega})^{-1}, \quad (20)$$

$$\tilde{\boldsymbol{x}} = (\bar{U}+\frac{D}{\omega})\boldsymbol{x}, \quad (21)$$

$$\tilde{\boldsymbol{r}} = (\frac{D}{\omega})(\bar{L}+\frac{D}{\omega})^{-1}\boldsymbol{r}. \quad (22)$$

と表される．このとき，前処理後の係数行列 $\tilde{A}$ は

$$\begin{aligned}\tilde{A} &= (\frac{D}{\omega})(\bar{L}+\frac{D}{\omega})^{-1}A(\bar{U}+\frac{D}{\omega})^{-1}\\ &= (\frac{D}{\omega})(\bar{L}+\frac{D}{\omega})^{-1}\\ &\quad\times(\bar{L}+\frac{D}{\omega}+\bar{U}+\frac{D}{\omega}+D-\frac{2D}{\omega}+R)(\bar{U}+\frac{D}{\omega})^{-1}\\ &= (\frac{D}{\omega})((\bar{U}+\frac{D}{\omega})^{-1}\\ &\quad+(\bar{L}+\frac{D}{\omega})^{-1}((I+(1-\frac{2}{\omega})D+R)(\bar{U}+\frac{D}{\omega})^{-1})) \quad (23)\end{aligned}$$

と変形できる．以上から，行列 $\tilde{A}$ とベクトル $\boldsymbol{v}$ の積 $\tilde{A}\boldsymbol{v}$ の計算は次の 5 段階の手順で行える．ここで，下線部は，従来の E-SSOR 前処理と異なるところである．

1. $\boldsymbol{y} = (\bar{U}+\frac{D}{\omega})^{-1}\boldsymbol{v}$ (24)
2. $\boldsymbol{z} = \boldsymbol{v} + (1-\frac{2}{\omega})D\boldsymbol{y}$ (25)
3. $\underline{\boldsymbol{u} = R\boldsymbol{y}}$ (26)
4. $\boldsymbol{w} = (\bar{L}+\frac{D}{\omega})^{-1}\underline{(\boldsymbol{z}+\boldsymbol{u})}$ (27)
5. $\tilde{A}\boldsymbol{v} = (\frac{D}{\omega})(\boldsymbol{y}+\boldsymbol{w})$ (28)

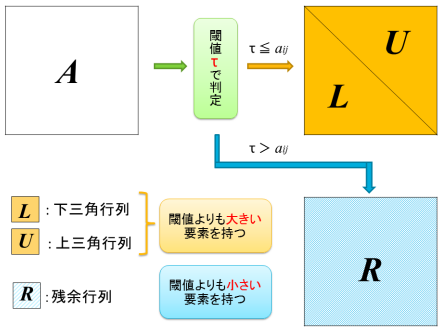


図 2 閾値つき E-SSOR(tol) 前処理の概略

Fig. 2 A diagram of E-SSOR preconditioning with tolerance value.

表 1 テスト行列の 11 個の主な特徴
Table 1 Characteristics of 11 test matrices.

行列	次元数	総非零要素数	平均非零要素数
bcircuit	68,902	375,558	5.5
big	13,209	91,465	6.9
bridge	64,461	4,373,817	67.9
k3plates	11,107	378,927	34.1
raefsky3	21,200	1,488,768	70.2
sme3Da	12,504	874,887	70.0
sme3Db	29,067	2,081,063	71.6
sme3Dc	42,930	3,148,656	73.3
water_tank	60,740	2,035,281	33.5
xenon1	48,600	1,181,120	24.3
xenon2	157,464	3,866,688	24.6

この閾値つき E-SSOR(tol) 前処理を，T 型 E-SSOR 前処理と呼ぶ．

4. 数値実験

計算機環境と計算条件を以下に示す．計算機は SR16000(CPU : IBM Power7(4.00GHz，32MB L3 Cache. 8cores × 32), OS : AIX) を使用した．プログラムは Fortran90，コンパイラは日立最適化 Fortran Compiler を使用した．最適化オプションは “-hf95,O3” を用いた．計算はすべて倍精度浮動小数点演算で行い，時間計測には時間計測関数 “getrusage” を用いた．収束判定条件は相対残差の 2 ノルム : $||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{r}_0||_2 \leq 10^{-12}$，初期近似解 $\boldsymbol{x}_0$ はすべて $\mathbf{0}$，最大反復回数は 10,000 回とした．行列は予め対角スケーリングによって対角項を 1 に正規化した．前処理は従来型，R 型，T 型の 3 種類の E-SSOR 前処理と ILU(0) 前処理を使用し，解法は BiCGSafe 法 [3] を用いた．緩和係数 ω は 0.5，0.8，1.0，1.2，1.5 の 5 通りとした．計算量削減のため，残差ノルムへの変換を伴う収束判定は，5 反復当たり 1 回行った．閾値 (τ と表記する) は 0.1，0.01 の 2 通りだけ調べた．

4.1 テスト行列

表 1 にテスト行列の 11 個の特徴を示す．テスト行列は，行列 bridge を除き，フロリダ大学のデータベース [6] から選出した．

4.2 T 型 E-SSOR 前処理の収束性調査

表 2 に 4 種類の E-SSOR 前処理の収束性を示す．ここで，“R 要素数” は残余行列 R の非零要素数を示し，“比率” は係数行列全体の総非零要素数を 1 としたときの残余行列 R の非零要素数の比を示す．“tot-t.”，“ave-t.” はそれぞれ合計計算時間と 1 反復当りの計算時間を示し，“ratio1”，“ratio2” は同様に，合計計算時間の比と 1 反復当りの計算時間の比を示す．また，“ω” は緩和係数を意味する．T 型

表 2 4 種類の前処理の収束性
Table 2 The convergence rate of some preconditionings.

(a) 行列 bcircuit(非零要素数 : 375,558)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.0	4,706	**32.175**	**0.81**	6.84	1.69
従来型	-	-	1.0	9,785	39.621	1.00	4.05	1.00
R 型	76,502	0.204	-	max	-	-	-	-
T 型	154,870	0.412	0.8	9,600	45.702	1.15	4.76	1.18

(b) 行列 big(非零要素数 : 91,465)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.0	974	1.288	1.62	1.32	1.77
従来型	-	-	1.0	1,065	**0.796**	**1.00**	0.75	1.00
R 型	40,184	0.439	1.5	1,800	1.544	1.94	0.86	1.15
T 型	0	0.000	1.2	1,070	0.908	1.14	0.85	1.14

(c) 行列 bridge(非零要素数 : 4,373,817)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	902	**42.653**	**0.78**	47.29	1.45
従来型	-	-	0.8	1,670	54.591	1.00	32.69	1.00
R 型	198,616	0.045	0.8	1,890	62.509	1.15	33.07	1.01
T 型	2,700,998	0.618	0.8	2,155	43.720	0.80	20.29	0.62

(d) 行列 k3plates(非零要素数 : 378,927)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	3,319	9.663	0.99	2.91	1.27
従来型	-	-	1.2	4,265	9.796	1.00	2.30	1.00
R 型	9,592	0.025	1.0	5,470	15.006	1.53	2.74	1.19
T 型	84,262	0.222	0.8	4,565	**8.297**	**0.85**	1.82	0.79

(e) 行列 raefsky3(非零要素数 : 1,488,768)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.0	141	**2.503**	**0.17**	17.75	1.62
従来型	-	-	0.8	1,335	14.591	1.00	10.93	1.00
R 型	102,208	0.069	0.8	1,515	15.149	1.04	10.00	0.91
T 型	1,191,734	0.800	0.8	1,365	7.260	0.50	5.32	0.49

E-SSOR 前処理の閾値は行列 bridge と行列 raefsky3，行列 xenon2 では $\tau = 0.1$，他の行列では $\tau = 0.01$ のときの結果を示す．太字表記は各行列における最速ケースを意味する．また，本実験では収束時の解ベクトルの精度の確認を行った．しかし，どの前処理も同様の解の精度で収束していたため，表中では記載していない．

表 2 から以下のことが分かる．

(1) T 型 E-SSOR 前処理は従来型と比較して，反復回数が増加するケースが多い．

(2) しかし，T 型 E-SSOR 前処理は従来型よりも平均計算

(f) 行列 sme3Da(非零要素数：874,887)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	791	6.783	1.23	8.58	1.35
従来型	-	-	1.2	870	5.510	1.00	6.33	1.00
R 型	650,453	0.743	1.2	1,645	7.329	1.33	4.46	0.70
T 型	217,471	0.249	1.0	925	**4.210**	**0.76**	4.55	0.72

(g) 行列 sme3Db(非零要素数：2,081,063)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.0	1,244	35.798	1.93	28.78	1.76
従来型	-	-	1.2	1,135	18.521	1.00	16.32	1.00
R 型	1,545,157	0.742	1.0	2,190	28.891	1.56	13.19	0.81
T 型	518,846	0.249	1.0	1,140	**16.144**	**0.87**	14.16	0.87

(h) 行列 sme3Dc(非零要素数：3,148,656)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	1,370	63.544	1.62	46.38	1.78
従来型	-	-	1.2	1,505	**39.132**	**1.00**	26.00	1.00
R 型	2,340,293	0.743	1.2	2,915	63.761	1.63	21.87	0.84
T 型	789,162	0.372	1.0	1,540	40.296	1.03	26.17	1.01

(i) 行列 water_tank(非零要素数：2,035,281)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.0	301	**5.904**	**0.36**	19.61	1.08
従来型	-	-	1.0	900	16.399	1.00	18.22	1.00
R 型	92,953	0.046	1.0	1,190	21.674	1.32	18.21	1.00
T 型	923,728	0.454	1.0	890	7.534	0.46	8.47	0.46

(j) 行列 xenon1(非零要素数：1,181,120)

前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	297	3.289	0.79	11.07	1.30
従来型	-	-	0.8	485	4.140	1.00	8.54	1.00
R 型	63,832	0.054	1.0	635	5.545	1.34	8.73	1.02
T 型	306,180	0.259	1.2	465	**2.852**	**0.69**	6.13	0.72

(k) 行列 xenon2(非零要素数：3,866,688)

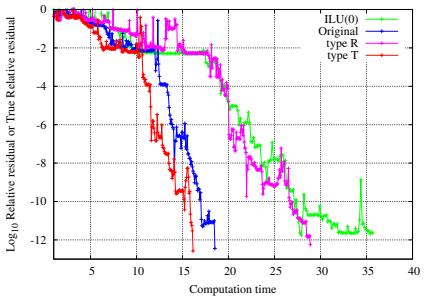
前処理	R 要素数	比率	ω	反復回数	tot-t. [s.]	ratio 1	ave-t. [ms.]	ratio 2
ILU(0)	-	-	1.2	351	13.330	0.80	37.98	1.33
従来型	-	-	1.0	585	16.670	1.00	28.50	1.00
R 型	102,280	0.026	0.5	850	26.033	1.56	30.63	1.07
T 型	2,516,566	0.651	0.8	885	**7.908**	**0.47**	8.94	0.31

時間が短く，従来型よりも速く収束した．

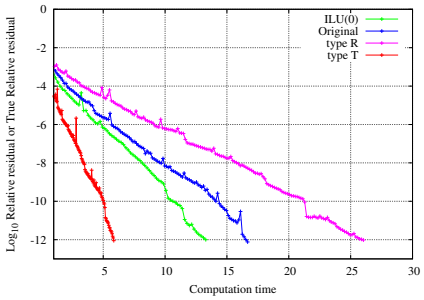
(3) 特に，行列 sme3Da に対しては従来型の 0.76 倍，行列 xenon2 に対しては従来型の 0.47 倍の合計時間で収束し，他の前処理よりも優れた収束性を示した．

(4) R 型 E-SSOR 前処理は，平均計算時間は短縮されるが，反復回数が増加するため，合計時間が増加するケースが多かった．

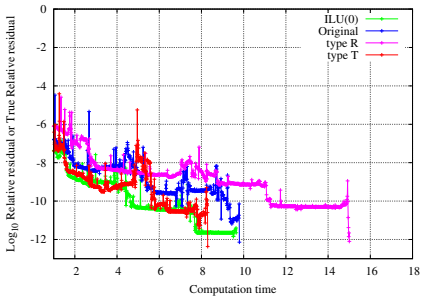
図 3 に行列 sme3Db と行列 xenon2 と行列 k3plates における 4 種類の前処理の残差履歴図を示す．ここで，横軸は計算時間 (単位:秒)，縦軸は相対残差の 2 ノルム $||\boldsymbol{r}_{k+1}||_2/||\boldsymbol{r}_0||_2$ の常用対数表示を意味する．図 3 から T 型 E-SSOR 前処理は従来型よりも速く収束していることが分かる．



(a) 行列 sme3Db



(b) 行列 xenon2



(c) 行列 k3plates

図 3 4 種類の前処理の残差履歴図

Fig. 3 History of relative residual of some preconditionings.

表 3 に閾値を変化させた時の T 型 E-SSOR 前処理の収束性を示す．表中では閾値 “tol.” を 0.001，0.005，0.01，0.05，0.1 の 5 通りに変化させた．表中の最上位の行は従来型の結果を示している．“pre-t.”，“itr-t.” はそれぞれ前処理時間と反復時間を示す．表 3 から以下のことが分かる．

(1) どちらの行列に対しても，閾値を 0.05 にした時に最速で収束した．

(2) 行列 k3plates では 21%，行列 xenon1 では 25%，合計時間が短縮された．

(3) 係数行列の全非零要素の約 1/2 から約 1/4 が残余行列に割り当てられた時に，最速で収束するケースが多い．

表 4 にテスト行列 11 個に対する 4 種類の前処理の合計

表 3 閾値を変化させた時の T 型 E-SSOR 前処理の収束性
Table 3 The convergence rate of E-SSOR preconditioning in several tolerance values.

(a) 行列 xenon1(非零要素数：1,181,120)

tol.	R 要素	比率	ω	反復回数	pre-t. [s.]	itr-t. [s.]	tot-t. [s.]	ratio
-	0	-	1.0	500	0.046	3.367	3.413	1.00
0.001	43,220	0.04	0.8	515	0.024	3.713	3.737	1.10
			1.0	490	0.024	3.512	3.536	1.04
			1.2	480	0.016	3.452	3.468	1.02
0.005	201,948	0.17	0.8	475	0.016	3.092	3.108	0.91
			1.0	530	0.032	3.428	3.460	1.01
			1.2	545	0.016	3.545	3.560	1.04
0.01	306,180	0.26	0.8	500	0.032	2.962	2.994	0.88
			1.0	505	0.032	2.990	3.022	0.89
			1.2	510	0.032	3.024	3.056	0.90
0.05	625,714	0.53	0.8	530	0.024	2.912	2.936	0.86
			1.0	500	0.024	2.764	**2.787**	0.82
			1.2	590	0.024	3.254	3.278	0.96
0.1	766,398	0.65	0.8	540	0.024	2.960	2.984	0.87
			1.0	545	0.016	3.000	3.016	0.88
			1.2	610	0.024	3.336	3.360	0.98

(b) 行列 xenon2(非零要素数：3,866,688)

tol.	R 要素	比率	ω	反復回数	pre-t. [s.]	itr-t. [s.]	tot-t. [s.]	ratio
-	0	-	1.0	530	0.138	16.017	16.155	1.00
0.001	142,596	0.04	0.8	610	0.065	18.942	19.007	1.18
			1.0	600	0.080	18.605	18.685	1.16
			1.2	640	0.081	19.868	19.949	1.23
0.005	667,348	0.17	0.8	540	0.081	14.342	14.423	0.89
			1.0	605	0.074	16.145	16.218	1.00
			1.2	665	0.081	17.690	17.771	1.10
0.01	1,008,900	0.26	0.8	610	0.073	13.411	13.485	0.83
			1.0	615	0.082	13.507	13.588	0.84
			1.2	605	0.081	13.320	13.402	0.83
0.05	2,063,098	0.53	0.8	635	0.082	12.384	12.466	0.77
			1.0	585	0.081	11.463	**11.543**	0.71
			1.2	665	0.082	13.027	13.109	0.81
0.1	2,516,566	0.65	0.8	575	0.082	11.191	11.273	0.70
			1.0	665	0.082	12.880	12.962	0.80
			1.2	655	0.074	12.747	12.820	0.79

時間による性能比較を示す．“最速ケース”，“最遅ケース”は各前処理が最も速く収束した行列，最も遅く収束した行列の数を示す．“従来型より高速”，“ILU(0) より高速” は各前処理が従来型 E-SSOR 前処理，ILU(0) 前処理よりも高速で収束した行列の数を示す．表 4 から以下のことが分かる．

(1) T 型 E-SSOR 前処理が 11 個中 5 個の行列で最速で収束し，4 種類の前処理の中で最多だった．
(2) 次いで，ILU(0) 前処理が 11 個 4 個の行列に対して最速で収束した．
(3) ILU(0) 前処理は最も遅く収束したケースがあり，行列によって収束性が大きく変化するが，T 型 E-SSOR 前処理は様々な行列に対して高速で収束している．
(4) T 型 E-SSOR 前処理は 8 個の行列に対して従来型 E-SSOR 前処理よりも高速で収束した．
(5) 同様に，T 型 E-SSOR 前処理は 7 個の行列に対して ILU(0) 前処理よりも高速で収束した．

表 4 4 種類の前処理の合計時間による性能比較
Table 4 The comparison of performance of some preconditionings from the viewpoint of total computation time.

前処理	最速ケース	最遅ケース	従来型より高速	ILU(0) より高速
ILU(0)	4	1	7	-
従来型	2	0	-	4
R 型	0	10	0	1
T 型	**5**	0	**8**	**7**
全体	11	11	-	-

5. まとめ

本論文では，残余行列の生成に閾値を導入することにより，E-SSOR 前処理を高速化する閾値つき E-SSOR(tol) 前処理を提案し，その収束性を調査した．数値実験から，閾値つき E-SSOR(tol) 前処理が 11 個中 5 個の行列で最速で収束し，高い収束安定性を示すことがわかった．

今後の課題は閾値つき E-SSOR(tol) 前処理の並列化を行い，並列性能を調査することである．

参考文献

[1] Chan, T. F., van der Vorst, H. A.: Approximate and Incomplete Factorizations, in D.E. Keyes, A. Samed and V. Venkatakrishnan (eds.), Parallel Numerical Algorithms, 1997.
[2] Eisenstat, S. C.: Efficient implementation of a class of preconditioned conjugate gradient methods, SIAM J. Sci. Stat. Comput., Vol.2, No.1, pp.1–4, 1981.
[3] Fujino, S., Fujiwara, M., Yoshida, M.: A proposal of preconditioned BiCGSafe method with safe convergence, Proc. of The 17th IMACS World Congress on Scientific Computation, Appl. Math. Simul., CD-ROM, Paris, France, 2005.
[4] 伊東千晶, 岩里洸介, 村上啓一, 藤野清次: Cache-Cache(カシュカシュ) Elements 並列化手法の提案, 日本計算工学会論文集, Vol.2014, No.20140013，2014.
[5] Y. Saad : Iterative Methods for Sparse Linear Systems, 2nd Ed, SIAM, Philadelphia, 2003.
[6] University of Florida Sparse Matrix Collection: http://www.cise.ufl.edu/research/sparse/matrices/index.html