数理解析研究所講究録 1320 巻 2003 年 179-189

近似逆行列による前処理の特性について

Convergence Property of Approximate Inverse Preconditioners

九州大学・情報基盤センター　藤野 清次 (Seiji Fujino)
Computing and Communications Center,
Kyushu University

概要:
本研究では, 大規模な正定値行列を係数行列に持つ連立1次方程式を共役勾配 (Conjugate Gradient) 法で解くための前処理技術: 従来の不完全コレスキー分解, Kaporin 型の U^TU 分解そして A-直交化 (A-orthogonality) を利用した近似逆行列 (Approximate Inverse) 法の収束特性について論じることにする [6].

1　はじめに

スパース (疎な) 連立1次方程式 $Ax = b$ の近似解を求めることを考える. A は $n \times n$ の大きさの対称正定値 (SPD) 行列 $A = (a_{ij})$ で, b, x は右辺ベクトルおよび解ベクトルを各々表す. 本研究では, 共役勾配 (CG) 法を使ってこの連立1次方程式を解くことにする.

一般に, SPD 行列に対する前処理として不完全コレスキー (IC) 分解が有効とされる. IC 分解は分解が容易にできるが, その有効性が理論的にわかっているのは行列が M 行列のときだけである [12]. それ以外の行列のとき場合によっては, 分解の途中でピボットが零または非常に小さな値になり破綻をきたすことがある.

一方, Benzi と Tuma によって提案された A-直交化を用いる SAINV 法は, 棄却許容用のパラメータ γ を使って A-直交化を不完全に行なっても破綻しないことが理論的にわかっている. しかし, 前処理のコストは通常の IC 分解の場合よりも大きい. また, Kaporin による $U^TU + U^TR + R^TU$ 分解は中間メモリを多く必要とするが, 収束性は優れており, 前処理によってその特徴は異なることが多い. そこで, 本研究ではその特徴を明らかにするために以下の前処理法について収束特性の評価を行なう.

1. 対角 (Diagonal) スケーリング
2. フィルインを考慮しない不完全コレスキー分解 : IC(0) 分解
3. Kaporin による $U^TU + U^TR + R^TU$ 分解の RIC2S 法 [9] ([13] 参考)
4. Ajiz と Jennings による RIC1 法 [1]
5. Benzi と Tuma による **SAINV**(Stabilized AINV) 法 [3]
6. Benzi と Tuma による **RIF**(Robust Incomplete Factorization) 法 [4]
7. Kolotilin による **FSAI**(Factorized Sparse Approximate Inverse) 法 [8]

1.1 Kaporin による $U^TU + U^TR + R^TU$ 分解の RIC2S 法

行列 A を次のように分解する.

$$A = U^TU + U^TR + R^TU \tag{1}$$

ここで, A は正値対称行列, U は正則な上三角行列そして R は厳密な上三角行列で誤差行列を表す. ここで, 厳密とは対角成分が 0 の上三角行列をさす. このとき, 行列 U と R は互いに" 構造的直交性" を持つものとする. すなわち,

$$(U)_{ij} \times (R)_{ij} = 0 \qquad \text{for all } i, j \tag{2}$$

である. したがって, U^TR の対角成分も 0 になる. この前提条件の下で, 構造的直交性を利用し, “中間行列” を経て $U^TU + U^TR + R^TU$ 分解が一意に決定される. Kaporin による $U^TU + U^TR + R^TU$ 分解を行なう RIC2S 法の算法を以下に示す. ここで, 行列は $A = (a_{ij})$ とし, n はその次元数, 定数 τ は棄却許容値を各々表す.

Kaporin による RIC2S 法の算法

```
for i = 1, ..., n
  for j = i, ..., n
    v_j = a_ij
  end for
  for k = 1, ..., i-1
    for j = i, ..., n
      v_j = v_j - u_ki u_kj - u_ki r_kj - r_ki u_kj
    end for
  end for
  u_ii = sqrt(v_i)
  for j = i+1, ..., n
    v_j = v_j / u_ii
  end for
  for j = i+1, ..., n
    if |v_j| >= τ then
      u_ij = v_j
    else if |v_j| > 0 then
      r_ij = v_j
    end if
  end for
end for
```

上の算法の中で, $\sigma_j(v)$ は棄却用のパラメータ τ を使って次のように定義される. すなわち, 分解後の行列 U の要素の値が棄却許容値 τ より小さいとき棄却される. しかし, 正定値性は保持されることが理論的に明らかになっている.

$$\begin{cases} \sigma_j(v) = 0, & \textbf{if} \quad |v_j| < \tau \\ \sigma_j(v) = 1, & \textbf{if} \quad |v_j| \geq \tau \end{cases}$$

1.2 Ajiz と Jennings による RIC1 法

構造工学の分野でよく知られた技法に Ajiz と Jennings によって開発された頑強な不完全コレスキー分解 RIC1 法がある [1]. この方法は「**安定化相殺**」法とも呼ばれる. すなわち, 不完全コレスキー因子 L の要素 l_{ij} が棄却されるとき, 対応する対角項 a_{ii} と a_{jj} が修正される. RIC1 法の算法は次のように表せる. ここで, 元の行列を $A = (a_{ij})$ とし, τ を棄却許容値とする.

Ajiz と Jennings による RIC 1 法の算法

```
for i = 1,···,n
  d_i = a_ii
end for
for i = 1,···,n
  for j = i+1,···,n
    v_j = a_ij
  end for
  for k = 1,···,i-1
    for j = i+1,···,n
      v_j = v_j - u_ki u_kj
    end for
  end for
  for j = i+1,···,n
    ξ = |v_j|/√(d_i d_j)
    if ξ ≤ τ then
      v_j = 0,  d_i = (1+ξ)d_i,  d_j = (1+ξ)d_j
    end if
  end for
  u_ii = √d_i
  for j = i+1,···,n
    u_ij = v_j/u_ii
  end for
  for j = i+1,···,n
    d_j = d_j - u_ij^2
  end for
end for
```

RIC1 法は分解する行列 A が対称正定値行列であれば分解することができる. また, 不完全コレスキー分解と同様に前処理に要する時間も非常に少ない. しかし, 行列積 U^TU の近似精度は, 棄却許容値 τ に関して 1 次のオーダ $O(\tau)$ で表され, そのため CG 法の収束に時間がかかる場合もある. RIC1 法と RIC2S 法の分解の形を以下に示す. ここで, E は誤差行列を表し保存されない. 一方, R も誤差行列を表すが分解が終了するまで保存される. また, D は対角行列, I は単位行列を各々表す. σ は定数で原論文では 2 程度の値とされる. τ は棄却許容値である. S は中間行

前処理	分解形	備考
RIC1 法	$A = U^TU - (E + E^T - D)$	
RIC2S 法	$A = U^TU + U^TR + R^TU - (\sigma\tau^2 I + S)$	,σ は定数, τ は棄却許容値

1.3 SAINV 法

Z は単位上三角行列, D は対角行列とする. すなわち, $Z = (z_1, z_2, \dots, z_n)$,

$$D = \begin{pmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_n \end{pmatrix} \tag{3}$$

である. z_i は行列 Z の第 i 番目の列を表す. 行列 Z および D を求める具体的な算法は, 初期値を単位基本ベクトル e_j として以下のように表される.

$$\begin{array}{rll} 1: & \textbf{let} & j = 1, \dots, n,\ z_j = e_j \\ 2: & \textbf{for} & i = 1, \dots, n \\ 3: & & v_i = Az_i \\ 4: & & \textbf{for}\ \ j = i, \dots, n \\ 5: & & p_j = v_i^T z_j \\ 6: & & \textbf{end}\ \ \textbf{for} \\ 7: & & \textbf{for}\ \ j = i+1, \dots, n \\ 8: & & z_j = z_j - \frac{p_j}{p_i} z_i \\ 9: & & \textbf{end}\ \ \textbf{for} \\ 10: & \textbf{end}\ \ \textbf{for} & \end{array} \tag{4}$$

この算法は, 元の AINV 法 [2] に対して Stabilized Approximate INVerse factorization (**SAINV**) 法と呼ばれる [3]. z_i の値が非常に小さくなりそれが棄却されても, 元の行列 A が正定値行列であれば p_j の値は負にならないので分解過程の途中で破綻も起こらない. そして求まった行列 Z, D を用いて前処理行列 M は次のように表せる.

$$M = ZD^{-1}Z^T (\approx A^{-1}) \tag{5}$$

1.4 RIF 法

A-直交化で得られた近似逆行列 $A^{-1} = ZD^{-1}Z^T$ を使用する方法といわゆる完全分解 $A = LDL^T$ を使用する方法とが融合したものがこの Robust Incomplete Factorization (**RIF**) 法である. 2 つの式の間には論文 [7] ですでに指摘されているように次の関係が成り立つ.

$$Z^T = L^{-1} \tag{6}$$

この関係式と式 $Z^TAZ = D$ を使って次の関係が導ける.

$$AZ = LD \text{ または } L = AZD^{-1} \tag{7}$$

したがって,

$$
\begin{aligned}
L &= AZD^{-1} \\
&= (Az_1, \cdots, Az_j, \cdots, Az_n)D^{-1} \\
&= \left(\frac{Az_1}{d_1}, \cdots, \frac{Az_j}{d_j}, \cdots, \frac{Az_n}{d_n}\right) \qquad (8)
\end{aligned}
$$

となり, 行列 L の要素 l_{ij} は次のように表せる. ただし, a_j^T は行列 A の第 j 番目の行を表す.

$$
\begin{aligned}
l_{ij} &= \frac{a_j^T z_i}{d_j} \\
&= \frac{(Az_j, z_i)}{(Az_j, z_j)}, \qquad i \geq j \qquad (9)
\end{aligned}
$$

この結果, 行列 A の因子 L が, A-直交過程のいわば「副産物」として余分なコストをかけずに得られたことになる. CG 法の反復計算の中では $A = LDL^T$ の因子 L が用いられる.

2 数値実験

テスト行列は, 3 種類の行列のデータベース [5] [10] [11] に収められたものから固体力学・構造力学系の行列を選んで使用した. 計算には九州大学情報基盤センターの Compaq GS320(クロック周波数 761MHz) の 1PE を使用した.

表 1 にテストした 8 個の行列の主な仕様を示す. 表中の第 2 欄の n(上段の数字) と nnz(下段の数字) は各々行列の次元数と狭義下三角行列 L 中の非零要素の個数を表す. 演算はすべて倍精度演算で行なった. CG 法の初期解はすべて 0, 最大反復回数は 10000 回, 収束判定は相対残差 L_2 ノルム $||r_k||_2/||r_0||_2 \leq 10^{-8}$ で行なった. また, 厳密解をすべて 1 として右辺項を定めた. 格子の順番つけは通常の" 自然なオーダリング" よりもいい結果が出た逆 Cuthill-McKee オーダリングの結果を示す. 表 1 の中で Diag. は対角スケーリングの, IC(0) は fill-in を考慮しない不完全コレスキ分解の, 前処理つき CG 法の結果を表す. 前処理の欄で, 上段は反復回数を下段は収束までの CPU 時間を表す. "over" は反復過程の途中でオーバフローのために計算が中断されたもの, "max" は最大反復回数までで収束しなかったもの, "stag." は残差が停滞し計算が途中で中断されたものを各々表す. 同様に, 表 2 に他の 5 つの前処理の CG 法の結果を示す. 太字で表した数字は各行列のテストの中で最も計算時間が少なかったものを表す.

表 3 と表 4 には, 行列 BCSSTK25 と NASASRB における 4 つの前処理のパラメータごとの収束性能を示す. 表中で, "itr." は反復回数を, "T-c" は計算時間の合計を,"pre-c" は前処理にかかった時間を, そして "itr-c" は CG 法の反復計算にかかった時間を各々表す. 表 5 に CG 法の収束までの計算時間が最も少ないときの棄却許容のパラメータ γ の値を示す. ただし, RIF 法において行列 Z と L に関する 2 つの棄却許容のパラメータは同じ値を使用した.

表1: テスト行列の仕様と対角スケーリング, IC(0) 分解つき CG 法の収束状況

Matrix	Source	n/nnz	Application	Diag.	IC(0)
BCSSTK17	M.M. [11]	10974 219812	Pressure vessel	2607 16.18	over
BCSSTK18	M.M.	11948 80519	Power plant	1007 1.599	596 2.315
BCSSTK25	M.M.	15439 133840	Skycraper	9388 22.10	5381 58.78
CT20STIF	Florida [5]	52329 1375396	Engine block	max	stag.
NASASRB	Florida	54870 1366097	Rocket booster	max	over
TUBE1-2	Kouhia [10]	21498 459277	Thin shell	max	9382 334.1
SMT	Kouhia	25710 1889447	Transistor	3386 173.9	over
ENGINE	Kouhia	143571 2424822	Engine head	2873 264.3	719 175.1

表2: 前処理つき CG 法の収束状況 (上段: 反復回数, 下段: CPU 時間 (sec.))

Matrix	RIC2S	RIC1	SAINV	RIF	FSAI
BCSSTK17	208 3.179	46 **2.977**	624 9.658	242 4.801	576 8.558
BCSSTK18	66 **0.513**	61 0.579	253 1.164	124 0.905	364 1.449
BCSSTK25	197 **3.87**	81 6.85	1520 12.46	711 8.18	2512 21.71
CT20STIF	752 **191.1**	872 249.8	4812 323.9	1503 211.9	2536 260.9
NASASRB	506 95.12	317 **91.25**	7100 556.72	2251 223.43	2143 223.24
TUBE1-2	505 41.77	646 44.13	3327 76.34	625 **37.12**	2230 70.89
SMT	274 **36.04**	249 59.97	763 61.32	430 41.81	850 108.54
ENGINE	120 **54.19**	89 73.65	754 149.15	335 102.68	957 203.84

表1から, 従来の対角スケーリングおよびIC(0) 分解ではこれらの問題に対して前処理の役目が十分果たされていないことがわかる. 一方, 表2から調べた5つの前処理を使って CG 法がすべて収束したことがわかる. 最も計算時間が短縮されたのは行列 TUBE1-2 の場合でちょうど 1/9 に短縮された. ただし, 行列 CT20STIF, NASASRB は対角スケーリングと IC(0) 分解は収束しなかったので比率の計算から除外した.

また, 表 3,4 から, Kaporin による RIC2S 法と Ajiz-Jennings による RIC1 法では, 棄却許容値 γ をだんだん小さくすると計算時間全体も短くなるのに対して, SAINV 法と RIF 法では計算時間が最も小さい棄却許容値 γ が存在することがわかる.

さらに, 表 5 から, CG 法の収束までの計算時間が最小のときの棄却許容 γ の値は, RIC2S 法と RIC1 法では非常に小さく 1/1000 から 1/10000 程度, 一方 SAINV 法と RIF 法では比較的大きな値で 1/10 から 1/100 程度の値であることがわかる. ここでは, 対角項の値をすべて 1 に揃えているので, 棄却許容 γ の値に関するこれらの知見は, 他の行列の性質を調べるときに有用であると思われる.

表 3：前処理 vs. 反復回数, CPU 時間 (sec.) との関係 (行列：BCSSTK25)

	RIC2S		RIC1		SAINV		RIF	
棄却許容値	itr.	pre-c	itr.	pre-c	itr.	pre-c	itr.	pre-c
γ	T-c	itr-c	T-c	itr-c	T-c	itr-c	T-c	itr-c
0.1	2386	0.08	2512	0.03	1520	0.31	1510	0.32
	17.09	17.01	18.33	18.30	12.77	12.46	12.00	11.68
0.05	1657	0.11	2082	0.04	1049	0.66	861	0.66
	13.52	13.41	19.90	19.86	13.84	13.18	8.55	7.89
0.01	872	0.41	1197	0.09	491	6.18	400	6.03
	10.83	10.42	15.69	15.60	26.02	19.83	10.74	4.70
0.001	197	1.32	623	0.32				
	5.19	3.87	11.57	11.25				
0.0001			220	0.97				
			9.28	8.30				

表 4：前処理 vs. 反復回数, CPU 時間 (sec.) との関係 (行列：NASASRB)

	RIC2S		RIC1		SAINV		RIF	
棄却許容値	itr.	pre-c	itr.	pre-c	itr.	pre-c	itr.	pre-c
γ	T-c	itr-c	T-c	itr-c	T-c	itr-c	T-c	itr-c
0.1	over	–	5814	0.40	7100	3.00	4237	3.01
			396.3	395.9	556.7	553.7	302.6	299.6
0.05	over	–	4013	0.51	6364	7.15	2811	7.15
			336.6	336.1	636.6	629.5	242.2	235.0
0.01	1967	3.49	2276	0.84	4050	69.75	1577	70.59
	190.3	186.8	237.5	236.3	1056.	986.5	252.9	182.8
0.005	1678	6.07	1928	1.18				
	195.5	189.4	235.6	234.4				
0.001	506	18.19	949	2.49				
	95.1	76.9	165.4	162.9				

表 5：CG 法の収束までの計算時間が最も少ないときの棄却許容値 γ の値

Matrix	RIC2S	RIC1	SAINV	RIF
BCSSTK17	0.01	0.0001	0.05	0.03
BCSSTK18	0.01	0.001	0.10	0.05
BCSSTK25	0.001	0.00001	0.10	0.03
CT20STIF	0.001	0.0005	0.10	0.01
NASASRB	0.001	0.0001	0.10	0.03
TUBE1-2	0.001	0.0005	0.10	0.01
SMT	0.01	0.0005	0.05	0.04
ENGINE	0.005	0.0001	0.05	0.03

図 1, 3, 5 に, 行列 BCSSTK25, NASASRB, SMT に対する, 棄却用のパラメータ γ と収束までの CG 法の計算時間との対応関係を表す. 棄却用のパラメータのない, 対角スケーリング, IC(0) 分解そして FSAI 法の結果は見やすくするために CPU 時間のところに横線を引いて表した. 横軸は棄却用のパラメータ γ の値, 縦軸は CPU 時間 (単位：秒) を表す.

同様に, 図 2, 4, 6 に 7 種類の前処理つきの CG 法の収束履歴を表す. プロットは反復 10 回おきに行なった. 横軸は CG 法の反復回数, 縦軸は相対残差の大きさを表す. 目盛は常用対数である.

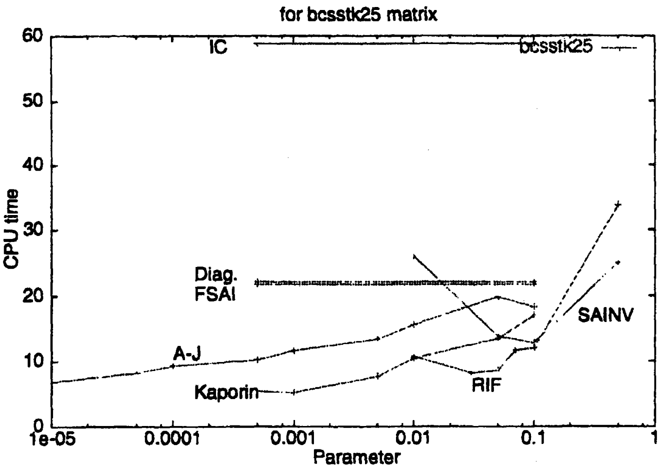

図 1: 行列：BCSSTK25 に対する CPU 時間 vs. 棄却パラメータ γ

図 1, 3, 5 の結果から, Kaporin による RIC2S 法と Ajiz-Jennings による RIC1 法では棄却用のパラメータ γ の値を小さくするに従って全体の計算時間はほぼ単調に, 特に前者は, 減少する傾向が見られることがわかる. 一方, 近似逆行列による SAINV 法と RIF 法では, 棄却用のパラメータ γ の値を小さくしすぎると全体の計算時間が逆に増加することがわかる. また, その増加の程度は RIF 法の方が緩やかであり, 実際に使用しやすいことがわかった.

また, 図 2, 4, 6 の収束履歴から, Kaporin による RIC2S 法と Ajiz-Jennings による RIC1 法の収束性が他の前処理と比較して非常に優れていることがわかる.

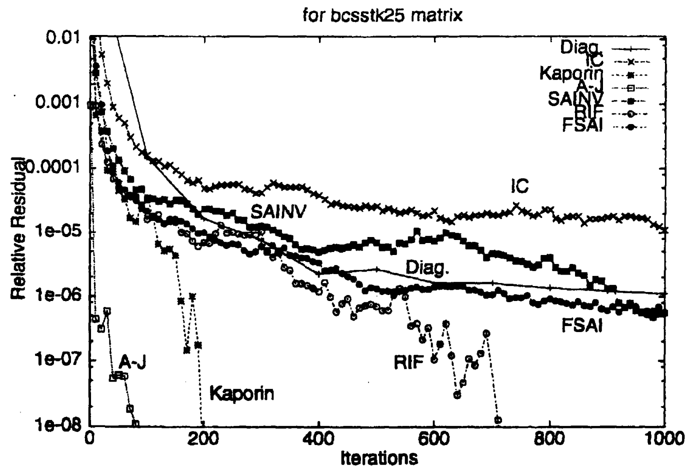

図 2: 行列 : BCSSTK25 に対する PCG の収束履歴

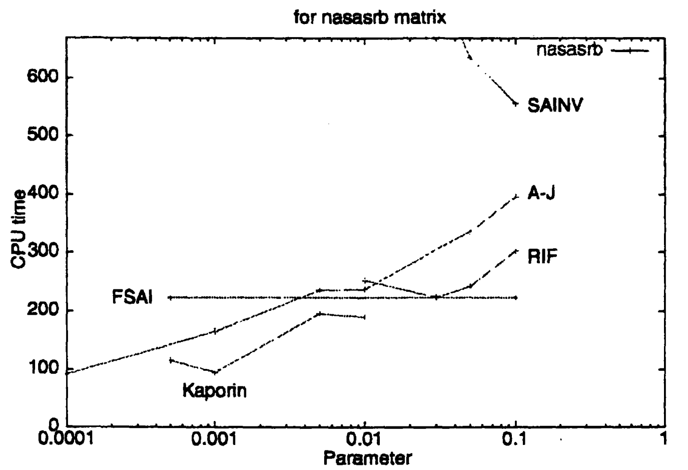

図 3: 行列 : NASASRB に対する CPU 時間 vs. 棄却パラメータ γ

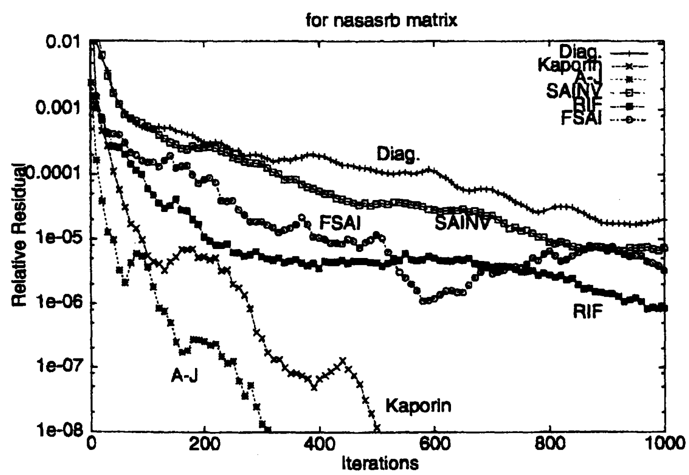

図 4: 行列 : NASASRB に対する PCG 法の収束履歴

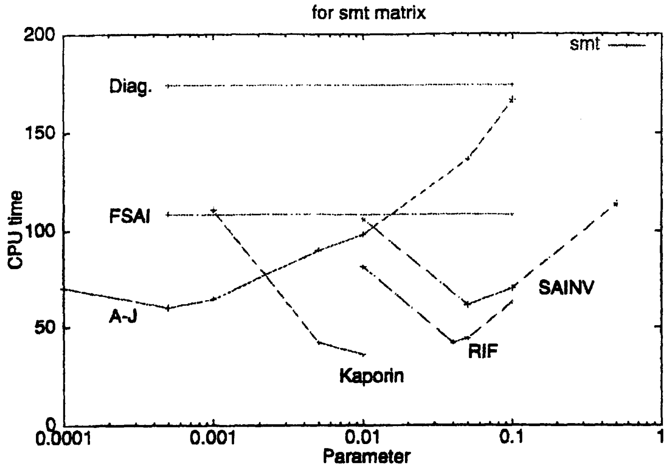

図 5: 行列：SMT に対する CPU 時間 vs. 棄却パラメータ γ

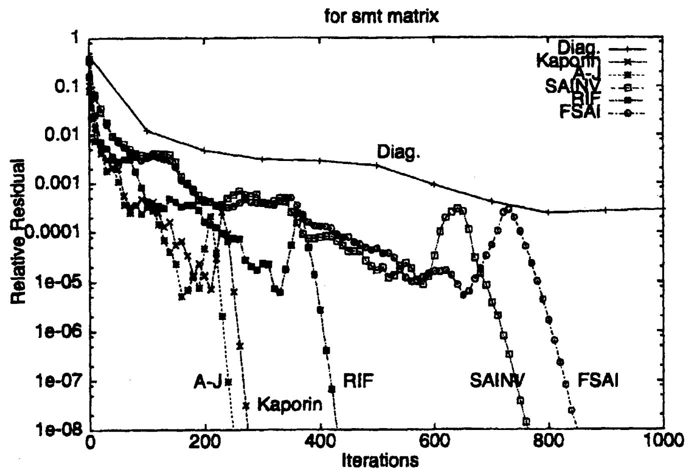

図 6: 行列：SMT に対する PCG 法の収束履歴

3 おわりに

Kaporin による RIC2S 法と Ajiz と Jennings による RIC1 法の収束性が極めて優れていることがわかった. また, 近似逆行列を利用した SAINV 法と RIF 法も収束の安定性は良好であるが, 全体の計算時間は RIC2S 法と RIC1 法に及ばず, これらについては今後研究の余地がまだあることがわかった. さらに, これらの前処理では, 全体の計算時間の中で前処理にかかる時間が占める割合が大きく, 従来のように CG 法の収束までの計算時間だけでは収束性について論じられないこともわかった.

今後は各前処理の収束性とそれに必要な作業用メモリの大きさとの関係を明らかにするとともに, メモリ節約のためのメモリの動的配置 (Dynamic Allocation) などについても検討していく予

本研究を進めるにあたり, 多くのご助言と協力を戴いた M. Benzi 助教授および M. Tuma 博士に心より感謝の意を表する.

参考文献

[1] Ajiz, MA., Jennings, A.: A robust incomplete Cholesky-conjugate gradient algorithm, Int. J. for Numerical Methods in Engineering, 20(1984), 949-966.

[2] Benzi, M., Meyer, CD., Tuma, M.: A Sparse Approximate Inverse Preconditioner for the Conjugate Gradient Method, SIAM 17(1996), 1135-1149.

[3] Benzi, M., Cullum, JK., Tuma, M.: Robust approximate inverse preconditioning for the conjugate gradient method, SIAM J. on Scientific Computing, 22(2000), 1318-1332.

[4] Benzi, M., Tuma, M.: A robust incomplete factorization preconditioner for positive definite matrices, Numerical Linear Algebra with Applications, 99(2001), 1-20.

[5] University of Florida Sparse Matrix Collection Web Page: http://www.cise.ufl.edu/research/sparse/matrices/

[6] 藤野清次: Tismenetsky-Kaporin による不完全分解について–CG 法の前処理の現状報告 (その 2)–, 応用数学合同研究集会報告集, 龍谷大学瀬田キャンパス, 2002.12.19-21, 147-152.

[7] Hestenes, MR., Stiefel, E.: Methods of conjugate gradients for solving linear systems, J. of Research of the National Bureau of Standards, 49(1952), 409-436.

[8] Kolotilina, L.Yu., Yeremin, A.Yu.: Factorized sparse approximate inverse preconditioning I, Theory, SIAM J. on Matrix Analysis and Applications, 14(1993), 45-58.

[9] Kaporin, IE.: High quality preconditioning of a general symmetric positive definite matrix based on its $U^TU + U^TR + R^TU$-decomposition, Numerical Linear Algebra with Applications, 5(1998), 483-509.

[10] Helsinki University of Technology, Kouhia R., Sparse Matrices Web Page: http://www.hut.fi/~kouhia/sparse.html

[11] Matrix Market Web Page: http://math.nist.gov/MatrixMarket/

[12] Meijerink, JA., van der Vorst, HA.: An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, Mathematics of Computation, 31(1977), 148-162.

[13] Tismenetsky, M.: A new preconditioning technique for solving large sparse systems, Linear Algebra and its Applications, 154-156(1991), 331-353.