数理解析研究所講究録
第514巻 1984年 110-134

ベクトル計算機向きＩＣＣＧ法

日立・ソフト　後　保範（Yasunori Ushiro）

1. はじめに

疎な対称正定値行列を係数とする連立一次方程式の反復解法において，ICCG法（Incomplete Cholesky Conjugate Gradient Method）は古典的なCG法やSOR法より収束が格段に良いことが知られてきた[1)2)]。最近さらにICCG法を改良したMICCG法[3)4)]（Modified ICCG法）が注目されてきている。一方高速計算の必要性からベクトル計算機が出現してきた。このため，ICCG法の計算をベクトル計算機向きにするためのアルゴリズムの検討[5)6)7)]がなされている。しかし，いずれも元のICCG法より収束速度が低下するという傾向がある。

ここ数年，収束速度を低下させないベクトル計算機向きICCG法を検討してきた。その結果，リストベクトルを使用すると同一な収束速度を保ったままベクトル化できることが分った。この方法はMICCG法にも疎な非対称行列用の解法

(不完全三角分解と組み合わせたBCG法[14)] [Bi Conjugate Gradient Method]やCR法[15)] [Conjugate Residual Method])にもそのまま応用できる。ここでは本検討結果のICCG法とMICCG法について報告する。

数値実験は三次元直方体領域における拡散方程式を7点中央差分公式で離散化した行列を対象に行った。ベクトル計算機としてHITAC S-810 モデル20[12)]を使用した。すでに提案されているベクトル計算機向きICCG法に類似な解法についても数値実験をしたので同時に報告する。

今回提案したベクトル計算機向きICCG法は有限要素法による離散化などで多く発生する不規則的スパース行列にも適用できる。ここではその計算方法についても報告する。

2. 反復解法の収束の速さの比較

ベクトル計算機向き解法の収束の評価をする前に，ICCG法及びMICCG法と古典的CG法，SOR法及びSLOR法の収束の速さを比較した。その結果を以下に示す。

収束の速さの比較には(2.1)式に示す三次元拡散方程式を中央差分で離散化して得られる行列を使用した。計算対象領域は直方体領域の左下奥1/8領域である。全体領域の表面は0に固定した境界とし，1/8領域への切断面は拡散による移動のない境界(Γ_2)とする。拡散係数と形状の収束性への影響を

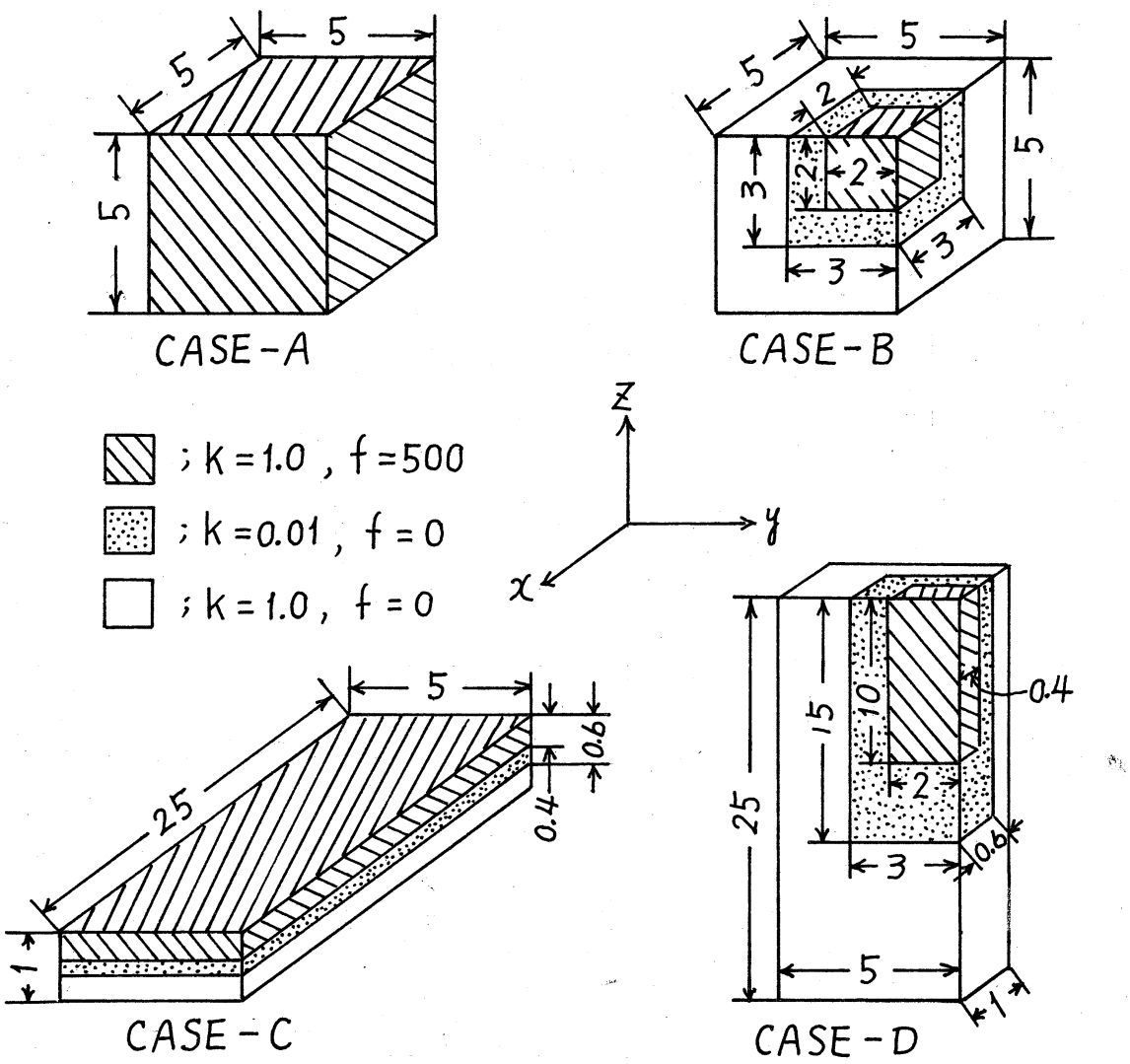


図 2.1　三次元熱伝導モデル

各モデルとも下記の拡散方程式及び境界条件に従う。

$$-div(k \cdot grad\,\phi) = f \qquad (2.1)$$

$$\left.\begin{aligned} \phi &= 0 && \text{on } \Gamma_1 \\ (k \cdot grad\,\phi)\cdot n &= 0 && \text{on } \Gamma_2 \end{aligned}\right\} \text{境界条件} \qquad (2.2)$$

考慮して計算対象モデルは図2.1に示すCASE-A, CASE-B, CASE-C及びCASE-Dの4種類のモデルを選定した。各モデルは直方体領域の1/8領域に対応するもので境界条件はすべて同一とする。

各モデルとも20x20x20に等分割し，中央差分公式を使用して次元数8000の次のような連立一次方程式を作成する。

$$A x = b \tag{2.3}$$

未知数となる節点の番号はx方向，y方向，z方向の順に付けた。x^{ν}をν回反復計算時の解ベクトルとするとき，解の収束を判定する基準として次式で表現される2乗ノルムの相対残差を使用した。

$$\text{相対残差} = \frac{\|A x^{\nu} - b\|_2}{\|b\|_2} \tag{2.4}$$

反復計算の初期値は$(0,0,\cdots,0)^T$なるゼロベクトルとした。CASE-Aのモデルにおけるmiccg法，iccg法，cg法，sor法，及びslor法の反復回数に対する収束の状態を図2.2に示す。同様にCASE-B, CASE-C及びCASE-Dに対するものをそれぞれ図2.3，図2.4及び図2.5に示す。cg法及びiccg法は加速係数を必要としない。miccg法の加速係数αは全モデルともα=0.975を使用した。一方sor法及びslor法は各モデルごとに加速係数αを与え，その値はYoung-Frankel[8]の理論

から(2.5)式で計算した。

$$\left.\begin{aligned} \alpha &= 1+(1-\sqrt{1-\lambda})/(1+\sqrt{1-\lambda}) \\ \omega_i &= \|A x^i - b\|_2 / \|b\|_2 \quad (i=\ell-3,\ell-2,\ell-1,\ell) \\ \lambda_i &= \omega_i / \omega_{i-1} \quad (i=\ell-2,\ell-1,\ell) \\ r &= (\lambda_\ell-\lambda_{\ell-1})/(\lambda_{\ell-1}-\lambda_{\ell-2}) \\ \lambda &= \lambda_\ell+(\lambda_\ell-\lambda_{\ell-1})\cdot r/(1-r) \end{aligned}\right\} \quad (2.5)$$

ℓはここでは25とした。

図2.2から図2.5のSOR法とSLOR法の加速係数αは$\alpha=1.0$でℓ回(ここでは25回)計算し，それ以後(2.5)式で算出したαを使用した。図2.6にSOR法の加速係数を算出した値の近傍で変化させた場合の相対残差を示した。同様に図2.7にSLOR法の相対残差を示す。両図ともαは0.02単位に変化させて測定したものである。これからSOR法及びSLOR法ともに(2.5)式で算出したαが最適加速係数に近いものであることが分かる。図2.8にMICCG法の加速係数αを変化させ相対残差が10^{-10}以下となる反復回数を示した。このときMICCG法は(3.2)式で示す方式のものを使用した。αを1/40単位に変化させると，各モデルとも$\alpha=0.95$か0.975で最適な加速係数となった。そこで今回は全モデルとも$\alpha=0.975$を採用した。

図2.2から図2.5よりMICCG法は4種類のモデルのすべてで，他解法より収束が速くかつ安定した解法であることが分かる。

全ケースとも 20x20x20 分割

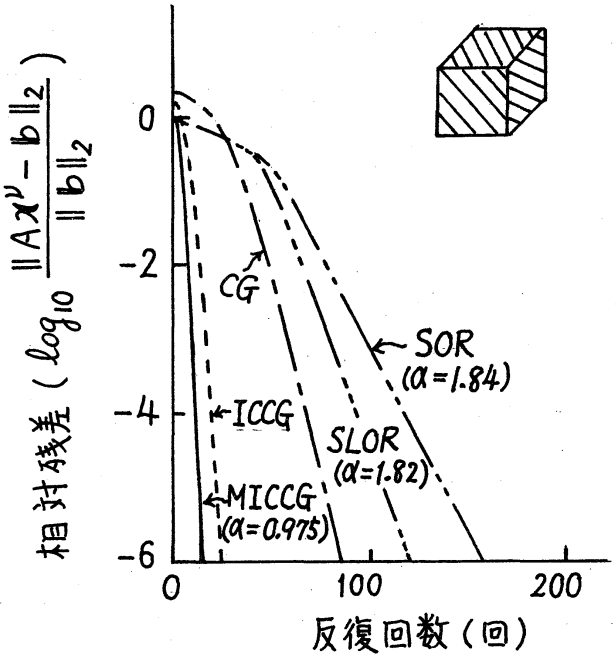


図2.2 CASE-Aの収束状況

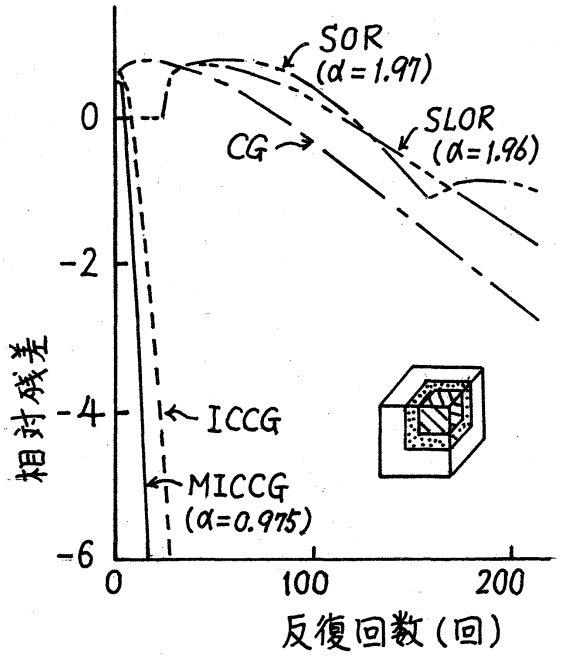


図2.3 CASE-Bの収束状況

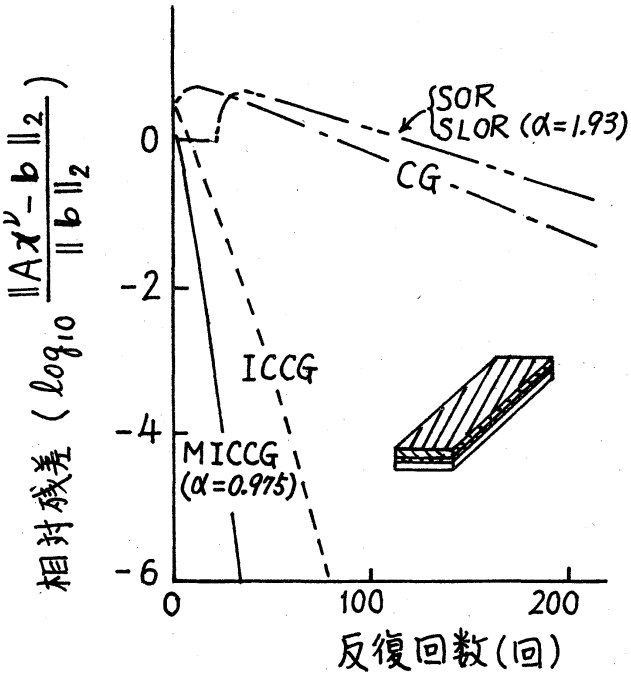


図2.4 CASE-Cの収束状況

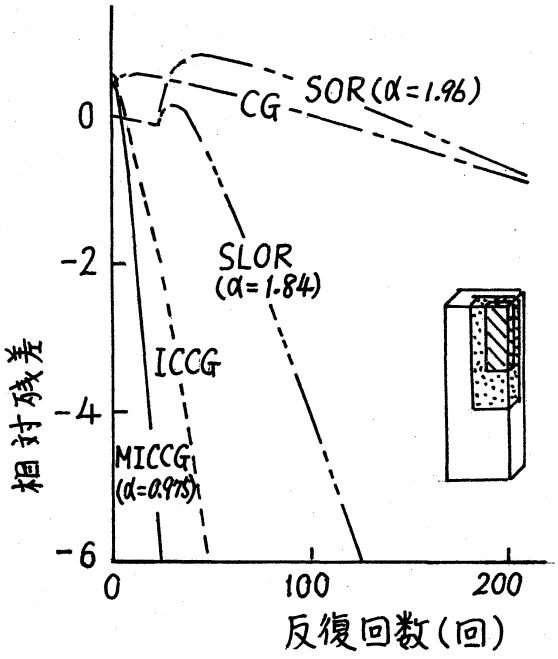


図2.5 CASE-Dの収束状況

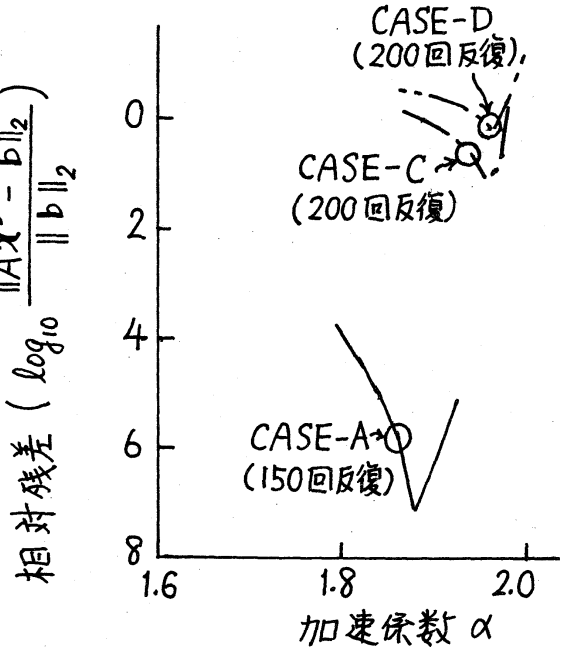


図2.6 SOR法の加速係数の影響

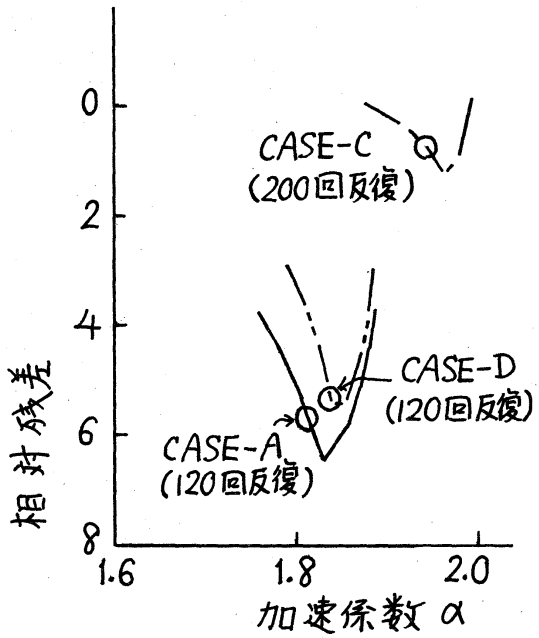


図2.7 SLOR法の加速係数の影響

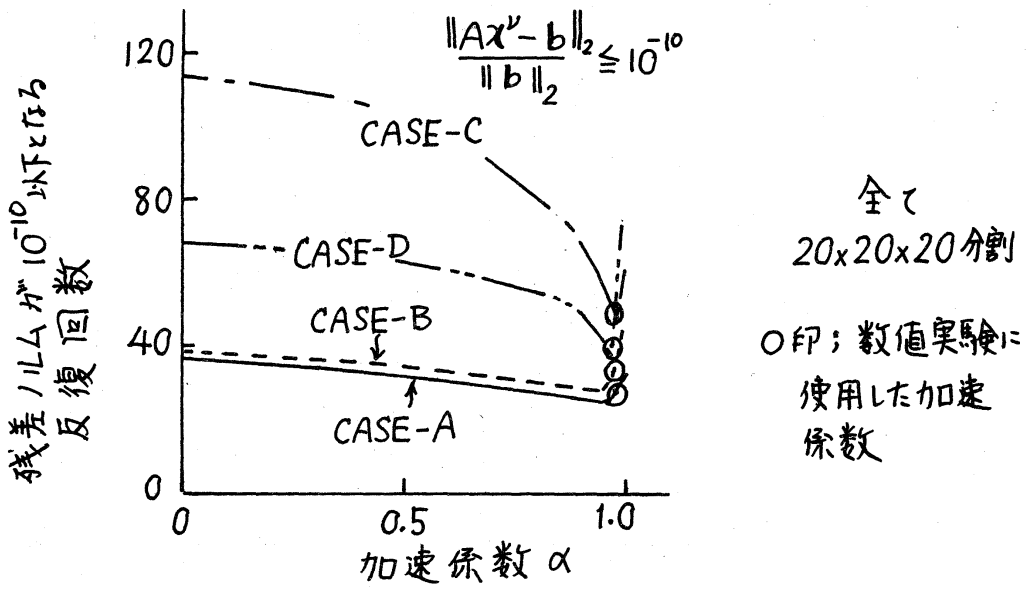


図2.8 MICCG法の加速係数の影響

3. 計算方法

3.1 ICCG法とMICCG法の計算方法

ICCG法（Incomplete Cholesky Conjugate Gradient Method）とMICCG法（Modified ICCG法）の計算方法を図3.1に示す。ICCG法とMICCG法が異なるのは行列AをLDL^{T}の形に不完全三角分解する部分だけである。行列Aの形を図3.2に，非ゼロ要素の形を変えないで不完全三角分解した後の下三角行列Lの形を図3.3に示す。不完全三角分解後の対角行列Dの要素を$\{dd_i\}$とする。ICCG法の不完全LDL^{T}分解の計算を(3.1)式に，MICCG法の計算を(3.2)式及び(3.3)式に示す。

ICCG法の不完全 LDL^T 分解

$$dd_i = 1/(d_i - c_i^2 \times dd_{i-1} - b_i^2 \times dd_{i-m_1} - a_i^2 \times dd_{i-m_2}) \qquad (3.1)$$

MICCG法の不完全 LDL^T 分解

〈方式1 --- 加速係数 α 〉

$$\begin{aligned} dd_i = 1/\{ & d_i - c_i \times dd_{i-1} \times [c_i + \alpha \times (a_{i+m_2-1} + b_{i+m_1-1})] \\ & - b_i \times dd_{i-m_1} \times [b_i + \alpha \times (c_{i+1-m_1} + a_{i+m_2-m_1})] \\ & - a_i \times dd_{i-m_2} \times [a_i + \alpha \times (b_{i+m_1-m_2} + c_{i+1-m_2})]\} \end{aligned} \qquad (3.2)$$

〈方式2 --- 加速係数 σ 〉

$$\begin{aligned} dd_i = 1.0/\{ & (1+\sigma) \times d_i - c_i \times dd_{i-1} \times (c_i + a_{i+m_2-1} + b_{i+m_1-1}) \\ & - b_i \times dd_{i-m_1} \times (b_i + c_{i+1-m_1} + a_{i+m_2-m_1}) \\ & - a_i \times dd_{i-m_2} \times (a_i + b_{i+m_1-m_2} + c_{i+1-m_2})\} \end{aligned} \qquad (3.3)$$

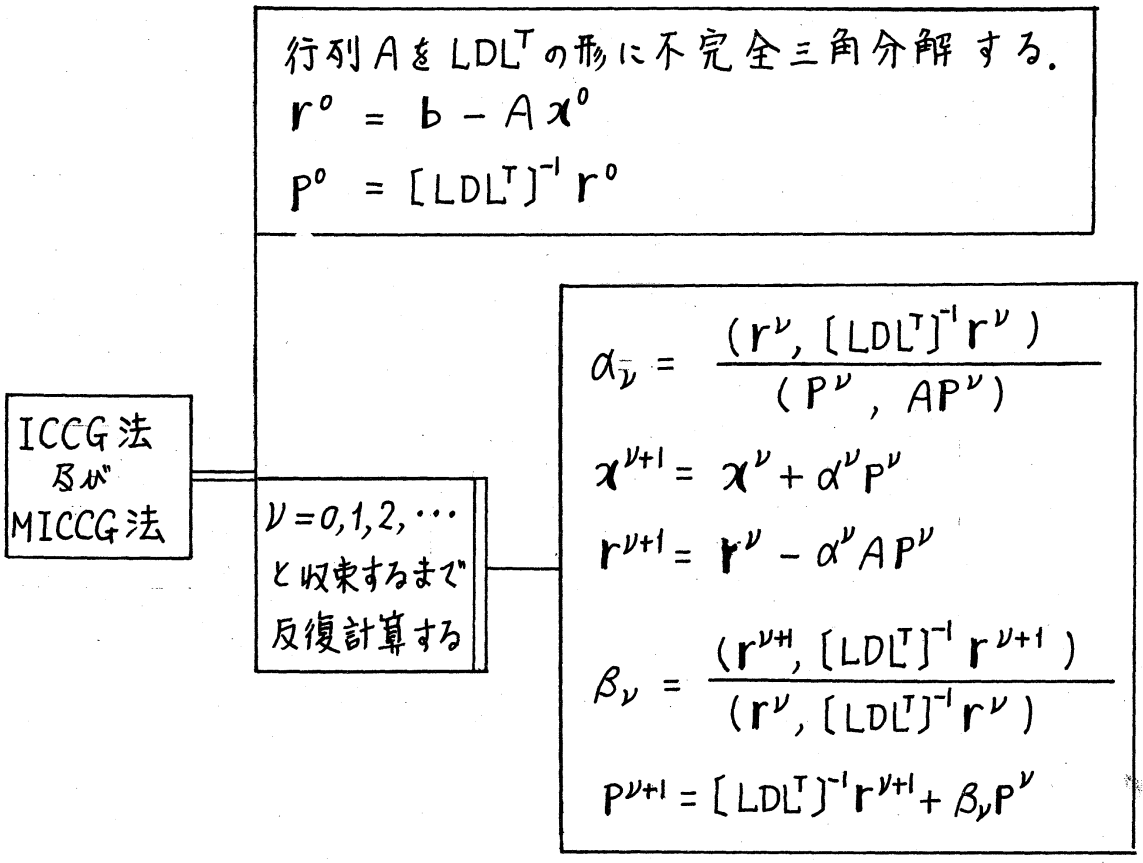


図3.1 ICCG法及びMICCG法の計算方法

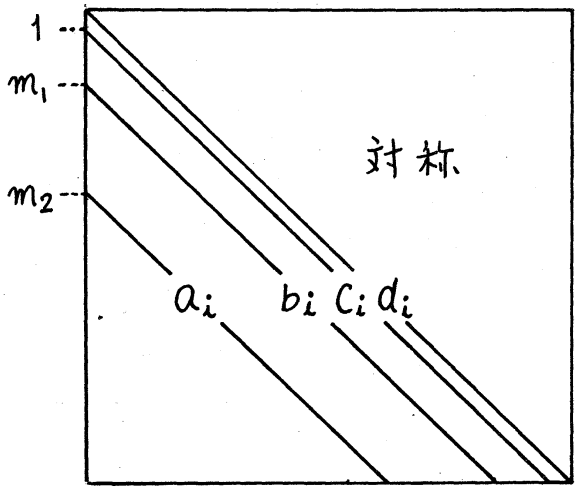


図3.2 行列Aの形

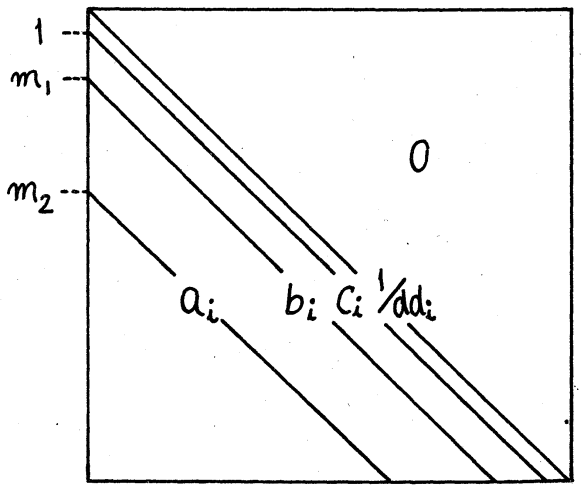


図3.3 分解後の行列Lの形

今回の数値実験では方式1のMICCG法を使用した。その理由は加速係数を一定($\alpha=0.975$)にして各モデルとも良好な収束を示したためである。

3.2 今回提案のベクトル計算機向き方式

CG法はベクトル計算機向き解法であるが，不完全三角分解と組み合わせたICCG法及びMICCG法は通常下記の二つの計算がベクトル計算に不適となる。

(a) 行列の不完全三角分解。(LDL^Tと表す)

(b) 不完全三角分解を使用した前進及び後退代入

($g=[LDL^T]^{-1}r$と表す)

上記二つの計算は通常の方法でプログラムすると，一つ前で計算した値を使用するというデータの依存関係があり，ベクトル計算機に適さない。即ち三次元の拡散方程式を離散化した行列では図3.4の関係がある。

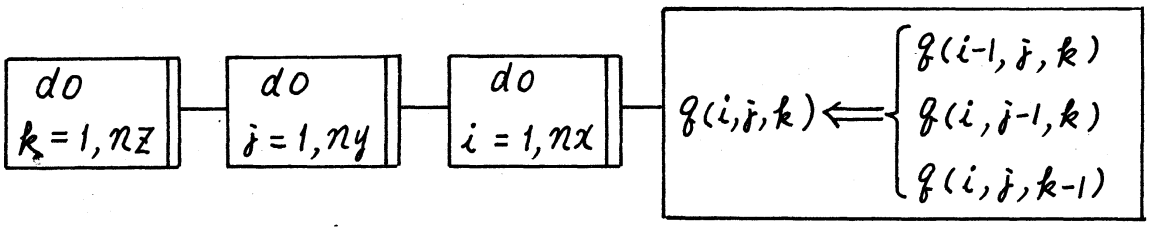


図3.4 不完全三角分解と$g=[LDL^T]^{-1}r$の計算におけるデータの依存関係

しかし，リストベクトルを使用して$i+j+k=3$から$n_x+n_y+n_z$まで$i+j+k$の値が一定となるものをまとめて計算する方法

を採用すると次のことが分かる。

(a) $g(i,j,k)$ を計算するための $g(i-1,j,k)$, $g(i,j-1,k)$ 及び $g(i,j,k-1)$ はすでに計算済みの値となりデータの依存関係がなくなり，ベクトル計算に適する。

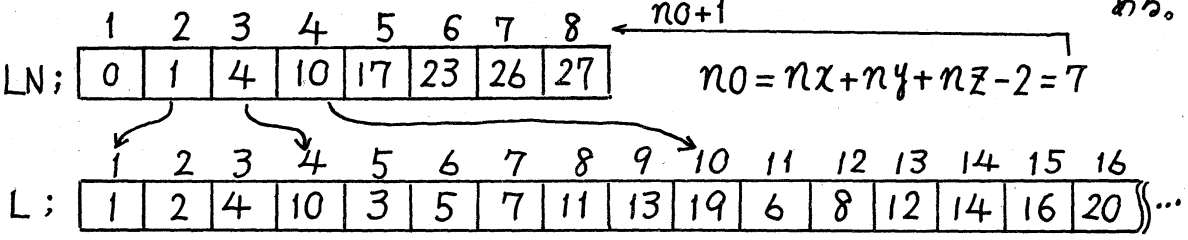
(b) $g(i-1,j,k)$, $g(i,j-1,k)$ 及び $g(i,j,k-1)$ の値の計算が完了後 $g(i,j,k)$ を計算するという関係は保持しているため収束速度は変化しない。

図3.5 にMICCG法の方式1を本方法でベクトル計算機向きにした場合の不完全三角分解を示す。図3.6 にICCG法及びMICCG法に共通なベクトル計算機向き $g=[LDL^T]^{-1}r$ の計算方法を示す。二次元問題と三次元問題の図3.5と図3.6の計算における平均ベクトル長は次のようになる。

$$\text{平均ベクトル長(二次元)} = \frac{nx \times ny}{nx+ny-1} \qquad (3.4)$$

$$\text{平均ベクトル長(三次元)} = \frac{nx \times ny \times nz}{nx+ny+nz-2} \qquad (3.5)$$

図3.5と図3.6に使用するリストベクトルは三次元問題で $nx=ny=nz=3$ とした場合の具体例を示すと下記のようになる。このとき，$m1=3$, $m2=9$ である。ここで nx, ny, nz は分割数である。



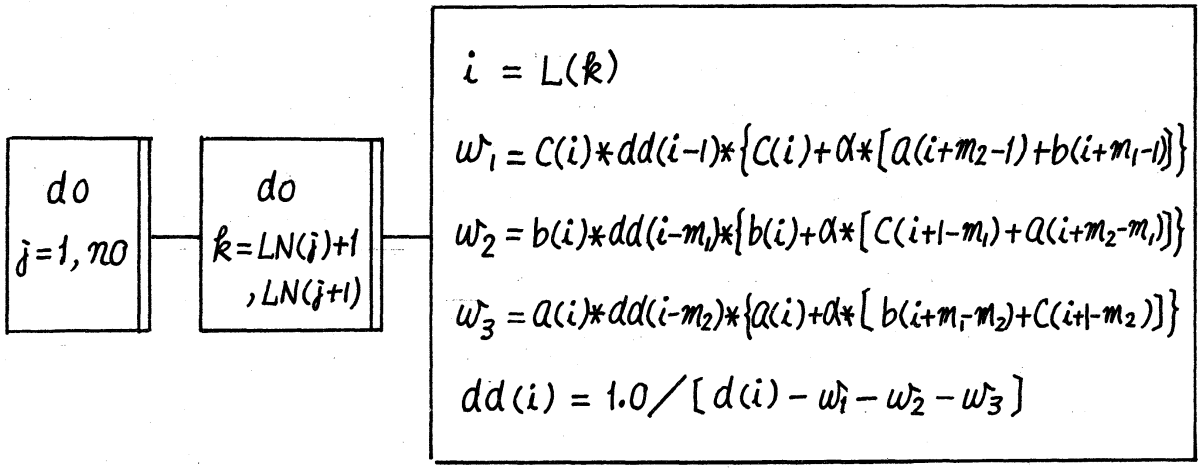


図 3.5　ベクトル計算機向き不完全三角分解（MICCG法の方式1）

注）二次元問題ではaの項目が不要

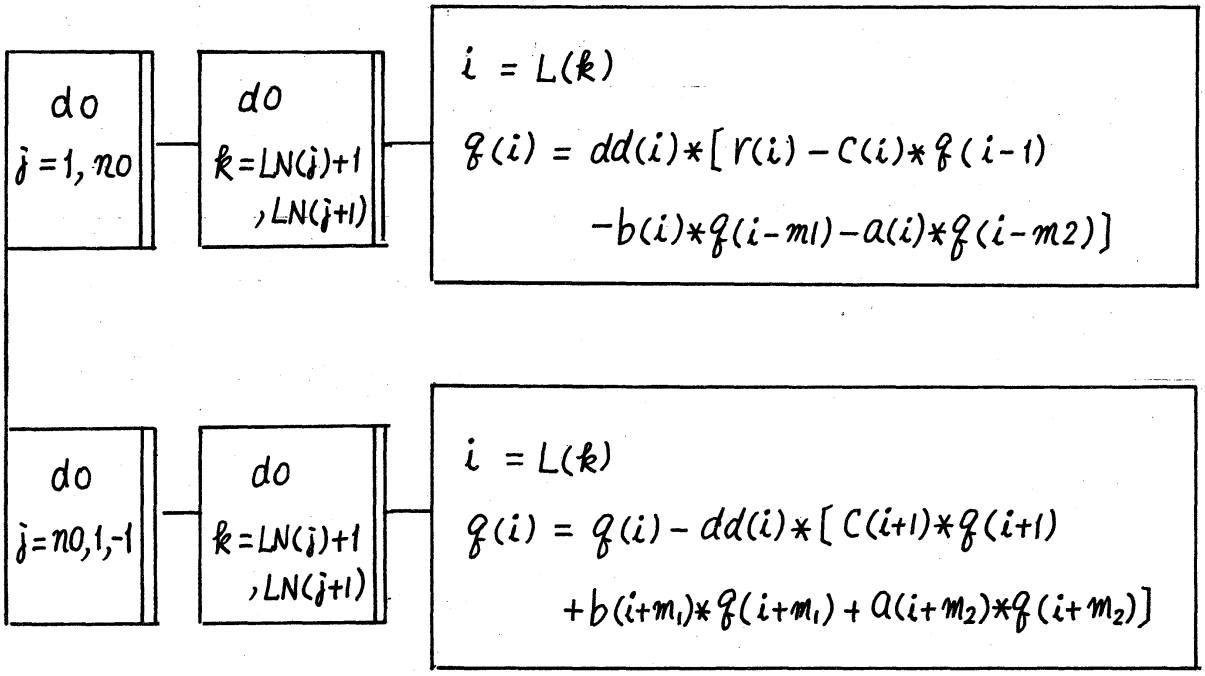


図 3.6　ベクトル計算機向き $q=[LDL^T]^{-1}r$ の計算

注）二次元問題ではaの項が不要

3.3 他のベクトル計算機向き解法

ベクトル計算機向きICCG法に類似な解法としてJohnson & Paulの提案した方法[5]とVan Der Vorstの提案した方法[6]を簡単に紹介する。

(1) Johnson & Paulの方法

IICGJ法(Incomplete Inverse Conjugate Gradient Jacobi Method)と名付ける。

連立一次方程式 $Ax=b$ に対して $A=D^{1/2}(I-G)D^{1/2}$, $z=D^{1/2}x$, $h=D^{-1/2}b$ を代入して整理すると次式を得る。

$$(I-G)z=h$$

そこで $(I-G)^{-1}$ の近似行列として $I+G+G^2$ をPreconditionerに使用する方法である。加速パラメータとしてu, v, wを使用して $u\cdot I+vG+w\cdot G^2$ とできるが, 数値実験はすべて1.0とした。

(2) Van Der Vorstの方法

不完全LDL^T分解は通常の方法と同一でありICCG法に対応する方法をICCG*とし, MICCG法に対応するものをMICCG*法とする。不完全LDL^T分解した行列Lを使用した $Ly=r$ の計算で

$L=I-C-F$ と置く。 ここでCは図3.3の要素C_iよりなる行列。各ブロックごとに下記のような計算をする。対角を1にスケール変更する。

$$\begin{aligned} y_\ell &= (I-C_\ell)^{-1}(r_\ell+F_\ell\cdot y_{\ell-1}) \\ &\fallingdotseq (I+C_\ell)(I+C_\ell^2)(r_\ell+F_\ell\cdot y_{\ell-1}) \end{aligned} \qquad (3.6)$$

4. 数値実験結果

今回提案したリストベクトルを使用して元と同一の収束を保つICCG法をICCG，MICCG法をMICCGと呼ぶ。Johnson & Paulの方法[5]をIICGJと呼ぶ。Van Der Vorstの方法[6]でICCG法に対応するものを$ICCG^*$，MICCG法に対応するものを$MICCG^*$と呼ぶ。収束の速さの比較には図2.1の三次元熱伝導モデルを使用した。各モデルとも30x30x30に等分割し，中央差分公式を使用して次元数27000の連立一次方程式を作成した。未知数となる節点の番号はx方向，y方向，z方向の順に付けた。解の収束を判定する基準として(2.4)式の2乗ノルムの相対残差を使用した。MICCG及び$MICCG^*$の加速係数は(3.2)式で示す方式を採用し，全モデルとも加速係数αは0.975を使用した。また反復計算の初期値は$(0,0,\cdots,0)^T$なるゼロベクトルとした。

図2.1のCASE-Aのモデルにおける，MICCG，ICCG，$MICCG^*$，$ICCG^*$及びIICGJの反復回数に対する収束の状態を図4.1に示す。同様にCASE-B，CASE-C及びCASE-Dに対するものをそれぞれ図4.2，図4.3及び図4.4に示す。いずれのモデルでもMICCGが収束が最も速くかつ安定した解法であることが分かる。ICCGはいずれのモデルでも収束が安定しているが，反復回数はMICCGの約2倍必要となる。CASE-AとCASE-Bに

全ケースとも 30x30x30 分割

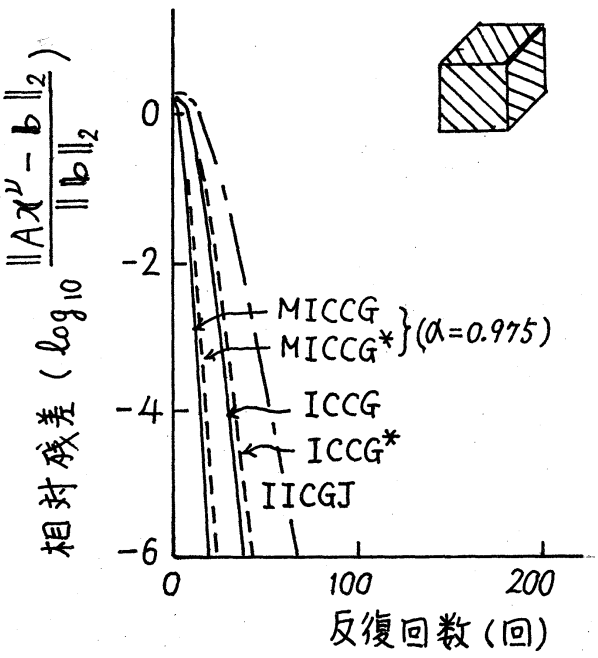


図4.1 CASE-Aの収束状況

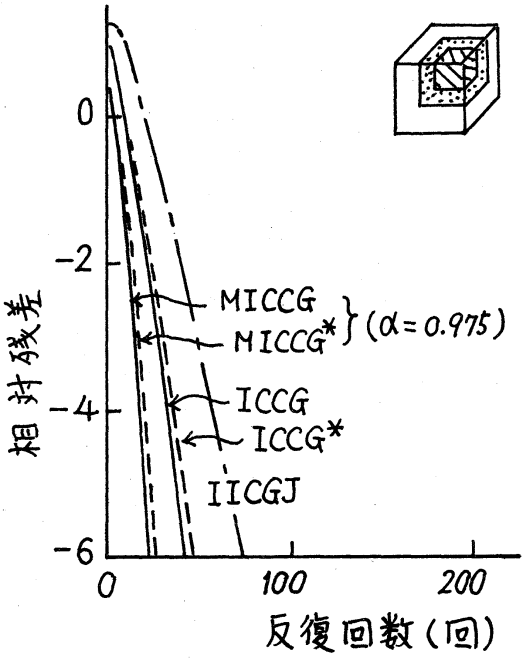


図4.2 CASE-Bの収束状況

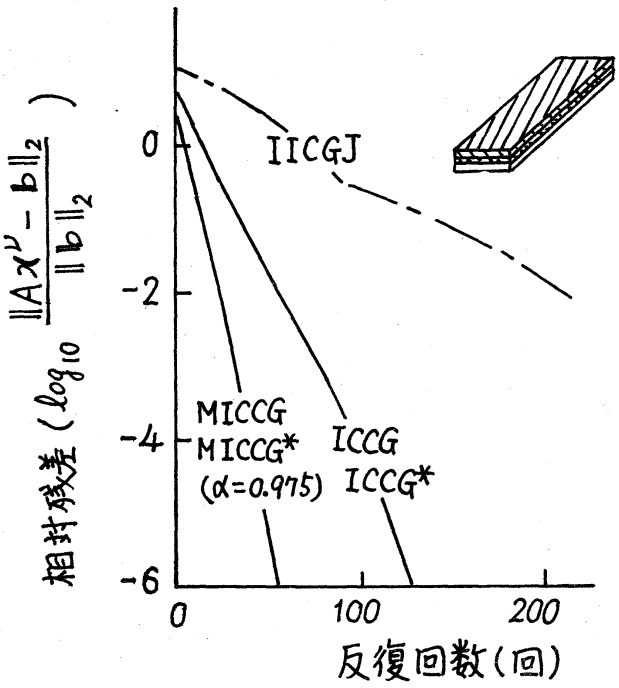


図4.3 CASE-Cの収束状況

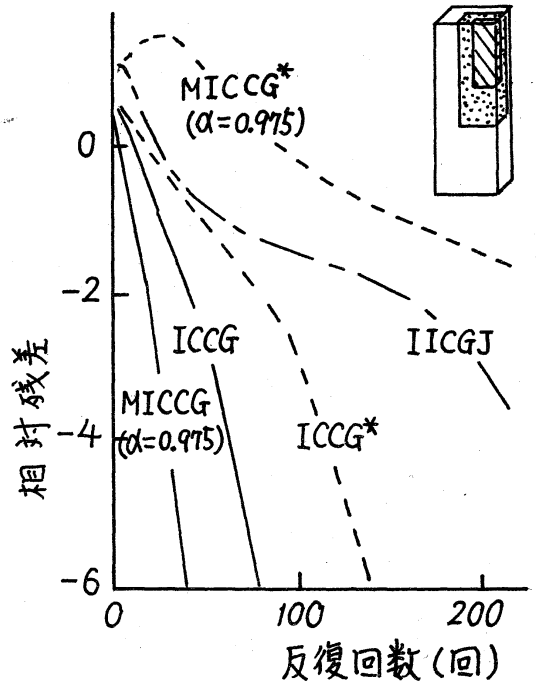


図4.4 CASE-Dの収束状況

おいてMICCG*はMICCGに，ICCG*はICCGに近い収束をしている。CASE-Cにおいては MICCG* はMICCGとICCG*はICCGと同一の収束をする。しかし，CASE-Dにおいて様子が一変する。ICCG*はICCGに対し約2倍収束が遅いだけであるが，MICCG*は収束が非常に悪い。その原因は(3.6)式で $(I-C_e)^{-1}$ を $(I+C_e)(I+C_e^2)$ と近似したときの近似度がCASE-Dのモデルでは悪いためである。即ち対角要素とその一つ下の C_i の要素の大きさが近いためである。この要素の値の近さは(3.2)式の方式1で不完全 LDL^T 分解するとさらに高まり，MICCG*の収束がICCG*より悪くなっているものと思われる。このCASE-Dのモデルのような場合は(3.3)式の方式2で加速係数 σ を適当に大きくするとよいと思われるが，今回は数値実験をしていない。いずれにしてもMICCG*では通常のMICCG法とは加速係数を変化させなければならないという問題が発生する。

IICGJも加速係数を与えることで収束を速くすることができるが今回の数値実験では加速係数なしのもので測定した。そのためIICGJは加速係数を持たないICCGと比較評価するのが妥当である。CASE-A及びCASE-BではIICGJの収束はICCGに対し約2倍遅い。CASE-CとCASE-DではIICGJの収束はICCGに対し約4倍遅い。

次にベクトル演算の効果を見るために，ベクトル計算機

HITAC S-810 モデル20と汎用計算機 HITAC M-280H でのCPU時間の比較をした。30x30x30 分割で反復回数50回でのS-810のM-280Hに対する計算速度向上比率は次の通りである。

(a) MICCG 及び ICCG --- 約 50 倍

(b) MICCG* 及び ICCG* --- 約 20 倍

(c) IICGJ ---- 約 80 倍

この差の原因は $[LDL^T]^{-1}r$ に関連する計算の平均ベクトル長が異なるためである。またMICCG* とICCG* は不完全 LDL^T 分解計算の一部がベクトル化できないことも原因している。

30x30x30分割したとき $[LDL^T]^{-1}r$ に関連する計算の平均ベクトル長は次の通りである。

(a) MICCG 及び ICCG --- 約 300

(b) MICCG* 及び ICCG* --- 30

(c) IICGJ ---- 27000

HITAC S-810 モデル 20 のCPU時間を横軸にした収束状況のグラフを次に示す。図2.1 のCASE-Aのモデルにおける MICCG, ICCG, MICCG*, ICCG* 及び IICGJ の計算時間に対する収束の状態を図4.5 に示す。同様にCASE-B, CASE-C, CASE-D に対するものをそれぞれ図4.6, 図4.7 及び図4.8 に示す。

ベクトル計算機のCPU時間で比較するとMICCGが最も収束が速くかつ安定した解法であることが分かる。次いでICCGが速い。

全ケースとも 30×30×30 分割（S-810 モデル20）

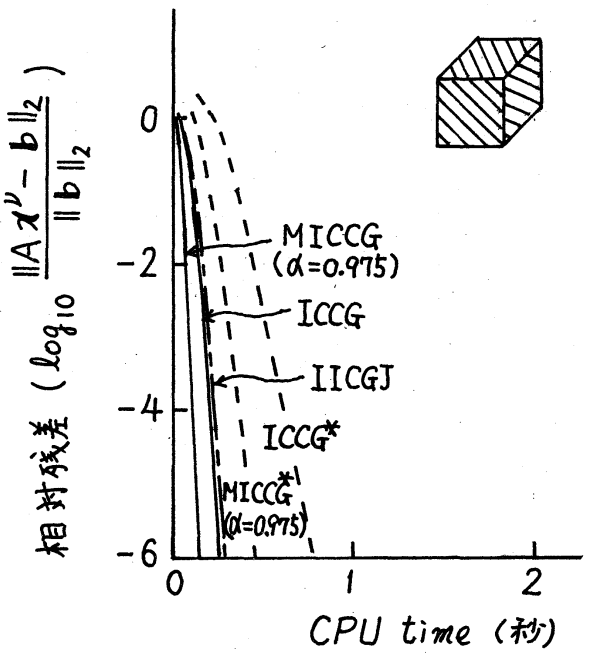


図4.5 CASE-Aの計算時間

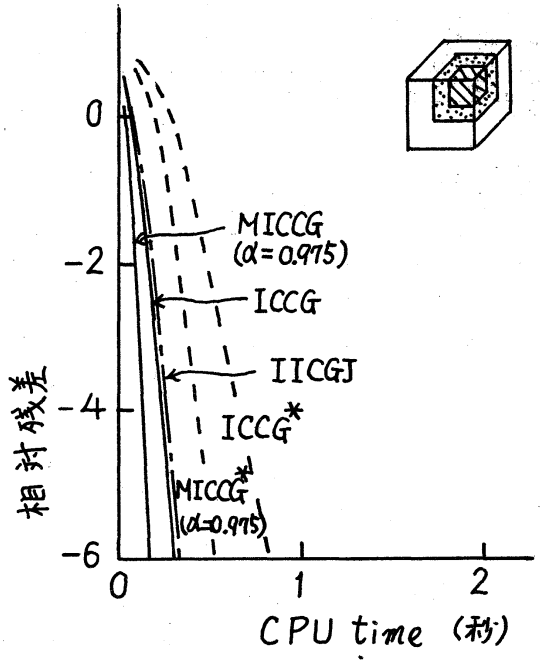


図4.6 CASE-Bの計算時間

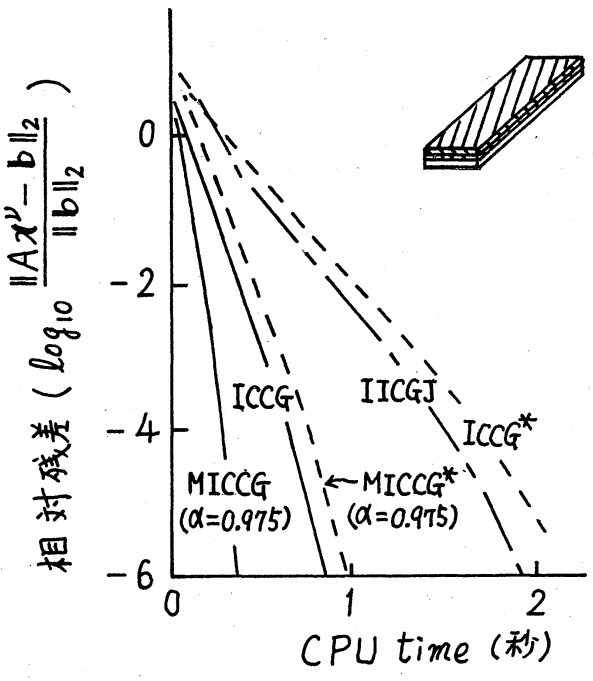


図4.7 CASE-Cの計算時間

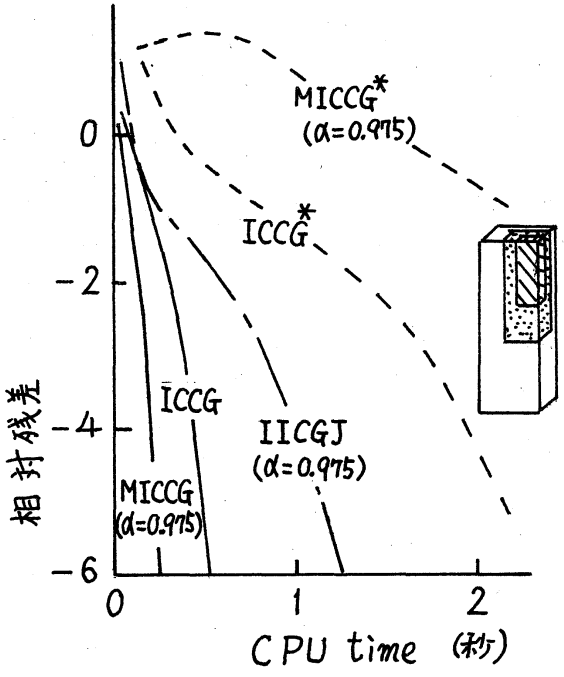
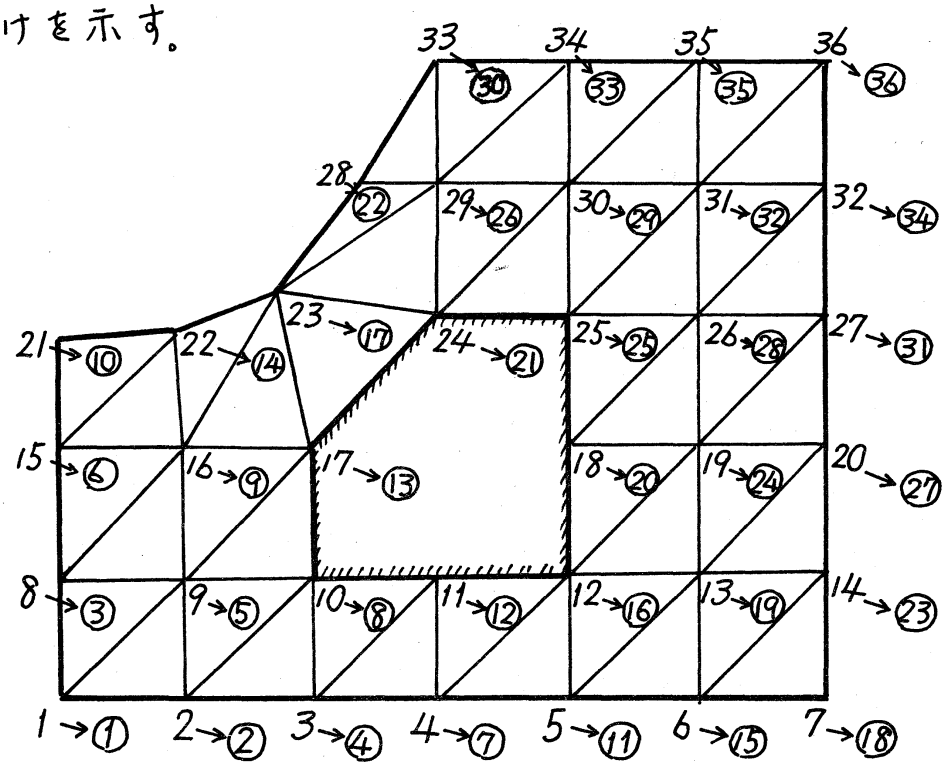


図4.8 CASE-Dの計算時間

5. 不規則的スパース行列のICCG法

汎用計算機向きICCG法と収束が同一なベクトル計算機向き計算方法[9]を示す。不完全LDL^T分解における行列Lの非ゼロ要素の位置は元の下三角部分の非ゼロ要素の位置と同じものとする。この計算はA·P及び$[LDL^T]^{-1}r$の計算をベクトル演算で高速処理することが問題となる。A·Pの計算はデータの独立性があるためベクトル長を長くする工夫だけすればよい。ここでは$[LDL^T]^{-1}r$のベクトル化方式について示す。図5.1に不規則的スパース行列を発生させる有限要素分割と節点の番号付けを示す。



注：1, 2, 3 --- オリジナル番号付け
①, ②, ③ --- ベクトル計算機向き番号付け

図5.1 有限要素分割と節点の番号付け

オリジナル番号付けからベクトル計算機向き番号付けへの変換は，プログラムで次の条件のもとで実行する。

(a) 直接結合する節点の番号の大小関係はオリジナルな番号とベクトル計算機向きの番号で一致させる。

(b) 連続する番号の節点ができるだけ直接結合しないようにベクトル計算機向き番号を付ける。

(a)はICCG法の収束率を変化させないための条件であり，(b)はベクトル長をできるだけ長くするための条件である。

図5.2にオリジナルの番号付けで作成したスパース行列の非ゼロ要素の列番号テーブルを示す。図5.3に上記条件のもとで作成したスパース行列の非ゼロ要素の列番号テーブルを示す。同図のブロック番号⑦を見ると，下三角テーブルLLは11から17の番号で構成されていることが分かる。すなわち，ブロック⑦の行番号に対応する18番から22番の未知数の計算は既に計算が完了している11番から17番の値を使用して実行できる。この関係を利用して，不規則的スパース行列の不完全LDL^T分解と$[LDL^T]^{-1}r$の計算がベクトル化できる。

オリジナルの番号付けからベクトル計算機向き番号付けへの変換経過を図5.4に示す。まず新旧番号対応テーブルの1番に1をセットする。図5.2の上三角LUテーブルより行番号1に対応する番号(2,8,9)の集合を取り出し図5.4のNW

1		
2		
3		
4		
5		
6		
1		
1	2	8
2	3	9
3	4	10
4	5	11
5	6	12
6	7	13
8		
8	9	15
9	10	16
12		
12	13	18
13	14	19
15		
15	16	21
16	17	22
17	23	
18	24	
18	19	25
19	20	26
23		
23	24	28
24	25	29
25	26	30
26	27	31
28	29	
29	30	33
30	31	34
31	32	35

下三角（LL）

行番号

1	2	8	9
2	3	9	10
3	4	10	11
4	5	11	12
5	6	12	13
6	7	13	14
7	14		
8	9	15	16
9	10	16	17
10	11	17	
11	12		
12	13	18	19
13	14	19	20
14	20		
15	16	21	22
16	17	22	23
17	23	24	
18	19	25	26
19	20	26	27
20	27		
21	22		
22	23		
23	24	28	29
24	25	29	30
25	26	30	31
26	27	31	32
27	32		
28	29	33	
29	30	33	34
30	31	34	35
31	32	35	36
32	36		
33	34		
34	35		
35	36		
36			

上三角（LU）

図5.2 オリジナル番号付け
非ゼロ要素列番号テーブル

			①
1			
1			
2			
1	2	3	③
3			
4			
2	4	5	④
3	5	6	
6			
7			
4	7	8	⑤
5	8	9	
6	9	10	
11			
7	11	12	⑥
9	13	14	
15			
11	15	16	⑦
16			
13	17		
17			
15	18	19	⑧
16	19	20	
20	21		
17	21	22	
19	23	24	
20	24	25	⑨
21	25	26	
22	26		
24	27	28	
25	28	29	⑩
26	29	30	
28	31	32	⑪
29	32	33	
32	34	35	⑫

下三角（LL）

ブロック番号

行番号

1	2	3	5
2	4	5	8
3	5	6	9
4	7	8	12
5	8	9	13
6	9	10	14
7	11	12	16
8	12	13	
9	13	14	17
10	14		
11	15	16	19
12	16		
13	17	21	
14	17		
15	18	19	23
16	19	20	24
17	21	22	26
18	23		
19	23	24	27
20	24	25	28
21	25	26	29
22	26	30	
23	27		
24	27	28	31
25	28	29	32
26	29	30	33
27	31		
28	31	32	34
29	32	33	35
30	33		
31	34		
32	34	35	36
33	35		
34	36		
35	36		
36			

上三角（LU）

図5.3 ベクトル計算機向き番号付け
非ゼロ要素列番号テーブル

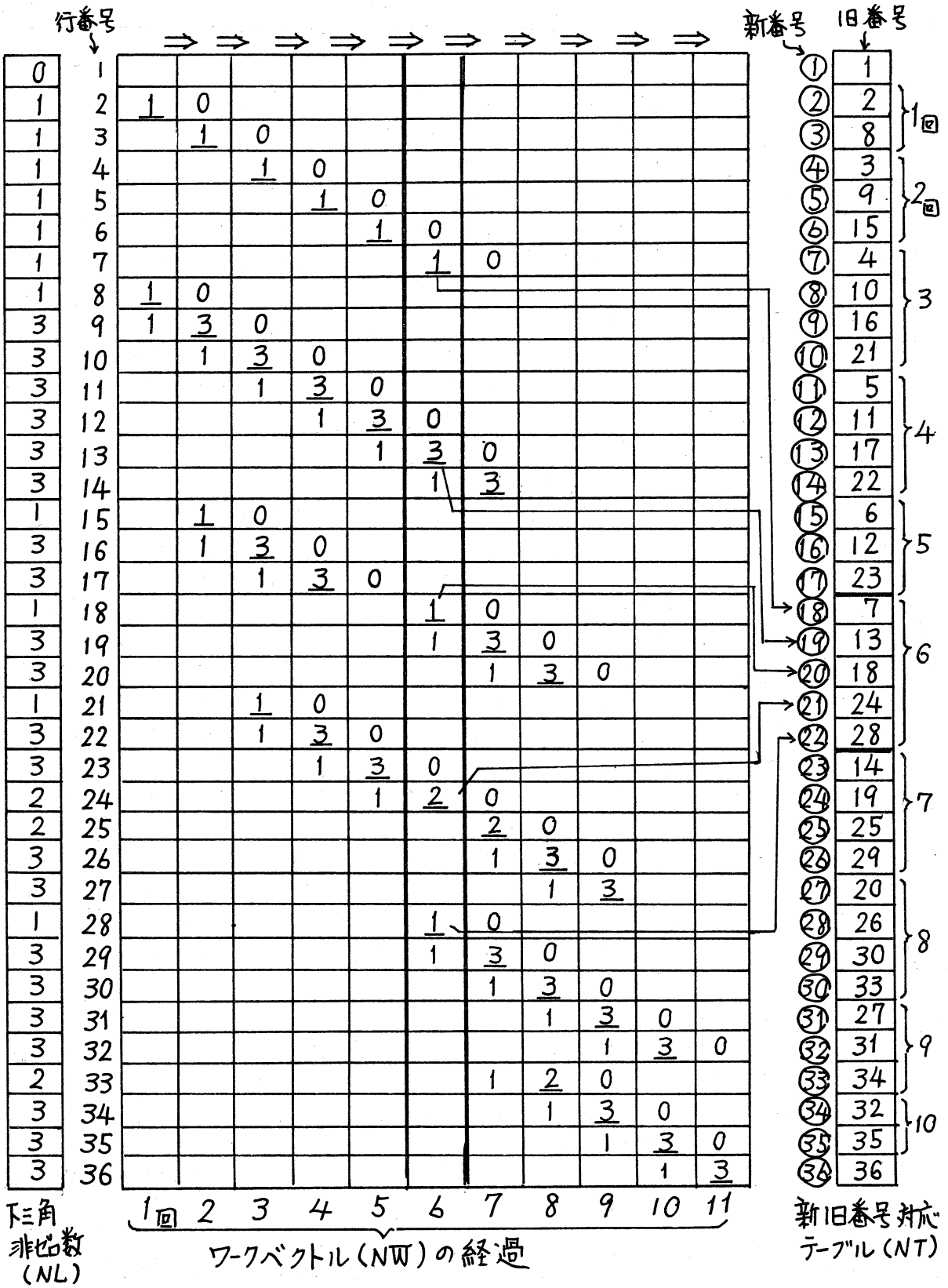


図5.4 ベクトル向き節点番号変換の途中経過

の2，8，9番に1を加算する。その結果，下三角非ゼロ要素数NLと一致した値を持つNWの番号（2,8）の集合を取り出し新旧番号テーブルに追加する。次に図5.2の上三角LUテーブルから，今追加した番号（2,8）に対応する番号（3,9,10,9,15,16）の集合を取り出し図5.4の第1回目のNWの3,9,10,9,15,16番に1を加算して第2回目のNWを得る。このときすでに新旧番号テーブルに追加した番号に対応する位置の値はゼロにしておく。その結果NLと一致した値を持つ第2回目のNWの番号（3，9，15）を取り出し新旧番号テーブルに追加する。上記を繰り返すことで新旧番号対応テーブルを作成する。

ここで工夫した不規則的スパース行列のICCG法をHITAC M-280H（スカラで計算）とHITAC S-810 モデル20（ベクトルで計算）で測定した。その結果収束は同一であり，S-810はM-280Hに比較して約30倍高速となった。測定に使用した行列の次元数は約5000，ICCG法の反復回数は100である。

6. おわりに

汎用計算機向きICCG法（MICCG法）と収束が同一なベクトル計算機向きICCG法（MICCG法）の計算方式を考案した。その結果，30x30x30分割した三次元問題でベクトル計算機HITAC S-810 モデル20を使用するとHITAC M-280Hの約50倍高速化できた。5000次元の不規則的スパース行列に適用し

たICCG法で約30倍高速化できた。

謝辞　御指導いただいた図書館情報大学・村田健郎教授，東京都立大学・小野寺嘉孝教授ならびに筑波大学・名取亮助教授に感謝します。

参考文献

1) J.A. Meijerink, H.A. Van der Vorst ; Iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, Math. Comp. Vol 31, pp. 148~162 (1977)

2) T.A. Manteuffel ; An incomplete factorization technique for positive definite linear systems, Math. Comp. Vol 34 pp. 473~497 (1980)

3) I. Gustafsson ; A class of first order factorization methods, BIT 18, pp. 142~156 (1978)

4) 野寺 隆；大型疎行列に対するPCG法，Seminar on Mathematical Sciences, NO.7 (1983)

5) Olin G. Johnson, George Paul ; Vector algorithms for elliptic partial differential equations based on the JACOBI method, Elliptic Problem Solvers, Academic Press, pp. 345~351 (1981)

6) Henk A. Van der Vorst ; A vectorizable variant of some ICCG methods, SIAM J. Comp. Vol 3, pp. 350~356 (1982)

7) 後 保範；ベクトル計算機向き不完全三角分解；本講究録453, pp 102~126 (1982)

8) フォーサイス，ワソー；藤野精一；偏微分方程式の差分法による近似解法，第3章，吉岡書店（1976）

9) 後，西方，長堀；スーパーコンピュータ“HITAC S-810”による行列計算，日立評論，Vol.65，NO.8，PP557～562（1983）

10) R.S.バーガ，渋谷政昭；計算機による大型行列の反復解法，サイエンス社（1972）

11) 戸川隼人；共役勾配法，教育出版（1977）

12) 小高 他3；最大性能が630 MFLOPSで1Gバイトの半導体拡張記憶が付くスーパコンピュータ HITAC S-810，日経エレクトロニクス，Vol. 314，PP. 159～184（1983）

13) T. Hoshino, T. Kawai, T. shirakawa ; PACS : A Parallel Microprocessor Array for Scientific Calculations

14) Fletcher ; Conjugate gradient methods for indefinite systems, Proc. of the Dundee Biennial Conf. on Num. Anal. Springer-Verlag (1975)

15) Y. Saad ; The Lanczos biorthogonalization Algorithm and other oblique Projection methods for solving large unsymmetric systems, SIAM J. Num. Anal., Vol. 19, PP. 485～506（1982）