数 値 解 析 17－4
（１９８６．７．４）

ベクトル計算機における Ｓｃａｌｅｄ ＣＧ法の有効性について

速水 謙 ， 原田 紀夫
日本電気（株） Ｃ＆Ｃシステム研究所

本報告では高度なベクトル処理能力を最大限活用するという立場から，ベクトル計算機向きの高速な連立一次方程式の解法として Ｓｃａｌｅｄ Ｃｏｎｊｕｇａｔｅ Ｇｒａｄｉｅｎｔ Ａｌｇｏｒｉｔｈｍ（ＳＣＧ法）を提案する．この解法は偏微分方程式の離散近似等で生じる正定値対称スパースな連立一次方程式の行列を対角項でスケーリングした後に共役勾配法(Conjugate Gradient Algorithm) を適用するもので，１００％ベクトル処理可能なので高速ベクトル計算が実現でき，高い収束性を保持しているという特徴をもつ．又，所要記憶容量も少なくて済む．
例えばスーパーコンピューターＳＸシステムを用いた数値実験で，従来高速算法として知られているＩＣＣＧ法(Incomplete Cholesky - Conjugate Gradient Algorithm)に対してもベクトル処理計算機において実行時間で約９～２０倍の高速が得られた．又，ＩＣＣＧ法をベクトル処理に適するようにするためリストベクトルを用いた方法に比べても実行時間で約１．６～５．０倍の高速が得られた．
これらの種々の数値実験を基にＳＣＧ法の有効性を示す．

THE USE OF THE SCALED CONJUGATE GRADIENT ALGORITHM ON VECTOR PROCESSORS

Ken HAYAMI and Norio HARADA

C&C Systems Research Laboratories, NEC Corporation
4-1-1, Miyazaki, Miyamae, Kawasaki, Kanagawa, 213 Japan

We propose the Scaled Conjugate Gradient (SCG) algorithm as an algorithm suited to vector processors for solving large sparse positive definite linear systems, which arise in the discrete approximation of partial differential equations.
The SCG, which applies the conjugate gradient algorithm after scaling the matrix by its diagonals, is 100% vectorizable, requires little memory and is simple to code.
Numerical experiments were done on the NEC supercomputer SX-2 for various large and ill conditioned problems arising from finite differencing of partial differential equations in 2 and 3 dimensions.
Results show that the SCG is 9 to 21 times as fast in execution time compared to the ICCG algorithm, which is considered to be a powerful algorithm for solving problems mentioned above, and 1.5 to 5 times as fast as the vectorized M/ICCG algorithms which utilize list vectors.
The SCG is also shown to be superior to the ICCG for linear systems with random sparse matrices arising in 3 dimensional finite element analysis.

1．はじめに

高度のベクトル処理能力をもつスーパーコンピューターの出現に伴い，大規模数値シミュレーションにおいて数値計算アルゴリズムがいかにそのベクトル処理能力を生かすかが重要になってきている．本報告では高いベクトル処理能力を生かす高速アルゴリズムを開発するという立場から連立一次方程式のベクトル計算機向き解法を提案する．

近年，偏微分方程式の離散近似に伴う大規模で条件数の大きい対称正定値な連立一次方程式の有力な反復解法として ＩＣＣＧ法 (Incomplete Cholesky - Conjugate Gradient Algorithm) [1),2)] が提案されている．この方法は従来のＳＯＲ法 (Successive Over Relaxation Method)，ＡＤＩ法 (Alternate Direction Implicit Method)等の反復法及びそれらの改良版に比べ，収束性が極めて良いため，広く使用されている．しかし，ベクトル処理を行うスーパーコンピューターのアルゴリズムという観点からは，ＩＣＣＧ法はその反復の中にベクトル処理に向かない前進後退代入を含むため，ベクトル処理能力を十分に生かし切れない面がある．この対策として近似逆行列の Neumann級数展開などの近似多項式を用いる方法 [10),11)] などがあるが，メモリーを多用し，複雑である割にはベクトル性能，収束性が思わしくない．また，リストベクトルを用いて前進後退代入をベクトル化する手法[3)] が提案されているが，間接アドレスを用いるため連続アクセスに比べてデータのアクセスが遅くなることと，元数に比べてベクトル長が短いため，ベクトル計算機の性能を十分に生かしていない．

このようにＩＣＣＧ法は反復数が少ないという極めて良い収束性を有するがベクトル処理に向かない部分を含んでいる．そこで我々は，アルゴリズムの実行時間の高速性は単なる反復数だけでなく反復数と反復当りの実行時間の積であるという視点に立ち，元の行列をその対角項によってスケーリングを施してからＣＧ法(Conjugate Gradient algorithm)[4)] を適用する Ｓｃａｌｅｄ　ＣＧ法(Scaled Conjugate Gradient algorithm, 以下ＳＣＧ法と呼ぶ) [5),6),7)] をベクトル計算機向きの有力なアルゴリズムとして提案する．ＳＣＧ法は１００%ベクトル処理可能で，所要記憶容量も少なくて済み，プログラムも簡単である．反復数という面ではＩＣＣＧ法には及ばないが高度のベクトル処理能力をもつ計算機での実行時間という面では，ＩＣＣＧ法，リストベクトルを用いたベクトル化Ｍ／ＩＣＣＧ法より数倍から数十倍の高速性が得られる事を，実際例を含めた多くの例題のスーパーコンピューターＳＸ上での数値実験によりしめす．

2．アルゴリズム

（ｉ）ＩＣＣＧ法

ＩＣＣＧ法 [1),2)] はＣＧ法の収束を加速するためにまず行列Ａの不完全コレスキー分解を行う．つまり，Ａの非零要素に対してのみコレスキー分解を行い，

$$A \sim LDL^T \qquad (3),$$

を得る．但し，Ｌ＝（l_{ij}） はＡの下対角と同じ非零パターンをもつ下三角行列，Ｄは対角行列，L^TはＬの転置行列であり，～は近似を表す．（３）より

$$A' \equiv (LD^{1/2})^{-1} A (LD^{1/2})^{-T} \sim I \qquad (4),$$

となり，Ａ’はＡより条件の良い正定値対称行列になると考えられる．そこで，

$$x' = (LD^{1/2})^T x$$
$$b' = (LD^{1/2})^{-1} b \qquad (5),$$

とおいてもとの方程式（１）と等価な方程式

$$A'x' = b' \qquad (6),$$

を得，（６）にＣＧ法（２）を適用し，もとの変数で表すことにより次ぎのＩＣＣＧ法のアルゴリズムを得る．

① Ａの不完全コレスキー分解
$A \sim LDL^T$ （３） を行う．

② $r = b - Ax_1$
$p = (LDL^T)^{-1} r_1$
$i = 1, 2, 3, \ldots$
$\alpha_i = \dfrac{(r_i,\ (LDL^T)^{-1} r_i)}{(p_i,\ Ap_i)}$
$x_{i+1} = x_i + \alpha_i p_i$
$r_{i+1} = r_i - \alpha_i Ap_i$
$\beta_i = \dfrac{(r_{i+1},\ (LDL^T)^{-1} r_{i+1})}{(r_i,\ (LDL^T)^{-1} r_i)}$
$p_{i+1} = (LDL^T)^{-1} r_{i+1} + \beta_i p_i$

（７），

但し，$(LDL^T)^{-1} r$ ，Ａｐ ，内積等は反復当たり１回行えばよい．

ところが，ＩＣＣＧ法（７）の反復毎に現れる $s = (LDL^T)^{-1} r$ の計算は前進後退代入を含んでおり，これらはｓ のデータの依存関係のため効率良くベクトル処理を行うのが困難である．

（ii）ＳＣＧ法

この困難を克服するため，我々は不完全コレスキー分解の代わりに，より単純に，行列Ａをその対角項

$$D = diag[a_{11}, a_{22}, \ldots, a_{nn}] \qquad (8),$$

によりスケーリングする．つまり，

$$A' \equiv D^{-1/2} A D^{-1/2} \qquad (9),$$

とすればＡ’はＡより条件の良い正定値対称行列になると考えられるから，

$$x' = D^{1/2} x$$
$$b' = D^{-1/2} b \ ,$$

とおき，もとの方程式（１）と同等な

$A'x' = b'$ にＣＧ法を適用することにより，次の Ｓｃａｌｅｄ　Ｃｏｎｊｕｇａｔｅ　Ｇｒａｄｉｅｎｔ （ＳＣＧ法）を得る．

① $D^{-1} = diag\,[1/a_{11}, 1/a_{22}, \ldots, 1/a_{nn}]$
を求める．

② $r = b - Ax_1$
$p = D^{-1} r_1$
$i = 1, 2, 3, \ldots$
$\alpha_i = \dfrac{(r_i,\ D^{-1} r_i)}{(p_i,\ Ap_i)}$
$x_{i+1} = x_i + \alpha_i p_i$
$r_{i+1} = r_i - \alpha_i Ap_i$
$\beta_i = \dfrac{(r_{i+1},\ D^{-1} r_{i+1})}{(r_i,\ D^{-1} r_i)}$
$p_{i+1} = D^{-1} r_{i+1} + \beta_i p_i$

（１０），

但し，Ａｐ，$D^{-1}r$，内積等は反復当たり１回計算すればよい．

ＳＣＧ法は，行列Ａの非零要素を格納する適切なデータ構造を考慮すれば容易に１００%ベクトル処理可能である．表１に一般の場合のＳＣＧ法とＩＣＣＧ法の反復当りの演算量と必要な記憶領域を比較する．

反復当りの演算量，記憶領域ともＳＣＧ法はＩＣＣＧ法より少なくて済む．

表1　ＳＣＧ法　と　ＩＣＣＧ法の 反復当りの演算量，記憶領域

	反復当りの演算量	記憶領域
ＳＣＧ法	２ｎｚ＋１２ｎ	０．５ｎｚ＋６ｎ
ＩＣＣＧ法	４ｎｚ＋１２ｎ	ｎｚ＋５．５ｎ

（注）　ｎ：元数，　ｎｚ：行列Ａの非零要素数

３．有限差分法に対する数値実験結果

ここでは，偏微分方程式の有限差分近似で生じる規則スパースな連立一次方程式に対してＳＣＧ法とＩＣＣＧ法をＮＥＣのスーパーコンピュータ　ＳＸ－２上の数値実験により比較する．

（ｉ）２次元非定常拡散方程式

ケース　①[2]

$$\partial u/\partial t = \partial/\partial x(\rho\,\partial u/\partial x) + \partial/\partial y(\rho\,\partial u/\partial y) \qquad (11),$$

を図１の境界条件（Ｄでｕ＝０，Ｎで$\partial u/\partial n = 0$）のもとで，時間刻み（$\Delta t = 1$）を１つ進めるために陰的に解く．まずめに変数を対応させた５点中心差分により１８７５元の正定値対称な連立一次方程式を得る．条件数は約8×10^6で，条件の悪い問題である．

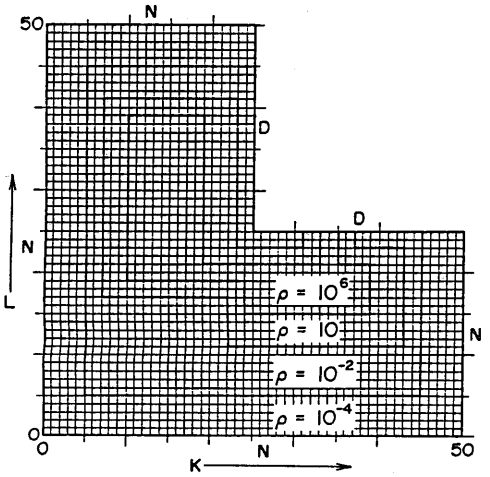

図１　２次元非定常拡散方程式の計算領域
（Ｋｅｒｓｈａｗ のモデル）

（ｉｉ）３次元拡散方程式

$$-\partial/\partial x(k_x\,\partial u/\partial x) - \partial/\partial y(k_y\,\partial u/\partial y) - \partial/\partial z(k_z\,\partial u/\partial z) = f \qquad (12),$$

を図２，図３の直方体領域で格子点を変数に対応させた７点中心差分で解く．境界条件は図２，図３で現れている面に対しては　$\partial u/\partial n = 0$，　現れていない面に対しては　ｕ＝０とする．
また，熱源は領域で一様に　ｆ＝５００　とする．

ケース　②　（メッシュ数ｍの変化）
各方向のメッシュ数を　ｍ＝１０，２０，３０，　４０，５０　と変化させる．従ってｍ³の元数の連立一次方程式を得る．（但し図２でｘ＝ｙ＝ｚ＝５，$k_x = k_y = k_z = 1$）

ケース　③　（係数ｋの空間変化）
係数$k = k_x = k_y = k_z$が図３の様に空間的に１からr^6まで変化する．但し，$r^6 = 10^3, 10^6, 10^9$．（但し，図３でｘ＝ｙ＝ｚ＝５，ｍ＝２０）

ケース　④　（係数ｋの異方性）
（１）$k_x = 1$，$k_y = 5$，$k_z = 25$
（２）$k_x = 1$，$k_y = 10$，$k_z = 100$
（但し，図２でｘ＝ｙ＝ｚ＝５，ｍ＝２０）

ケース　⑤　（メッシュ間隔の異方性）
（１）ｘ＝２，ｙ＝５，ｚ＝１０
（２）ｘ＝１，ｙ＝５，ｚ＝２５
（但し，図２で$k_x = k_y = k_z = 1$，ｍ＝２０）

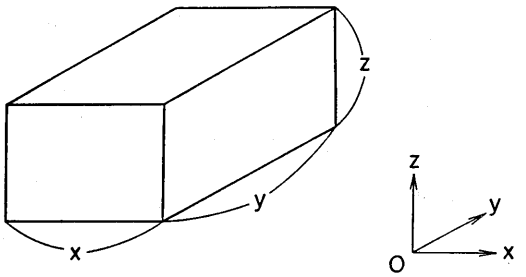

図２　３次元拡散方程式の計算領域

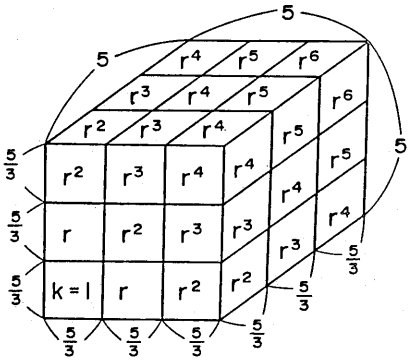

図３　拡散係数　ｋが空間的に変化する例

以上でメッシュの番号付けはｘ，ｙ，ｚ方向の順に付け，行列Ａの非零要素の値は，対角方向の一次元配列に格納し，行列とベクトルの積のベクトル処理を容易にした．また解の初期ベクトルは全て$x_1 = 0$とした．

代表的な例として図４に，ケース②，ｍ＝３０の場合のＳＣＧ法とＩＣＣＧ法の相対残差L_2ノルム（対数スケール）の推移をＳＸ－２でベクトル処理した実行時間を横軸にとったグラフに示す．ベクトル処理におけるＳＣＧ法の優位性が明らかである．

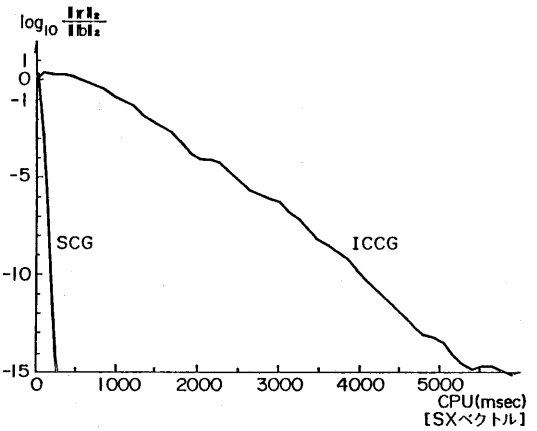

図４　ＳＣＧ法とＩＣＣＧ法の　ＳＸ－２　ベクトル処理実行時間
vs.　相対残差L_2ノルム　（ケース③，ｍ＝３０）

次に，ケース①～⑤に対してＳＣＧ法とＩＣＣＧ法が相対残差L_2ノルム

$\|b-Ax_1\|_2 / \|b\|_2 < 10^{-6}$，

を達成するのに要したＳＸ－２ベクトル処理の実行時間，反復数，ベクトル処理による加速率（スカラー処理の実行時間／ベクトル処理の実行時間），及びＩＣＣＧ法とＳＣＧ法のベクトル処理の実行時間の比を表２に示す．

表２より，ＳＣＧ法はＩＣＣＧ法に比べて，ベクトル処理の実行時間で，普通の問題で１８～２１倍，条件の悪い問題でも８．６～１８倍速い．また，ＳＣＧ法のベクトル処理による加速率が５０倍であるのに対して，ＩＣＣＧ法では２倍にとどまっておりＳＸ－２の性能が十分に生かされていない．

これは先に述べた様にＩＣＣＧ法の前進後退代入がベクトル化されなかったためである．（不完全コレスキー分解に要する時間は全体の１％以下である．）　実際，ＩＣＣＧ法のプログラムのプログラムのベクトル化率は４８％前後であった．従ってＩＣＣＧ法はベクトル処理によって２倍以上は速くならない．

以上によりＳＣＧ法のベクトル計算機における有効性が明らかにされたが，ＳＣＧ法におけるスケーリングの前処理としての効果を評価するために，ただのＣＧ法と比較する．表３に，同じケース①～⑤の問題に対してＣＧ法で打切り相対残差L_2ノルム10^{-6}を達成するのに要したＳＸ－２ベクトル処理の実行時間と反復数，及びＳＣＧ法のベクトル処理実行時間との比較を示す．

普通の問題ではＳＣＧ法はＣＧ法の２倍速い程度だが，条件の悪い問題，特に係数の空間変化の激しい問題ではＣＧ法では収束しない場合でもＳＣＧ法は比較的少ない反復数で収束しており，スケーリングが前処理として有効に働いている事がわかる．

表3　SCG法　と　CG法 の比較　（SX－2，ベクトル実行）

ケース	CG法		CG／SCG（CPU時間比）
	CPU時間（msec）	反復数	
① Kershaw のモデル	144	1709	13
② メッシュ数の変化			
m＝10	3．74	78	1．8
m＝20	54．4	164	2．0
m＝30	272	249	2．0
m＝40	861	336	1．9
m＝50	2140	423	1．9
③ 係数 k の空間変化			
$r^6=10^3$	699	2111	17
$r^6=10^6$	収束せず	20000	—
$r^6=10^9$	収束せず	20000	—
④ 係数 k の異方性			
$k_x=1$，$k_y=5$，$k_z=25$	130	393	2．5
$k_x=1$，$k_y=10$，$k_z=100$	193	583	3．2
⑤ メッシュ間隔の異方性			
x＝2，y＝5，z＝10	88．8	268	1．7
x＝1，y＝5，z＝25	145	439	2．5

（注）打切り相対残差L_2ノルムは　10^{-6}

表2　SCG法　と　ICCG法 の比較　（SX－2，ベクトル実行）

ケース	SCG法			ICCG法			ICCG／SCG（CPU時間比）
	CPU時間（msec）	反復数	ベクトル加速率	CPU時間（msec）	反復数	ベクトル加速率	
① Kershaw のモデル	11．1	114	42	159	35	1．9	14
② メッシュ数の変化							
m＝10	2．08	37	37	38．4	16	1．9	18
m＝20	27．4	72	49	565	28	1．9	21
m＝30	138	111	51	2870	43	2．0	21
m＝40	442	152	51	8870	56	2．0	20
m＝50	1120	194	50	22000	71	2．0	20
③ 係数 k の空間変化							
$r^6=10^3$	40．2	106	49	723	36	1．9	18
$r^6=10^6$	46．3	122	49	841	42	1．9	18
$r^6=10^9$	50．4	133	49	920	46	1．9	18
④ 係数 k の異方性							
$k_x=1$，$k_y=5$，$k_z=25$	52．4	138	48	743	37	1．9	14
$k_x=1$，$k_y=10$，$k_z=100$	61．0	161	49	703	35	1．9	12
⑤ メッシュ間隔の異方性							
x＝2，y＝5，z＝10	53．8	142	49	723	36	1．9	13
x＝1，y＝5，z＝25	59．1	156	49	506	35	1．9	8．6

（注）打切り相対残差L_2ノルムは　10^{-6}

4．ベクトル化 M／ＩＣＣＧ法とＳＣＧ法の比較

ＩＣＣＧ法をベクトル化する試みも多くなされており[10),11)]，その中でもリストベクトルを用いる方法[3),8)]が有効とされている．これは前進後退代入などで，互いにデータ依存関係の無い節点（有限差分メッシュの場合 ｘ，ｙ，ｚ方向の格子座標 ｉ，ｊ，ｋ が ｉ＋ｊ＋ｋ＝一定 を充たす面内の節点）に関してリストベクトルを用いてベクトル化する手法である．

また，有限差分に対するＩＣＣＧ法の収束を加速する手法として，　ＭＩＣＣＧ法（ Modified　ICCG algorithm)[9),3),8)] が提案されている．　これは，不完全コレスキー分解 Ａ～ＬＤＬᵀ で得られる　Ｄ：ｄｄᵢ　を

$$dd_i = 1/\left[d_i - c_i*dd_{i-1}*\{c_i+\theta*(a_{i+m2-1}+b_{i+m1-1})\} - b_i*dd_{i-m1}*\{b_i+\theta*(c_{i+1-m1}+a_{i+m2-m1})\} - a_i*dd_{i-m2}*\{a_i+\theta*(b_{i+m1-m2}+c_{i+1-m2})\}\right] \quad (13),$$

の様に修正したものである．但し（１３）は３次元の有限差分で生じる図４の様な行列Ａの第ｉ行の非零要素に対して，対角をd_i，第１下対角をc_i，第ｍ１下対角をb_i，第ｍ２下対角をa_i（但しｍ１＜ｍ２），加速パラメターをθとおいたものである．

表４にケース②～⑤の３次元問題に対してＳＣＧ法と，リストベクトルを用いたＩＣＣＧ法及びＭＩＣＣＧ法のＳＸ－２ベクトル処理の実行時間を比較する．　ＭＩＣＣＧ法は加速パラメターθ＝ 0.975, 0.95, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0(ICCG)　のうち最適であったものを示した．打ち切り相対残差L_2ノルムは10^{-6}である．

ほとんどの場合ＳＣＧ法はＭ／ＩＣＣＧ法の３～５倍速く，異方性のある場合でも１．５～２．６倍速い．　ちなみに，ケース②，ｍ＝５０でのＳＣＧ法の演算処理速度は７８８ＭＦＬＯＰＳであった．

これは，ＳＣＧ法ではベクトル長が元数 m^3＝１０００～１２５０００であるのに対し，リストベクトルを用いたＭ／ＩＣＣＧ法ではベクトル長はデータの依存関係がない面上の節点数（m^2のオーダー）しかとれないこと，またリストベクトルＭ／ＩＣＣＧ法では間接アドレス参照があるため，ＳＣＧ法の連続アドレス参照に比べてアドレス参照の負荷が大きいためである．

以上の議論により，３次元の有限差分で生じる問題で，ＳＣＧ法はリストベクトルを用いたベクトル化Ｍ／ＩＣＣＧ法に対しても，ＳＸ－２上で優位であることが示された．・・・

（注１：表４のＳＣＧ法のプログラムは３次元有限差分専用に作ってあり，メモリーへのストア回数が少ないため，表２の汎用規則スパース用に作ったＳＣＧ法のプログラムに比べて３割ほど速くなっている．）

（注２：リストベクトルＭ／ＩＣＣＧ法 のプログラムは日本電気技術情報システム（株）　亘　紀子氏に作成して頂いたものである．）

5．デバイス・シミュレーションへの適用

デバイス・シミュレーションで生じる２次元ポアソン方程式の有限差分近似（５０×５０の不等間隔メッシュ）で生じる連立一次方程式に対してＳＣＧ法とリストベクトルを用いたＩＣＣＧ法をＳＸ－２上で比較した結果を表５に示す．但し，打切り相対L_2残差は10^{-15}とした．

なお，ＳＣＧ法とＩＣＣＧ法で得た解ベクトルの差の相対L_2誤差は４．９×10^{-15}で，ＳＣＧ法の解はＩＣＣＧ法の解と比べて精度は変わらないと考えられる．

（解析データの使用を快諾して頂いた　日本電気　（株）マイクロエレクトロニクス研究所　福間主任に感謝致します．）

表５．デバイス・シミュレーションでの
ＳＣＧ法とリストベクトルＩＣＣＧ法の比較

	ＳＣＧ法	ＩＣＣＧ法 リストベクトル
ＣＰＵ 時間	１３．３ｍｓ	４６．２ｍｓ
反復数	１６２回	４３回

表4　SCG法 と ベクトル化 M／ICCG法 の比較 （SX－2 ベクトル実行）

ケース	SCG法		ベクトル化 ICCG法			ベクトル化 MICCG法			
	CPU時間 (msec)	反復数	CPU時間 (msec)	反復数	ICCG / SCG (CPU)	CPU時間 (msec)	反復数	最適 θ	MICCG / SCG (CPU)
② メッシュ数の変化									
m=10	1.57	37	7.79	16	5.0	7.33	15	0.2	4.7
m=20	19.9	72	76.1	28	3.8	73.5	27	0.5	3.7
m=30	99.7	111	364	43	3.7	339	40	0.8	3.4
m=40	323	152	1100	56	3.4	1020	52	0.9	3.2
m=50	801	194	2690	71	3.4	2420	64	0.95	3.0
③ 係数 k の空間変化									
$r^6=10^3$	30.1	106	97.2	36	3.2	92.0	34	0.5	3.1
$r^6=10^6$	34.5	122	113	42	3.3	105	39	0.8	3.0
$r^6=10^9$	36.5	133	124	46	3.4	118	44	0.4	3.2
④ 係数 k の異方性									
$k_x=1,\ k_y=5,\ k_z=25$	37.8	138	99.9	37	2.6	92.0	34	0.7	2.4
$k_x=1,\ k_y=10,\ k_z=100$	44.1	161	94.6	35	2.1	84.0	31	0.6	1.9
⑤ メッシュ間隔の異方性									
x=2, y=5, z=10	38.9	142	97.2	36	2.5	89.3	33	0.4	2.3
x=1, y=5, z=25	42.8	156	68.2	25	1.6	62.9	23	0.7	1.5

（注1）打ち切り相対残差L_2 ノルムは10^{-6}　　（注2）　θはMICCG法のパラメター

6．ランダム・スパース行列への ＳＣＧ法の適用

次に，ランダム・スパースで正定値対称な大規模連立一次方程式へのＳＣＧ法の適用を検討する．

有限要素法などで生じるランダム・スパース行列に対してＳＣＧ法を適用する際，ベクトル計算機上で効率良く行列とベクトルの積を計算するための行列のデータ構造が重要となる．

従来，対称なランダム・スパース行列を格納するのに行方向のリストがよく用いられてきた．しかし，有限要素法などでは１節点につながる節点数は限られているので，行方向リストを用いて行列とベクトルの積を計算すると最深ｄｏｌｏｏｐ長が短いため，高いベクトル性能を引き出しにくい．

そこで，ベクトル計算機向きのデータ構造として，次の様な対角方向リストを提案する：

ＡＤ（ｊ）：行列Ａの対角要素，
ＡＵ（ｉ）：行列Ａの上対角非零要素，
ＪＣ（ｉ）：ＡＵ（ｉ）の列番号，
ＫＤ（ｉ）：ｉ本目の対角方向リストは第ＫＤ（ｉ）上対角上にある，
ＩＤ（ｉ）：ｉ本目の対角方向リストの先頭要素の要素番号，
ＮＴ　　：対角方向リストの本数，
Ｎ　　：元数，

対角方向リストで行列とベクトルの積ｓ＝Ａｐを計算するには，

```
j=1,2,...,N
  S(j)=AD(j)*P(j)

i=1,2,...,NT
  k=KD(i)
  l=ID(i),...,ID(i+1)-1
    j=JC(l)
    S(j-k)=S(j-k)+AU(l)*P(j)
    S(j)=S(j)+AU(l)*P(j-k)
```

（１４），

とすればよい．

有限要素メッシュが規則的な場合は行列の非零要素は対角方向に並ぶ．また複雑なメッシュでも空間的に規則的に（例えばｘ，ｙ，ｚ方向の順に）番号付けを行えば，対角方向に並ぶ傾向を持つ．

従って，（１４）の対角方向リストに沿った　ｌによる最深ｄｏ　ｌｏｏｐ長は，行方向リストの場合に比べて長い．また，最深ｄｏ　ｌｏｏｐ内ではＳ（ｊ），Ｐ（ｊ）等の引数は単調増加であるから，ベクトル処理に際するデータ参照関係の問題も生じない．

７．有限要素法に対する数値実験結果

３次元の複雑な形状の静電界の有限要素解析に対して，ＳＣＧ法及びＩＣＣＧ法を適用した数値実験結果を表６に示す．ＳＣＧ法は行方向，対角方向リストについて，ＩＣＣＧ法は行方向リストについて，打切り相対残差L_2ノルムを10^{-15}にとったときのＳＸ－２ベクトル処理の実行時間と反復数を示した．

ランダム・スパース行列においても，ＳＣＧ法はＩＣＣＧ法に対して優位である．但し，行方向リストのＩＣＣＧ法では，実行時間の９９％以上は不完全コレスキー分解に費やされてる．従って，ＩＣＣＧ法はより適切なデータ構造の検討が必要である．

対角方向リストを用いたＳＣＧ法は，行方向リストを用いたＳＣＧ法に比べてＳＸ－２ベクトル処理の実行時間で２～３倍速い．これは，行方向リストの平均ベクトル長が６であるのに対して，対角方向リストの平均ベクトル長が２５～６４と長くなっているのが主な原因と考える．なお表６の解析データは節点番号が空間的に規則的につけられており，半バンド巾が小さいので，対角方向リストに向いている．

（　表６の３次元静電界有限要素法の解析データの使用を快諾して頂いた　日本電気（株）　Ｃ＆Ｃシステム研究所　土肥主任に感謝致します．）

表６　３次元有限要素法で生じるランダム・スパース行列に対するＳＣＧ法（対角方向，行方向リスト），ＩＣＣＧ法（行方向リスト）の比較

アルゴリズム		ケース１	ケース２	ケース３	ケース４
	元数	1167	2039	3575	6309
対角方向 ＳＣＧ法	ＳＸ－２ベクトル 実行時間（ｍｓｅｃ）	24.1	36.9	55.9	94.3
行方向 ＳＣＧ法	ＳＸ－２ベクトル 実行時間（ｍｓｅｃ）	50.4	93.9	157	278
両ＳＣＧ法	反復数	19	20	20	21
行方向 ＩＣＣＧ法	ＳＸ－２ベクトル 実行時間（ｍｓｅｃ）	4610	17500	49400	
	反復数	5	5	5	

（注）　打切り相対残差L_2ノルムは10^{-15}．

８．むすび

本論文では，偏微分方程式の離散近似などで生じる正定値対称で大規模スパースな連立一次方程式のベクトル計算機向きの反復解法としてＳＣＧ法を提案した．　ＳＣＧ法は１００％ベクトル処理可能で容易に高速ベクトル計算が実現でき，収束性も良いのでベクトル計算機上の実行時間の面で有効である．また，所要記憶容量も少なくて済む．

スーパーコンピュータＳＸ－２上の数値実験により，２，３次元の有限差分法で生じる幅広い問題に対してＳＣＧ法は，従来のＩＣＣＧ法に比べて実行時間で約９～２１倍速く，リストベクトルを用いたベクトル化Ｍ／ＩＣＣＧ法に対しても１．５～５倍速い事をしめした．

また，３次元の有限要素法で生じるランダムスパースな問題に対して，ＳＣＧ法に適した対角方向リストによるデータ格納方式を提案し，ＳＣＧ法がＩＣＣＧ法に比べて優位な事を示した．

以上により，ＳＣＧ法は高度なベクトル処理能力をもつスーパーコンピュータでの有力なアルゴリズムと考える．

参考文献

1) Meijerink, J.A. and Van der Vorst, H.A.: 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) Kershaw, D.S.: The incomplete Cholesky - conjugate gradient method for the iterative solution of systems of linear equations, J. Comp. Phys., Vol.26, pp.43-65 (1978).

3) 後　保範：ベクトル計算機向きＩＣＣＧ法，京都大学数理解析研究所講究録，No.514, pp.110-134, 京都　(1984).

4) Hestenes, M.R. and Stiefel, E: Method of conjugate gradients for solving linear systems, J. Res. Nat. Bur. Standards, No.49, pp.409-436 (1952).

5) Fox, R.L. and Stanton, E.L.: Developments in structural analysis by the direct energy minimization, AIAA, Vol.6, No.6, pp.1036-1042 (1968).

6) 速水　謙，原田　紀夫：Ｓｃａｌｅｄ　ＣＧ法再考，情報処理学会第３０回全国大会予稿集，pp.1731-1732 (1985).

7) Hayami, K. and Harada, N: The scaled conjugate gradient method and vector processors, Proceedings of the First International Conference on Supercomputing Systems, pp.213-221, Florida (1985).

8) 村田，小国，唐木：スーパーコンピュータ（科学技術計算への応用），丸善，東京 (1985).

9) Gustafsson, I: A class of first order factorization methods, BIT 18, pp.142-156 (1978).

10) Johnson, O.G. and Paul, G.: Optimal parametrized inverse preconditioning for conjugate gradient calculations, Res. Rep., RC8644, IBM Research Center, Yorktown Heights, NY (1981).

11) Van der Vorst, H.A.: A vectorizable variant of some ICCG methods, SIAM J. Sci. Stat. Comput., Vol.3, No.3, Sept. (1982).