

超平面法とベクトル計算機

藤野清次 (計算流体力学研究所) 森 正武 (東京大学工学部) 竹内敏己 (花王文理科学研究所)

超平面法は従来その計算機が持つリスト・ベクトル機能の性能を元にベクトル計算機単位で、例えばリスト・ベクトルに強い計算機あるいは弱い計算機だと、議論されることが多かった。また超平面法について知られていることも、その平均ベクトル長や各指標間の関係式などに過ぎず、理論的な立場から超平面法の性能評価が余りなされていなかった。本論文ではベクトル計算機の高速性能を大きく左右するバンク・コンフリクトの影響を出来るだけ少なくするという立場から、超平面法のベクトル計算機上の有効利用方法を理論的にかつ具体的に提言するものである。最初に本研究では、超平面法の性能がメモリーへアクセスする間隔の大きさとそれが起こる回数に非常に依存していることを示す。次にアクセスする間隔とその回数を理論的に見積り、それらが各軸方向の格子点数によってすべて決まることを明かにする。そして各軸方向の格子点数を最も適切に選んだときとそうでないときは計算効率に実に2.8倍の差がでることを示す。

Performance of the hyperplane ordering on vector computers

Seiji Fujino

Institute of Computational Fluid Dynamics
2-1-4, Harmachi, Meguro-ku, Tokyo 152, Japan

Masatake Mori

Faculty of Engineering, University of Tokyo
7-3-1, Hongo, Bunkyo-ku, Tokyo 113, Japan

Toshiki Takeuchi

Institute of mathematical Science of KAO Corporation
2-1-3, Bunka, Sumida-ku, Tokyo 131, Japan

We consider the hyperplane ordering vectorization with the least effect of memory bank conflict on vector computers. We show that the performance of the hyperplane ordering on recently used vector computers depends upon the magnitude of the access stride to memory and the total count of the access strides. We estimate precisely the magnitude of the access stride and its total count which are determined by the numbers of gridpoints in the three directions. We can gain the highest performance of the vector computers if and only if we have the best choice of the numbers of the gridpoints in the 3 directions. The worst choice requires 2.8 times as much CPU time as the best choice.

1 はじめに

超平面法は前処理付き共役傾斜法 (*Incomplete Cholesky Conjugate Gradient method*、以下 *ICCG* 法と略する。) などのベクトル化技術の1つとして知られよく利用されている。2次元長方形領域で5点差分近似式を超平面法でベクトル化する場合、格子点の計算順序を通常の辞書式に付けるとメモリーへのアクセスは等間隔、すなわち X 方向の格子点数を N_x としたとき、 $N_x - 1$ になる。大抵のベクトル計算機ではバンク・コンフリクトなどの影響により、連続アクセスするときよりも離れた番地へアクセスするとき効率が落ちることが多いが、2次元の超平面法の場合は間隔が一定なのであらかじめその影響が最小限になるように対策を立てることが出来る。しかしながら、3次元の超平面法の場合、2次元のときと大きく違うところは、アクセスされる配列のアドレスが2次元のように等間隔でなく、そのために間接指標アドレス (いわゆるリスト・ベクトル) にせざるを得ない点にある。いままでその平均ベクトル長などについては知られていたが、正確なアクセス間隔の大きさやそれらが起こる回数合計などについてはわからなかった。そのためどの程度バンク・コンフリクトの影響を受けているのか、あるいはそれを最少限にするにはどうしたらよいかなど、具体的なことは何一つ知られていなかった。

本論文では第3章でベクトル計算機上のバンク・コンフリクトの存在をテストプログラムを使って提示し、効率に対する影響の度を調べる。そして、第4章で2次元長方形領域の境界値問題をいわゆる超平面法 (この場合はアドレスを直接指定してリスト・ベクトルを使わない) でベクトル化した *ICCG* 法で解き、 X 方向の格子点数を変化させてどのくらいバンク・コンフリクトの影響を実際に受けているか調べる。第5章では、3次元7点差分近似のときの各軸方向の格子点数とメモリーへアクセスする間隔との間に成り立つ定理を導出する。第6章では、3次元立方体領域で定義された境界値問題を取扱う。これを超平面法 (リスト・ベクトルを使用) でベクトル化した *ICCG* 法で解き、第5章で導いた定理をもとに各軸方向の格子点数を変化させてバンク・コンフリクトの影響を数値実験で調べる。最後に、3次元問題において効率的に解を得るために、 X, Y, Z 方向の格子点数をどのように定めたらよいか指針を与える。

2 超平面法について

よく知られているように2次元5点差分近似の場合、各格子点の指標を (i, j) で表すとき計算する順序を $i+j$ = 一定のグループ毎に行う方法を2次元超平面法と言う。この場合メモリーへアクセスする間隔は $N_x - 1$ になる (N_x は X 方向の格子点数そして格子点の順番は辞書式とする)。(図1参照)

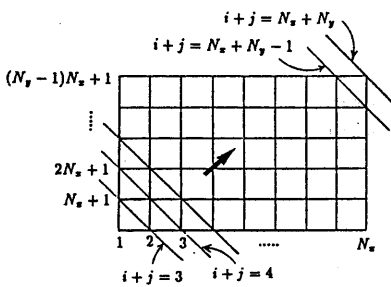


図1 2次元超平面法

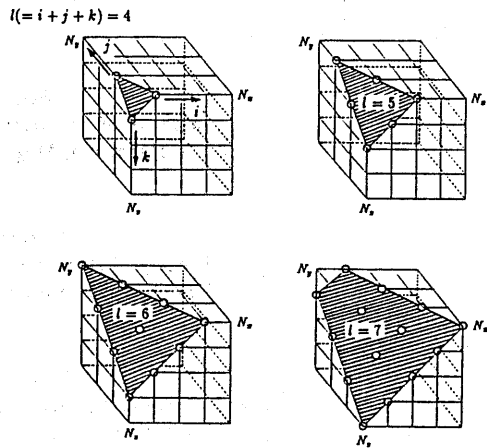


図2 3次元超平面法

一方、3次元7点差分近似の場合、各格子点の指標を (i, j, k) で表すとき計算する順序を $i+j+k =$ 一定のグループ毎に行う方法が3次元超平面法である。図2に $L = 4, 5, 6, 7$ の超平面を示す。この場合、第 $L-1 (= i+j+k)$ 番目の超平面上の格子点のグループが同時に更新(ベクトル化更新)されると、次に第 $L (= i+j+k)$ 番目の超平面上の格子点のグループが同時に更新される。したがって、ICCG法などを使用する際には、通常最初に上の関係式を満足する格子点をグループ単位で配列に収納する。そしてCG法の反復計算の中のLU分解の前進・後退代入計算において計算に必要な番地を、直接隣に書き表す代わりに、最初に上の点列の配列を参照し次に指示された番地に格納されている内容を参照するという間接的な手順を踏んで、始めて必要な計算が出来ることになる。しかし、ベクトル化に関する問題点としてグループ分けされた第 L 番目の超平面上の格子点がメモリーの中で等間隔に位置していないことが挙げられる。したがって超平面法がうまく働くかどうかは、与えられたベクトル計算機上で間接アドレスの処理がどれだけ効率よく行えるかに大きく依存する。そこで次の章で実際にどのくらいバンク・コンフリクトが計算の効率に影響するのかを測定してみることにする。

3 バンク・コンフリクトとは？

通常使用されるベクトル計算機は主記憶領域(メインメモリー)は幾つかのバンクで構成されている。表1に我々がテストで使用したベクトル計算機の主な性能を示す。

製造元	機種	clockcycle	最高速度	バンク数
富士通	VP-200	7 ns	570 Mflops	256
富士通	VP-400E	7 ns	1700 Mflops	256
日本電気	SX-2	6 ns	1300 Mflops	512
日立	S820/80	4 ns	2000 Mflops	128

表1 ベクトル計算機の主な性能

次にメモリー衝突(いわゆるバンク・コンフリクト)について簡単に説明する。たとえば同じバンクが同時にアクセスされるとデータの供給に遅れが生じるため、演算に待ちの状態が発生し、その計算機が本来持つ高性能に比較して計算速度が大幅に低下することをバンク・コンフリクトと呼ぶ。各計算機のハードウェアの構造によって多少その用語なり正確な意味合いが異なることがあるが概ね上の様な現象を指す。それ故、バンク・コンフリクトは避けられるものであればそれに越したことはない。それには、メモリーへのアクセス間隔に注意を払わざるを得ない。さらに、メモリーへのアクセスは連続アドレスよりも間隔が開いたアドレスへのアクセスの方が落ちることは一般的に知られているが、どの程度低下するか？ どの程度我々は注意を払ったらよいのか？ あるいはそれを回避するにはどうしたらよいのか？ などと言う点については書くことをどうも避けている資料が多い。そこで以下の簡単な(2つのベクトル乗算と加算だけを含む)プログラム(図3参照)でアクセス間隔の違いによる効率の低下を調べた。

```

N=100000
KANKAK=24
CALL TIMER(TS)
DO 100 K=1,KANKAK
DO 100 I=1, N*KANKAK, K
A(I)=B(I)+C(I)*D(I)
100 CONTINUE
CALL TIMER(TE)
CPU=TE-TS

```

図3 アクセス間隔と計算速度の関係を調べるプログラム

サブプログラム TIMER は CPU 計測のルーチンである。計算は倍精度で行った。間隔が1(連続アクセス)から24までの結果のみを掲載するが、それ以上の大きさの間隔のときも同様の傾向が現れた。結果を表2に示す。

表の中で*印は4の倍数、#印は8の倍数を表す。どの計算機でも間隔が奇数(各欄の左側)のときは、それが偶数のとき(各欄の右側)よりも効率がよいことがこの結果から分かる。連続アクセスが最も効率がよいのは当然としても、間隔が8の倍数のとき大幅に効率が低下し、ついで4の倍数のときも非常に効率が悪い。機種によって低下率に少し違いがあるが、この数値実験から得られる結論として、我々はメモリーへアクセスするとき8や4の倍数の間隔は極力避け、出来るだけ間隔が奇数になるように工夫すべきだということが分かる。

間隔		VP-200		VP-400E		SX-2		S-820/80	
奇数	偶数	奇数	偶数	奇数	偶数	奇数	偶数	奇数	偶数
1	2	241	70	276	141	454	402	781	398
3	4*	136	35	141	71	402	277	506	199
5	6	136	70	141	141	402	402	535	351
7	8#	136	35	141	36	402	151	615	101
9	10	136	70	141	141	293	293	397	365
11	12*	136	35	141	71	293	277	610	198
13	14	136	70	141	141	293	293	534	362
15	16#	136	33	141	34	293	77	614	51
17	18	137	71	141	141	293	293	619	396
19	20*	137	35	141	71	293	277	614	201
21	22	137	71	141	141	293	293	623	366
23	24#	137	36	141	36	293	151	558	101

表2 等間隔アクセスの計算速度 (Mflops 単位)

4 2次元超平面法でベクトル化した ICCG 法の性能評価

ここでは2次元境界値問題を超平面法でベクトル化した ICCG 法で解いたときの実際のバンク・コンフリクトの影響を論じる。支配方程式と境界条件を式(1),(2),(3)に示す。

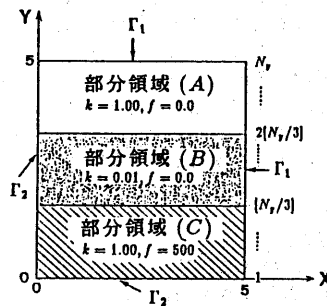
$$\text{div}(-k\nabla\phi) = f \quad (1)$$

$$\phi = 0 \text{ on } \Gamma_1 : \{x = 5, 0 \leq y \leq 5\} \text{ and } \{y = 5, 0 \leq x \leq 5\} \quad (2)$$

$$(-k\nabla\phi) \cdot n = 0 \text{ on } \Gamma_2 : \{x = 0, 0 \leq y \leq 5\} \text{ and } \{y = 0, 0 \leq x \leq 5\} \quad (3)$$

ここで n は単位外向き法線ベクトルを表す。2次元正方形領域 $[0, 5] \times [0, 5]$ は図4に示すような3つの異なる物理定数を持つ部分領域に分割されている。

図4 2次元問題の解析領域



支配方程式を中心差分で離散化すると式(4)のように5点差分近似式で表される。

$$c_{i,j}\phi_{i,j-1} + b_{i,j}\phi_{i-1,j} + d_{i,j}\phi_{i,j} + b_{j,i}\phi_{i+1,j} + c_{j,i}\phi_{i,j+1} = f_{i,j} \quad (4)$$

ICCG 法の収束条件は相対残差 2 乗ノルムで 10^{-6} とし、反復計算の初期値はすべて 0 とした。また格子幅は $\Delta X = 5/(N_x - 1)$, $\Delta Y = 5/(N_y - 1)$, $N_x = N_y$ とし、4 通り ($N_x, N_y = 249, 250, 251, 253$) の格子点数を選んで効率を調べた。演算はすべて倍精度演算である。アクセスする間隔 ($N_x - 1$) はそれぞれ 8 の倍数、奇数、偶数 (4 の倍数ではない)、そして 4 の倍数である。

また Y 方向の格子点の番号は部分領域毎に次のように付けた。

部分領域 (A)	$2[N_y/3]$ から	N_y まで
部分領域 (B)	$[N_y/3]$ から	$2[N_y/3]-1$ まで
部分領域 (C)	1 から	$[N_y/3]-1$ まで

ここで $[n/3]$ は $n/3$ を越えない最大の整数を表すものとする。計算結果を表3に示す。表中で CPU 時間は秒単位である。また比率とは格子点数 $N_x = 250$ のときの CPU 時間を 1 としたときの各割合を表す。4 通りの総格子点数はほとんど等しいので、この比率は $N_x = 250$ の場合 (すなわち $N_x - 1$ が奇数になるとき) に対

する効率の低下度と見なすことが出来る。5点差分近似のICCG法の反復1回当たりの演算量はおよそ $31N_x N_y$ で見積られる。

	格子点数	$N_x - 1$	VP-200	VP-400E	SX-2	S-820/80
(A)	249 × 249	8の倍数	119(0.50)	122(0.50)	363(0.67)	475(0.68)
(B)	250 × 250	奇数	236(1.00)	245(1.00)	538(1.00)	703(1.00)
(C)	251 × 251	偶数	183(0.78)	245(1.00)	539(1.00)	671(0.95)
(D)	253 × 253	4の倍数	119(0.50)	187(0.76)	474(0.88)	612(0.87)

表3 2D問題のバンクコンフリクトの影響 (Mflops単位)

VP-200においては $(N_x - 1)$ が4の倍数のときと8の倍数のときの効率は、 $(N_x - 1)$ が奇数のときの効率のちょうど半分に落ちる。また $(N_x - 1)$ が偶数(4の倍数でない)のときの効率は、それが奇数のときの効率より少し落ちて約78%になる。

同様にVP-400Eでも、 $(N_x - 1)$ が8の倍数のときの効率が $(N_x - 1)$ が奇数のときの効率より非常に悪い。しかし、 $(N_x - 1)$ が偶数のときの効率はそれが奇数のときの効率に等しい。この傾向はSX-2でも現れた。最後に、S-820/80では $(N_x - 1)$ を偶数、4の倍数、そして8の倍数と変えたとき、効率は $(N_x - 1)$ が奇数のときの効率の95%, 87%そして68%と段々落ちてくる。

以上の結果から、 $(N_x - 1)$ が奇数となるようにX方向の格子点数を選ぶことがバンク・コンフリクトの影響を最小限にするであることが分かった。

5 同一超平面上の格子点の番号の間隔について

前章の考察の結果、超平面法でベクトル化したICCG法を使うときX方向の格子数を適切に選べば効率向上につながる事が分かった。そこでこの章では3次元7点差分近似の場合に起こるアクセス間隔について理論的に考察することにする。

いま、各格子点の指標を (i, j, k) で表し格子点の番号は辞書式に付け、そして N_x, N_y, N_z はそれぞれX, Y, Z方向の格子点数を表すものとする。同一超平面上の第1格子点 (i_1, j_1, k_1) から次の第2格子点 (i_2, j_2, k_2) へ移るとき、その格子点の番号の増分値とそれが起こる回数に関する定理を以下証明する。最初に二つの補助定理を証明する。

[補助定理1]

いま、全体集合Xを

$$X = \{(i, j, k) \mid i = 1, \dots, N_x, j = 1, \dots, N_y, k = 1, \dots, N_z, (i, j, k) \neq (N_x, N_y, N_z)\}$$

とおく。そして指標 j, k, l が増加するとき、第1格子点 (i_1, j_1, k_1) の集合をそれぞれ A_1, A_2, A_3 とおく。このとき、この三つの集合の間には次の関係式が成り立つ。

$$A_1 = \bar{C}_1, \tag{5}$$

$$A_2 = C_1 \cap C_2 \cap C_3, \tag{6}$$

$$A_3 = C_1 \cap (\bar{C}_2 \cup \bar{C}_3), \tag{7}$$

ただし、集合 C_1, C_2, C_3 は次のように表される。

$$C_1 = \{X \mid i = 1 \text{ or } j = N_y\}, C_2 = \{X \mid i \neq 1 \text{ or } j \neq 1\}, C_3 = \{X \mid k \neq N_z\}$$

[証明]

最初に、指標 j が増加するときの集合 A_1 を考える。指標 k, l は固定されているから、第1格子点を (i_1, j_1, k_1) とすると第2格子点は $(i_1 - 1, j_1 + 1, k_1)$ となる。したがって、集合 A_1 に含まれる格子点 (i_1, j_1, k_1) は、 $i \neq 1$ かつ $j \neq N_y$ でなければならない。それ故、次の(8)式が成り立つ。

$$A_1 = \bar{C}_1 \tag{8}$$

次に指標 k が増加するときを考える。全体集合Xは三つの集合 A_1, A_2, A_3 の直和で表されるので、

$$A_2 = \overline{A_1 \cup A_3} \subset \bar{A}_1 = C_1$$

が成り立つ。また指標 l は固定されているので第1格子点 (i_1, j_1, k_1) と第2格子点 (i_2, j_2, k_2) の間には、 $i_1 + j_1 = i_2 + j_2 - 1$ の関係が成り立つので、集合 A_2 に含まれる格子点 (i_1, j_1, k_1) は、 $k_1 \neq N_x$ かつ $(i, j) \neq (1, 1)$ でなければならない。すなわち $A_2 \subset (C_3 \cap C_2)$ である。したがって

$$A_2 \subset (C_1 \cap C_2 \cap C_3) \quad (9)$$

が成立する。さらに、この(9)式を使って次の(10)式を証明する。

$$A_2 = (C_1 \cap C_2 \cap C_3) \quad (10)$$

最初に、その準備として集合 A_3 を評価する。集合 A_3 は、

$$A_3 = \overline{A_1 \cup A_2} = \overline{A_1} \cap \overline{A_2} \supset C_1 \cap \overline{(C_1 \cap C_2 \cap C_3)} = C_1 \cap (\overline{C_2} \cup \overline{C_3})$$

と表される。すなわち、

$$A_3 \supset (C_1 \cap (\overline{C_2} \cup \overline{C_3})) \quad (11)$$

が成り立つ。次にこの(11)式の両辺の集合の濃度(すなわち要素の個数)を比較する。いま、集合 A_3 の濃度を $|A_3|$ のように書くと、第3の集合 A_3 の濃度即ち超平面の枚数は、

$$|A_3| = N_x + N_y + N_z - 3 \quad (12)$$

であることが分かる。また、(7)式の中の集合 C_1 と集合 C_2 の要素を実際に調べてみると、

$$C_1 \cap \overline{C_2} = \overline{C_2} \quad (13)$$

の関係が導ける。したがって、

$$C_1 \cap (\overline{C_2} \cup \overline{C_3}) = \overline{C_2} \cup (C_1 \cap \overline{C_3}) \quad (14)$$

が成り立つ。それ故、その濃度は

$$|C_1 \cap (\overline{C_2} \cup \overline{C_3})| = |\overline{C_2}| + |C_1 \cap \overline{C_3}| - |C_1 \cap \overline{C_2} \cap \overline{C_3}| \quad (15)$$

と表される。右辺の三つの集合の各濃度は、第1項の集合は N_x 、第3項の集合は、 $C_1 \cap \overline{C_2} \cap \overline{C_3} = \{(1, l, N_x)\}$ より $|C_1 \cap \overline{C_2} \cap \overline{C_3}| = 1$ である。また、第2項の集合の濃度 $|C_1 \cap \overline{C_3}|$ を見積るために二つの集合 $B_1 = \{X \mid i = 1\}$ と $B_2 = \{X \mid j = N_y\}$ を新たに定義すると、次の4つの関係式が成り立つことがわかる。すなわち、(a) $C_1 = B_1 \cup B_2$, (b) $B_1 \cap \overline{C_3} = \{X \mid i = 1 \text{ and } k = N_x\}$, (c) $B_2 \cap \overline{C_3} = \{X \mid j = N_y \text{ and } k = N_x\}$, そして (d) $B_1 \cap B_2 \cap \overline{C_3} = \{(1, N_y, N_x)\}$ である。

ここで、(15)式の第2項の集合の濃度は上の関係(a)~(d)を使うと、

$$\begin{aligned} |C_1 \cap \overline{C_3}| &= |(B_1 \cup B_2) \cap \overline{C_3}| = |B_1 \cap \overline{C_3}| + |B_2 \cap \overline{C_3}| - |B_1 \cap B_2 \cap \overline{C_3}| \\ &= N_y + (N_x - 1) - 1 = N_x + N_y - 2 \end{aligned} \quad (16)$$

のように表せる。したがって、

$$|C_1 \cap (\overline{C_2} \cup \overline{C_3})| = N_x + (N_x + N_y - 2) - 1 = |A_3| \quad (17)$$

が成り立つ。すなわち、

$$A_3 = C_1 \cap (\overline{C_2} \cup \overline{C_3})$$

である。それ故、最終目標である(10)式が以下のように導ける。

$$\begin{aligned} A_2 &= \overline{A_1 \cup A_3} = \overline{C_1} \cup \overline{(C_1 \cap (\overline{C_2} \cup \overline{C_3}))} \\ &= C_1 \cap C_2 \cap C_3 \end{aligned} \quad (18)$$

Q. E. D.

一方、第2格子点 (i_2, j_2, k_2) の集合についても第1格子点と同様に次の補助定理2が成立する。

[補助定理2]

いま、全体集合 X^* を

$$X^* = \{(i, j, k) \mid i = 1, \dots, N_x, j = 1, \dots, N_y, k = 1, \dots, N_x, (i, j, k) \neq (1, 1, 1)\}$$

とする。そして、指標 j, k, l が増加するとき、第2格子点 (i_2, j_2, k_2) の集合をそれぞれ A_1^* , A_2^* として A_3^* とおくと、この三つの集合の間には次の関係式が成り立つ。

$$A_1^* = \overline{D_1}, \quad A_2^* = D_1 \cap D_2 \cap D_3, \quad A_3^* = D_1 \cap (\overline{D_2} \cup \overline{D_3})$$

ここで、 $D_1 = \{X^* \mid i = N_x \text{ or } j = 1\}$, $D_2 = \{X^* \mid i \neq N_x \text{ or } j \neq N_y\}$, $D_3 = \{X^* \mid k \neq 1\}$ である。

[証明]

補助定理1の証明と本質的に同一であるので、証明は省略する。 Q. E. D.

次にこの二つの補助定理を利用して二つの格子点の番号の差（すなわちメモリーへアクセスする間隔の大きさ）とその回数合計を求める。いま、指標 l を固定すると図5に示すように

(A) 指標 j が増加するとき、(B) 指標 k が増加するとき、の二つの場合に分けて考える。

指標 i については、 $i = l - k - j$ から求めればよい。

(A) 指標 j が増加するとき、補助定理1の集合 A_1 からわかるように $i \neq 1$ かつ $j \neq N_y$ でなければならないので、格子点の番号の増分値 Δd_1 (図5参照) と、それが起きる回数合計 c_1 は次のように表わされる。

$$\begin{aligned} \Delta d_1 &= N_x - 1 \\ c_1 &= (N_x - 1)(N_y - 1)N_x \end{aligned} \quad (19)$$

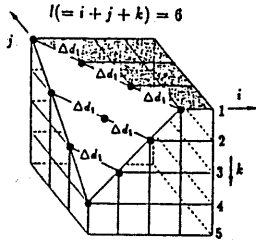


図5 増分 Δd_1 と Δd_2

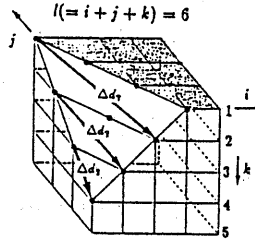
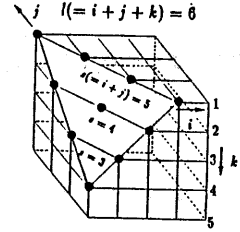


図6 同一超平面上的パラメータ s



$$\begin{cases} N_x = 5 \\ N_y = 4 \\ N_z = 5 \end{cases}$$

(B) 一方、指数 k が増加するとき次の定理が成り立つ。

[定理]

指標 k が増加するとき、第1格子点 (i_1, j_1, k_1) と第2格子点 (i_2, j_2, k_2) の間の番号の増分値 Δd_2 は、

$$\Delta d_2 = N_x N_y - 1 + m(N_x - 1), \quad (20)$$

$$m = \max(1, s - N_x - 1) - \min(s - 1, N_y), \quad s = 3, 4, \dots, N_x + N_y \quad (21)$$

で表される。またその発生回数合計は次のようになる。

$$c_2 = (N_x + N_y - 2)(N_x - 1) \quad (22)$$

[証明]

指標 k が増加したとき、それが属する集合は補助定理1の(6)式より

$$(i = 1 \text{ または } j = N_y) \text{ かつ } (i, j) \neq (1, 1) \text{ かつ } k \neq N_z \quad (23)$$

である。この関係式から指標 i, j と指標 k とは独立の関係で、指標 k を固定して指標 i, j だけで考えてもよいことがわかる。いま、第1格子点 (i, j, k) から第2格子点 $(i^*, j^*, k+1)$ へ移るときの格子の番号の増分値を考える。第1格子点の指標 i, j で $s = i + j$ とおく。このパラメータは図6に示すように指標 k 一定の超平面上的格子点の集合を意味する。このとき指標 l は固定されているから、 $i^* + j^* = s - 1$ が成り立つ。ここで、以下の三つの集合 (ただし $Y = Y_1 \cup Y_2$ である)

$$Y = \{(i, j) \mid i = 1 \text{ or } j = N_y, (i, j) \neq (1, 1)\} \quad (24)$$

$$Y_1 = \{Y \mid i = 1\}, \quad Y_2 = \{Y \mid j = N_y\} \quad (25)$$

を定義する。まず、集合 Y_1 と Y_2 は、

$$Y_1 = \{(1, 2), (1, 3), \dots, (1, N_y)\}, \quad Y_2 = \{(1, N_y), (2, N_y), \dots, (N_x, N_y)\} \quad (26)$$

であるから、これらをパラメータ s を用いて表すと、

$$Y_1 = \{(1, s-1) \mid s = 3, \dots, N_y + 1\}, \quad (27)$$

$$Y_2 = \{(s - N_y, N_y) \mid s = N_y + 1, \dots, N_x + N_y\} \quad (28)$$

となるので、二つの集合 Y_1 と Y_2 を一つにまとめると、

$$\begin{aligned} Y &= Y_1 \cup Y_2 \\ &= \{(\max(1, s - N_y), \min(s - 1, N_y)) \mid s = 3, 4, \dots, N_x + N_y\}. \end{aligned} \quad (29)$$

と書ける。次に、第2格子点の指標 (i^*, j^*) は補助定理2より、

$$i^* = N_x \text{ または } j^* = 1 \text{ かつ } (i^*, j^*) \neq (N_x, N_y) \quad (30)$$

の関係がある。ここで、以下の三つの集合 (ただし $Y^* = Y_1^* \cup Y_2^*$ である)

$$Y^* = \{(i^*, j^*) \mid i^* = N_x \text{ or } j^* = 1, (i^*, j^*) \neq (N_x, N_y)\}, \quad (31)$$

$$Y_1^* = \{Y^* \mid i^* = N_x\}, \quad Y_2^* = \{Y^* \mid j^* = 1\}. \quad (32)$$

を定義する。また集合 Y_1^* と Y_2^* は、

$$Y_1^* = \{(N_x, 1), (N_x, 2), \dots, (N_x, N_y - 1)\} \quad (33)$$

$$Y_2^* = \{(1, 1), (2, 1), \dots, (N_x, 1)\} \quad (34)$$

であるから、これらをパラメータ s の関係式、すなわち $i^* + j^* = s - 1$ を用いて表すと、

$$Y_1^* = \{(N_x, s - 1 - N_x) \mid s = N_x + 2, \dots, N_x + N_y\}, \quad (35)$$

$$Y_2^* = \{(s - 2, 1) \mid s = 3, 4, \dots, N_x + 2\} \quad (36)$$

となる。同様に二つの集合 Y_1^* と Y_2^* を一つにまとめると次のようになる。

$$\begin{aligned} Y^* &= Y_1^* \cup Y_2^* \\ &= \{(\min(N_x, s - 2), \max(1, s - N_x - 1)) \mid s = 3, 4, \dots, N_x + N_y\} \end{aligned} \quad (37)$$

したがって、第1格子点 (i, j, k) から第2格子点 $(i^*, j^*, k+1)$ へ指標 k が増加するとき、すべて次の形

$$\left(\max(1, s - N_y), \min(s - 1, N_y), k \right), \quad (38)$$

$$\rightarrow \left(\min(N_x, s - 2), \max(1, s - N_x - 1), k + 1 \right), \quad (39)$$

$$3 \leq s \leq N_x + N_y, \quad 1 \leq k \leq N_x - 1 \quad (40)$$

に整理できることがわかる。それ故、そのときの増分値は、

$$\begin{aligned} \Delta d_2 &= N_x N_y + N_x(j^* - j) + (i^* - i) \\ &= N_x N_y + N_x(\max(1, s - N_x - 1) - \min(s - 1, N_y)) \\ &\quad + \min(N_x, s - 2) - \max(1, s - N_y) \\ &= N_x N_y + N_x(\max(1, s - N_x - 1) - \min(s - 1, N_y)) \\ &\quad + (s - 1 - \max(1, s - N_x - 1)) - (s - \min(s - 1, N_y)) \\ &= N_x N_y - 1 + (N_x - 1)(\max(1, s - N_x - 1) - \min(s - 1, N_y)) \end{aligned} \quad (41)$$

で見積られる。またこの増分値が発生する回数の合計はつぎのようになる。

$$c_2 = (N_x + N_y - 2)(N_x - 1) \quad (42)$$

Δd	(偶数、偶数)	(偶数、奇数)	(奇数、偶数)	(奇数、奇数)
奇数間隔の 合計回数	$N_x N_x (N_y - 1)$ $-N_y - N_x + 2$	$(N_y N_x - 1)$ $\times (N_x - 1)$	$(N_x + N_y - 2)$ $\times (N_x - 1)$	0
偶数間隔の 合計回数	$N_x (N_x - 1)$	$(N_y - 1)$ $\times (N_x - 1)$	$(N_x - 1)$ $\times (N_y - 1) N_x$	$N_x N_y N_x + 2$ $-N_x - N_y - N_x$

表4 (a) $N_x \geq N_y$ のときの奇数間隔と偶数間隔の合計回数

Δd	(偶数、偶数)	(偶数、奇数)	(奇数、偶数)	(奇数、奇数)
奇数間隔の 合計回数	$N_x N_x (N_y - 1)$ $-N_y - N_x + 2$	$N_x N_x (N_y - 1)$ $-N_y - N_x + 2$	$(N_x + N_y - 2)$ $\times (N_x - 1)$	0
偶数間隔の 合計回数	$N_x (N_x - 1)$	$N_x (N_x - 1)$	$(N_x - 1)$ $\times (N_y - 1) N_x$	$N_x N_y N_x + 2$ $-N_x - N_y - N_x$

表4 (b) $N_x \leq N_y$ のときの奇数間隔と偶数間隔の合計回数

表4 (a) (b) に増分 Δd が奇数あるいは偶数の合計の回数を示す。この表の中で (偶数、偶数)、(偶数、奇数)、(奇数、偶数) そして (奇数、奇数) は、それぞれ ($N_x =$ 偶数、 $N_y =$ 偶数)、($N_x =$ 偶数、 $N_y =$ 奇数)、($N_x =$ 奇数、 $N_y =$ 偶数) そして ($N_x =$ 奇数、 $N_y =$ 奇数) を意味する。

この表の中で (偶数、偶数)、(偶数、奇数) の2つの場合が奇数の間隔の回数が、偶数の間隔の回数に比べて非常に多いことが分かる。我々は前の章でメモリーにアクセスする間隔は奇数にすべきだと、あるいは間隔は出来るだけ多く奇数にした方がよいことを知っている。したがって、この表4の結果は我々に各軸方向の格子点数の適切な選び方のヒントを与えてくれる。すなわち、 N_x 、 N_y 両方とも偶数あるいは N_x を偶数、 N_y は奇数とすればベクトル計算機の高性能を得ることができる。第6章で数値実験を行いこれを検証する。

6 3次元超平面法でベクトル化した ICCG 法の性能評価

ここでは3次元境界値問題を超平面法でベクトル化した ICCG 法で解いたときの実際のバンク・コンフリクトの影響を第5章の定理の結論を踏まえて論じることにする。ここで取り扱う問題は第4章の式 (1), (2), (3) を拡張したものである。

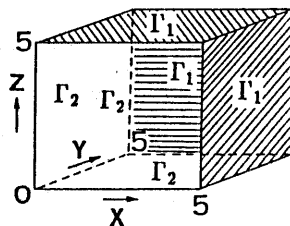
$$\text{div}(-k\nabla\phi) = f, \quad k = 1.0, \quad f = 500 \tag{43}$$

$$\phi = 0 \text{ on } \Gamma_1 : \{x = 5, 0 \leq y, z \leq 5\}, \{y = 5, 0 \leq x, z \leq 5\} \text{ and } \{z = 5, 0 \leq x, y \leq 5\} \tag{44}$$

$$(-k\nabla\phi) \cdot n = 0 \text{ on } \Gamma_2 : \{x = 0, 0 \leq y, z \leq 5\}, \{y = 0, 0 \leq x, z \leq 5\} \text{ and } \{z = 0, 0 \leq x, y \leq 5\} \tag{45}$$

図7に示す3次元立方体領域 $[0, 5] \times [0, 5] \times [0, 5]$ の中で k は1に固定した。

図7 3次元問題の解析領域



支配方程式は中心差分で離散化され、次の式のような形の7点差分近似式で表される。

$$e\phi_{i,j,k-1} + c\phi_{i,j-1,k} + b\phi_{i-1,j,k} + d\phi_{i,j,k} + b\phi_{i+1,j,k} + c\phi_{i,j+1,k} + e\phi_{i,j,k+1} = f_{i,j,k} \tag{46}$$

ICCG法の収束条件は相対残差2乗ノルムで 10^{-6} 、反復計算の初期値は全て0、演算は倍精度、また格子幅 $\Delta X = 5/(N_x - 1)$, $\Delta Y = 5/(N_y - 1)$, $\Delta Z = 5/(N_z - 1)$ とした。格子点数は6通り調べた。総格子点数は約10万点である。最初の(A)と(B)は $(N_x - 1)$ が奇数、次の(C)と(D)は $(N_x - 1)$ が偶数(4の倍数ではない)、最後の(E)と(F)は $(N_x - 1)$ が4の倍数とした。

	N_x	N_y	N_z	総格子点数
(A)	50 (奇数)	50 (奇数)	39	97500
(B)	50 (奇数)	49 (4の倍数)	40	98000
(C)	51 (偶数)	50 (奇数)	39	99450
(D)	51 (偶数)	51 (偶数)	38	98838
(E)	49 (4の倍数)	50 (奇数)	40	98000
(F)	49 (4の倍数)	49 (4の倍数)	41	98441

表5 3D問題の格子点数

計算結果を表6に示す。3次元7点差分式で離散化を行い得られた連立1次方程式をICCG法で解く。計算効率(Mflops)は次の式で計算した。但しNは総格子点数、Loopは反復回数を表す。

$$Mflops = \frac{41N + 39N \times Loop}{CPU \times 10^6} \quad (47)$$

括弧の中の数値は最も速度が遅い(F)を1.0としたときの比率を表す。

	格子点数	VP-200	VP-400E	SX-2	S-820/80
(A)	50 × 50 × 39	255(2.34)	177(1.53)	133(1.02)	724(2.76)
(B)	50 × 49 × 40	248(2.28)	177(1.53)	133(1.02)	695(2.65)
(C)	51 × 50 × 39	196(1.80)	177(1.53)	133(1.02)	697(2.66)
(D)	51 × 51 × 38	187(1.72)	177(1.53)	133(1.02)	684(2.61)
(E)	49 × 50 × 40	121(1.11)	127(1.09)	133(1.02)	414(1.58)
(F)	49 × 49 × 41	109(1.00)	116(1.00)	130(1.00)	262(1.00)

表6 3D問題の計算速度(Mflops単位)の比較

全ての計算機において(F)のケースが最も効率が悪く、ついで(E)のケースが悪い。これらはいずれも $N_x - 1$ が8の倍数にあたる。(F)はその上 $(N_y - 1)$ も8の倍数である。

またVP-200とS-820/80の結果はだいたい同じ傾向を示している。しかしVP-200では $N_x - 1$ が4の倍数になる(A),(B)のケースが、(C),(D)の $N_x - 1$ が偶数(4の倍数でない)になるケースより効率がよいのに対して、S-820/80では(C),(D)は(A),(B)とだいたい同じ効率であった。一方、VP-400では(A)から(D)までが同じ効率であった。しかしSX-2ではケース(F)は(A),(B)の結果と大差なく(A)や(B)を選ぶ限り効率的に何ら問題はない。

7 まとめ

超平面法を使うとき、各軸方向の格子点数の選択はベクトル計算機を有効利用するために重要な要素である。本論文ではメモリーにアクセスする間隔の大きさを正確に見積り、また奇数の間隔と偶数の間隔の合計回数も正確に見積った。結論として、バンク・コンフリクトの影響を少なくするためには出来るだけ多く奇数間隔でメモリーへアクセスするとよい。3次元7点差分のとき $N_x - 1$ を奇数(N_x はX方向の格子点数)にすることが最も重要である。

8 参考文献

- [1] H.A. Van der Vorst, HIGH PERFORMANCE PRECONDITIONING, SIAM J. SCI. STAT. COMPUT, Vol.10, 6(1989), pp.1174-1185
 [2] 後 保範、ベクトル計算機向き不完全三角分解、京都大学数理解析研究所、講義録 No.514, 1984