数理解析研究所講究録
第717巻 1990年 58-67

一般曲線座標系における反復解法（ＳＯＲ法）の高速化について

藤野清次（計算流体力学研究所）(Fujino Seiji)
田村哲郎（清水建設大崎研究室）(Tamura Tetsurou)
森　正武（東京大学　工学部　）(Mori Masatake)

1．はじめに

スーパーコンピュータの最高性能を発揮させるには、

（１）出来るだけ長いベクトル長の配列を確保すること

（２）再帰的な演算を出来るだけ排除すること

（３）メモリーへのデータの参照・格納を連続的にすること

などが特に重要である。更に出来上がったプログラムの汎用性やわかり易さ等も考慮に入れると

（４）配列のアドレス計算が簡明であること

等も効率を上げるためには必要になってくる。また数値解法が本来持つ精度や収束性自身をベクトル化等のために落とさないことも非常に重要である。
これら基本的な原則からはずれた計算技法を使っている場合、その計算技法の理論面及びプログラミング両面から検討が必要になってくる。

またスーパーコンピュータは機種ごとの「個性」が強く、その計算技法の有効性は機種によって大きな差が生じることがある。したがってより汎用性のある計算技法の開発が望ましい。

本論文では流体解析等ですでに確立されている一般曲線座標系で現れる連立１次方程式について考えることにする。一般曲線座標系とは任意形状の物体に沿って計算格子を張った物理空間から計算空間に支配方程式を写像して計算を行う方法の一つである。通常の直交座標系に比べて一般曲線座標系では交差微分項が現れ、離散化のための点数が増加する。更に例えばポアソン方程式から得られる連立１次方程式の係数行列なども対称性が崩れ非対称行列になる。したがってよく研究されている直交座標系の場合はともかく、一般曲線座標系においてもそこで現れる大規模スパース連立１次方程式の反復解法（ここではＳＯＲ法を取り上げる）の効率化について研究することは十分意義があると考える。

2．MULTI-COLOR技法によるＳＯＲ法の高速化

(2-1)計算順序の並替え

よく用いられているＳＯＲ法はそのままの計算順序ではベクトル化が出来ないことはよく知られている。そこで色々なベクトル化技法が提案されてきた。上で述べた基本原則（１）を主眼に考え出された擬似ＳＯＲ法［１］や、（２）を中

心に考え出されたhyper-plane法［２］や，直交座標系でのRed-Black法ＳＯＲ法［３］などがある。これらは本来のＳＯＲ法の計算の順序を各々の目的に合わせて並替えて計算を進めるものと言える。この並替えを行ってもＳＯＲ法本来の収束性は変わらないことは理論的に証明されている。［4］,［5］そこで得られた結論を要約すると以下のようになる。

(a)ＳＯＲ法は種々の並び替えをおこなっても最適な加速係数は等しい。

(b)僅かに生ずる収束状況の違いは、初期残差ベクトルの固有ベクトル成分が異なった分布をするために起こる。

これらの研究の成果から、ベクトル計算機用の効率を重視した並替えを行なっても数学的な収束率を損なうことは無いとの裏付けが得られたので、今後は安心して並替えの議論を進めることが出来る。

(2-2)一般曲線座標系でのＳＯＲ法の並替えについて

例えば３次元一般曲線座標系に於けるポアソン方程式は図１に示されるように２次精度の中心差分で離散化すると１９点近似式になる。この場合近似式に使われる点の数が増えれば増える程効率的な計算（即ちベクトル化）は難しくなる。何故なら、更新された新しいデータを順番に使えば使う程速く収束するが、それと引き換えに演算は再帰的になりベクトル化が困難になる。一方古いデータだけを使うとベクトル化は何ら問題なく行なえるが収束は極端に遅くなってしまう。たとえばJacobi法はその代表的なものである。是非とも数学的な収束率を保持したままベクトル長の長い計算技法を考案する必要がある。

本論文で提案する方法は、再帰的な演算が無いように３次元配列を１次元的に表現した技法と言えよう。　即ち各々の格子点を色分けし、その格子点の色と関係する周囲の１８個の格子点の色とが全て違うように、即ち再帰的関係が無いようにしたものである。種々の色分けのうち最少の分け方が７色であるという原理に本技法は基づく。したがってベクトル長は反対に最も長いＯ（Ｎ／７）が確保出来る。これが本技法の最大の特長である。（図２参照）

更にアドレス計算を単純化するために、各軸方向の格子点数に以下の様な約束を設けることにする。

(1)第１軸の格子点数：　７ｍ＋Ｐ　（ｍ＝1,2,3　）
(2)第２軸の格子点数：　７ｎ－Ｐ　（ｎ＝1,2,3　）
(3)第３軸の格子点数：　任意

但しＰ＝2,3,4,5 とする。（［6］,［7］参照）この様にすることによって初めて、ベクトル長がＯ（Ｎ／７）と非常に長いベクトル計算機向きＳＯＲ法が実現する。格子の番号付けは自然なOrdering、即ち(1,1,1),(2,1,1),(3,1,1,),…,(NX,1,1),(1,2,1),(2,2,1),……,(Nx,NY,1),(1,1,2),……,(NX,NY,NZ)

とする。この時、７色ＳＯＲ法の計算順序は

(1)第１番目の色の格子点、1,8,15,22,29,……

(2)第２番目の色の格子点、2,9,16,23,30,……

(3)第３番目の色の格子点、3,10,17,24,31,……

（第４、５、６番目の色の格子点も同様とする。）

(7)第７番目の色の格子点、7,14,21,28,35,……

となる。<Appendix 1.にプログラム例を示す>

この計算技法は最初に述べた高速化の基本原則（1），(2)，(4)を主眼に考えたアルゴリズムである。

(2-3)７色ＳＯＲ法の効率の数値実験

式（１）に示す２階の微分項を含む次の恒等式を考え、７色ＳＯＲ法の効率の数値実験を行なった。

$$U_{xx}+U_{yy}+U_{zz}+U_{xy}+U_{yz}+U_{zx}=f \qquad (1)$$

解析領域は [0,1] × [0,1] × [0,1] であり、右辺項ｆは以下の様に与えられる。

$$f(x,y,z)=e^{xyz}\{(xy)^2+(yz)^2+(zx)^2+(x+y+z)(1+xyz)\} \qquad (2)$$

この時真の解は $U=e^{xyz}$ となる。各項を２次精度の中心差分で離散化する。例えば $U_{xy}=0.25\,(U_{i+1,j+1}+U_{i-1,j-1}-U_{i+1,j-1}-U_{i-1,j+1})$ となる。(kは省略する）離散化方程式式は全体として１９点近似式になる。収束条件は相対残差L_2ノルムで4×10^{-4}、初期値は全て0、計算は単精度演算、格子点数は約１万点から約１０万点まで１０ケースについてテストを行なった。計算結果を表１に示す。

表１．７色ＳＯＲ法の計算速度（単位:Mflops）

格子点数	VP-200	VP-400E	SX-2	S-820/80
(1) 30x19x18	278	363	844	852
(2) 30x26x26	280	361	854	809
(3) 37x26x31	280	362	868	869
(4) 37x33x33	282	370	870	859
(5) 44x33x34	281	368	872	919
(6) 44x40x34	281	367	874	865
(7) 51x40x34	282	371	875	824
(8) 51x47x33	281	370	874	872
(9) 58x47x33	282	370	876	828
(10) 58x54x32	282	372	880	886
平均速度	281	367	869	858

各ベクトル計算機の最大演算速度は、VP-200は 570Mflops, VP-400Eは1700 Mflops, SX-2は 1300Mflops, そしてS-820/80は 2000Mflopsである。
７色ＳＯＲ法の平均速度が、それぞれの計算機の最大演算速度に対してどのくらいの比率に相当するかを計算してみると、それぞれVP-200は49%, VP-400Eは22%, SX-2は67% そして S-820/80は43%になった。
一般に数値解法の計算速度を比較する場合（精度を比較する場合と違い）、どうしても相対的な比較になることが多い。この様な場合、効率の良し悪しの判断は比較する相手の選び方によって大きく左右される。そこで本論文ではより普遍的で、かつ実際的な有効性の議論をするために、絶対的な基準に近いものとして計算速度のMflops値を採用することにした。３次元の１９点近似式をＳＯＲ法で解く場合、四則演算の回数は一定である。即ちＮを未格子点数とした時、誤差見積も含めて２１Ｎ個の加減算と２１Ｎ個の掛算、合計４２Ｎ個の演算が必要になり、次の（３）式でMflops値は計算出来る。

$$Mflops=42N \times Iter/(CPU\text{-}time \times 10^6) \qquad (3)$$

ここでＮは総格子点数,Iterは反復回数そしてCPU-timeは秒単位を表すものとする。
参考までに他の解法の結果を次の表２に示す。使用した計算機がVP-400Eと S-820/80の場合のみを示すが他の計算機でも同様の結果が得られた。（［８］参照）

表２．他のＳＯＲ法の計算速度（単位:Mflops）

格子点数	擬似SOR法	Checker-Board SOR法
(1) 30x19x18	181 [82]	93 [50]
(2) 30x26x26	172 [89]	89 [54]
(3) 37x26x31	204 [94]	109 [58]
(4) 37x33x33	196 [98]	105 [63]
(5) 44x33x34	232 [105]	127 [65]
(6) 44x40x34	230 [111]	126 [69]
(7) 51x40x34	265 [118]	146 [74]
(8) 51x47x33	257 [127]	142 [82]
(9) 58x47x33	291 [133]	161 [87]
(10) 58x54x32	295 [136]	164 [93]
平均速度	232 [109]	126 [70]

*各欄で左側の数値はS-820/80, [ ] 内の数値はVP-400Eの結果を表す。

表１、表２から７色ＳＯＲ法が他のＳＯＲ法に比べて圧倒的に効率がよいことがわかる。　ただしこの表の中で、擬似ＳＯＲ法はChecker-Board SOR法よりMflops値が大きく一見効率的に見える。しかしこの方法は１方向のみJacobi法を適用

しベクトル化したものなので、この方法の収束率はＳＯＲ法本来の収束率と比べて大幅に低下している。［１］ したがって収束までの反復回数は通常のＳＯＲ法より非常に多くなり効率が落ちる。

しかし表１からわかる様に各ベクトル計算機の最大演算速度との比ではまだ低率なので、この７色ＳＯＲ法はまだ改良の余地が残されていると推測出来る。（計算速度の絶対的評価基準導入のポイントがここに有る。）

３．データを等間隔で参照・格納するときの効率

第２章で述べた７色ＳＯＲ法の特長の１つは、アドレス計算の簡略化のために採用した７つおきに同じ色の格子点を配置することであった。このことはプログラムの中では例えば解ベクトル用の配列は１つで済む代わりに、データの参照や格納は当然７つおきになることを意味している。ところが汎用のベクトル計算機（正確には、数値実験で使用した上記の計算機では）に於いてはデータを連続的に参照・格納した時にその計算機が持つ最高性能が発揮されるように設計されている。したがって、上の数値実験で得られた７色ＳＯＲ法の効率はその影響を少なからず受けている（低下している）と推察される。（どの程度効率が低下するか知りたいのだが、残念ながらそれを記述した資料は余り多くない。） そこで以下の様な簡単なプログラムを作成し、その低下率を見積った。

```
（例）      L=N * m                （m個おきの参照・格納）
            DO 100 i=1,L,m
            A(i)=b(i)+C(i)*D(i)
  100       continue
```

図３．等間隔データの参照・格納の検証用プログラム
（演算は単精度、データ数は１０万個x５００回）

結果を表４にまとめた。連続的にデータを参照・格納した時に比べてどの程度効率が低下するかが知りたいポイントで、Mflops値そのものには大した意味は無い。

表４．等間隔データの参照・格納時の計算速度

	VP-200	VP-400E	SX-2	S-820/80
連続	350(100%)	445(100%)	469(100%)	517(100%)
２個おき	290	320	454	563
３個おき	190	185	398	450
４個おき	110	160	398	253
５個おき	205	210	398	520
６個おき	220	225	398	517
７個おき	110(31%)	185(42%)	398(85%)	526(102%)

表４からわかるように、連続的に計算した場合を各々100%とした時、計算機によっては２個以上等間隔にデータを参照・格納すると効率がかなり落ちる。特に７個おきの場合効率が約30%と大幅に落ちてしまう。特にVPの２機種はその傾向が目だつ。またS-820/80の場合はCPU計測のばらつきが大きく再考の余地が有る。それに対してSX-2の場合はほとんど効率が低下しないことがわかる。この結果を元に７色ＳＯＲ法に次のような改良を施した。

４．新しい７色ＳＯＲ法の効率について

本論文では高速化が主目的であるため、第３章の得られた知見から判断して、汎用性を少し犠牲にして等間隔アドレッシングを見送った方がよいことがわかる。そこで第１章で述べた高速化のための基本原則（１），（２）は従来通り踏襲し、更に（３）の指針を取り入れた新しい７色ＳＯＲ法を考えることにする。

そのためには、各色に対応してそれぞれ配列を持たせさえすれば解決する。すなわちプログラムは、係数行列用配列が７個、右辺定数用配列も７個そして解ベクトル用配列も７個必要になってくる。そうすれば各配列へのデータの参照・格納は連続的に出来ることになる。唯、この場合プログラムのステップ数は当然改良前の７倍に増大してしまい、配列毎のアドレス指定が煩雑になる。
(2-3)と同じ問題を新しい７色ＳＯＲ法で解いた。結果を表５に示す。

表５．新しい７色ＳＯＲ法の計算速度（単位:Mflops）

格子点数	VP-200	VP-400E	SX-2	S-820/80
(1) 30x19x18	441	568	996	1504
(2) 30x26x26	458	561	1002	1479
(3) 37x26x31	453	562	1017	1506
(4) 37x33x33	458	570	1030	1496
(5) 44x33x34	461	568	1024	1542
(6) 44x40x34	458	570	1028	1555
(7) 51x40x34	460	571	1029	1573
(8) 51x47x33	457	570	1025	1503
(9) 58x47x33	459	570	1035	1560
(10) 58x54x32	461	572	1034	1567
平均速度	457	568	1022	1529

新しい７色ＳＯＲ法の平均速度は、表１の結果と比較して、VP-200では1.63倍、VP-400Eでは1.55倍、SX-2では1.17倍、そしてS820/80では1.78倍と速くなった。したがって計算機の最大演算速度に対する比率は、VP-200は80%，VP-400Eは33%，SX-2は79% そして S-820/80は76%となり、VP-400Eを除きかなりの高率を達成することが出来た。
なお、今回のテストに使用したプログラムは、CPU計測のルーチンと強制ベクトル命令以外は同一である。また一般曲線座標系で解析するプログラムに、新しい７色ＳＯＲ法のプログラムを実際に組込むには若干の注意が必要になる。
Appendix 2.にまとめたので参照されたい。

5. まとめ

一般曲線座標系の下で現れる３次元の１９点近似のポアソン方程式に対してベクトル長がＯ（Ｎ／７）で見積られる７色ＳＯＲ法を提案した。この解法に基づいたプログラムを複数の汎用ベクトル計算機上で効率を測定し、一部を除いて最大演算能力の75%から80%の効率を達成することが出来た。またベクトル計算機を使用する上で有効な技法も報告した。

6. 参考文献

［１］杉原、小柳、森、「ベクトル計算機におけるＳＯＲ法的方法について」、数値解析研究会、情報処理学会、1987.3

［２］村田、小国、唐木、「スーパコンピュータ（科学技術計算への適用）」、丸善、1985

［３］M.Ortega,'Introduction to Parallel and Vector solution of Linear Systems', Pleunum Press, 1988

［４］横川三津夫、「逐次過大緩和法（ＳＯＲ法）のベクトル・パラレル処理」JAERI-M,日本原子力研究所、1988

［５］Loyce M.Adams, Harry F.Jordan, 'Is SOR color-blind ?',SIAM.J.Sci. Stat. Comput., Vol.7, No.2, April, 1986

［６］Fujino, S., Tamura,T. and Kuwahara,K.,"Application of the RAINBOW SOR technique to Fluid Flow Analysis in the 3-D generalized Curvilinear Coordinate System", Proceedings of 6th International Conference on 'Numerical methods in Laminar and Turbulenct Flow', Swansea, U.K., July, 1989

［７］Fujino, S., Tamura,T. and Kuwahara,K.,"Multicolored Poisson Solver for Fluid Flow Problems", Proceedings of the 8th GAMM Conference on Numerical Methods in Fluid Mechanics, University of Technology Delft, The Netherlands, September ,1989

［８］藤野、田村、「一般座標系２、３次元に於けるＳＯＲ法のベクトル化について」、数値解析研究会、情報処理学会、1989.10

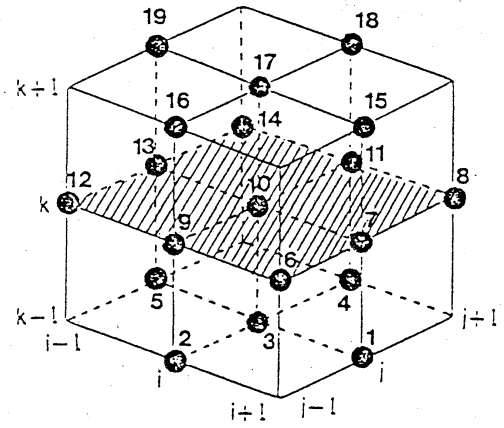

図１３次元一般曲線座標系に於ける１９点近似

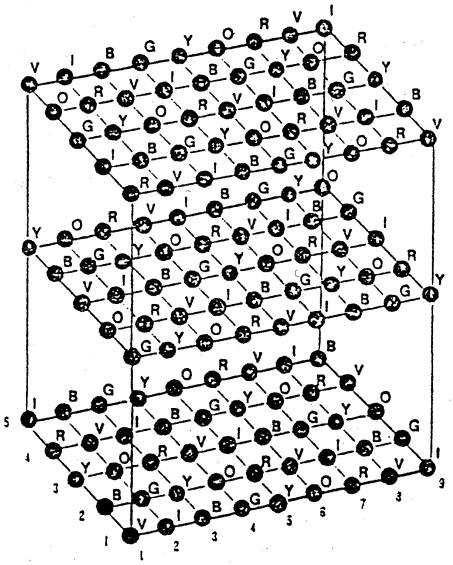

図２　７色の色分け

<Appendix 1.>

７色ＳＯＲ法のプログラム例

外側のループが７色の色に対応し、内側のループは先頭番地と計算間隔を表している。

```
          PARAMETER (MX=58,MY=54,MZ=32)
          PARAMETER (N=MX*MY*MZ,M1=MX,M2=MX*MY)
          DIMENSION  XS(1-M2:N+M2),AS(N+M2,19),BS(N)

          C ******  RAINBOW SOR  ******
000273    400    CONTINUE
000274           ER=0.0
000275 S         DO 410 K=1,7
          *VOCL LOOP,NOVREC
000276 V         DO 410 L=K,N,7
000277 V         GOSA=BS(L)-(AS(L,1)*XS(L-M2-M1)+AS(L,2)*XS(L-M2-1)+
                +      AS(L,3)*XS(L-M2)    +AS(L,4) *XS(L-M2+1)+
                +      AS(L,5)*XS(L-M2+M1)+AS(L,6) *XS(L-M1-1)+
                +      AS(L,7)*XS(L-M1)    +AS(L,8) *XS(L-M1+1)+
                +      AS(L,9)*XS(L-1)     +AS(L,10)*XS(L)+AS(L,11)*XS(L+1)+
                +      AS(L,12)*XS(L+M1-1)+AS(L,13)*XS(L+M1)+
                +      AS(L,14)*XS(L+M1+1)+AS(L,15)*XS(L+M2-M1)+
                +      AS(L,16)*XS(L+M2-1)+AS(L,17)*XS(L+M2)+
                +      AS(L,18)*XS(L+M2+1)+AS(L,19)*XS(L+M2+M1))
000278 V         ER=ER+GOSA*GOSA
000279 V         XS(L)=XS(L)+OMEGA*GOSA
000280 V 410     CONTINUE
```

<Appendix 2.>

新しい７色ＳＯＲ法の実際のImplementation時の注意

実際に新しい７色ＳＯＲ法を一般曲線座標系の解析プログラムの中にImplementする場合以下の事に留意する必要がある。本文で述べたように、新しい７色ＳＯＲ法では効率化を図るために、連立１次方程式用の配列として、係数行列用に７個、右辺項用に７個、そして解ベクトル用に７個それぞれ１次元配列を準備する必要がある。

ところが,一般的に３次元の物理空間に対応して配列を３次元配列:P(i,j,k)にとった方がわかり易くかつ便利なことが多い。一方本方法ではX1(i),X2(i),X3(i),X4(i),X5(i),X6(i),X7(i)という各１次元配列で計算を行なう。そのため配列Pから配列（X1,X2, ...,X7）へのデータ転送（逆方向へのデータ転送も含む）も効率よくした方がよい。ＳＯＲ法の反復計算の最初と最後の２回だけ転送すればよいケースではその負荷は大したことはない。しかし反復の途中で、何回も右辺項を修正したり境界条件の処理をしたりするケースも多く、元の物理空間に対応した３次元配列を直接取り扱い出来る方（アドレス計算などの手間のため）が都合が良い。その場合データ転送のオーバヘッドはかなり大きくなる。そこで考えられる３つのデータ転送方法の性能比較を行なった。

(1)番地逐次加算型

```
482          CALL  XCLOCK(SEC0,5)
483          S1=0.0
484          DO 120 LL=1,100
485          L=0
486          DO 20 K=1,MZ
487          DO 20 J=1,MY
488          DO 20 I=1,MX
489  V       L=L+1
490  V       X(L)=P(I,J,K)
491      20  CONTINUE
492          S1=S1+1.0
495      120 CONTINUE
498          CALL  XCLOCK(SEC1,5)
499          CPU1=SEC1-SEC0
```

(2)間接番地使用

```
501          CALL  XCLOCK(SEC0,5)
502          S2=0.0
503          DO 130 LL=1,100
504          DO 30 K=1,MZ
505          DO 30 J=1,MY
506          DO 30 I=1,MX
507  V       X(LIS(I,J,K))=P(I,J,K)
508      30  CONTINUE
509          S2=S2+1.0
510      130 CONTINUE
511          CALL  XCLOCK(SEC1,5)
512          CPU2=SEC1-SEC0
```

(3)サブプログラム使用

```
541          CALL  XCLOCK(SEC0,5)
542          S4=0.0
543          DO 150 LL=1,100
545  S       CALL  NEW(P,N7)
546  S       S4=S4+1.0
547      150 CONTINUE
548          CALL  XCLOCK(SEC1,5)
549          CPU4=SEC1-SEC0
```

```
800          SUBROUTINE NEW(X,M7)
900          DIMENSION X(7,M7)
921          PARAMETER (MX=080,MY=080,MZ=080)
925          PARAMETER (N=MX * MY * MZ)
930          PARAMETER (N7=(N+6)/7)
950          COMMON/COM1/X1(N7),X2(N7),---X7(N7)
1000         DO 10 I=1,N7
1100 V       X1(I)=X(1,I)
1200 V       X2(I)=X(2,I)
1300 V       X3(I)=X(3,I)
1310 V       X4(I)=X(4,I)
1320 V       X5(I)=X(5,I)
1330 V       X6(I)=X(6,I)
1380 V       X7(I)=X(7,I)
1385     10  CONTINUE
1390         RETURN
1395         END
```

数値実験結果を表Ａに示す。方法(3)が最も効率がよいことがわかる。
ただしこの場合ベクトル長が（Ａ）５０、（Ｂ）８０と非常に短いため、下の結果は必ずしもその機種の性能全体を表しているわけではない。ここでの関心は３次元配列から１次元配列へのデータ転送時間を短くする方法を見つけることにある。

表Ａ．３次元配列から１次元配列へのデータの転送速度

（Ａ）データ数が50×50×40個の場合（単位：mili sec）

	VP-200	VP-400E	SX-2	S-820/80
(1)番地逐次加算型	2.72	2.16	0.94	0.87
(2)間接番地使用	19.0	18.9	2.16	3.09
(3)サブプログラム使用	0.93	0.56	0.29	0.27

（Ｂ）データ数が80×80×80個の場合（単位：mili sec）

	VP-200	VP-400E	SX-2	S-820/80
(1)番地逐次加算型	12.0	9.31	4.50	3.86
(2)間接番地使用	94.7	94.0	9.83	14.7
(3)サブプログラム使用	4.68	2.77	1.49	1.38