LISTVEC 指示行を使った多粒子シミュレーションの大規模化 ——主メモリを節約し，かつ高速化を可能にする 1 つの方法

杉山 徹[†,☆] 寺田直樹[††] 村田健史[†††]
大村善治[†,☆☆] 臼井英之[†,☆☆] 松本紘[†,☆☆]

宇宙空間プラズマを粒子的に扱うシミュレーションを行う場合は，主に PIC（Particle-In-Cell）法と呼ばれる手法を用いて実行される．すなわち，場の量（電流，電磁場など）を空間格子点上で定義し，プラズマ粒子を空間にランダムに配置する．その中で，粒子の運動と場の量を結びつけるために粒子の速度モーメントを計算するが，この計算をベクトル化して行うには，粒子がランダムに分布しているため工夫を要する．本論文では，従来の方法と，地球シミュレータなどに使われている SX ベクトル計算機で使用可能なコンパイラ指示オプションを用いた方法とを比較した結果を報告する．なお，実際のアルゴリズムは，NEC が保持する特許事項である．

Vectorized Particle Simulation Using “LISTVEC” Compile-directive on SX Super-computer

Tooru Sugiyama,[†,☆] Naoki Terada,[††] Takeshi Murata,[†††]
Yoshiharu Omura,[†,☆☆] Hideyuki Usui[†,☆☆]
and Hiroshi Matsumoto[†,☆☆]

PIC (Particle-In-Cell) method is frequently used for space plasma particle simulations. In this method, the field components (i.e. current, electric/magnetic field) are defined on grids and plasma particles are randomly distributed in space. To obtain the current, we need to calculate the velocity moment of the particles. The calculation is hard to perform on vector-type computers. Here, we introduce a new method where “LISTVEC” compile-directive on SX super-computer is used. We compare it with the conventional robust method.

1. はじめに

宇宙空間プラズマを対象としたシミュレーションの 1 つとして，粒子法と呼ばれる手法がある．そこでは，プラズマ粒子の運動論効果を解明するために，プラズマ粒子 1 つ 1 つの運動と，周囲の電磁場との相互作用を self-consistent に計算する．粒子的に行うシミュレーションとしては，天文の分野で行われている多体問題のシミュレーション（物体と重力場）があるが，宇宙空間プラズマに関しては，そのような手法はあまり用いられず，代わりに PIC（Particle-In-Cell）と呼ばれる手法が使われる．つまり，場の量（電流，電磁場など）を空間格子点上でのみ定義し，プラズマ粒子を空間にランダムに配置する．粒子のモーメント値（電荷密度，電流密度）は，粒子位置に隣接する格子点にのみ配分する．ここで問題となるのが，ランダムな位置に分布する粒子データに対し，どのようにモーメント値をベクトル計算機でベクトル計算を行うかである．後述のような従来の方法で行うと，最大限のベクトル長で，99%を超えるベクトル化率で計算が可能であるが，作業用の主メモリを大量に使うため，実行するシミュレーションの規模が制限される．本論文では，新たな方法を紹介し，従来の方法と比較した結果を紹介する．実際にプラズマ粒子計算を行ったコードは，イオンのみを粒子として扱い，電子を電荷中性を満たす慣性のない流体として扱う HYBRID コードを用い，2 次元コードに対して検証した．また使用したベクトル計算機は，京都大学宇宙空電波科学研究センターで運用されている NEC 社製の SX-5 である．

† 京都大学宇宙電波科学研究センター
RASC, Kyoto University
†† 名古屋大学太陽地球環境研究所
STEL, Nagoya University
††† 愛媛大学総合情報メディアセンター
Ehime University
☆ 現在，海洋研究開発機構
Presently with JAMSTEC
☆☆ 現在，生存圏研究所
Presently with RISH, Kyoto University

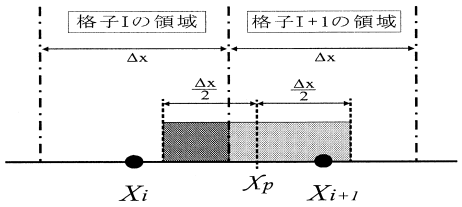

図 1 粒子の形状と密度配分の関係．粒子の密度は格子領域に占める粒子の大きさに比例して分配される（注：粒子の形状は，長方形以外にも三角形やスプライン関数型などがあり，分配される格子の数も 2 つ以上となるものもある）．

Fig. 1 Shape function of the super particle and area sharing method.

<プログラム 1>
```
! N: 粒子の背番号, Nmax: 総粒子数
! DNS: 密度
do 100 N=1, Nmax
   I   = INT( x(N) )
   S1 = Shape_Func1( x(N) )
   S2 = Shape_Func2( x(N) )
   DNS(I   ) = DNS(I   ) + S1
   DNS(I+1) = DNS(I+1) + S2
100 continue
```

2. 粒子モーメントの計算法

粒子法では，位相空間での分布関数が十分滑らかでなければ，正しい運動論を議論することができない．そのためには，格子点あたりの粒子数が数百個以上であることが望ましい[1]．よって，粒子法の計算では，格子点上の電磁場に対する計算量（流体的な計算）に対し，粒子に関する計算量が粒子数に応じて大きくなる．このことから，計算効率の向上には，粒子計算部を効率良く計算する必要がある．

粒子計算部では，超粒子として扱われる**図 1** にあるような格子間隔と同じ長さの長方形の粒子に対し，以下の 2 つの計算が行われる（簡単のため説明は 1 次元モデルで行う）．(1) 運動方程式から，粒子の速度と位置を更新する．ここでは，格子上（Xi）で定義されている電磁場の値を，粒子の位置（Xp）に内挿した値が用いられる．この計算では，特に工夫することなくベクトル化された計算が行われる．(2) 粒子の速度と位置から，電荷密度と電流密度を計算する．〈プログラム 1〉にあるようなループで密度計算を行う場合，粒子の背番号 N と粒子の位置の関係がランダムなため，N に対し格子点の位置 I の依存関係が不明となり，この DO ループはベクトル計算ができない．すなわち**図 2** にあるように，粒子背番号 2 と 3，6 と 7 が同じ格子間に存在している場合は，配列 DNS がベクトル計算できない．そこで考え出されたベクトル化の方法は，作業用配列 WRK_N(1:Gmax,1:Lv) や WRK_V(1:Gmax,1:Lv) を成分の数だけ用意し，〈プログラム 2〉にあるように，内側の 200 番 DO ループ

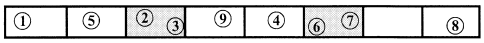

図 2 ベクトル化を阻止する粒子の分布例．配列 DNS の箱に粒子が分布している概念図

Fig. 2 Exsample for an unvectorized particle distribution.

<プログラム 2>
```
real  X(N2max,Lv), V(N2max,Lv)
real  WRK_N(Gmax,Lv), DNS(Gmax)
real  WRK_V(Gmax,Lv), FLX(Gmax)
do 100 N=1, N2max
do 200 L=1, Lv
   I   = INT( x(N,L) )
   S1 = Shape_Func1( x(N,L) )
   S2 = Shape_Func2( x(N,L) )
   WRK_N(   I,L) = WRK_N(   I,L)+S1
   WRK_N(I+1,L) = WRK_N(I+1,L)+S2
   WRK_V(   I,L) = WRK_V(   I,L)+S1*V(N,L)
   WRK_V(I+1,L) = WRK_V(I+1,L)+S2*V(N,L)
200 continue
100 continue
do 300 I=1,Gmax
   DNS(I) = SUM( WRK_N(I,1:Lv) )
   FLX(I) = SUM( WRK_V(I,1:Lv) )
300 continue
```

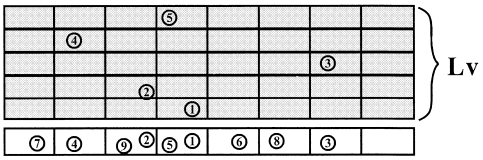

図 3 ベクトル化を可能にした密度計算方法の概念図（格子点数 × Lv）というサイズの作業用 2 次元配列用主メモリ量が必要．

Fig. 3 Example for a vectorized calculation.

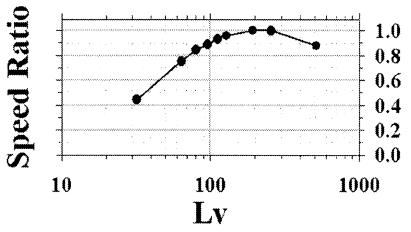

図 4 Lv に対する計算速度依存．Lv = 256 での速度で規格化してある．

Fig. 4 Dependence of the calculation speed on Lv. The speed is normalize by that of Lv = 256.

でベクトル計算を行う[2]．その概念図を**図 3** に示す．この方法では，たとえ図 3 にあるように，粒子背番号 1と5が同じ格子間に存在しても，ループ変数Lの値が異なるためベクトル計算が可能である．Lv に対する計算速度の依存性を**図 4** に示す（後述の RUN1 による測定）．計算速度を測定した 2 次元コードのパラメータは，格子数が，X と Y 方向にそれぞれ 512 であり（バッファ用 2 つ），格子点あたり，128 個の粒子を初期状態に分布させている（総粒子数 33,292,800 個）．SX 計算機では，最大ベクトルレジスタ長が 256 であり，256 要素に区切ってベクトル処理をしている

ため，Lv = 256 で最速となる．しかし，この方法では，作業用配列分の主メモリが余計に消費されるため，シミュレーションの規模に制限を与えてしまう．Lv を 256 にすることは，格子点あたり，256 個の粒子を分布させることと同じである．実際の計算では，4 種類（位置と速度 3 成分）の作業用配列が必要となり，それだけで，格子点あたり，粒子 $256 \times 4 = 1024$ 個分の主メモリを消費することとなる．

3. 主メモリを節約したモーメント計算法

3.1 作業用配列の再利用

作業用配列を 1 つだけ用意し，〈プログラム 2〉の 100 番と 200 番のループを，位置と速度 3 成分についてそれぞれ行い，密度を求めれば主メモリを節約することができる．**表 1** に，実行速度がどの程度変わるかを測定した計算結果を示す．100 ステップと 2000 ステップの計算に要した時間を示す．従来の方法で行った計算を RUN1，再利用し 4 度ループをまわした計算を RUN2 とする．RUN2 では，4 倍多くループをまわしているが，約 1.2 倍の時間増で終わるため，主メモリ量と速度の比を考えると，有用な方法の 1 つである．

3.2 コンパイル指示行 LISTVEC の利用

次に，本論文の主となるコンパイル指示行によるベクトル化の結果を示す．図 2 にあるように，同じ配列に背番号の近い粒子が入った場合，ベクトル計算ができないが，入らない場合はベクトル計算が可能である．よって，密度計算のループを粒子の背番号でまわしているときに，ベクトル計算が可能な間はベクトル計算をし，不可能なときは行わないということができれば，作業配列を用意しなくても高速計算が可能となる．そのような計算を実行させるコンパイル指示行 LISTVEC が，SX のコンパイラに用意されている．〈プログラム 1〉において，異なる N について，同じ値を持つ I（= INT(x(N))）がある場合，この DO ループでは，配列 DNS の 1 個の要素が複数回参照と定義が行われる．これを DNS の要素の衝突が発生すると呼ぶ．このようにループの異なる繰返しで定義と参照が行われる場合，そのままベクトル化すると参照と定義の順番が保存されないため，ベクトル化不可の依存関係がある状態になる．LISTVEC 指示行でベクトル化する場合は，衝突が発生しない DNS の要素についてはベクトル命令を用いて演算した結果を DNS に格納するが，衝突が発生した DNS の要素については，スカラ命令を用いて計算し直す．これを補正と呼ぶ．補正のための再計算にかかる時間が，衝突した場合のオーバヘッドとなる．この方法の良いところは，各要素が参照されたかどうかのマスクをレジスタで保持して，衝突があるかどうかを判定しているため，余分に主メモリを消費することがないという点である．SX 計算機では，ループを最大ベクトルレジスタ長である 256 要素に区切って処理しているため，その区切られた 256 要素内に同一の値 I を持つ x(N) が現れたときだけ補正を行う．たとえば，ループの繰返し数が 512 で，IX(1) と IX(300) が同じ値であった場合，IX(1:256) と IX(257:512) に分割した区間内には同じ値の要素が存在しないため，補正作業は必要ない．使用例を〈プログラム 3〉に示し，表 1 に，実行速度がどの程度変わるかを測定した計算結果を示す．本計算を RUN3 とする．また，比較のため〈プログラム 1〉でスカラ実行させた計算 RUN4 の結果も示す．RUN3 は，RUN1 に対して，約 1.57 倍，RUN2 に対しては，約 1.29 倍の時間を要したが，使用する主メモリ量が，それぞれ 2.4 倍，1.33 倍小さいことを考えれば，実用可能な方法と考えられる．

3.3 コンパイル指示行 LISTVEC の使用制限

LISTVEC 指示行を使用したベクトル化では，現在までに，我々の使用法の下で，以下の 3 つの使用制限があることが分かった．

(1) 補正作業が必要となるため，1 つのループの中で同じ名前の配列を複数回使用することはできない．

(2) 補正作業を必要とする依存関係の発生を確率的に下げるため，格子点の数は，最大ベクトル長の 256 より十分大きくなければならない．

表 1 計算に要した時間（秒）とベクトル化率（%）
Table 1 Calculation time (sec.) and vector ratio (%).

	RUN1	RUN2	RUN3	RUN4
Size	3.65 GB	2.04 GB	1.50 GB	1.50 GB
100 step	475.7	586.3	753.7	14510.9
V. Ratio	99.7	99.5	99.7	0.0
2000 step	9473.6	11541.5	14888.5	291891.9
V. Ratio	99.7	99.5	99.7	0.0

<プログラム 3>
```
!CDIR LISTVEC
do 100 N=1, Nmax
   I  = INT( x(N) )
   S1 = Shape_Func1( x(N) )
   DNS(I   ) = DNS(I   ) + S1
   FLX(I   ) = FLX(I   ) + S1 * V(N)
100 continue
!CDIR LISTVEC
do 110 N=1, Nmax
   I  = INT( x(N) )
   S2 = Shape_Func2( x(N) )
   DNS(I+1) = DNS(I+1) + S2
   FLX(I+1) = FLX(I+1) + S2 * V(N)
110 continue
```

<プログラム 4>
```
do 100 N=1, Nmax
   I   = INT( x(N) )
   S1 = Shape_Func1( x(N) )
   S2 = Shape_Func2( x(N) )
   DNS (I    ) = DNS (I    ) + S1
   DNS2(I+1) = DNS2(I+1) + S2
   FLX (I    ) = FLX (I    ) + S1 * V(N)
   FLX2(I+1) = FLX2(I+1) + S2 * V(N)
100 continue
do 300 I = 1, Gmax
   DNS(I) = DNS(I) + DNS2(I)
   FLX(I) = FLX(I) + FLX2(I)
300 continue
```

表 2 計算に要した時間（秒）とベクトル化率（%）
Table 2 Calculation time (sec.) and vector ratio (%).

	RUN3	RUN5	RUN6
Size	1.50 GB	1.53 GB	1.53 GB
100 step	753.7	547.1	17039.8
V. Ratio	99.7	99.8	96.3
2000 step	14888.5	10907.8	201229.05
V. Ratio	99.7	99.8	95.8

(3) 補正作業の発生確率を下げるために，粒子を空間にランダムに分布させなければならない．

〈プログラム 3〉において，ループを 2 つに分けている理由は，上記制限 (1) のためである．また，配列名が異なるため，電荷密度と電流密度を同じループで計算することが可能である．

一方，配列名が異なればよいということから，110 番ループで使われる配列の名前を変え，100 番ループの中で計算することも可能である（RUN5）．その例を〈プログラム 4〉に示す．この方法では，1 つの粒子のモーメント値を分配する格子数分の主メモリが消費されるが，配分される格子点数は，1 次元計算で格子点番号 I と $I+1$ の 2 つ，2 次元計算で格子点番号 (I, J)，$(I+1, J)$，$(I, J+1)$，$(I+1, J+1)$ の 4 つ，3 次元計算では同様に 8 つであるため，従来の方法に比べ消費される主メモリ量は少ない．**表 2** に，実行速度がどの程度変わるかを測定した計算結果を示す．特記すべきことは，本方法で行うと，RUN2 よりも高速に計算できることである．また，RUN1 に対しても約 1.15 倍の時間で実行できる．

次に，制限 (2) に関して，格子点数に対する計算速度を測定した結果を**表 3** に示す．システムサイズに 16 倍の差があるにもかかわらず，計算時間が 9.8 倍であることから，補正する確率が下がった効果が現れている．初期状態として，粒子をシミュレーション空間に分布させるときに関する注意点として制限 (3) があげられる．プログラミングを簡単にするためや，乱数発生を極力抑えたい場合，隣り合う背番号の粒子を順番に並べ，近接した初期位置を持たせることがある．しかし，この方法では，補正作業を必要とする衝突が多発する．表 2 の RUN6 として，順番に粒子を配置した場合の計算に要した時間を示す．実行速度がスカラ計算速度以下になり，補正作業が多発していることが分かる．また，100 ステップと 2000 ステップの計算時間を比べて，計算に要する時間がタイムステップ数に単純に比例せず，大きいステップ数の方が，計算速度が速くなっている．これは，粒子位置の coherency が時間の経過につれてなくなっていき，補正作業量が少なくなっていくためである．一方，RUN1〜RUN5 の速度測定では，粒子の初期位置は乱数を発生させ空間にランダムに分布させている．その効果は，計算に要する時間がタイムステップ数に単純に比例することに見られる（表 2）．

表 3 2000 ステップの計算に要した時間（秒）
Table 3 Calculation time (sec.) for 2000 step.

格子点数	Size	RUN3
128×128	189 MB	1514.9
512×512	1.50 GB	14888.5

4. 考　　察

従来のベクトル化のための手法に比べ，格段に主メモリ節約し，指示行を利用したベクトル化手法を紹介した．本計算コードで粒子データに必要とされる主メモリ量は，約 1.29 GB（〜8 (byte) × 512 (grid) × 512 (grid) × 128 (particle/cell) × 5 (成分)）であるため，表 1 の RUN1 で示されている従来の方法において作業用配列のサイズがいかに大きいかが分かる．1 つのノード中の主メモリサイズを大きくせずノード数を多くして大規模なシミュレーションを行うシステムを構築し，従来の方法でベクトル化を行うと，作業配列に主メモリを消費されてしまい，肝心の粒子の数が十分大きくとれない問題が発生する．しかし，本論文で紹介した方法を用いれば，十分な量の粒子を入れたシミュレーションが実行可能となり，また，計算に要する時間の伸びが約 1.15 倍程度に抑えられることが判明した．この速度は，RUN1 による計算で，Lv = 96 にした値と同程度であるため（図 4），格子点あたり約 384 (= 96×4) 個分の粒子に相当する主メモリを節約できたことを意味する．一方，作業用配列を使う場合には，すべての密度成分に対して主メモリを用意しなければ，LISTVEC 指示行を用いた方法より高速に実行できない．このため，十分大きな共有メモリをノード内に持たないベクトル計算機では，LISTVEC 指示行を使う方が有効である．さらに，計算ノード数を増やすことにより，必要な主メモリを確保することも可

能であるが，並列化にともなう計算速度の低下を起こし，LISTVEC 指示行を用いた方が計算速度が上がる可能性がある．この点に関しては，地球シミュレータの MPI の特性を考慮し[3] 実際に計算速度を測定する予定である．

5. おわりに

LISTVEC 指示行を用いたベクトル化手法は，十分に主メモリを節約できるだけでなく，ベクトル化率も，従来の方法と同じく 99.7〜99.8%に達成している．このため，SX 計算機を利用するときには，LISTVEC 指示行を使用するコードが有用であることが分かった．

参考文献

1) 村田健史，上岡功治，高橋誠治，岡田雅樹，上田裕子，大村善治，松本　紘：プラズマ電磁粒子コードの並列化手法と速度向上率の評価，情報処理学会論文誌：数理モデル化と応用，Vol.43 No.SIG7 (TOM6), pp.118–131 (2002).
2) Computer Space Plasma Physics: *Simulation Techniques and Software*, Matsumoto, H. and Omura, Y. (Eds.), Terra Scientific Publishing Company (1993).
3) 上原　均，田村正典，板倉憲一，横川三津夫：地球シミュレータの MPI 性能評価，情報処理学会論文誌：ハイパフォーマンスコンピューティングシステム，Vol.44 No.SIG1 (HPS6), pp.24–34 (2003).

(平成 15 年 10 月 6 日受付)
(平成 16 年 1 月 15 日採録)

杉山　徹

平成 10 年東京大学大学院理学系研究科にて博士（理学）取得．平成 16 年より海洋研究開発機構研究員．主に，計算機実験，人工衛星データ解析による宇宙プラズマの研究に従事．地球電磁気・地球惑星圏学会，アメリカ地球物理学会（AGU）各会員．

寺田　直樹

平成 14 年京都大学大学院理学研究科にて博士（理学）取得．平成 15 年より名古屋大学太陽地球環境研究所日本学術振興会特別研究員（PD）．主に，計算機実験による宇宙プラズマ，惑星超高層大気の研究に従事．地球電磁気・地球惑星圏学会，アメリカ地球物理学会（AGU）各会員．

村田　健史（正会員）

平成 7 年京都大学大学院工学研究科研究指導認定退学．博士（工学）．平成 15 年より愛媛大学工学部助教授，愛媛大学総合情報メディアセンター助教授．宇宙情報工学，宇宙電波科学，福祉情報工学の研究等に従事．電子情報通信学会，応用数理学会，地球電磁気・地球惑星圏学会，アメリカ地球物理学会（AGU）各会員．

大村　善治

昭和 60 年京都大学大学院工学研究科研究指導認定退学．工学博士．平成 12 年より京都大学宙空電波科学研究センター教授．専門は，宇宙プラズマ波動の計算機実験および宇宙環境シミュレータシステムの開発．地球電磁気・地球惑星圏学会，アメリカ地球物理学会（AGU）各会員．

臼井　英之

平成 4 年京都大学大学院工学研究科研究指導認定退学．工学博士．平成 10 年より京都大学宙空電波科学研究センター助教授．電磁粒子コードを用いた計算機実験による宇宙プラズマ電磁気環境の研究に従事．地球電磁気・地球惑星圏学会，アメリカ地球物理学会（AGU）各会員．

松本　紘

昭和 42 年京都大学大学院工学研究科修士課程修了．昭和 52 年京都大学超高層電波研究センター教授，センター長．平成 12 年京都大学宙空電波科学研究センター長．平成 16 年より京都大学生存圏研究所所長，教授．宇宙航空研究開発機構客員教授併任．中国武漢大客座教授．工学博士．地球電磁気・地球惑星圏学会，国際電波科学（URSI）会長歴任．アメリカ地球物理学会（AGU），IEEE フェロー．英国王立天文学協会（RAS）アソシエイト．