反復法における 4倍精度演算と倍精度演算の組み合わせ法について

A MIXED PRECISION ITERATIVE SOLVER FOR A LINEAR SYSTEM

長谷川秀彦[1], 小武守恒[2]
Hidehiko HASEGAWA and Hisashi KOTAKEMORI

[1]博士（工学）筑波大学教授　図書館情報メディア研究科（〒 305-8550 つくば市春日 1-2, hasegawa@slis.tsukuba.ac.jp)
[2]博士（理学）元（株）TCAD インターナショナル（kota@is.s.u-tokyo.ac.jp）

The convergence of Krylov subspace methods are much influenced by the rounding errors. The high precision operation is effective to improve convergence, however the arithmetic operation cost is high. We propose a DQ-SWITCH algorithm which is a restart algorithm with different precision arithmetic. This method effectively uses both fast double-double and double precision operations in order to reduce the computation time with keeping robustness.
Key Words : *Krylov subspace method, double-double precistion, iterative refinement, restart*

1. はじめに

CG, BiCG 法などのクリロフ部分空間法は，理論的には高々n 回（n は係数行列の次元数）の反復で収束する．しかしながら，浮動小数点数を用いた近似計算では丸め誤差の影響のため収束までに多くの反復が必要となったり，停滞したりする．収束性を改善するには高精度演算が有効である．

IEEE754 で規定されている倍精度浮動小数点数の演算を標準としたとき，高精度演算としては

- 計算機代数（有理数演算）
- 任意多倍長演算
- ４倍精度演算 [1]
- 区間演算
- 精度保証アルゴリズム [2]

などが考えられる．ハードウェアによって高速に実行される倍精度演算と比較すると，ハードウェアサポートがないこれらの方法は非常に低速である．いっぽう，クリロフ部分空間法による連立一次方程式の解法を考えた場合，４倍精度であっても有用な場合は少なくないはずである．

反復解法ライブラリ Lis [3] では，SSE2 の SIMD 命令を用いて高速な double-double 精度演算 [4] を実装し，実行時間を倍精度演算の約 3.5 倍に収めている．Lis では，ライブラリに与えられる係数行列 A と右辺ベクトル b，初期ベクトル x_0 は倍精度であることに着目し，大量のメモリが必要な係数行列 A は倍精度変数のままで，反復法内部の処理のみを 4 倍精度化し安定かつ高速な収束をねらっている．目的は反復法の内部処理の高精度化，高速化であり，外部とのデータのやり取りには倍精度を用いているため実装法には自由度がある．

このようなアプローチによって，倍精度演算では解けなかった問題が解けるというメリットはあるものの，大多数の問題は倍精度演算でじゅうぶんなため，広く使われるにはいたっていない．

2. 反復改良法

連立一次方程式の解の精度を改善する方法として反復改良法（Iterative Refinement Method) が知られている．反復改良法は，求められた解から残差を計算し，残差を右辺とする方程式を解いて解を補正する方法である．

Factorize $A = LU$ (A)
Solve $(LU)x_0 = b$ (B)
for(k=1;k< 最大反復回数;k++)
　$b_k = b - Ax_{k-1}$
　Solve $(LU)y_k = b_k$ (C)
　$x_k = x_{k-1} + y_k$
　x_k の収束判定

図–1 反復改良法 Iterative Refinement Method

反復改良法は，直接法を前提に同じ演算精度のもとで精度改善を図るために使われてきた．最近では，演算量が $O(n^3)$ の分解ステップ (A) を高速な単精度演算で行い，反復改良法を行うことで倍精度演算と同程度の解が高速に求められるという報告がある [5]. 方程式の条件が悪くなると倍精度よりも劣るが，良条件ならば同品質の解が短時間で求められる．

図-1 のアルゴリズムで，(B) と (C) を異なる演算精度で実行してもよいし，反復解法に適用することもできる．反復法に反復改良法を適用した場合，

- リスタート
- 初期解に粗い精度の解を使う
- 前処理の演算精度を変える [6]
- 行列積の演算精度を変える [3]

などのアプローチが考えられる．そもそも，反復法そのものが反復改良となっているため，反復の途中で何かを変化させることが反復改良法の考えにつながるため，数多くの組み合わせが存在し，ストーリーはそう単純でない．また，混合精度演算の導入法として，高精度と低精度での反復を繰り返す方法 [7] も考えられる．

3. 混合精度算法 DQ-SWITCH

われわれは混合精度の反復解法として DQ-SWITCH アルゴリズムを提案している [8]．DQ-SWITCH は，倍精度演算で解いた解（あるいは途中の値）を初期値として高速な 4 倍精度演算で反復を行う手法であり，演算精度を変えたリスタート法とみなすことができる．リスタート時に渡すのは生成された解 x_k のみで，その他の中間変数は渡さない．このアルゴリズムを図-2 に示す．nrm2 は相対残差ノルム，restart_tol はリスタートの判定基準，tol は収束判定基準である．

Lis では，4 倍精度のベクトルの hi と lo が別々の配列に格納されているため，hi のみ用いれば倍精度として扱える．したがって，倍精度と 4 倍精度のベクトルを個々に用意する必要も，リスタート時に解 x_k を倍精度から 4 倍精度へコピーする必要もない．

DQ-SWITCH で問題になるのはリスタートのタイミングで，ここではrestart_tol の決め方である．GCR 法などでは，残差最小性を利用することでより合理的な判定が可能になるが，ここではそのような条件を用いずにどれだけのことができるかに興味がある．

```
for(k=0;k<最大反復回数;k++) {
  倍精度演算の反復解法
  if( nrm2<restart_tol ) break;
}
x 以外の作業変数をゼロクリア
for(k=k+1;k<最大反復回数;k++) {
  4 倍精度演算の反復解法
  if( nrm2<tol ) break;
}
```

図–2 DQ-SWITCH のアルゴリズム

4. まとめ

われわれが反復解法ライブラリ Lis に実装した高速な double-double 精度演算によって，4 倍精度演算相当の処理が倍精度演算の約 3.5 倍で実行できるようになった．演算精度の向上によって今まで解けなかった問題が解けるようになるものの，倍精度でも解ける多くの問題に対しては過剰スペックであり，なんとかして計算時間の短縮と丸め誤差の補償の両立を図りたい．

そこで，異なる演算精度を用いた反復改良法，あるいは演算精度の異なるリスタート法として，われわれは DQ-SWITCH を提案している [8]．DQ-SWITCH は前半の反復を倍精度演算で行い，倍精度演算の限界となる点を検出し，後半の反復を 4 倍精度で行う．高速化には，できるだけ多くの反復を倍精度で行うことが重要だが，限界を越えてしまうと 4 倍精度演算に切り替えても収束しないという問題点がある．

いくつかの数値実験で，DQ-SWITCH は適切なリスタート基準を決定できれば計算時間を短縮できることが示されている．また，Lis における doble-double の実装では，余計なオーバーヘッドなしに DQ-SWITCH アルゴリズムが利用できる．

演算精度を上げることで倍精度のときよりも反復回数が少なくなるのであれば，ILU 前処理のような並列化困難な重い前処理を利用しなくとも負荷の軽い簡便な前処理でもよくなる可能性がある．並列環境では，データ量（メモリ容量と通信量）の増加は大問題だが，PE 内での演算量の増加は問題になりにくい．そのため，高精度化による反復回数の減少は，(1) 通信量の減少，(2) 通信量あたりの演算数の増加という二つの面で並列処理にとって効果的に働くはずである．しかし，解けなかった問題がとけるようになることはあっても，時間差が逆転するような問題は多くないようである．

今後の課題として，より汎用的なリスタート基準決定法の開発，分散並列環境におけるより多くの大規模テスト行列に対する検証などがあげられる．

参考文献

1) Y. Hida, X. S. Li and D. H. Bailey. Algorithms for quad-double precision floating point arithmetic. Proceedings of the 15th Symposium on Computer Arithmetic, pp.155–162, 2001.
2) 荻田 武史, 大石 進一. 大規模連立一次方程式のための高速精度保証法. 情報処理学会論文誌: 数理モデル化と応用, 46:SIG10 (TOM12), pp. 10–18, 2005.
3) http://www.ssisc.org/lis.
4) D. H. Bailey. A fortran-90 double-double library. http://www.nersc.gov/~dhbailey/mpdist/mpdist.html.
5) J. Langou, et al.. Exploiting the Performance of 32 bit Floating Point Arithmetic in Obtaining 64 bit Accuracy (Revisiting Iterative Refinement for Linear Systems). SC06 Technical Paper, 2006. http://sc06.supercomputing.org/ schedule/pdf/pap254.pdf
6) H. Tadano and T. Sakurai. On Single Precision Preconditioners for Krylov Subspace Iterative Methods. Proc. 6th International Conference on Large-Scale Scientific Computations (LSSC'07), LNCS, Vol. 4818, pp. 721–728, 2008.
7) 小武守恒，藤井昭宏，長谷川秀彦，西田晃. 倍精度と 4 倍精度の混合型反復法の提案. HPCS2007, pp. 9–16, 2007.
8) 小武守恒，藤井昭宏，長谷川秀彦，西田晃. 反復法ライブラリ向け 4 倍精度演算の実装と SSE2 を用いた高速化. 情報処理学会論文誌:コンピューティングシステム, Vol.1, No.1, pp.73–84, 2008.