

## CG法の逆べき乗法への応用

7L-2

斎藤 知哉, 亘 紀子, 坪田 美佐, 熊本 武夫  
日本電気技術情報システム開発(株)

## 0. 序

2, 3次元の有界領域上での汎関数の最小値問題から得られる線形作用素の固有値問題を差分法・有限要素法で離散化すると、行列の一般化固有値問題

$$A\phi = \lambda B\phi \quad (1)$$

が得られる。行列A, Bは正定値対称で、特に差分法の場合Bは対角になる。近似固有値が求められている、或は固有値を低いほうから幾つか求める、といった場合にしばしば逆べき乗法が用いられる。このときの $A^{-1}$ を作用させるところにCG法を用いた高速化が本論の主題である。以下で、逆べき乗法と境界値問題の場合とでは有効なプリコンディショニングが異なることを論じ、また逆べき乗法用のプリコンディショニングを提案する。

## 1. CG法に関しての考察

AをN次元の正定値対称行列とする。連立方程式  $Ax = b$  をCGで解くときの残差ベクトル $r_k$ はAの多項式をbに乗じたものである。故に、bが異なる固有値を持つn個の固有ベクトルのみを成分に持てば、 $r_k$ はこれらの固有ベクトルで張られる部分空間内を動くから、 $r_k$ の線形独立性によりn回の反復で求められることがわかる。したがってbが、主成分のn個の固有ベクトルに比べて十分小さい成分を他の固有ベクトルに持つ場合も、ある誤差内ではnに近い反復回数で求められることが期待できる。したがって  $n \ll N$  の場合はCG法が有効な線形方程式解法と言える。そこで逆べき乗法用のプリコンディショニングとして以下のものを提案する。

## 2. 逆べき乗法用CG法のプリコンディショニング

逆べき乗法で $\phi_{k+1}$ について 繰り返し  $A\phi_{k+1} = B\phi_k$  を解いてゆくと $\phi_k$ は固有値問題(1)の小さい固有値に対する固有ベクトルだけに成分を持つようになる。そこで、差分で離散化した場合は

$$\sqrt{B}A\sqrt{B}^{-1}\sqrt{B}x = \sqrt{B}b$$

とプレコンディショニングを施す。

有限要素法の場合領域が十分細分されていれば実効上Bは $D_i = \sum B_{ij}$  で定義される対角行列とほぼ等しいのでこれを用いて

$$\sqrt{D}A\sqrt{D}^{-1}\sqrt{D}x = \sqrt{D}^{-1}Bb$$

とプレコンディショニングを施す。bが固有ベクトルに

かなり近い場合は  $D_i = \sum B_{ij} * b_j / b_i$  で定義される対角行列でプレコンディショニングを施すことも有効と思われる。(図3参照)

このプレコンディショニングにより行列Aの条件数は一般に悪化するが左辺のベクトルを行列の固有ベクトルに近づける効果を持つ。この点でこの方法は他のプレコンディショニングと異なる。

## 3. 原点移動による加速

逆べき乗法では固有値の原点を、求めたい固有値の近くにとると収束が速くなる。CG法を用いる場合、固有値の原点を求める固有値よりやや小さ目にとることによって行列の正定値性を保つ必要がある。(2番目以降の固有値を求めるときはより小さな固有値に対応する固有ベクトルの成分を予め抜いておくとする。)

求めたい固有値の次の固有値のおおよその値がわかっている場合は、上で述べた条件を満たす固有値の原点を以下のように見積ることができる。 $\phi$ を近似固有ベクトルとし、 $\phi = \eta + \sum \eta_i$ と固有ベクトルで展開されるとする。ここで $\varepsilon_i^2 = (\eta_i, B\eta_i) / (\eta, B\eta)$ は微小な数である。また $\eta$ に対応する固有値を $\lambda$ とする。このとき $\lambda$ は $\varepsilon_i$ の2次のorderで  $\lambda_{\text{cal}} = (\phi, A\phi) / (\phi, B\phi)$  で求められ、その誤差は  $\text{err} = \|B^{-1}A\phi - \lambda_{\text{cal}}\phi\|_B^2 / \|\phi\|_B^2$  を用いて $\varepsilon_i$ の2次のorderでまで、 $\lambda_{\text{err}} = \text{err} / |\lambda_{\text{cal}} - \lambda^*|$  で抑えられる。ここで $\lambda^*$ は $\lambda$ の次の固有値の近似値である。(  $B^{-1}A\phi$  として反復の一つ前の近似ベクトルが利用できる。)そこで固有値の原点として $\lambda_{\text{cal}} - \lambda_{\text{err}}$ を用いることができる。

## 4. CGの打ち切り誤差

逆べき乗法に於ていて、 $A^{-1}B$ を乗ずるのは高次モードの成分を落とす為であるから、必ずしもcg法で正確に連立方程式  $A\phi_{k+1} = B\phi_k$  解く必要はない。この事に着目し、mをCG反復の添え字として  $(B\phi_k, \phi_k) / (B\phi_k, \phi_{k+1}^{(m)})$  の変動が $\lambda_{\text{err}} * (\lambda_{\text{cal}} / \lambda^*)^2$  に比べかなり小さくなったら反復を打ち切ることが考えられる。ここで $\lambda_{\text{err}}$ は前回計算した固有値の誤差で  $\lambda_{\text{err}} * (\lambda_{\text{cal}} / \lambda^*)^2$  は現在のCG反復が終わった時点で予想される誤差である。但し原点移動法を用いるときは $\lambda_{\text{cal}}, \lambda^*$ とも実際の値から原点の値を引いておかないはならない。

5. 数値実験例

差分法では、単位正方形(立方体)での井戸型ポテンシャルの最小固有値問題を例に取った。固有値方程式は  $-\Delta u + (p(x)+p(y)+p(z))u = \lambda u$  (或は  $-\Delta u + (p(x)+p(y)+p(z))u = \lambda u$ )。ここで  $p(x) = 320*(\theta(0.475-x)+\theta(x-0.625))$  である。また境界条件は  $u = 0$ 。領域の分割は  $|\Delta u|$  に比例して細かくするようにした。分割は  $64*64$  で、初期ベクトルとして分割  $32*32$  の結果を用いた。

有限要素法では単位円の井戸型ポテンシャルの最小固有値問題で方程式は同じ  $p$  を用いて  $-\Delta u + (p(r))u = \lambda u$ 。境界条件は  $u=0$ 。実際の数値実験では対称性を考慮して、半円領域で行なった。領域を112の三角形に分割して3次の基底関数を用いたものが図3、7532の三角形に分割して1次の基底関数を用いたものが図4である。三角形密度は井戸付近で大きくなるように取った。初期ベクトルは何れも四分の一の三角数で分割したときの結果を用いた。

比較の対象として、3.で提案したプレコンディショニングによるもの(仮にこれをB-SCG、特に  $D_i = \sum B_{ij} * b_j / b_i$  によるものをB-scg2と呼ぶ)、この他に通常のScaled CG法(A-scg)、通常のCG法(CG)、不完全コレスキー分解でプレコンディショニングした(fill-inは考

慮しない)CG法(iccg)、Rayleigh商の最小化の方法、但しBradburyの直行パラメータによるもの(MIN R.Q.)、完全コレスキー分解(LL<sup>t</sup>)を選んだ。

各図は各方法のCPU timeと固有値の誤差の関係で、CPU timeには前処理の時間も含めてある。図1、2の使用計算機はNECのスーパーコンピュータSX-2である。図1ではLL<sup>t</sup>は初期の分解だけで200msecを越えた。また図2の三次元ではLL<sup>t</sup>は行っていない。

有限要素法は汎用機のACOS 1000でしか実験していない。

(図3、4)これらはすべての最深DOループがベクトル化可能であるから、vector計算機でも同様な傾向を持つと思われる。なお、図3、4ではiccg及びLL<sup>t</sup>は行っていない。また図3のcg、図4のB-scg2は収束が非常に悪かったので図に載せなかった。

6. 今後の課題

4. で  $(B \phi_k, \phi_k) / (B \phi_k, \phi_{k+1}^{(m)})$  の変動と  $\lambda_{err} * (\lambda_{cal} / \lambda^*)^2$  を比較してCG法を打ち切ることを提案したが、現在は前者を適当な小さな数を後者に掛けたものと比較している。5.で述べたcg、B-scg2法の不安定さは主にCGの打ち切りが早過ぎるのが原因で、また無駄にcg反復を繰り返している場合もあると考えられる。それ故、この点は理論的に詰める必要がある。

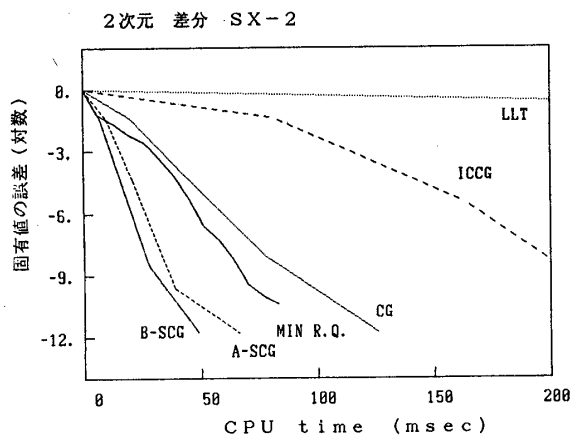


図1

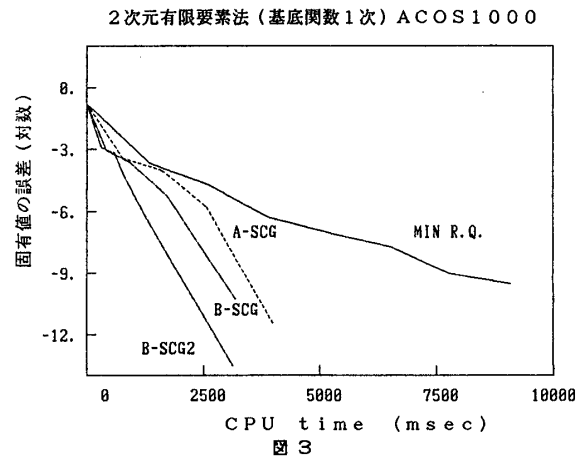


図3

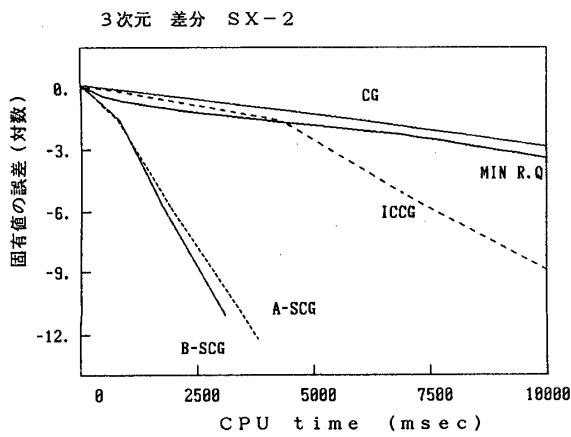


図2

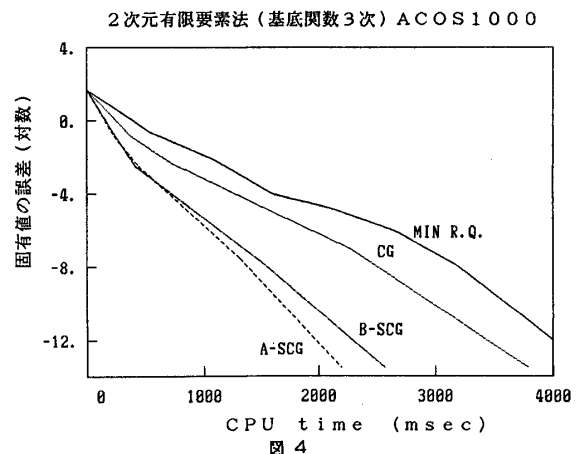


図4