数理解析研究所講究録
第613巻 1987年 135-143

BCG法とCGS法

筑波大 電情 名取 亮 (Makoto Natori)

1. BCG法

非対称 $n \times n$ 行列 A を係数とする連立一次方程式

$$Ax = b \qquad (1)$$

を解くためのBCG法 (Biconjugate Gradient Method, 双対共役勾配法) は, (1) と双対な方程式

$$A^T x^* = b^* \qquad (2)$$

を組合せて計算する方法である. 普通は $b^* = b$ とするが, ここでは一般に b^* としておく. アルゴリズムは次の通りである [1].

初期値 x_0 と x_0^* を用意
$r_0 = b - Ax_0$; $r_0^* = b^* - A^T x_0^*$
$p_0 = r_0$; $p_0^* = r_0^*$; $k = 0$
while $\|r_k\| > eps * \|b\|$ do

$$\alpha_k = (r_k^*, r_k) / (p_k^*, Ap_k) \qquad (3a)$$

$$x_{k+1} = x_k + \alpha_k p_k \qquad (3b)$$

$$r_{k+1} = r_k - \alpha_k A p_k \; ; \quad r^*_{k+1} = r^*_k - \alpha_k A^T p^*_k \qquad (3c)$$

$$\beta_k = (r^*_{k+1}, r_{k+1}) / (r^*_k, r_k) \qquad (3d)$$

$$p_{k+1} = r_{k+1} + \beta_k p_k \; ; \quad p^*_{k+1} = r^*_{k+1} + \beta_k p^*_k \qquad (3e)$$

$$k = k+1$$

このアルゴリズムは次のようにして導出できる．$2n \times 2n$ の行列とベクトルを

$$\tilde{A} = \begin{bmatrix} A & O \\ O & A^T \end{bmatrix}, \quad \tilde{x} = \begin{bmatrix} x \\ x^* \end{bmatrix}, \quad \tilde{b} = \begin{bmatrix} b \\ b^* \end{bmatrix}$$

として，方程式

$$\tilde{A}\tilde{x} = \tilde{b} \qquad (4)$$

を考える．これは，(1) と (2) をまとめたものである．

ここで，

$$\tilde{H} = \begin{bmatrix} O & I \\ I & O \end{bmatrix}$$

を用いて，内積

$$\langle \tilde{x}, \tilde{y} \rangle_{\tilde{H}} \equiv (\tilde{x}, \tilde{H}\tilde{y}) = (\tilde{H}\tilde{x}, \tilde{y})$$

を定義する．$\tilde{H}\tilde{A}$ が対称であることから，$\tilde{A}$ がこの内積に関して自己随伴：

$$\langle \tilde{x}, \tilde{A}\tilde{y} \rangle_{\tilde{H}} = \langle \tilde{A}\tilde{x}, \tilde{y} \rangle_{\tilde{H}}$$

となることに注意すると，関数

$$f(\tilde{x}) = \frac{1}{2} \langle \tilde{x}, \tilde{A}\tilde{x} \rangle_{\tilde{H}} - \langle \tilde{x}, \tilde{b} \rangle_{\tilde{H}}$$

を停留にする $\tilde{x}$ が(4)の解となることがわかる．これに，CG法を適用すると次のアルゴリズムが得られる．

初期値 $\tilde{x}_0$ を用意
$\tilde{r}_0 = \tilde{b} - \tilde{A}\tilde{x}_0$
$\tilde{p}_0 = \tilde{r}_0$; $k = 0$
while $\| r_k \| > eps * \| b \|$ do

$$\alpha_k = \langle \tilde{r}_k, \tilde{r}_k \rangle_{\tilde{H}} / \langle \tilde{p}_k, \tilde{A}\tilde{p}_k \rangle_{\tilde{H}} \qquad (5a)$$
$$\tilde{x}_{k+1} = \tilde{x}_k + \alpha_k \tilde{p}_k \qquad (5b)$$
$$\tilde{r}_{k+1} = \tilde{r}_k - \alpha_k \tilde{A}\tilde{p}_k \qquad (5c)$$
$$\beta_k = \langle \tilde{r}_{k+1}, \tilde{r}_{k+1} \rangle_{\tilde{H}} / \langle \tilde{r}_k, \tilde{r}_k \rangle_{\tilde{H}} \qquad (5d)$$
$$\tilde{p}_{k+1} = \tilde{r}_{k+1} + \beta_k \tilde{p}_k \qquad (5e)$$

$k = k + 1$

このアルゴリズムが(3)に示したBCG法と同一であることは次のようにしてわかる．まず，

$$\langle \tilde{r}_k, \tilde{r}_k \rangle_{\tilde{H}} = (\tilde{r}_k, \tilde{H}\tilde{r}_k) = \left(\begin{bmatrix} r_k \\ r_k^* \end{bmatrix}, \begin{bmatrix} r_k^* \\ r_k \end{bmatrix} \right)$$
$$= 2(r_k^*, r_k)$$
$$\langle \tilde{p}_k, \tilde{A}\tilde{p}_k \rangle_{\tilde{H}} = (\tilde{p}_k, \tilde{H}\tilde{A}\tilde{p}_k) = \left(\begin{bmatrix} p_k \\ p_k^* \end{bmatrix}, \begin{bmatrix} A^T p_k^* \\ A p_k \end{bmatrix} \right)$$
$$= 2(p_k^*, A p_k)$$

を用いると，(5a)(5d)がそれぞれ(3a)(3d)と同一である

ことがわかる．(5b)(5c)(5e) が (3b)(3c)(3e) と同じであることは明らかであろう．

$\tilde{r}_k$ と $\tilde{p}_k$ の生成多項式をそれぞれ $R_k(\tilde{A})$, $P_k(\tilde{A})$ とすると

$$\tilde{r}_k = R_k(\tilde{A})\,\tilde{r}_0 = \begin{bmatrix} R_k(A)\,r_0 \\ R_k(A^T)\,r_0^* \end{bmatrix}$$

より

$$\left.\begin{aligned} r_k &= R_k(A)\,r_0 \\ r_k^* &= R_k(A^T)\,r_0^* \end{aligned}\right\} \quad (6)$$

となり，同様にして

$$\tilde{p}_k = P_k(\tilde{A})\,\tilde{r}_0$$

より

$$\left.\begin{aligned} p_k &= P_k(A)\,r_0 \\ p_k^* &= P_k(A^T)\,r_0^* \end{aligned}\right\} \quad (7)$$

となる．これを用いると，直交性

$$\langle \tilde{r}_i, \tilde{r}_j \rangle_{\tilde{H}} = 0 \quad (i \neq j)$$

から

$$(r_i^*, r_j) = 0 \quad (i \neq j)$$

が，また共役直交性

$$\langle \tilde{p}_i, \tilde{A}\tilde{p}_j \rangle_{\tilde{H}} = 0 \quad (i \neq j)$$

から

$$(p_i^*, Ap_j) = 0 \quad (i \neq j)$$

が導かれる.

2. CGS法

つぎに，最近デバイスシミュレーションの分野で話題になっているCGS法(CG Squared Method，自乗共役勾配法)を紹介する[2]. アルゴリズムを示すと，

初期値 $\hat{x}_0$ を用意

$\hat{r}_0 = b - A\hat{x}_0$

$\hat{p}_0 = e_0 = \hat{r}_0$; $k=0$

while $\|\hat{r}_k\| > eps * \|b\|$ do

$$\alpha_k = (\hat{r}_0, \hat{r}_k) / (\hat{r}_0, A\hat{p}_k) \tag{8a}$$

$$h_{k+1} = e_k - \alpha_k A\hat{p}_k \tag{8b}$$

$$\hat{r}_{k+1} = \hat{r}_k - \alpha_k A(e_k + h_{k+1}) \tag{8c}$$

$$\hat{x}_{k+1} = \hat{x}_k + \alpha_k (e_k + h_{k+1}) \tag{8d}$$

$$\beta_k = (\hat{r}_0, \hat{r}_{k+1}) / (\hat{r}_0, \hat{r}_k) \tag{8e}$$

$$e_{k+1} = \hat{r}_{k+1} + \beta_k h_{k+1} \tag{8f}$$

$$\hat{p}_{k+1} = e_{k+1} + \beta_k (h_{k+1} + \beta_k \hat{p}_k) \tag{8g}$$

$k = k+1$

となる．これは，BCG法における r_k, p_k の生成多項式を $R_k(A)$, $P_k(A)$ として，新しいベクトル

$$\hat{r}_k = R_k^2(A)\, r_0$$
$$\hat{p}_k = P_k^2(A)\, r_0$$
$$e_k = P_k(A) R_k(A)\, r_0$$
$$f_{k+1} = P_k(A) R_{k+1}(A)\, r_0$$

を導入し，

$$\hat{r}_k = b - A\hat{x}_k \qquad (9)$$

とすることにより得られる．ただし，$\hat{x}_0 = x_0$, $\hat{r}_0 = r_0$ さらに $r_0^* = r_0$ であるものとする．まず，(6)(7) を用いて

$$(r_k^*, r_k) = (R_k(A^T) r_0, R_k(A) r_0)$$
$$= (r_0, R_k^2(A) r_0)$$
$$= (\hat{r}_0, \hat{r}_k)$$
$$(p_k^*, Ap_k) = (P_k(A^T) r_0, A P_k(A) r_0)$$
$$= (r_0, A P_k^2(A) r_0)$$
$$= (\hat{r}_0, A\hat{p}_k)$$

となり，(3a) から (8a)，(3d) から (8e) が導かれる．(8b) は，(3c) を (6)(7) を用いて書き直し，

$$R_{k+1}(A) r_0 = R_k(A) r_0 - \alpha_k A P_k(A) r_0 \qquad (10)$$

この両辺に $P_k(A)$ を掛けることによって得られる．

(8c) は，(3c) を書き直した (10) 式 の両辺に $R_{k+1}(A)$ を

掛けて

$$R_{k+1}^2(A)\,r_0 = R_k(A)R_{k+1}(A)\,r_0 - \alpha_k A P_k(A)R_{k+1}(A)\,r_0$$

とし，右辺第1項に(10)式を適用すると得られる．

(8d)は，(8c)と(9)から明らかである．

(8f)は，(3e)を(6)(7)を用いて書き直し，

$$P_{k+1}(A)\,r_0 = R_{k+1}(A)\,r_0 + \beta_k P_k(A)\,r_0 \qquad (11)$$

この両辺に $R_{k+1}(A)$ を掛ければよい．

(8g)は，(11)の両辺に $P_{k+1}(A)$ を掛けて

$$P_{k+1}^2(A)\,r_0 = P_{k+1}(A)R_{k+1}(A)\,r_0 + \beta_k P_k(A)P_{k+1}(A)\,r_0$$

とし，右辺第2項にもう一度(11)式を適用すれば得られる．

3. BCG法とCGS法の比較

前節で述べたように，CGS法はBCG法と本質的には同じであるが，BCG法ではベクトル列 $\{r_k\}$ を計算するのに対してCGS法では $\{\hat{r}_k\}$ を計算する点が異なる．

記憶容量

計算に必要な作業ベクトルの本数は等しい．

演算量

反復1回当りの演算量も等しい．ただし，BCG法では Av と A^Tv が1回ずつであるのに対してCGS法では

$A\nu$ が2回で，A^T は不要である．

実用的には，CG法のときと同じように前処理によって収束を速める．Aの不完全LU分解を LDU とすると (1) の代りに

$$(LDU)^{-1}Ax = (LDU)^{-1}b$$

を解くことになる．したがって，アルゴリズム中のAは，$(LDU)^{-1}A$ におきかわり，

$A\nu$ は $(LDU)^{-1}A\nu$

$A^T\nu$ は $A^T(LDU)^{-T}\nu$

となって，いずれにしても前進，後退代入が必要になる．その際，CGS法では $A^T(LDU)^{-T}\nu$ の計算が不要であるためプログラムが簡単になるという利点がある．

収束性

もし，

$$\|r_k\| = \|R_k(A)r_0\| \ll \|b\|$$

となったとすると

$$\|\hat{r}_k\| = \|R_k^2(A)r_0\|$$

は、$\|r_k\|$ よりも小さくなるので，CGS法の方がBCG法より速く収束する．

実際，3次元移流拡散方程式を $40\times20\times20$ の網目で差分法により離散化し，Gustafsson流の補正をした不完全

LU分解（MILU分解）で前処理をすると，相対残差が 10^{-8} 以下になるまでの反復回数は，BCG法が約30回に対してCGS法は約20回と2/3に減少する．この比率2/3は，セルペクレ数が 2^{-5}～2^{3} の範囲ではほぼ一定である．

参考文献

[1] R. Fletcher, Conjugate gradient methods for indefinite systems, G. A. Watson, ed., Numerical Analysis (Proc. of the Dundee Conference on Numerical Analysis, 1975), Lecture Notes in Math., 506, Springer-Verlag, 1976, pp. 73－89.

[2] C. den Heijer, Preconditioned Iterative Methods for Nonsymmetric Linear Systems, Proc. of the Intern. Conference on Simulation of Semiconductor Devices and Processes, 1984, pp. 267－285.