数理解析研究所講究録
第483巻 1983年 40-62

CG法と同時逆反復法の組合せによる固有値計算

日立ソフト　後 保範（Yasunori Ushiro）
日立中研　村田健郎（Kenro Murata）

1. はじめに

対称帯固有値問題 $Ax = \lambda x$ を解くアルゴリズムとして，スツルム・同時逆反復法[1]を検討してきた。それは密集固有値をスツルムを使用して合理的に区分けして，そこに対しては同時逆反復が働くようにしたものである。即ちスツルム・逆反復法と同時逆反復法の長所を生かし，短所を表面化させないよう両者を統合したものである。その結果

(イ) 帯幅が広くかつ密集固有値があるため，スツルム・逆反復法では難儀な問題

(ロ) 所望の固有値数が多くかつ分布が精疎交々，しかも所望精度が精しいため同時逆反復法では難儀な問題

の双方に対し，CPU，USEタイム共にそれなりに耐えるようになった。

しかし，偏微分方程式の離散化により発生する対称スパー

ス行列の場合でも，スツルム・同時逆反復法は三角分解を行うためスパース性を失うという欠点がある。この場合，スツルム過程におけるＵＳＥタイムは小さくできるが，同時逆反復過程におけるページスワップの負担が大きい。このため元の行列のスパース性を保持したまま同時逆反復法を適用することを考えた。

ここでは同時逆反復過程の連立一次方程式の解の計算に対応する部分をＣＧ法及びＩＣＣＧ法（不完全コレスキー分解とＣＧ法を組合せた解法）で置き換えることを試みた。ＣＧ法及びＩＣＣＧ法は同時逆反復の初期段階で適用するもの２種類と，一定の収束の後で適用するもの１種類を検討対象とした。拡散方程式を離散化して得られるスパース行列を数値実験の対象とした。

通常の連立一次方程式の計算においては，ＩＣＣＧ法はＣＧ法に比較して収束が格段に速いことが知られている。しかし同時逆反復法と組合せて使用する場合はＣＧ法の方が収束が安定することが判明した。このため数値実験の多くはＣＧ法と同時逆反復法の組合せで行った。

2. 同時逆反復法

ここではスツルムにより固有値を合理的に区分けし，その区分けした区間に対して同時逆反復法を適用するものとする。

同時逆反復法を適用する区間ごとのシフト点αは区分けにより自動的に決定する。さらにシフト点αの値の小さい区間から先に固有ベクトルを同時逆反復法で求めるため，既に求めた固有ベクトル成分は直交化により抜き取ることができる。

今回は各区分ごとの固有値の個数をPとし，シフト点αにおける $(A-\alpha I)x=\lambda x$ に対する同時逆反復法として次のものを適用した。(a)から(e)まで収束するまで反復する。

(a) αの近傍の固有ベクトルの濃縮　$Y=Y(X_{\nu-1}, A-\alpha I)$

(b) 正規直交化(Gram-Schmit)　$Y=Q\cdot R$

(C) P次元に縮小した行列の作成　$H=Q^T(A-\alpha I)Q$

(d) P次元の固有値計算 $(i=1,2\cdots P)$　$Hv_i=\lambda_i v_i$

(e) X_νの作成 $(V=(v_1,v_2,\cdots,v_P)^T)$　$X_\nu=QV$

従来の同時逆反復法と異なるのは，項番(a)におけるシフト点αの固有ベクトルの濃縮方法である。この部分に対し，CG法及びICCG法を適用した。

3. CG法及びICCG法

シフト点αの近傍の固有ベクトルの濃縮方法として次の3種類の方法を検討した。方式1及び方式2は同時逆反復の初期段階で適用し，方式3は一定の収束の後で適用する。

各方法とも～～部分を取り除いたものがCG法で，～～部分を含むものがICCG法である。

ＩＣＣＧ法のためのスパース行列の不完全ＬＤＬ^{T}分解には種々の方法があるがここでは元の行列と同じスパース性を保持する[4]もので数値実験を行った。また対角行列Ｄの計算で，Ｄの要素の絶対値が10^{-5}以下となった場合は，その要素の値を符号$\times 10^{-5}$で置き換えた。

3.1　方式1

連立一次方程式$(A-\alpha I)y_i = x_i$の解y_iの計算をＣＧ法及びＩＣＣＧ法で行う。ここで，$i=1,2,\cdots,P$でＰは各区分ごとの固有値数である。ℓ回反復計算する場合のＩＣＣＧ法を1-ICCG(ℓ)と名付け図3.1に示す。ＣＧ法のときは1-CG(ℓ)と名付け～～部分を除いたもので示す。

$\hat{A} = A - \alpha I$ とし $\hat{A}$ を不完全ＬＤＬ^{T}分解する

$i=1$ から各区分の固有値数Ｐまで繰り返す

　$y_i = 0$, $r = x_i$, $P = [LDL^T]^{-1} \cdot r$, $\mu_1 = (r, P)$

　指定回数ℓだけ反復

　　$\theta = \mu_1 / (P, \hat{A} \cdot P)$

　　$y_i = y_i + \theta \cdot P$, $r = r - \theta \hat{A} \cdot P$

　　$\mu_2 = \mu_1$, $\mu_1 = (r, [LDL^T]^{-1} r)$

　　$P = [LDL^T]^{-1} \cdot r + \mu_1 / \mu_2 \cdot P$

図3.1　1-ICCG(ℓ)及び1-CG(ℓ)のアルゴリズム

3.2 方式2

$\hat{A} = A - \alpha I$ とシフト点 α だけシフトした行列 $\hat{A}$ を作る。関数 $f(x) = (\hat{A}x, \hat{A}x)/(x, x)$ を最小化するようなCG法及びICCG法を考える。関数 $f(x)$ は行列 $\hat{A}^T\hat{A}$ とベクトル x のレーリー商であり，その値を最小化すると行列Aの α の近傍の固有ベクトルが濃縮される。本来なら関数 $f(x)$ として $\hat{A}$ と x のレーリー商を採用したいが，それではシフト点 α の近傍の固有ベクトルの濃縮ができない。

ここで関数 $f(x)$ を最小化するアルゴリズムは図3.2に示すFletcher-Reevers[2]のものを使用した。

$r_0 = \nabla f(x^{(0)})$

$P_0 = -r_0$

$\nu = 1, 2, \cdots$ と収束するまで反復

- $f(x^{(\nu-1)} + \theta P_{\nu-1})$ を最小（停留）にする θ を求める
- $x^{(\nu)} = x^{(\nu-1)} + \theta \cdot P_{\nu-1}$
- $r_\nu = \nabla f(x^{(\nu)})$
- $\beta = (r_\nu, r_\nu)/(r_{\nu-1}, r_{\nu-1})$
- $P_\nu = -r_\nu + \beta P_{\nu-1}$

図3.2 関数 $f(x)$ の最小化アルゴリズム

ℓ回反復計算する場合のICCG法を2-ICCG(ℓ)と名付け図3.3に示す。CG法のときは2-CG(ℓ)と名付け～～部分を除いたもので示す。

$\widehat{A} = A - \alpha I$ とし $\widehat{A}$ を不完全LDL^T分解する

$i = 1$ から各区分の固有値数Pまで繰り返す

　指定回数ℓだけ反復

$$\lambda = (\widehat{A}x_i, \widehat{A}x_i)/(x_i, x_i)$$

$$r = (\lambda x_i - \widehat{A}^T A x_i)/(x_i, x_i)$$

$$\mu_1 = ([LDL^T]^{-1} r, [LDL^T]^{-1} r)$$

$$\beta = \mu_1/\mu_2 \text{（但し1回目は0）}, \quad \mu_2 = \mu_1$$

$$P = [LDL^T]^{-1}[LDL^T]^{-1} \cdot r + \beta \cdot P$$

$$a = (\widehat{A}P, \widehat{A}P)(x_i, P) - (\widehat{A}x_i, \widehat{A}P)(P, P)$$

$$b = (\widehat{A}P, \widehat{A}P)(x_i, x_i) - (\widehat{A}x_i, \widehat{A}x_i)(P, P)$$

$$c = (\widehat{A}x_i, \widehat{A}P)(x_i, x_i) - (\widehat{A}x_i, \widehat{A}x_i)(x_i, P)$$

$$\theta = (-b + \sqrt{b^2 - 4ac})/2a$$

$$x_i = x_i + \theta P$$

図3.3　2-ICCG(ℓ)及び2-CG(ℓ)のアルゴリズム

3.3　方式3

本方式は関数$f(x) = (x, Ax) - (x, \lambda x)$の停留点を求めるよう，図3.2のアルゴリズムにより定式化したものである。

ここで $\lambda = (x, Ax)/(x, x)$ なるレーリー商である。本方式はレーリー商の精度が元の近似固有ベクトルの精度より2乗のオーダーで良くなることを利用したものである。

ℓ 回反復計算する場合のICCG法を3-ICCG(ℓ, h)と名付け図3.4に示す。CG法のときは3-CG(ℓ, h)と名付け~~~を除いたもので示す。ただし h は方式1及び方式2から方式3に切り換える判定の精度を示す。

$i=1$ から各区分の固有値数Pまで繰り返す

$\lambda_i = (x_i, Ax_i)/(x_i, x_i)$

$\widetilde{A}_i = A - \lambda_i I$ とし $\widetilde{A}_i$ を LDL^T 分解する

指定回数 ℓ だけ反復

$\Delta\lambda_i = (x_i, \widetilde{A}_i x_i)/(x_i, x_i)$

$r = \Delta\lambda_i x_i - \widetilde{A}_i x_i$

$\mu_1 = (r, [LDL^T]_i^{-1} r)$

$\beta = \mu_1/\mu_2$ (但し1回目は0), $\mu_2 = \mu_1$

$P = [LDL^T]_i^{-1} \cdot r + \beta P$

$\theta = (r, P)/(P, [\widetilde{A}_i - \Delta\lambda_i I] \cdot P)$

$x_i = x_i + \theta P$

図3.4 3-ICCG(ℓ, h)及び3-CG(ℓ, h)のアルゴリズム

4. 数値実験結果

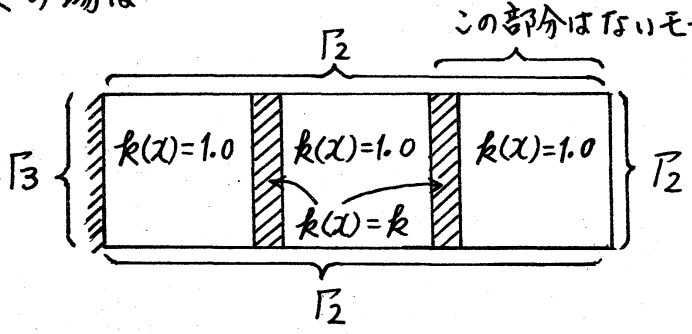
4.1 テスト問題

2次元拡散方程式を有限要素法で離散化した行列をテスト行列とした。

支配方程式は $\nabla\cdot(-k(x)\nabla u) = \lambda u$

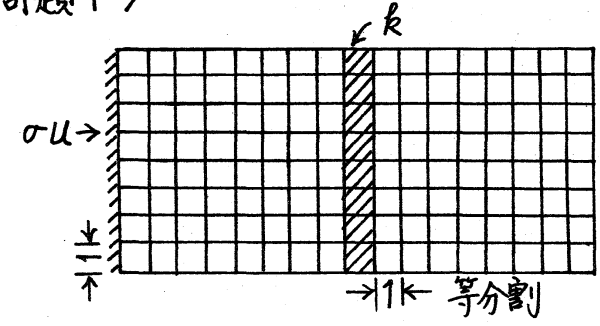
境界条件は Γ_2 で $(\nabla u)\cdot n = 0$, Γ_3 で $(k(x)\nabla u)\cdot n = \sigma u$

解析対象の場は



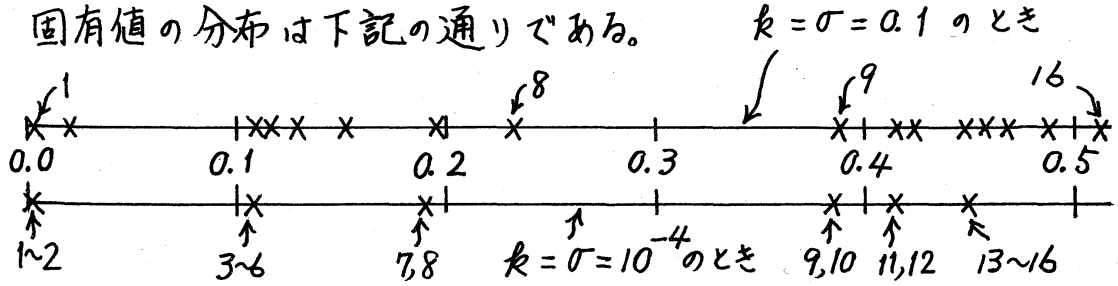
離散化して得られた固有値問題を $Au = \lambda u$ とした。

〈問題1〉

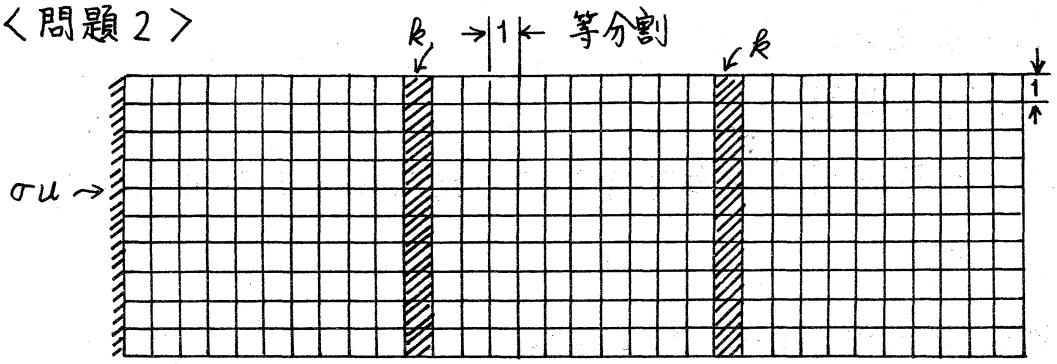


8x8 分割したものを2個連結したものであり，(8x8)x2 と表す。

固有値の分布は下記の通りである。

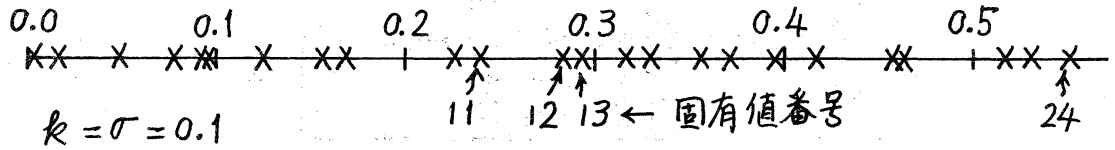


〈問題2〉

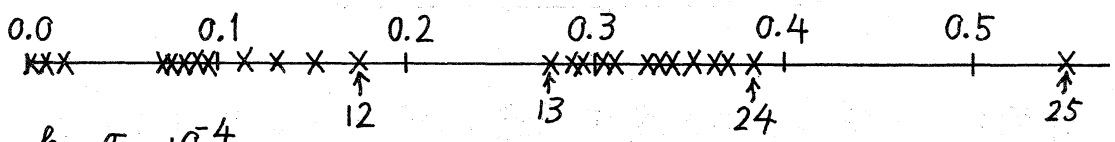


10×10分割したものを2個連結したものであり，(10×10)×3と表す。固有値の分布は下記の通りである。

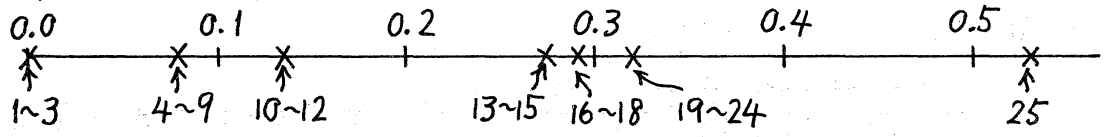
$k=\sigma=1.0$



$k=\sigma=0.1$



$k=\sigma=10^{-4}$



$k=\sigma=10^{-4}$ とすると固有値は 10^{-4} の程度で最大6個近接する。$k=\sigma=1.0$ とすると固有値は平均的に分布する。

4.2 方式1と方式2及びCG法とICCG法の比較

問題1のテスト行列を使用して，方式1と方式2の収束速度の比較及びCG法とICCG法の収束速度を比較するための数値実験を行った。

図4.1に $k=\sigma=0.1$ でシフト点を0.0とし8個同時逆反

復した場合の収束速度の比較を示す。図4.2にシフト点を0.4にし8個同時逆反復した場合の結果を示す。図4.2では前に計算した1から8番までの固有ベクトルと，同時逆反復を行うごとに直交させて計算した。　図4.3に$\varepsilon=\sigma=10^{-4}$でシフト点を0.0とし8個同時反復した場合の結果を示す。図4.1，図4.2及び図4.3では8個同時逆反復したベクトル中，目的の固有ベクトルへの収束が最も遅いものの精度を示した。図4.4は図4.1と同一の同時逆反復で，目的の固有ベクトルへの収束が最も速いものの精度を示した。

これらの図より方式2は方式1より収束が遅いことが分かる。その理由として，次の2つが考えられる。

(a) シフト点αの近傍の固有ベクトルを濃縮するのに，行列AではなくA^TAを対象に処理を行っている。このため処理対象の最小固有値と最大固有値の比が2乗で大きくなりCG法での収束が遅い。図4.4はこのことを良く示している。シフト点αに最も近い固有ベクトルでさえも2-CG(8)では収束が遅い。2-ICCG(8)ではこの問題は図4.4に示すように改善される。

(b) 方式2のICCG法はシフト点αに最も近い固有ベクトルの収束は良いが，離れると急に悪くなる。それは図3.3の方向ベクトルPを作成するときに，不完全分解した

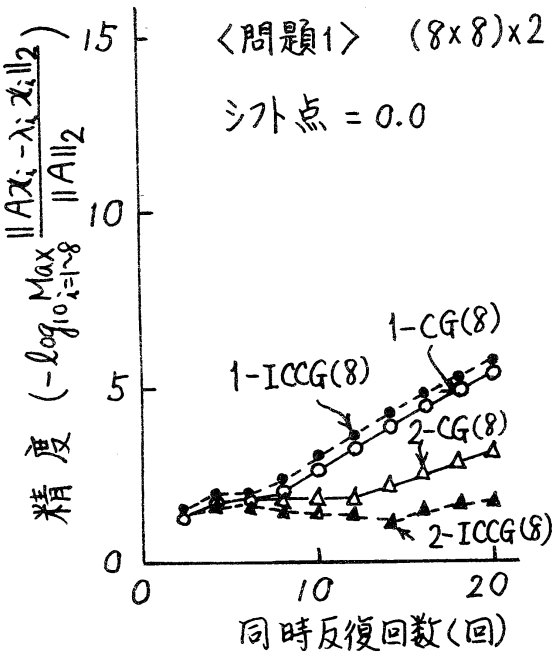


図4.1 最も収束の遅い固有ベクトルの精度
(最小より8個同時反復, $\varepsilon=\sigma=0.1$)

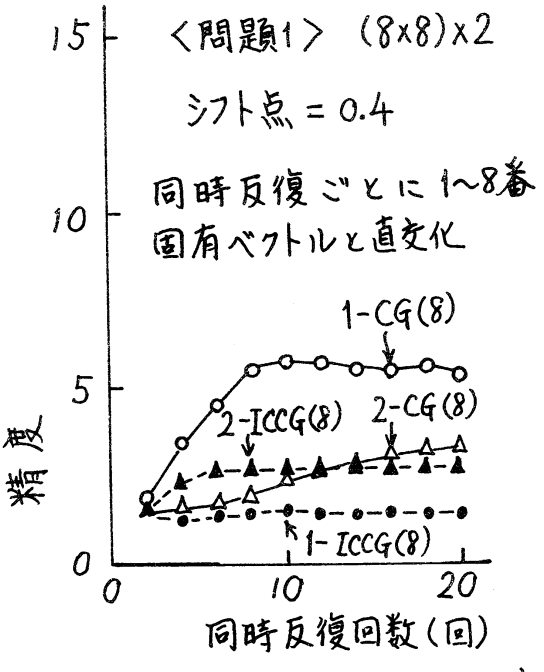


図4.2 最も収束の遅い固有ベクトルの精度
(9~16番の同時反復, $\varepsilon=\sigma=0.1$)

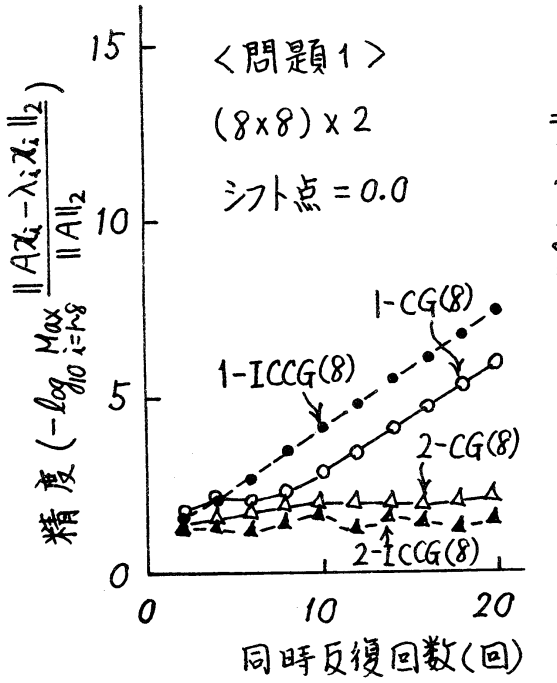


図4.3 最も収束の遅い固有ベクトルの精度
(最小より8個同時反復, $\varepsilon=\sigma=10^{-4}$)

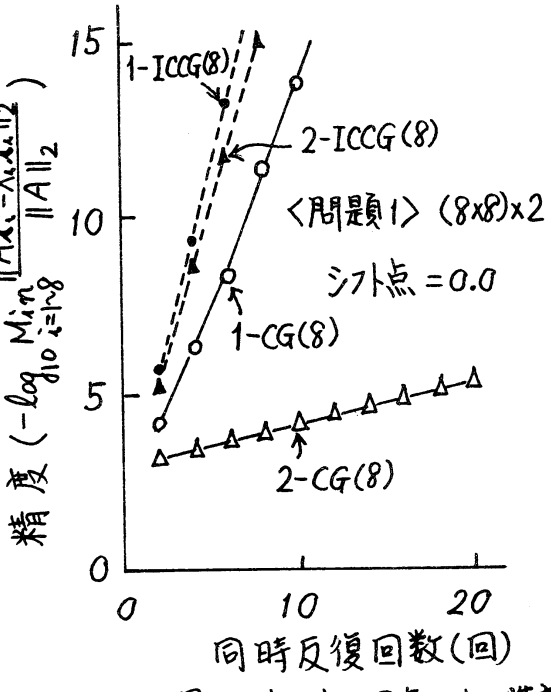


図4.4 最も収束の速い固有ベクトルの精度
(最小より8個同時反復, $\varepsilon=\sigma=0.1$)

行列 $(LDL^T)^{-1}$ を残差ベクトル r に2度作用させていることにより，一度取り除いた α に近い固有ベクトルの成分が総ての反復ベクトルに混入し，目的とする固有ベクトル成分を押し退けて成長するためと考える。

方式1のICCG法とCG法を比較すると，図4.1，図4.2及び図4.4ではICCG法の方がCG法より収束が速い。しかし図4.2ではICCG法の方が極端に収束が遅いことが分かる。図4.2でICCG法の収束が遅い原因は，一度取り除いた固有ベクトルの成分がICCG法の中の反復計算で混入し成長するためである。このことをより明確にするため，図4.2と同じ同時逆反復を行ったときの，前の区間に最も近い9番固有値の計算値のふるまいを図4.5に示す。

ICCG法では9番固有値の計算値が8番固有値と9番固有値の間をうろうろしている。これは直交化により反復ベクトルから取り除いた8番固有ベクトルの成分がICCG法の中の反復計算

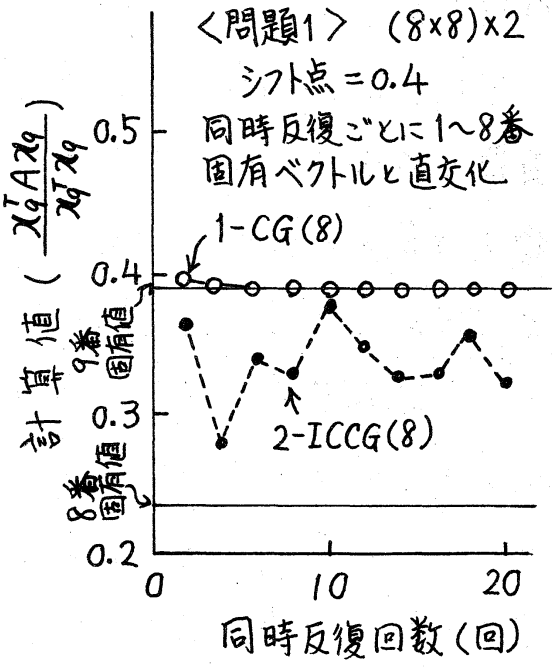


図4.5 計算した9番固有値のふるまい
(9~16番の同時反復，$k=\sigma=0.1$)

で混入したことを意味する。

以上の結果より同時逆反復の初期段階では方式1のCG法が最も安定した方法であると考え，方式3の前段階の計算として方式1のCG法を使用して数値実験を行った。また，ICCG法は直交化により取り除いた固有ベクトルの成分が原理的に混入してしまうため方式3の数値実験もCG法を採用して行った。

4.3 方式3による収束の加速

問題2のテスト行列を使用して，方式1のCG法で一定の精度まで収束させた後，方式3のCG法に引き継いだ場合の数値実験を行った。

図4.6に$k=\sigma=0.1$でシフト点を0.0とし12個同時逆反復した場合の収束速度を示す。方式1から方式3への切り換えは最も収束が遅いものの精度($-\log_{10} \max_i \|A\chi_i-\lambda_i\chi_i\|_2/\|A\|_2$)が2.5以上になったとき行った。図4.7にシフト点を0.3にし12個同時逆反復した場合の結果を示す。図4.2では，前に計算した1から12番までの固有ベクトルと同時逆反復を行うごとに直交させて計算した。図4.6と図4.7ではCG法内の反復計算回数を10回，20回及び30回と変化させたものを示す。図4.8は図4.7で$k=\sigma=10^{-4}$としたものの結果である。図4.9は図4.6で$k=\sigma=1.0$としたものである。

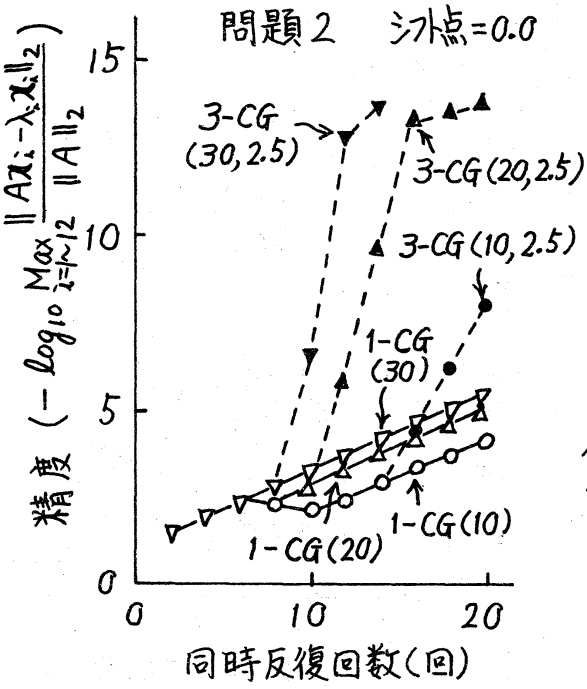


図4.6 最も収束の遅い固有ベクトルの精度
(最小より12個同時反復, $k=\sigma=0.1$)

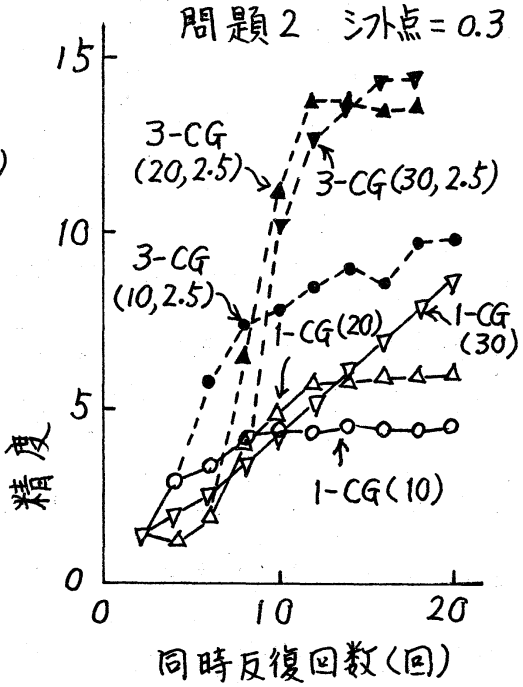


図4.7 最も収束の遅い固有ベクトルの精度
(13~24番の同時反復, $k=\sigma=0.1$)

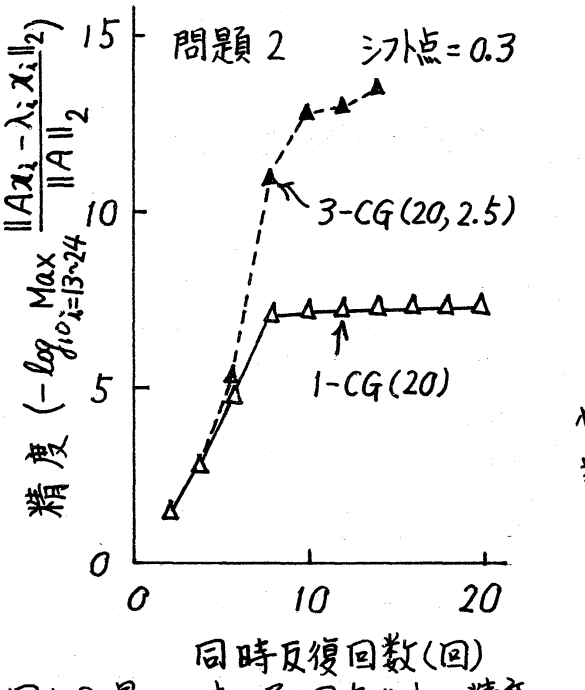


図4.8 最も収束の遅い固有ベクトルの精度
(13~24番の同時反復, $k=\sigma=10^{-4}$)

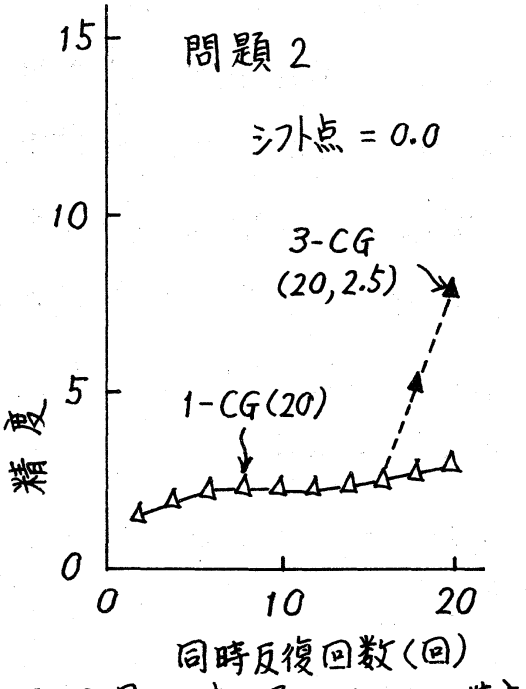


図4.9 最も収束の遅い固有ベクトルの精度
(最小より11個同時反復, $k=\sigma=1.0$)

図4.8と図4.9ではCG法内の反復回数を20回としたものだけ示した。図4.10は図4.6でCG法内の反復計算回数を20回に固定し，方式1から方式3への切り換え点を2.0，2.5，3.0及び4.0と変化させたものである。いずれの図も方式1だけで実行した結果も示した。

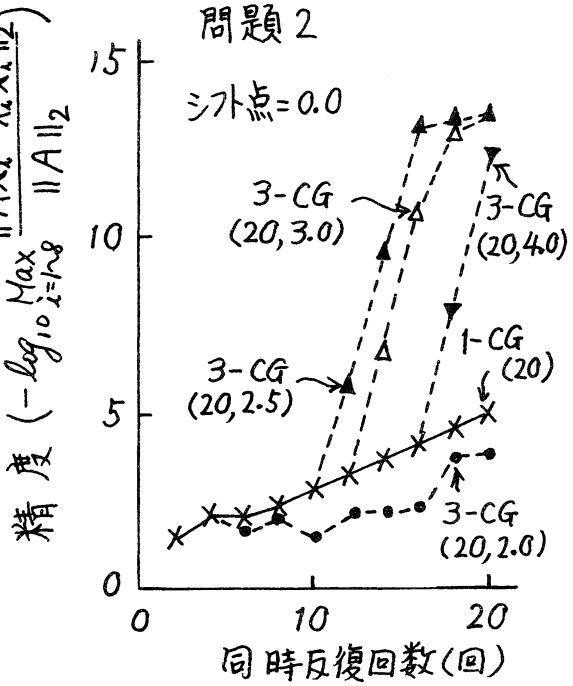


図4.10 最も収束の遅い固有ベクトルの精度
(最小より8個同時反復，$k=\sigma=0.1$)

いずれの場合もi番固有ベクトルの計算のための初期ベクトルはi/2の周期の一次元の正弦波で作成した。

図4.6から図4.10まですべて，方式3への切り換えを行った方が方式1より圧倒的に収束が速いことを示している。ただし，図4.10より方式1から方式3への切り換えは，早すぎると逆効果となることが分かる。

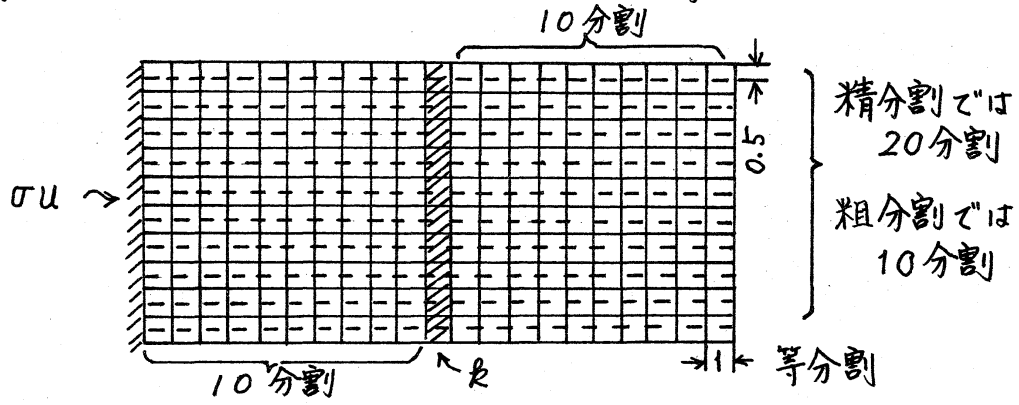
図4.6及び図4.7から方式1のCG法では，CG法内の反復回数を増加させていくと，収束速度は一定値に漸近していくことが分かる。その収束速度は同時逆反復計算中の連立一次方程式を直接解法で計算した場合と一致する。方式3の

CG法では，CG法内の反復回数を増加するとそれに比例して収束が速くなることが分かる。図4.9で示すように方式1では収束が非常に遅い場合でも，方式3に切り換えると収束は急激に速くなることが分かる。

方式1での同時逆反復計算回数を減少させるアイデアの一つとして，粗分割で作成した行列の固有ベクトルを精分割の行列の固有ベクトルの初期値とする方法[6]が考えられる。今回は方式1から方式3への切り換えの適切なタイミングを詳細に検討することより，上記の方法で方式1の反復回数の減少の可能性を検討することにした。

4.4 二段階メッシュ分割による効果

粗分割で作成した行列の固有ベクトルを，線形補間して精分割の初期ベクトルとした場合の数値実験結果を示す。
テスト問題は4.1節の拡散方程式を離散化したもので，次のように(10×20)×2に精分割した。粗分割は点線部分の分割を行なわない(10×10)×2に分割した。



粗分割と精分割による固有値の分布は次の通りである。

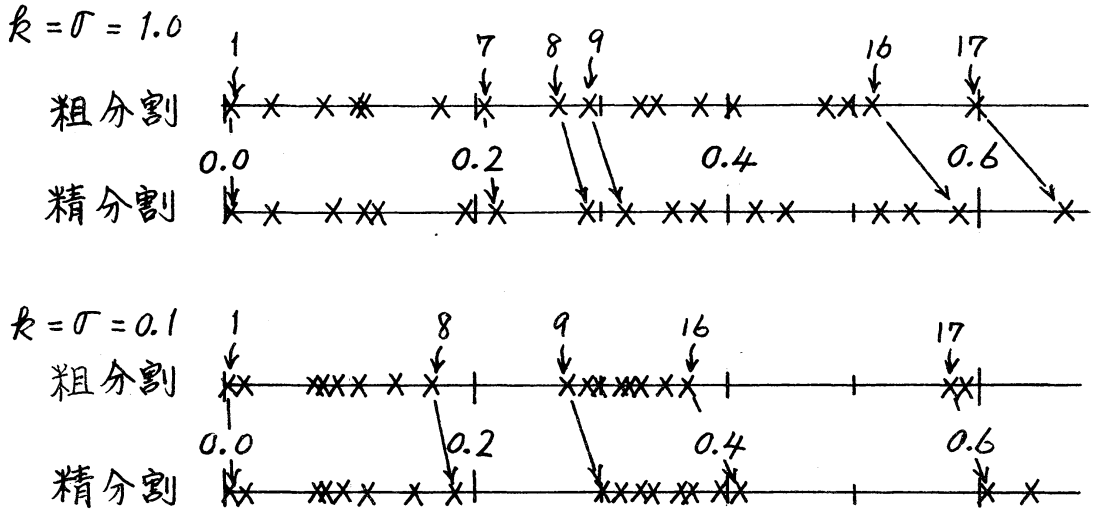


図4.11に $k=\sigma=0.1$ でシフト点を0.0とし8個同時逆反復した場合の収束速度を示す。図4.12にシフト点を0.3にし8個同時逆反復した場合の結果を示す。図4.12では前に計算した1から8番までの固有ベクトルと同時逆反復を行うごとに直交させて計算した。図4.13に $k=\sigma=1.0$ でシフト点を0.0とし7個同時逆反復した場合の結果を示す。いずれも精分割の結果である。初期ベクトルは粗分割の固有ベクトルを線形補間して作成したものである。比較のためにi番固有ベクトルの初期ベクトルはi/2の周期の一次元の正弦波で作成した場合の結果を示す。方式1から方式3への切り換えは，最も収束の遅いものの精度 $(-\log_{10} \max_i \|A x_i - \lambda_i x_i\|_2 / \|A\|_2)$ が4.0以上で行った。

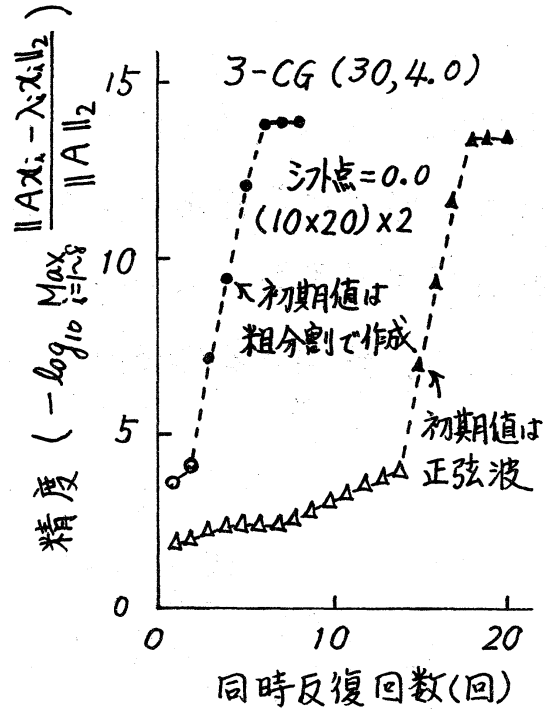


図4.11 最も収束の遅い固有ベクトルの精度
(最小より8個同時反復, $k=\sigma=0.1$)

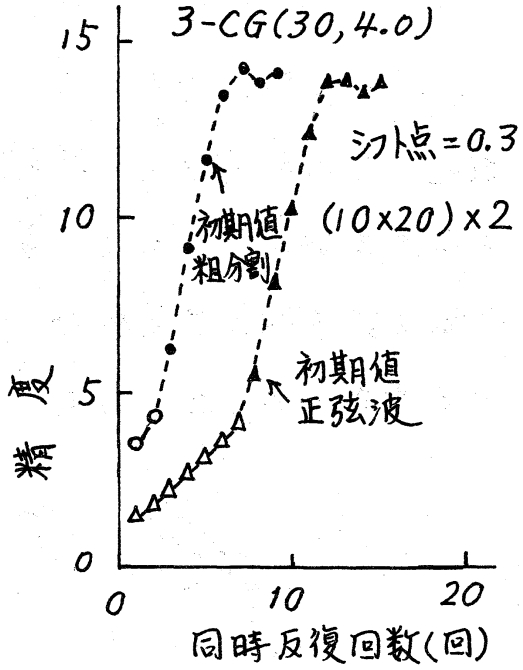


図4.12 最も収束の遅い固有ベクトルの精度
(9~16番の同時反復, $k=\sigma=0.1$)

粗分割の固有ベクトルを使用して, 精分割の初期ベクトルを作成すると収束が圧倒的に速くなることがこれらの図より分かる。

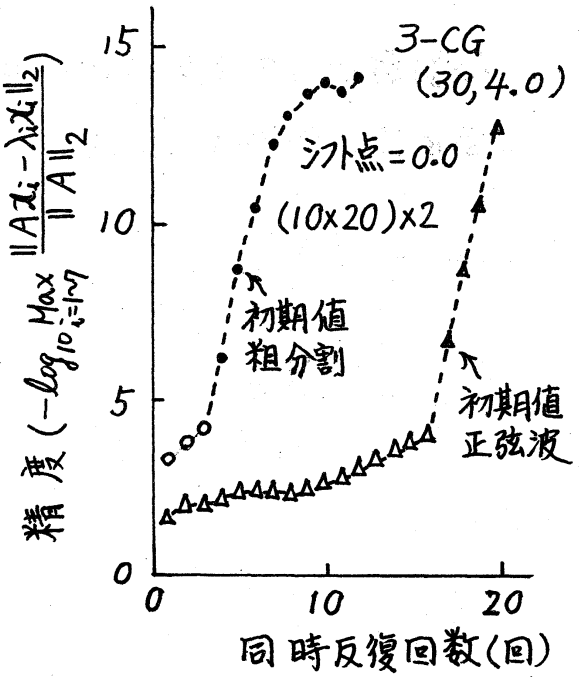
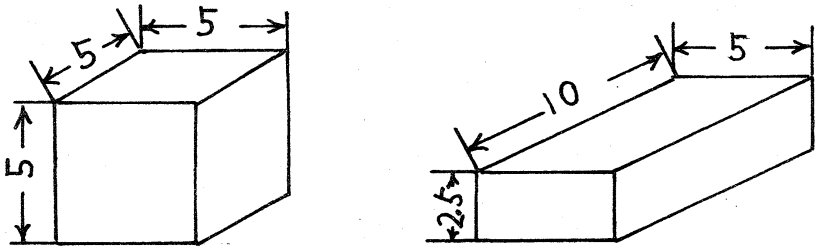


図4.13 最も収束の遅い固有ベクトルの精度
(最小より7個同時反復, $k=\sigma=1.0$)

4.5 3次元問題でのテスト例

3次元拡散方程式 $\nabla\cdot(-k(x)\nabla u) = \lambda u$ を中心差分で離散化した行列をテスト問題とした。解析対象領域の表3面は自由境界条件 $\nabla u = 0$, 裏3面は固定境界条件 $u=0$ とした。拡散係数 $k(x)$ は全領域で 1.0 とし, 解析対象領域は次の二つを使用してテストした。



ケース1　　ケース2

分割数はいずれも 20×20×20 で次元数 8000 である。固有値の分布は次の通りである。

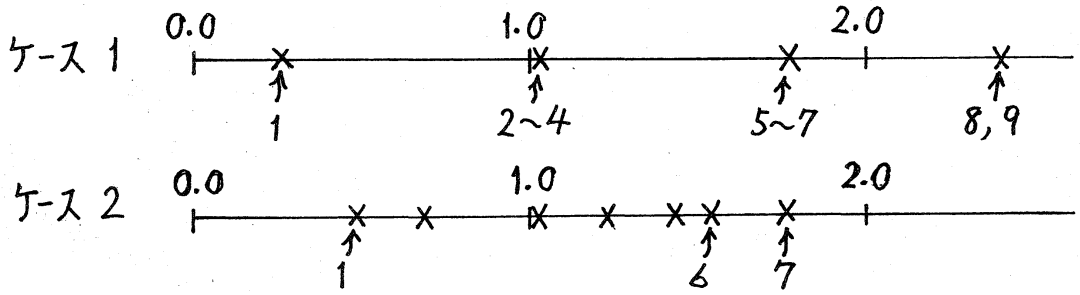


図4.14にケース1で8個同時反復した場合の収束速度を示す。図4.15にケース2で6個同時反復した場合の収束速度を示す。初期ベクトルはいずれも 10×10×10 分割で計算した

固有ベクトルから線形補間で作成した。計算は最初方式1のCG法で計算し，途中で方式3に切り換えた。また初期値を正弦波で作成したものも比較のために示した。

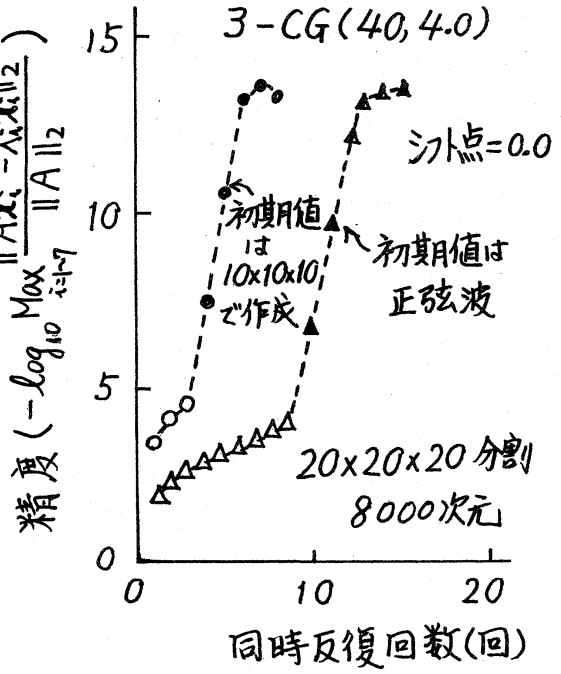


図4.14 最も収束の遅い固有ベクトルの精度
（最小より7個同時反復，ケース1）

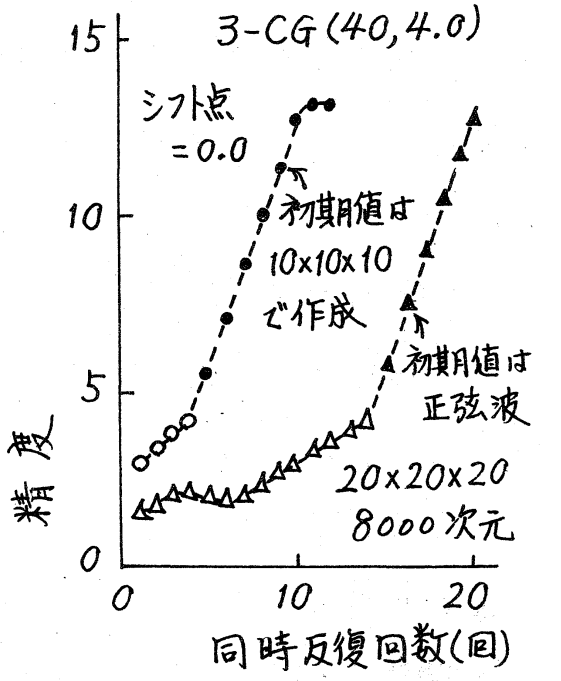


図4.15 最も収束の遅い固有ベクトルの精度
（最小より6個同時反復，ケース2）

本計算はHITAC M-280Hの倍精度演算で，1.2Mバイトの仮想記憶容量を使用して計算した。ケース1で7個の固有ベクトルをすべて10桁の精度まで計算するCPU時間は90秒であった。

5. おわりに

スッルム・同時逆反復法においてUSEタイムの負担を小さくするため，同時逆反復過程にICCG法及びCG法を適用することを検討した。通常の連立一次方程式の計算においては，ICCG法はCG法に比較して収束が格段に速いことが知られている。しかし同時逆反復法と組合せて使用する場合はCG法の方が収束が安定することが判明した。(附記1参照)

CG法及びICCG法は同時逆反復の初期段階で適用するもの2種類と，一定の収束の後で適用するもの1種類を検討対象とした。その結果，初期段階に適用する方式としては連立一次方程式の計算をCG法で単純に置き換えたものの収束が最も安定していた。一定の収束の後では，レーリー商入に対して関数 $f(x)=(x,Ax)-(x,\lambda x)$ の停留点を求めるようにCG法で定式化した。この方式は初期段階で適用した方式より格段に収束が速かった。

偏微分方程式の離散化で得られる行列においては，初期ベクトルの作成に工夫ができる。粗分割で作成した行列の固有ベクトルを，線形補間して精分割の初期ベクトルとすると本方式の場合収束が格段に速くなることが分った。(附記2参照)

本方式は初期段階の方式から加速段階の方式への切り換えタイミング等まだ詳細には検討していない部分も多いが，3

次元問題ではUSEタイムを減少させる有力な方法であると考える。

謝辞　適切な御助言をいただいた筑波大学　名取亮助教授及び東京都立大学　小野寺嘉孝教授に感謝します。

参考文献

(1) 村田，後；スツルム・同時逆反復法，本講究録453，PP 60～88（1982）

(2) Kowalik and Osborne, 山本，小山訳；非線形最適化問題，培風館（1970）

(3) Parlett ; The Symmetric Eigenvalue Problems, Prentice H. (1980)

(4) Meijerink and Vander Vorst ; An Iterative Solution Method for Linear Systems of which the Coefficient Matrix is a Symmetric M-Matrix, Math. Comp. Vol.31, N 137, PP148~162 (1977)

(5) 戸川；共役勾配法，教育出版（1977）

(6) 村田；二段階メッシュ法によるモード解析法，本講究録463，PP33~55（1982）

(附記1)

a) ICCG法でも，ICCG法内の1回の反復毎に既に求まっている固有ベクトルと直交化させてやらせるようにすれ

ばよいわけであるが，それでは毎回既に求めた固有ベクトルをディスクから呼び戻すことになって，現在の目的に沿わない。

b) もっと強いICCG法を適用すれば $(LDL^T)^{-1}$ が A^{-1} に近くなって，初めに一度だけ直交化しておけばあとで混入する率は減ると考えられる。しかしこういう方法を汎用化するのはむずかしいであろう。

c) ここで使うCG法は，以前に求めた固有ベクトルと直交する部分空間内で動作させるわけゆえ，収束が速いわけである。

(附記2)

a) 粗い網目によるところの固有ベクトルの低次のものは精しい網目による固有ベクトルに近い。CG法は

$$e = A^{-1}b - x \quad , \quad f(x) = e^T A e$$

とするとき e^TAe を最小にしようと働らく.

$$e = \sum c_i v_i \text{ のとき } e^TAe = \sum c_i^2 \lambda_i$$

ゆえ，高次のものを強く意識して e^TAe を最小化することになる。従って低次の固有ベクトル成分があるとき収束が遅い。そこで，精しい網目の初期ベクトルは誤差に含まれる低次成分をできるだけすくなくするために，粗い網目による固有ベクトル成分から生成するのである。