

二段階メッシュによるモード解析法

—— スツルム・サブスペース法とスツルム・ランチョス法の
比較に関連して ——

日立 中研 村田 健郎

1 はじめに

対称帯行列固有値問題 $Ax = \lambda x$ に対し、‘スツルム・同時逆反復法’⁽¹⁾ は、与えられた区間 $au < \lambda < al$ にある固有値 λ_i ; $au < \lambda_i < al$, $i = 1, \dots, nc$ と対応する固有ベクトル v_i , $i = 1, \dots, nc$ を所望の精度で求める汎用的な強力な方法である。この方法は、スツルム・逆反復法（日本ではグプタ法としばしば呼ばれる⁽⁴⁾）と、同時逆反復法（サブスペース法⁽⁴⁾）のそれぞれがもっている長所を生かし、短所を表面化させないように、特徴的なグループ分け法の採用によってひとつのプログラムにまとめ上げたものである。・本文を通じて帯半巾を m , 元数を n とする。

一方、スツルム・ランチョス法⁽²⁾ は、最近話題になっている Parlett⁽³⁾ の方法、即ち選択的直交化つきランチョス法の考え方に従って作成したランチョスプログラム LANSSO によって、上述の同時逆反復法のところを置き換えたもので

ある。こちらは全く試作的なもので、もし汎用化しようとするところから多くのことをしなければならぬものではあるが、比較的固有値の分布が素直な問題分野に於いては、スツルム・同時逆反復法よりも強力となるかも知れぬという感觸を得ている。

そこで、スツルム・同時逆反復法の方にもう一息肩入れする意味から、チェビシェフ加速⁽⁵⁾を加えることと並んで、同時逆反復法がもっているところの潜在的な長所：

‘よい初期ベクトル系を手えることができれば非常に効率的’を積極的にひき出すことを考えた*。それが表記の、二段階メッシュによるモード解析法である。

尚、前報のスツルム・ランチョス法⁽²⁾には、不備のところがあって、スツルム・同時逆反復法との比較についての議論も歯切れの悪いところがあったので、そのところをクリアーにする必要がある。その他、読者の通読の便宜を考えて、文献(1), (2)に書いたことの、現時点から見ての（改められた）要約から始めることにする。

* じつは、ランチョスは魅力はあるけれども、汎用化のむづかしさはまだまだ格別との予感から、これを深追いすることは、できれば敬遠したいと思ったからである。

2 スツルム・同時逆反復法の大筋

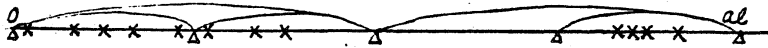
$\left\{ \begin{array}{l} Ax = \lambda x, A: \text{real symmetric positive def. band matrix の} \\ 0 < \lambda_i < al \text{ なる } \lambda_i \text{ と } v_i (i=1, \dots, nc) \text{ を 所望の精度 } eps \text{ で求める} \end{array} \right\}$

$al, eps, eps0, llm$ を与える.

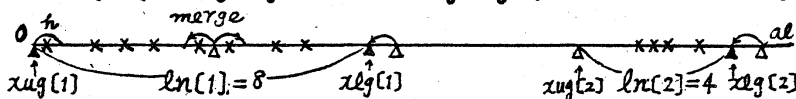
Call SSTRM (al , nc) { al を与えて $A - al * I = U^T D U$ の D の
input output
要素 d_i の負のものの個数 nc を求める } (1)

$epsbi = (al/nc) * 5$ { 但し今回のテストでは $epsbi$ を外から与える. }
 $h = (al/nc) * 0.5$

Repeat BISECT until $(xl[k] - xu[k]) < epsbi$
{ SSTRM 使用の二分法によつて粗いグループ分けを行う } (2)



合理的なグループ分けを行なう. (SSTRM を再び使用)
グループ番号 ig , 固有値個数 $ln[ig]$,
グループ下限 $xug[ig]$, 上限 $xlg[ig]$, グループ個数 $igmax$ を決定. (3)



$k=1$
do $ig=1, igmax$

$kp1 = k-1 + ln[ig]$ { $\frac{\lambda_{k-1}}{xug[ig]} \times \dots \times \frac{\lambda_{kp1}}{xlg[ig]}$ }

{ Subspace Iter. (or Lanczos w. Select. Orth.) }

get $\bar{\lambda}_i; \bar{v}_i; difi0 = \|A \bar{v}_i - \bar{\lambda}_i \bar{v}_i\| / al$ ($i = k$ to $kp1$) (4)

do $i = k, kp1$

$dif = difi0$; $ll = 0$; $\lambda = \bar{\lambda}_i$; $v = \bar{v}_i$

while $(dif > eps) \wedge (ll \leq llm)$ do { Inv. Iter. w. Rayleigh Shift }

Solve $\tilde{v}$; $(A - \lambda I) \tilde{v} = v$

$v = \tilde{v} / \|\tilde{v}\|_2$; $\lambda = v^T A v$ { Rayleigh Q. }

$dif = \|A v - \lambda v\| / al$; $ll = ll + 1$

$\lambda_i = \lambda$; $v_i = v$; $difi = dif$ (5)

check routine; write routine

$k = k + ln[ig]$

(1) ~ (5) についての概略説明

(1) SSTRM : ‘数個 (4 個程度) 先までの対角要素の絶対値最大のものをピボットとする行, 列交換のガウス’である。オーソドックスな Martin-Wilkinson の特殊ガウスによるのでは CPU タイムの負担があまりにも大きいからこれによって代用したのである。次頁にアルゴリズムの主要部を示す。(実際には $(m+kad)^2$ 語の中で処理するようプログラムを書く。)
 $\uparrow_4$

(2) BISECT : 通常の Bisection 法と比べると、甚だしく粗い $epsbi$ 値を使うところが特徴である。このプログラムによって粗い区分けを行なうのであるが、このまゝでは隣り合う区間の境界の両側に甚だしく近接した固有値が存在していることがある。それが検知されない。そこで次の (3) が必要となるのである。(図例は後述の問題 A1, $epsbi=0.4$ のもの。)

(3) 合理的なグループ分け : この部分が、スツルム・同時逆反復法の核心部である。アルゴリズムの大筋を次々頁に示す。re-grouping by SSTRM と呼ぶ。

(図例は後述の問題 A1, $epsbi=0.4$ のもの)

10 Continue {repeat} の部分が一般のグループに関するところである。その前の部分は左端のグループに対する特別処置、後の部分は、右端のグループに対する特別処置である。

main part of the SSTRM {アルゴリズムの大筋だけ。詳しくは文献(1)}

$m = m1 + kad$; $ir = 0$; $epsir = 0.2 \times 10^{-15}$; $nct = 0$; $anorm = 0.0$

do $k=1, n$

$ipk = k$; $amax = abs(aa[k, k])$

do $i=k+1, \min(k+kad, n)$

$aik = abs(aa[i, i])$

if $aik > amax$ then

$amax = aik$; $ipk = i$

if $ipk \neq k$ then

$aa[k, k] \rightleftharpoons aa[ipk, ipk]$

do $j=k+1, ipk-1$

$aa[k, j] \rightleftharpoons aa[j, ipk]$

do $j=ipk+1, \min(k+m, n)$

$aa[k, j] \rightleftharpoons aa[ipk, j]$

$anorm = \max(anorm, \sum_{j=k}^{jmax} |aa[k, j]|)$

if $amax < epsir * anorm$ then

$ir = ir + 1$; $\alpha = \alpha + epsir * 100^{ir}$; go to 500 {A- α Iの作り直し}

do $j=k+1, \min(k+m, n)$

$ak[j] = aa[k, j]$

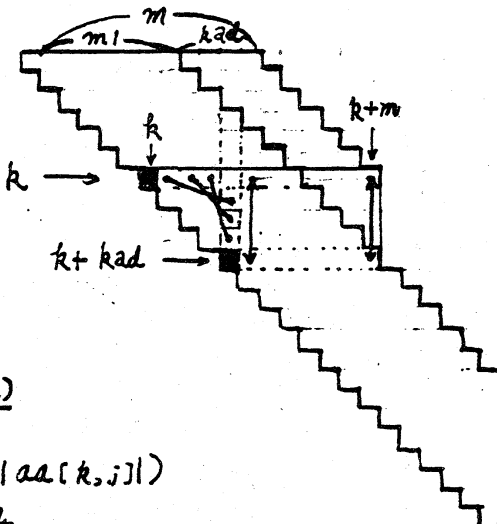
do $i=k+1, \min(k+m, n)$

$t = ak[i] / aa[k, k]$

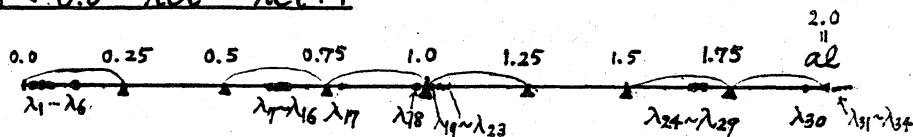
do $j=i, \min(k+m, n)$

$aa[i, j] = aa[i, j] - t * ak[j]$

if $aa[k, k] < 0.0$ $nct = nct + 1$



BISECT



$k = 1$

50 continue {start pt. of repeat}

if $xl[k] - xu[k] < epsbi$ go to 200

60 continue {start pt. of repeat}

$alfa = (xu[k] + xl[k]) / 2.0$

call SSTRM (alfa, nct)

do $i=k, nct$

$xl[i] = alfa$, $nc[i] = nct$

do $i=nct+1, nc$

$xu[i] = \max(alfa, xu[i])$

if $(xl[k] - xu[k]) > epsbi$ go to 60

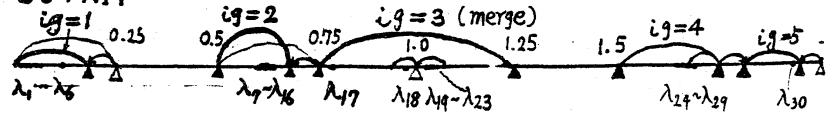
200 continue

$k = nc[k] + 1$

if $k \leq nc$ go to 50

k	nc[k]	xu[k]	xl[k]
1	6	0.0	0.25
2	6	0.0	0.25
3	6	0.0	0.25
4	6	0.0	0.25
5	6	0.0	0.25
6	6	0.0	0.25
7	16	0.5	0.75
8	16	0.5	0.75
9	16	0.5	0.75
10	16	0.5	0.75
11	16	0.5	0.75
12	16	0.5	0.75
13	16	0.5	0.75
14	16	0.5	0.75
15	16	0.5	0.75
16	16	0.5	0.75
17	18	0.75	1.00
18	18	0.75	1.00

re-grouping by SSTRM



```

k = nc[1] ; lnig = nc[1] ; igadv = 1
ln[1] = lnig ; xug[1] = xu[k] ; xlg[1] = xl[k]
nls = nc[k] ; ig = 1

```

{ここまで、左端グループに対する 特別処理 }

```
k = k + 1
```

10 continue { repeat }

```
k = nc[k] ; lnig = nc[k] - nls ; igadv = 1
```

```
if xu[k] = xlg[ig] then
```

```
  alfa = xlg[ig] - h
```

```
  call SSTRM (alfa, nct)
```

```
  if nct = nls then xlg[ig] = alfa
```

```
else
```

```
  if nct = nls - ln[ig] then xug[ig] = alfa
```

```
  alfa = xlg[ig] + h
```

```
  call SSTRM (alfa, nct)
```

```
  if nct = nls then xu[k] = alfa
```

```
else
```

```
  igadv = 0
```

```
  if nct = nc[k] then xl[k] = alfa
```

```
if igadv = 1 then
```

```
  ig = ig + 1
```

```
  ln[ig] = lnig ; xug[ig] = xu[k] ; xlg[ig] = xl[k]
```

```
else
```

```
  ln[ig] = ln[ig] + lnig ; xlg[ig] = xl[k]
```

```
nls = nc[k]
```

```
k = k + 1
```

```
if k ≤ nc go to 10
```

```
alfa = al - h
```

{この後は、右端グループに対する 特別処理 }

```
call SSTRM (alfa, nct)
```

```
if nct = nc then xlg[ig] = alfa
```

```
else
```

```
  alfa = al + h
```

```
  call SSTRM (alfa, nct)
```

```
  if nct ≠ nc then
```

```
    ln[ig] = ln[ig] + nct - nc ; xlg[ig] = alfa
```

k	nc[k]	xu[k]	xl[k]
1	6	0.0	0.25
2	6	0.0	0.25
3	6	0.0	0.25
4	6	0.0	0.25
5	6	0.0	0.25
6	6	0.0	0.25
7	16	0.5	0.75
8	16	0.5	0.75
9	16	0.5	0.75
10	16	0.5	0.75
11	16	0.5	0.75
12	16	0.5	0.75
13	16	0.5	0.75
14	16	0.5	0.75
15	16	0.5	0.75
16	16	0.5	0.75
17	18	0.75	1.00
18	18	0.75	1.00

(4) Subspace Iteration (同時逆反復法)

Rutishauser (1969) のアルゴリズムによった。(文献(3)に見通しのよい講義がある。) 下記の do l=1, lp のループが行列 $(A - \alpha I)^{-1}$ に対して Rutishauser のアルゴリズムを適用したところであるが、この中で $A - \alpha I$ をコレスキー $U^T D U$ 分解するプログラムについては注意が要る。(詳細は文献(1), 要は, α がたまたま固有値に甚だしく近接するなどのケースがあるのを警戒しての処置が必要なのである。) また, ページ交換をすくなく抑えるためのプログラム上の工夫が要る。

SUBSPACE

```

 $\alpha = 0.6 * xug[ig] + 0.4 * xlg[ig]$ 
set initial vectors  $v_k, v_{k+1}, \dots, v_{kp1}$  ( $v_i \perp v_j$ ,  $j = k - \ln[ig-1] \text{ to } kp1$ )
 $j \neq i$ 
"  $X = \{v_k, \dots, v_{kp1}\}$ 
"  $lp = \max(\ln[ig]/2 + 1, 2)$ 

|    |   |   |   |   |   |   |   |   |   |    |
|----|---|---|---|---|---|---|---|---|---|----|
| ig | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| lp | 2 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6  |

 $lg = 0$ 
199 continue {repeat}
     $lg = lg + 1$ 
    do  $l = 1, lp$ 
        solve  $\bar{X} : (A - \alpha I) \bar{X} = X$ 
         $H = \bar{X}^T X$ 
        solve  $P : HP = P\Delta$  (Eigen Solution)
        form  $X = \bar{X} P$ 
        normalize  $X$ 
    do  $i = k, kp1$ 
         $\bar{\lambda}_i = v_i^T A v_i$ 
         $difio = \|Av_i - \bar{\lambda}_i v_i\| / \alpha l$ 
        if  $(lg < lgm) \wedge (difio > eps0) \wedge (i \neq kp1)$  go to 199

```

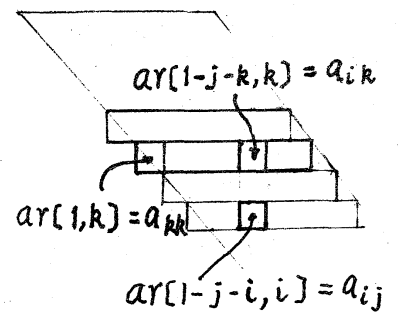
(5) レーリ商シフトつき逆反復 : 殆んどの $\bar{u}_i, \bar{v}_i$ が、
 (4) によって所望の精度に収まるのであるが、(4) で収束し
 損ねたものを (5) によって所望の精度に収めるわけである。
 ここで使うガウスも、オーソドックスなものでは負担が大き
 いので、SSTRM で採用した '対角上軸選択の対称ガウス'
 と本質的には同じものを使う :

GUSBR ($b = \bar{v}_i, \alpha = \bar{u}_i$ を与えて $(A - \alpha M) \bar{v}_i = \bar{v}_i$ なる $\bar{v}_i$ を b に立てる.)

* | ここまでは SSTRM と同じ。但し、 $ar[1,k] = aa[k,k], ar[1+j-i, i] = aa[i, j]$
 etc.

```

ip[k] = ipk
if ipk ≠ k then
  ar[1, k] ⇐ ar[1, ipk]
  do j = k+1, ipk-1
    ar[1+j-k, k] ⇐ ar[1+ipk-j, j]
  do j = ipk+1, min(k+m, n)
    ar[1+j-k, k] ⇐ ar[1+j-ipk, ipk]
  b[k] ⇐ b[ipk]
if ·amax < epsir ar[1, k] = SIGN(epsir, ar[1, k])
do j = k+1, min(k+m, n)
  ak(j) = ar[1+j-k, k]
do i = k+1, min(k+m, n)
  t = ak(i) / ar[1, k]
  b[i] = b[i] - t * b[k]
  do j = i, min(k+m, n)
    ar[1+j-i, i] = ar[1+j-i, i] - t * ak[j]
do k = n, 1, -1
  s = 0.0
  do j = k+1, min(k+m, n)
    s = s + ar[1+j-k, k] * b[j]
  b[k] = (b[k] - s) / ar[1, k]
  if ip[k] ≠ k then
    B[k] ⇐ B[ip[k]]
  
```



3 スツルム・ランチョス法の大筋

2の初めに示したスツルム・同時逆反復法の Subspace Iter. とあるところ即ち(4)の部分を、Parlett流の選択的直交化つきランチョスにとり換えたものである。但し、各グループ毎にシフト実 α をえらんでの $(A - \alpha \times I)^{-1}$ に対するランチョスとしたから、ParlettのLANSOと区別するためShiftのSを追加してLANSSOと名付けることにした。(次頁)

【コメント】

1) このアルゴリズムの目的は、スツルム・ランチョスの潜在的可能性を調べるのが主たる目的ゆえ、ParlettのLANSOとくらべると選択的直交化の‘選択’に相当する手順が甚だしく粗くなっているが、その実は今回の比較では不問に付す。

2) 文献(2)執筆の時実では、密集度の甚だしい固有値系をもつ問題を精度よく求めようとするとき、うまく行かないことがあった。(即ち後述の問題A1, C1のうちA1がうまく解けなかった。)理由は、LANSSOに使用のコレスキー分解プログラムと、後続のレーリー商シフトつき逆反復に使用したコレスキー分解に不備な実があったためである。今回はその実を改良して、データをとり直した。

LANSSO

```

 $\alpha = (xug[iq] + xlg[iq]) / 2.0$  ;  $h = (xlg[iq] - xug[iq]) / 2.0$ 
 $B = A - \alpha I$ 
 $B = U^T D U$  (cholesky)
 $X = (DU)^{-1} e$  { initial vector ; by wilkinson }
 $jmax = 16 + 8 * \ln[iq]$  ;  $jmax0 = 8 + 4 * \ln[iq]$ 
 $\beta = (X^T X)^{\frac{1}{2}}$  ;  $q_0 = 0$  ;  $norm = 0.0$ 
do j = 1, jmax

```

```

 $q = X / \beta$ 
 $u = B^{-1} q$ 
 $X = u - \beta q_{j-1}$  ;  $\alpha = q^T X$ 
 $X = X - \alpha q$  ;  $\beta = (X^T X)^{\frac{1}{2}}$ 
 $q_j = q$  ;  $ta[j] = \alpha$  ,  $tb[j] = \beta$ 
 $norm = \max(norm, |\alpha| + 2|\beta|)$ 

```

```

 $T_j = S \Lambda S^{-1}$  { eigenvalue only }  $\Lambda = \begin{bmatrix} \lambda_{j1} & 0 \\ 0 & \ddots & \lambda_{jj} \end{bmatrix}$ 
 $\mu_{ji} = 1/\lambda_{ji}$  ,  $|\mu_{j1}| \leq |\mu_{j2}| \leq \dots \leq |\mu_{jj}|$ 

```

```

 $ngd = \max \{ i ; -h - epsq < \mu_{ji} < h + epsq \}$ 
 $difom = 0.0$  ;  $count = 0$ 
do i = 1, ngd

```

```

compute  $s_i = (s_{i1}, \dots, s_{ji})^T$  { eigenvector of  $\mu_{ji}$  }
 $b_{ji} = |tb[j]| * |s_{ji}|$  {  $s_i = (s_{i1}, \dots, s_{ji})^T$  }
 $epsqn = epsq * norm$ 
if  $b_{ji} < epsqn$  then

```

```

 $y_i = Q_j s_i$  {  $Q_j = \{q_1, \dots, q_j\}$  }

```

```

 $\lambda_i = \alpha + \mu_{ji}$ 

```

```

 $X = X - (X^T y_i) y_i$  { re-orth. }

```

```

 $difo = \|A y_i - \lambda_i y_i\|$ 

```

```

 $epsc = epse * norm$  ;  $difom = \max(difo, difom)$ 

```

```

if  $(difo < epsc) \wedge (-h \leq \mu_{ji} \leq h)$  count = count + 1

```

```

if  $(count > \ln[iq]) \wedge ((difom < eps0) \vee (j > jmax0))$  go to 400

```

```

if count <  $\ln[iq]$  ifsok =  $\ln[iq] - count$ 

```

```

write iq, j, difom, ifusok

```

```

400 continue

```

4 スツルム・同時逆反復法とスツルム・ランチョス法の比較

Test Matrix : 下記の A, C ニ種の二次元場の、 $\text{div}(-k \nabla u)$ を有限要素法で離散化した行列をテスト行列とした。

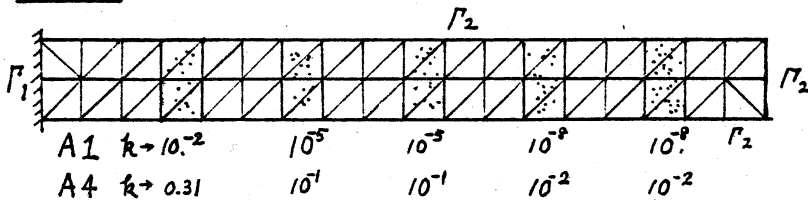
A1 と C1 : $k(x)$ 値の変化が大きく、甚だしい密集固有値をもつ問題。ランチョスにとって苦手な問題である。

(問題 A1 は、前報⁽²⁾のスツルム・ランチョス法では、LANSSO で収束しない固有値が甚だ多く、かつその中には後続のレーリー商逆反復が停滞するものまであったものである。今度は、それなりのデータを得た。)

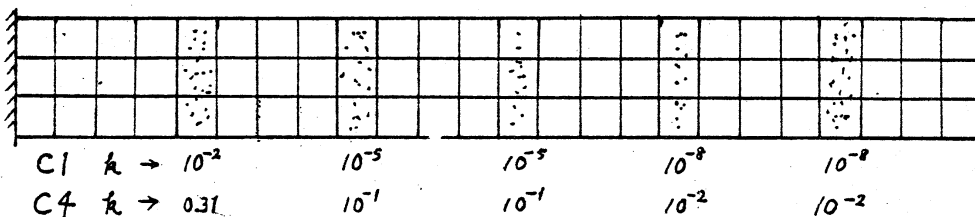
A4 と C4 : $k(x)$ 値の変化が小さく、甚だしい密集固有値は無い問題。ランチョスにとって得意な問題である。

(境界条件: $\Gamma_1 : u=0$, $\Gamma_2 : \nabla u=0$)

Prob.A



Prob.C



eps 値 は、 10^{-9} , 10^{-14} の二ケース、epsbi 値 は 0.4 と 0.2 の二ケースについてデータを示した。(次頁以降)
CPU カウント, USE カウント のとり方については後述する。

Prob. A1 $\epsilon = 10^{-9}$ $\alpha = 2.0$ ($n_c = 30$)
 $LQM = 5$

RL: L-リ商逆反復の略
 $L = 4$

epsbi	Common		SUBSPACE				TOTAL		LANSSO				TOTAL	
	SSTRM	LU	Iter.	$4P_i V_i / m$	$P_i V_i / m$	RL	CPU	USE	Iter.	$4P_i V_i / m$	$P_i V_i / m$	RL	CPU	USE
0.4	14	5	72	11	3	1	31	81	121	3	1	0	22	127
0.2	20	7	76	10	3	0	37	81	117	3	1	0	30	125

Prob. C1 $\epsilon = 10^{-9}$ $\alpha = 1.5$ ($n_c = 30$)
 $LQM = 5$

0.4	9	4	60	12	3	2	27	69	138	3	1	14	30	159
0.2	13	6	61	7	2	3	29	72	126	3	1	1	23	134

mean → 31 76

2425 135.75

Prob. A1 $\epsilon = 10^{-14}$ $\alpha = 2.0$ ($n_c = 30$)
 $LQM = 5$

0.4	14	5	90	15	4	8	42	107	163	4	1	16	39	185
0.2	20	7	97	13	3	1	41	108	165	4	1	11	42	184

Prob. C1 $\epsilon = 10^{-14}$ $\alpha = 1.5$ ($n_c = 30$)
 $LQM = 5$

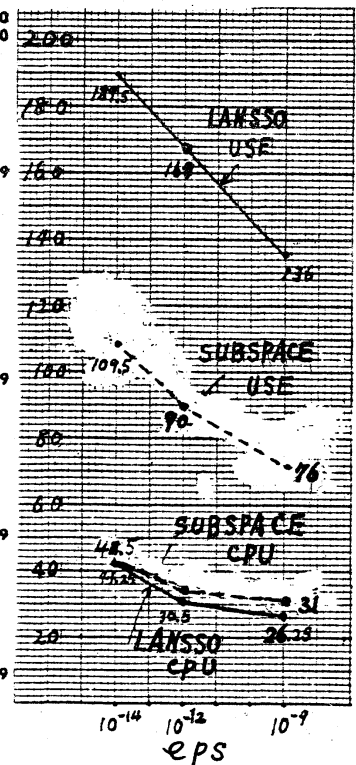
0.4	9	4	95	25	6	7	45	112	159	4	1	24	41	188
0.2	13	6	97	15	4	4	38	111	174	4	1	20	43	201

A1 $\epsilon = 10^{-9}$ output

mean → 42.5 109.5

41.25 189.5

EPSO.LQM	DEL	0.10000-09	5	0.200
1	4	0.00214487952685	0.00214487952685	0.50-15
2	4	0.00000581674257	0.00000581674257	0.20-14
3	4	0.00000084977085	0.00000084977085	0.30-13
4	4	0.00000000581462	0.00000000581462	0.20-13
5	4	0.00000000084758	0.00000000084758	0.10-12
6	4	0.13457983425647	0.13457983425647	0.50-10
7	10	0.5000000000000000	0.5919999999999998	0.729999
8	4	0.63397459755095	0.63397459755095	0.20-11
9	4	0.63397459934207	0.63397459934207	0.10-11
10	4	0.63397460438446	0.63397460438446	0.80-12
11	4	0.63397460491805	0.63397460491805	0.30-11
12	4	0.63397593459094	0.63397593459094	0.30-11
13	4	0.63397772606152	0.63397772606152	0.40-12
14	4	0.63398276779036	0.63398276779036	0.60-12
15	4	0.63398329929128	0.63398329929128	0.10-11
16	4	0.63715601382387	0.63715601382387	0.20-11
17	4	0.63780612130754	0.63780612130754	0.10-11
18	7	0.7500000000000000	0.8579999999999996	1.01999
19	4	0.76728046769175	0.76728046769175	0.10-11
20	4	0.98365489920253	0.98365489920253	0.50-13
21	4	1.00000000396426	1.00000000396426	0.20-11
22	4	1.00000001103431	1.00000001103431	0.10-11
23	4	1.00000396859763	1.00000396859763	0.10-11
24	4	1.00001103528705	1.00001103528705	0.70-12
25	4	1.00509312702005	1.00509312702005	0.30-11
26	10	1.5000000000000000	1.5919999999999998	1.72999
27	4	1.65714795272553	1.65714795272553	0.10-12
28	4	1.69722436754223	1.69722436754223	0.40-11
29	4	1.69722437466897	1.69722437466897	0.20-10
30	4	1.69722964174447	1.69722964174447	0.20-10
31	4	1.69723676343086	1.69723676343086	0.20-10
32	4	1.70294056116073	1.70294056116073	0.20-09
33	7	1.7500000000000000	1.8419999999999996	1.97999
34	1	1.94104742546787	1.94104742395532	0.90-05
35	6	0	0	10
36	8	GINVRY SUBSPACE	LF	10
37	14	72	1	10



Prob. A4 $\text{eps} = 10^{-9}$, $\text{al} = 1.5$ ($\text{nc} = 23$)

*RL: レーリー-商逆交復の略号

epsbi	Common		SUBSPACE				TOTAL		LANSSO (L=4)				TOTAL	
	SSTR	LU	Iter.	4-V ₁ /m	P-V ₁ /m	RL*	CPU	USE	Iter.	4-V ₁ /m	V ₁ /m	RL	CPU	USE
0.4	10	4	84	15	4	6	35	98	90	2	1	0	16	95
0.2	15	6	88	12	3	6	39	103	96	2	1	2	25	105

Prob. C4 $\text{eps} = 10^{-9}$, $\text{al} = 1.5$ ($\text{nc} = 29$)

0.4	10	4	104	25	6	6	45	120	96	2	1	1	17	102
0.2	15	6	84	12	3	3	36	96	110	3	1	1	25	118

mean $\rightarrow$ 38.75 104.25

20.75 105

Prob. A4 $\text{eps} = 10^{-14}$, $\text{al} = 1.5$ ($\text{nc} = 23$)

0.4	10	4	92	15	4	18	47	118	128	3	1	8	25	141
0.2	15	6	100	12	3	9	42	118	146	4	1	8	33	161

Prob. C4 $\text{eps} = 10^{-14}$, $\text{al} = 1.5$ ($\text{nc} = 29$)

0.4	10	4	108	25	6	10	49	128	152	4	1	23	41	179
0.2	15	6	120	18	5	14	53	145	170	4	1	11	33	185

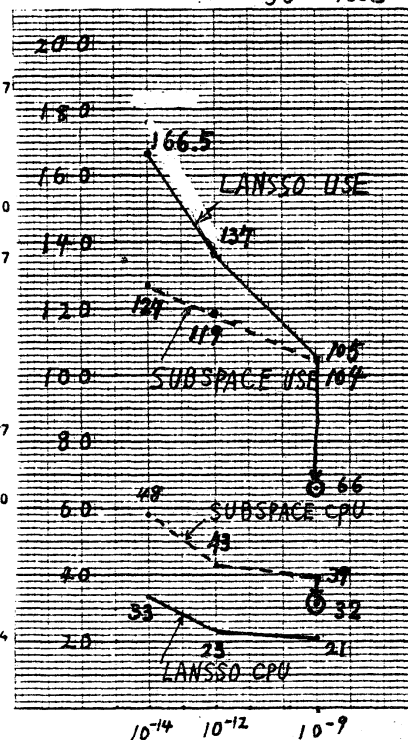
A4 $\text{epsbi} = 0.2$ $\text{eps} = 10^{-12}$ (10^{-14} で同じ) 47.75 127.25

33 166.5

Out put

EPS0, LQM														
1	3	0	0.06958233613252	0.06958233613252	0.20-15	0.1679347								
2	3	0	0.03604532610513	0.03604532610513	0.30-15	0.30-15								
3	3	1	0.00732362476562	0.00732362476562	0.40-09	0.40-15								
4	3	1	0.00450670238372	0.00450670238372	0.40-08	0.30-15								
5	3	1	0.00066163028734	0.00066163028734	0.30-07	0.30-15								
6	2	0	0.1875000000000000	0.2624999999999999	0.20-14	0.3750000								
7	5	0	0.5625000000000000	0.029673913043478237	0.20-15	0.7304347								
8	5	0	0.63517812091124	0.63517812091124	0.20-15	0.20-15								
9	5	0	0.63703731155864	0.63703731155864	0.20-14	0.20-14								
10	5	0	0.64151514789898	0.64151514789898	0.10-15	0.10-15								
11	5	0	0.64245992387869	0.64245992387869	0.10-14	0.10-14								
12	5	0	0.64887717635201	0.64887717635201	0.20-15	0.20-15								
13	5	0	0.66502564282909	0.66502564282909	0.70-15	0.70-15								
14	5	1	0.70710818723366	0.70710818723366	0.20-07	0.40-15								
15	2	1	0.70844914085996	0.70844914085996	0.30-08	0.20-15								
16	2	1	0.7500000000000000	0.817173913043478237	0.10-09	0.9179347								
17	4	0	0.76823559635370	0.76823559635370	0.30-15	0.30-15								
18	4	0	0.76189722384915	0.76189722384915	0.20-06	0.60-15								
19	4	0	0.9375000000000000	1.0124999999999999	0.10-14	1.125000								
20	4	0	1.01097989344215	1.01097989344215	0.10-14	0.10-14								
21	4	0	1.00398808190031	1.00398808190031	0.10-14	0.10-14								
22	4	0	1.04756803514056	1.04756803514056	0.40-14	0.40-14								
23	2	0	0.96028138099873	0.96028138099873	0.60-14	0.60-14								
24	2	0	1.08512399123548	1.08512399123548	0.20-14	0.20-14								
25	2	0	1.11444785532518	1.11444785532518	0.20-10	0.80-15								
26	2	0	1.14456521739130435	1.27891304347826074	0.80-15	1.480434								
27	2	0	1.25326002723534	1.25326002723534	0.80-15	0.80-15								

BISECT 8 7 100 6 LF 8



◎ EP: 4Eビシ7加速付加

CPU カウント : コレスキ分解のための $\frac{1}{2}m^2n$ 積和を 1 カウントとする。分解結果を使っての前進・後退代入は $2mn$ 積和ゆえ $4/m$ カウント, 従って p 個のベクトルを ν 回同時逆反復するには, $4p\nu/m$ カウントとなる。

SSTRM 等で使用する対称ガウスは $\frac{1}{2}(m+4)^2n$ 積和 であるが, $m \geq 80$ の大型問題に興味がある当面の場合, これを 1 カウントとして大勢を誤ることはない。

SUBSPACE の内訳 : $p = \ell n[ig], \nu = \ell q * \ell p$ のとき,

$(A - \alpha I) \bar{X} = X$	$H = \bar{X}^T \bar{X}$	$H = P \Delta P^T$	$X = \bar{X} Q$
$2mnp\nu$	np^2	$\sim p^3$	np

$$\underline{2mnp\nu} / (\frac{1}{2}m^2n) = 4p\nu/m \text{ カウント だけでよい.}$$

USE カウント : コレスキー分解結果を使っての前進・後退代入の一往復に $2mn$ 語のメモリ参照を行なう。我々の興味は、実メモリ容量 $\ll mn$ 語 という大型問題にあるから、 $2mn$ 語分のページ交換が行なわれるとして、 $2mn$ を USE の 1 カウントとする。

SUBSPACE の内訳 p, ν は上と同じとして、

$2(m+p)n\nu$ (語)	$np\nu$	$\sim p^2\nu$	$np\nu$
------------------	---------	---------------	---------

$$\{2(m+p)n\nu + np\nu + np\nu\} / 2mn = \nu + (2p\nu/m) \text{ カウント}$$

SSTRM は、 $(m+4)^2$ 語の小さなワークメモリを使って処理することができるから、USE カウントは無視される。

論評 : 12, 13 頁右下のグラフによって全体像を見ることができる。ここでは $m=160$ として、表、グラフ共調製している。 $m=80$ としたならば、SUBSPACE の $4P_i v_i/m$ 項の方が LANSSO の $4v_i/m$ 項よりも大巾に増す等の事情があらわれて、LANSSO の方が SUBSPACE とくらべ、ここに示した比較より若干有利の方向に変わる。しかし大勢を変えるほどではない。さて、

- (1) 問題 A1, C1 即ち甚だしい密集固有値を持った問題においては LANSSO の方が (CPU はとも角) USE カウントにおいて甚だ不利である。
- (2) 問題 A4, C4 即ち甚だしい密集固有値を持たない問題においては、LANSSO の方が (USE はとも角) CPU カウントにおいて甚だ有利である。

じつは、上記の SUBSPACE には、まだチェビシェフ加速が入っていない。チェビシェフ加速を入れると SUBSPACE の反復回数が 40% 程度減る。ところが 40% 減っても、A4, C4 系の、Total の CPU カウントは 20% 程度減るにすぎず、スツルム・ランチョス系のそれにくらべ 50% 程度多い。(13 頁右下のグラフの中の ⊙ 印を見よ。これは推定値。)

そこでもっと積極的な、‘二段階メッシュ法による肩入れ’を試みようという気になったのである。

5 二段階メッシュによるモード解析法

応用分野: いわゆるモード解析法, 例えば時間依存問題

$$\frac{\partial u}{\partial t} + \operatorname{div}(-k(x) \nabla u) = -c(x)u + f(x, t)$$

を離散固有値問題 $Au = \lambda u$ に帰着させて解く問題分野.

方法の大筋:

- (1) 粗いメッシュ系による離散固有値問題 $A^c u^c = \lambda^c u^c$ の αl 以下の固有値 λ_i^c , $0 < \lambda_i^c < \alpha l$; $i = 1, \dots, n_c$ をスツルム・同時逆反復法によって求める. (c は coarse の c)
- (2) 精しいメッシュ系による $A^f u^f = \lambda^f u^f$ を解くために.

- グループは粗い系で定めた通りのものを使用
- 各グループ g 毎の同時逆反復を行うに際し、 g グループ内の $i = i_{g1}, \dots, i_{gm}$ につき初期ベクトル v_i^{f0} を、 v_i^c からの interpolation によって作る. 次いで λ_i^{f0} をレーリ商:

$$\lambda_i^{f0} := (v_i^{f0})^T A^f (v_i^{f0})$$

によって作り、この λ_i^{f0} , $i = i_{g1}, \dots, i_{gm}$ の平均値を α_g とする.

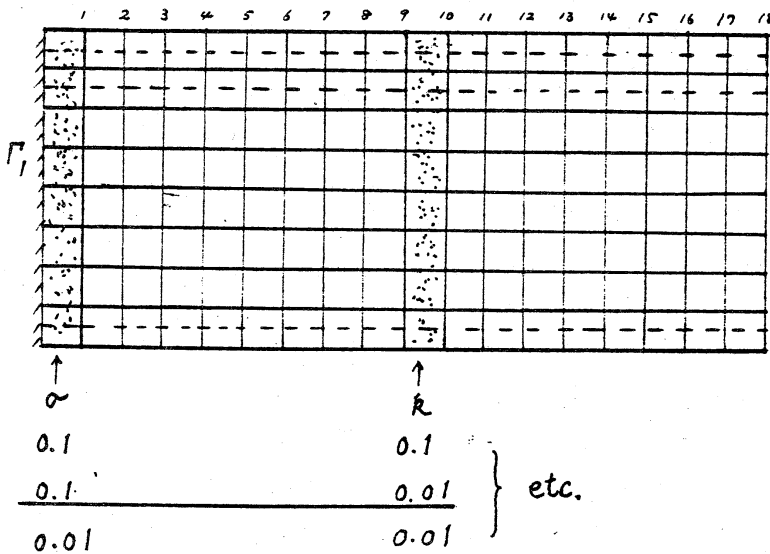
- α_g をシフト点とする同時逆反復:

$$\begin{pmatrix} (A^f - \alpha_g I) X^{(\nu+1)} \\ \vdots \end{pmatrix} = \bar{X}^{(\nu)}$$

を行なう. (f は fine の f)

- 最後に、収束し損ねたものを、個別のレーリ商シフトつき逆反復によって仕上げる.

テスト問題 H : $\text{div}(-k(x) \nabla u)$ より生成される行列.



粗系は $9 \times 18 = 162$ 元

精系は $17 \times 18 = 306$ 元

固有値の分布図を次頁に示す。何れも、0.25 の近くと 0.5 の近くに密集固有値があらわれる。従って

$0 < \lambda < \alpha l$ の αl を 0.5 前後にとってやるときわじいことが起り易くて面白いゆけである。そう考えて αl をえらび、データを取った。上図で左端の  部の $k(x)$ 値を σ 、中央部の  部の $k(x)$ 値を k とし、 σ と k を 0.1 とか 0.01 にとる。白ぬき部の $k(x)$ 値は 1 にとる。

$\sigma = k = 0.01$, $\alpha l = 0.5$ のとき、

$nc = 24$, 従って標準の $epsbi$ は、

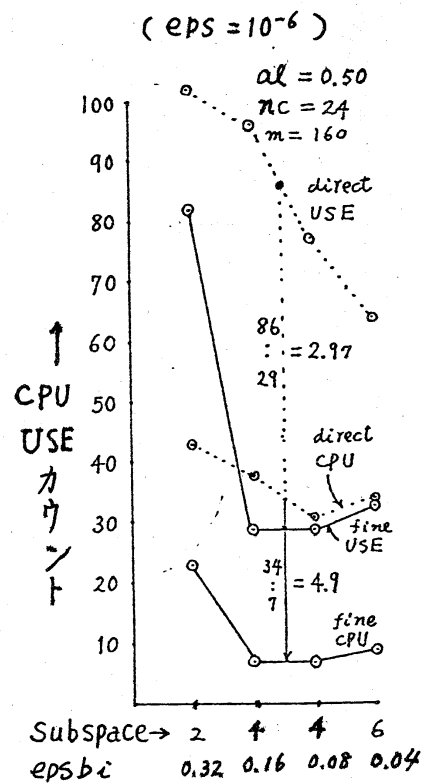
$$(0.5/24) * 5 \doteq 0.1$$

であるが、 $epsbi$ を外から与えて

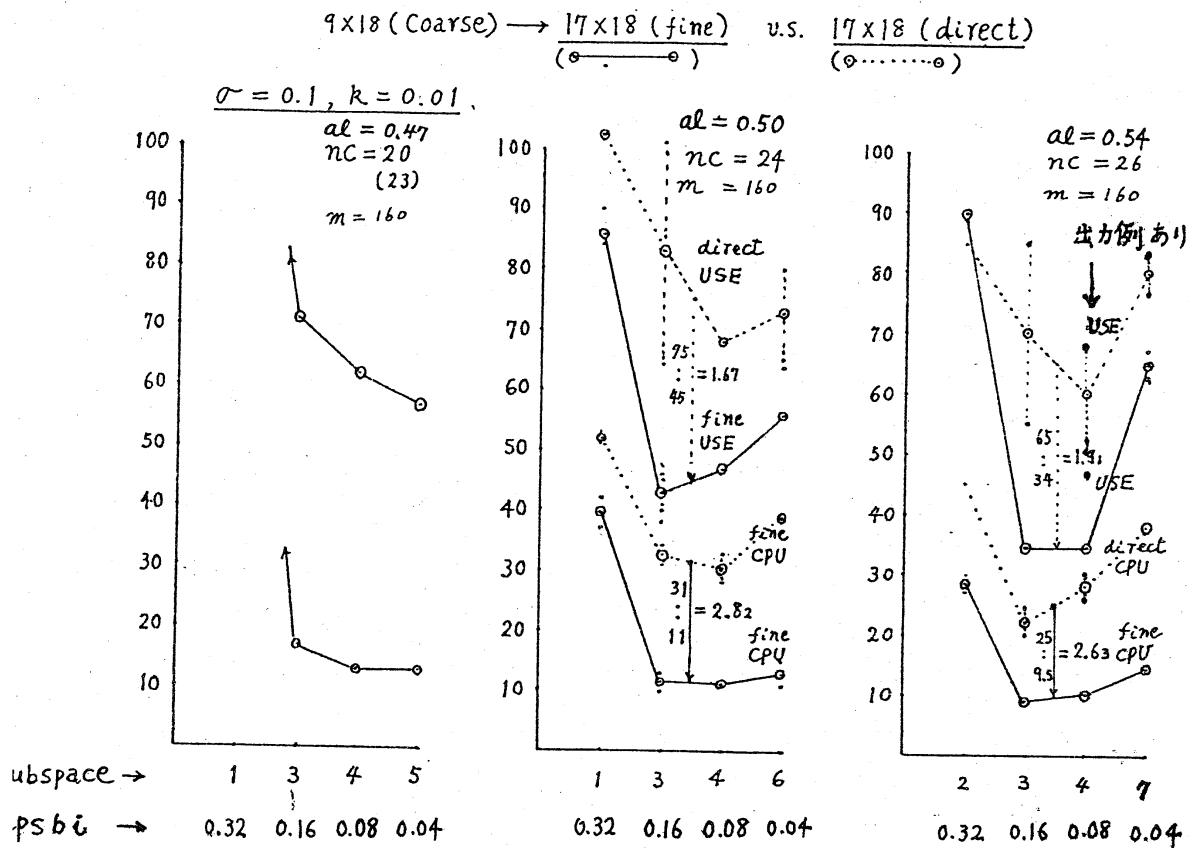
$$epsbi = 0.32, 0.16, 0.08, 0.04$$

の 4 ケースについてデータを取った。

(右図) 点線が直接に解いたもの、実線が二段階メソッドによるものである。



$\sigma = 0.1, \sigma = 0.01$ の場に対しては、 $al = 0.47, 0.50, 0.54$ の三ケースについてとったデータを示そう。それぞれにつき $epsbi$ は $0.32, 0.16, 0.08, 0.04$ の4ケースである。 $epsbi$ の最適設計値 $(al/nc)*5$ はやはり 0.1 前後ゆえ、図の 0.16 と 0.08 の間に最適値があることと、つちがまが合う。



以上、チェビシェフ加速は附加していない。それでも、二段メッシュ法によれば、スツルム・ランチョス法とくらべて、CPUカウント、USEカウント共に文句なく小さい。特に問題視されていたCPUカウントが、直接スツルム・同時逆反復をほどこすときとくらべ、 $1/2.5$ 乃至 $1/5$ になる。

細かいコメント

(1) ここに例示したような‘意地の悪い’問題では、スツルム・同時逆反復法を直接適用するときの USE カウントは、 al をちよつと動かしたとき、可なり違う。例えば前頁右端のグラフ、即ち $al = 0.54$ の場合、 al を 0.01 程度動かしたとき、 $epsbi = 0.08$ のケースで USE カウントが 49 ~ 67 の間にばらついた。(そういうことは、スツルム・ランゲヨスでも同様に起る。) 図の $\vdots$ 線はそのバラツキの範囲を示している。○ Ep は平均値を示している。

(2) $al = 0.47$ (前頁左端) のときは、0.47 という値は、 $\lambda_{20} = 0.46546\dots$, $\lambda_{21} = 0.47003\dots$, $\lambda_{22} = 0.47412\dots$, $\lambda_{23} = 0.47789\dots$ という密集固有値の間である。当然最初の SSTRM カウントによって $nc = 20$ と出るが、regrouping のとき 23 番目のものまで仲間に入れるようになる。しかし λ_{24} も、すぐ右にあるわけゆえ、このときの CPU, USE カウントが他の場合とくらべ特に大きくなるわけである。

$al = 0.50$ のとき $nc = 24$ となり λ_{24} まで求めるが λ_{25} λ_{26} は λ_{24} とくらべまだ可なり近接している。 $al = 0.54$, $nc = 26$ となると、 $\lambda_{26} = 0.5164\dots$ に対し λ_{27} は 0.672... であって、 λ_{26} と λ_{27} とが離れているから、この場合が一番成績が良いわけである。

出力例を1つだけ例示しよう. $al = 0.54$, $epsbi = 0.08$.

上段がスツルム・同時逆反復法を直接ほどこしたもの、下段が、2段階メシユ法によるものである。

Prob. H 17X18 (direct)

$\alpha_l = 0.54$, $\epsilon_{psbi} = 0.08$.

$$\sigma = 0.1, \quad k = 0.01, \quad \text{eps} = 10^{-6}, \quad \text{eps0} = 10^{-7}, \quad \text{LQM} = 5$$

EPSO.LQM.			- 3.0000-04			5		
8 0.0			0.539999999999999968E-01			0.13499999999999995		
1	5	0	0.05750250288748	0.05750250288748	0.20-14	0.20-14	0.20-14	
2	5	0	0.06039854049116	0.06039854049116	0.40-14	0.40-14	0.40-14	
3	5	0	0.06509907101929	0.06509907101929	0.30-14	0.30-14	0.30-14	
4	5	0	0.06751919909665	0.06751919909665	0.20-14	0.20-14	0.20-14	
5	5	0	0.00482384434455	0.00482384434455	0.20-10	0.20-10	0.20-10	
6	5	0	0.10695153548022	0.10695153548022	0.60-09	0.60-09	0.60-09	
7	5	0	0.00042725006469	0.00042725006469	0.30-08	0.30-08	0.30-08	
8	5	0	0.11717878372461	0.11717878372461	0.50-07	0.50-07	0.50-07	
0 0			0 8			20 160		
8 0.202499999999999986			0.231576923076923055			0.275192307692307681		
9	5	0	0.23115836094885	0.23115836094885	0.10-10	0.10-10	0.10-10	
10	5	0	0.22590909481745	0.22590909481745	0.40-09	0.40-09	0.40-09	
11	5	0	0.23783240353611	0.23783240353611	0.50-10	0.50-10	0.50-10	
12	5	0	0.21688821269918	0.21688821269918	0.60-08	0.60-08	0.60-08	
13	5	0	0.25809199696860	0.25809199696860	0.60-10	0.60-10	0.60-10	
14	5	0	0.25942077947446	0.25942077947446	0.10-10	0.10-10	0.10-10	
15	5	0	0.26788081909681	0.26788081909681	0.10-09	0.10-09	0.10-09	
16	5	0	0.27140637233400	0.27140637233400	0.90-08	0.90-08	0.90-08	
8 0			0 8			10 80		
10 0.404999999999999985			0.454846153846153803			0.529915384615384599		
17	6	0	0.45780835733505	0.45780835733505	0.50-11	0.50-11	0.50-11	
18	6	0	0.46546082127325	0.46546082127325	0.40-10	0.40-10	0.40-10	
19	6	0	0.47003307736395	0.47003307736395	0.40-09	0.40-09	0.40-09	
20	6	0	0.43724449666209	0.43724449666209	0.20-09	0.20-09	0.20-09	
21	6	0	0.47412804463708	0.47412804463708	0.20-09	0.20-09	0.20-09	
22	6	0	0.47789865784922	0.47789865784922	0.50-09	0.50-09	0.50-09	
23	6	0	0.43019819409561	0.43019819409561	0.70-09	0.70-09	0.70-09	
24	6	0	0.48940703264051	0.48940703264051	0.30-09	0.30-09	0.30-09	
25	6	0	0.50280573290504	0.50280573290504	0.20-08	0.20-08	0.20-08	
26	6	0	0.51648349038081	0.51648349038081	0.80-08	0.80-08	0.80-08	
8 0			0 10			12 120		
BISECT			GINVRY SUBSPACE			LF TOTALF TOTALC		
7 8			3 0			360 9 4		
42			45			69		

Prob. H 9×18 (coarse) $\rightarrow 17 \times 18$ (fine)

$$\sigma = 0.1, \quad k = 0.01, \quad \overline{eps} = 10^{-6}, \quad eps0 = 10^{-7}, \quad LQM = 5$$
[illegible]

6 結言 (まとめ)

1. スツルム・同時逆反復法

従来のスツルム・逆反復法，単なる同時逆反復法の何れとくらべても、多くの場合 CPU，USE カウント両面において有利である。汎用性の点で抜群と言って良い。帯半中 160 で、

CPU カウント： 1 固有値当り

$\text{eps} = 10^{-14}$ のとき	1.8 前後 (1.4~2.2 が殆んど)
$\text{eps} = 10^{-9}$ のとき	1.4 前後 (1.0~1.8 が殆んど)
$\text{eps} = 10^{-6}$ のとき	1.2 前後 (1.0~1.6 が殆んど)

USE カウント： 1 固有値当り (こちらはやゝばらつきが多い)

$\text{eps} = 10^{-9}$ のとき 2.5 ~ 5.5

2 スツルム・ランチョス法： こちらは試作の段階で、汎用化のためには未だ多くのことを為さねばならぬが、見込みとしては、密集固有値の甚だしいものがない、比較的素直な問題の場合、スツルム・同時逆反復法とくらべ、USE カウントは免も角として、CPU カウントにおいて著しく有利 (30%~50% 小) となる可能性がある。

3 スツルム・同時逆反復法の二段階メッシュ法： (これは、汎ゆる帯固有値解析に使えるというわけでは勿論無いが、) CPU カウントを、通常のスツルム・同時逆反復法と比べ、 $1/2.5 \sim 1/5$ に減らす効果がある。

4 スツルム・同時逆反復法に関するコメント (附記の程度)

合理的なグループ分けがなされるために、各グループ毎のチェビシェフ加速⁽⁵⁾(2次)が、比較的安定に利く筈である。

(問題Hにおいて一部試みている。それによると、同時逆反復の回数が、20%及至50%減る。全体的なCPUカウントでみると、10%~20%の減の程度である。)

同じく、合理的なグループ分けと、合理的なシフト量の修正手法などによって同時逆反復 $(A - \alpha I)X^{(v)} = X^{(v-1)}$ のところを、不完全LU分解とCG法の組合せ⁽⁶⁾によって攻めることが、比較的安定に可能になるであろうと期待している。これについての予備的報告が文献(7)になされている。

文献

- (1) 村田・後 'スツルム・同時逆反復法' 京大数理解析研 講究録 453号
(1982年2月) 60p-88p
- (2) 村田・梅谷 'スツルム・ランチョス法' 全上 89p-101p
- (3) Parlett 'The Symmetric Eigenvalue Problems' (1980, PrenticeH)
- (4) 戸川 '振動解析' (1975) サイエンス社
- (5) Wilkinson・Reinsch Linear Algebra (1971) Springer の中の
Rutishauser: 'Simultaneous Iteration Methods' (Contribution II/8)
- (6) Meijerink & Van der Vorst, Math. Comp. 31 (1977) 148p-162p,
全上 J. Computational Phys. (1981) 134p-155p
- (7) 後: '不完全三角分解と共役傾斜法による大次元行列の固有値計算'
(1981) 情報処理学会第22回全国大会 講演論文集 811p-812p