数理解析研究所講究録
463巻 1982年 33-55

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 ; $au<\lambda_i<al$, $i=1,\cdots,nc$ と対応する固有ベクトル v_i , $i=1,\cdots,nc$ 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れ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触を得ている。

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与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、そこ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

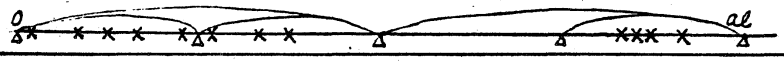
{ $Ax = \lambda x$, A : real symmetric positive def. band matrix の
$0 < \lambda_i < al$ なる λ_i と v_i $(i=1,\cdots,nc)$ を所望の精度 eps で求める }

al , eps , epso , llm　　を与える.

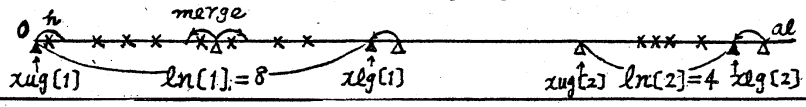
Call SSTRM (al , nc) { al を与えて $A - al*I = U^T D U$ の D の要素 d_i の負のものの個数 nc を求める } (1)
(al : input , nc : output)

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 使用の二分法によっ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]　　{ λ_{k-1} λ_k λ_{kp1} ; xug[ig] xlg[ig] }

{ Subspace Iter. (or Lanczos w. Select. Orth.)}
get $\bar{\lambda}_i$; $\bar{v}_i$; difi0 = $\|A\bar{v}_i - \bar{\lambda}_i \bar{v}_i\|/al$ $(i = k \text{ to } kp1)$ (4)

do i = k, kp1
　dif = difi0 ; ll = 0 ; $\lambda = \bar{\lambda}_i$; $v = \bar{v}_i$
　while (dif > eps) ∧ (ll ≤ 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 = $\|Av - \lambda v\|/al$; 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。)
↑4

(2) BISECT : 通常の Bisection 法と比べるとき、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、このまま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.）

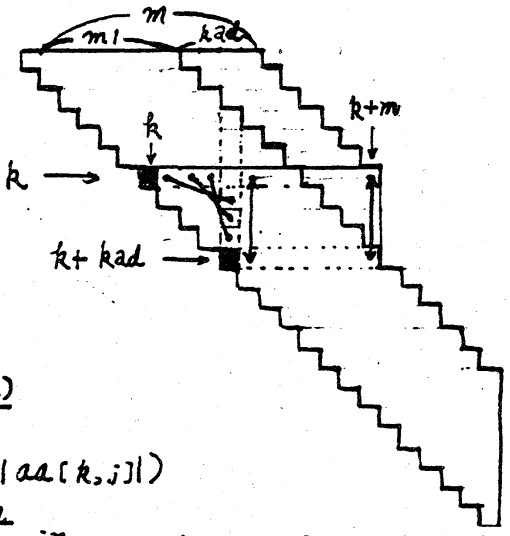
(3) 合理的なグループ分け : この部分が、スツルム・同時逆反復法の核心部である。アルゴリズムの大筋を次々頁に示す。re-grouping by SSTRM と呼ぶ。

（図例は後述の問題A1, epsbi=0.4のもの）

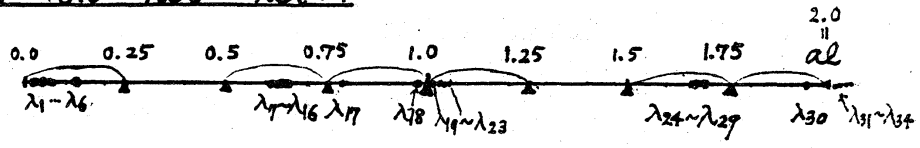
10 continue {repeat} の部分が一般のグループに関するところである。その前の部分は左端のグループに対する特別処置,後の部分は、右端のグループに対する特別処置である。

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*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 ≠ k then
        aa[k,k] ⇄ aa[ipk, ipk]
        do  j = k+1, ipk-1
            aa[k,j] ⇄ aa[j,ipk]
        do  j = ipk+1, min(k+m,n)
            aa[k,j] ⇄ aa[ipk,j]
    anorm = max(anorm, Σ_{j=k}^{jmax} |aa[k,j]|)
    if amax < epsir * anorm then
        ir = ir+1 ; α = α + epsir*100^ir ; go to 500 {A-αI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
```



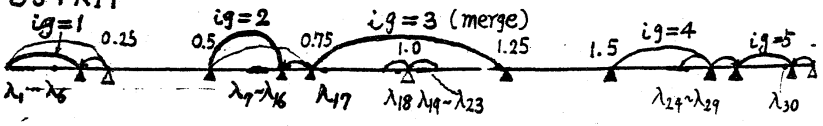
BISECT




```
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 ≤ 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
        {ここまでは、左端グループに対する特別処置}
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置}
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）

Rutischauser (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 Rutischauser 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^TDU 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
  α = 0.6*xug[ig] + 0.4*xlg[ig]
  set initial vectors v_k, v_{k+1}, ..., v_{kp1}   ( v_i ⊥ v_j , j = k - ln[ig-1] to kp1 , j≠i )
  "   X = { v_k, ..., v_{kp1} }
  "   lp = max(ln[ig]/2 + 1, 2)
  lq = 0
199 continue {repeat}
      lq = lq+1
      do l = 1, lp
        solve X̄ : (A-αI) X̄ = X
        H = X̄^T X̄
        solve P : HP = PΔ      ( Eigen Solution )
        form X = X̄ P
        normalize X
      do i = k, kp1
        λ̄_i = v_i^T A v_i
        difi0 = ‖ A v_i - λ̄_i v_i ‖ / al
        if (lq < lqm) ∧ (difi0 > epso) ∧ (i ≠ kp1) go to 199
```

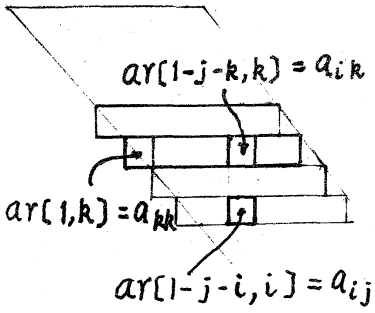
ig	1	2	3	4	5	6	7	8	9	10
lp	2	2	2	3	3	4	4	5	5	6

(5) レーリ商シフトつき逆反復： 殆んどの λ_i, 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 ($b=\bar{v}_i$, $\alpha=\bar{\lambda}_i$ を与えて $(A-\alpha M)v_i=\bar{v}_i$ なる 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 流の選択的再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点 α をえらんでの $(A-\alpha*I)^{-1}$ 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・ランチョスの潜然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、その点は今回の比較では不問に付す。

2)　文献(2)執筆の時点では、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点があったためである．今回はその点を改良して、データをとり直した．

LANSSO

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

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

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

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

    compute $s_i = (s_{1i}, \cdots, s_{ji})^T$ { eigenvector of μ_{ji} }
    $b_{ji} = |tb[j]| * |s_{ji}|$ { $s_i = (s_{1i}, \cdots, s_{ji})^T$ }
    $epsqn = epsq * norm$
    if $b_{ji} < epsqn$ then

      $y_i = Q_j s_i$ { $Q_j = \{q_1, \cdots, q_j\}$ }
      $\lambda_i = alfa + \mu_{ji}$
      $\mathcal{X} = \mathcal{X} - (\mathcal{X}^T y_i) y_i$ { 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 $(count > ln[ig]) \wedge ((difom < epso) \cup (j > jmax0))$ go to 400
if count < ln[ig] ifsok = ln[ig] - count
write ig, 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二種の二次元場の、$div(-k(x)\nabla u)$ を有限要素法で離散化した行列をテスト行列とした。

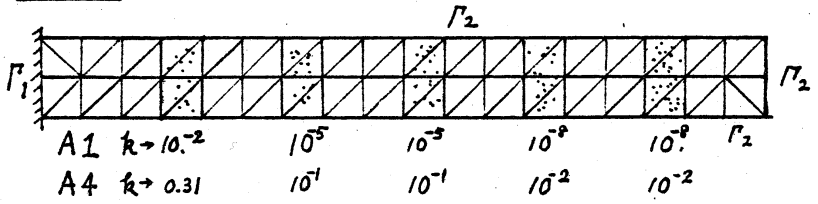
A1とC1 : $k(x)$ 値の変化が大きく、甚だしい密集固有値をもつ問題。ランチョスにとって苦手な問題である。

(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。)

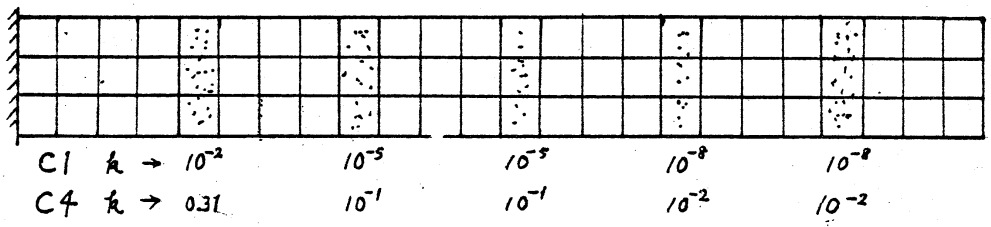
A4とC4 : $k(x)$値の変化が小さく、甚だしい密集固有値は無い問題。 ランチョスにとって得意な問題である。

(境界条件: Γ_1 : $u=0$, Γ_2 : $\nabla u=0$)

Prob.A



Prob.C



eps値は、10^{-9}, 10^{-14} のニケース, epsbi値は 0.4と0.2のニ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 eps $=10^{-9}$ al $= 2.0$ (nc $= 30$) LQM=5 RL : レーリ商逆反復の略 L=4

epsbi	Common		SUBSPACE				TOTAL		LANSSO				TOTAL	
	SSTRM	LU	Iter.	$4P_i\nu_i/m$	$P_i\nu_i/m$	RL	CPU	USE	Iter.	$4\nu_i/m$	ν_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 eps $=10^{-9}$ al $= 1.5$ (nc $= 30$) LQM=5

0.4	9	4	60	12	3	2	27	69	138	3	1	14	30	157
0.2	13	6	61	7	2	3	29	72	126	3	1	1	23	134

mean → 31 76 26.25 135.75

Prob A1 eps $=10^{-14}$ al $= 2.0$ (nc $= 30$) 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

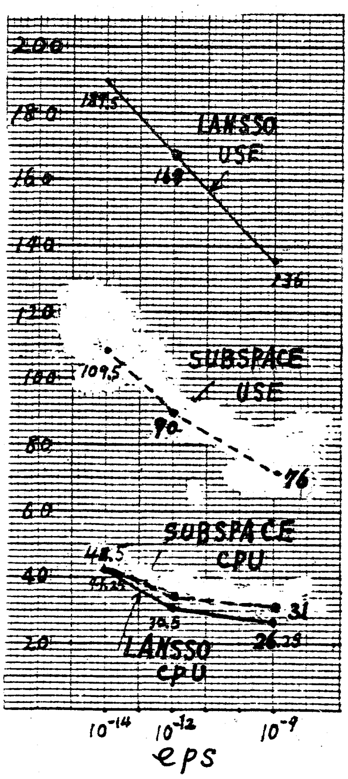
Prob C1 eps $=10^{-14}$ al $= 1.5$ (nc $= 30$) LQM=5

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 eps $=10^{-9}$ output mean → 42.5 109.5 41.25 189.5

```
   EPSO,LQM            DEL=       0.1000D-09    5                        0.200
          6  0.0                              0.49999999999999993E-01  0.250000
 1     4  0   0.00214687952685   0.00214687952685   0.5D-15   0.5D-15
 2     4  0   0.00000581674257   0.00000581674257   0.2D-14   0.2D-14
 3     4  0   0.00000084977085   0.00000084977085   0.3D-13   0.3D-13
 4     4  0   0.00000000581462   0.00000000581462   0.2D-13   0.2D-13
 5     4  0   0.00000000084758   0.00000000084758   0.1D-12   0.1D-12
 6     4  0   0.13457983425647   0.13457983425647   0.5D-10   0.5D-10
          0          0           0             6          12           72
         10  0.5000000000000000         0.59199999999999985         0.729999
 7     6  0   0.63397459755095   0.63397459755095   0.2D-11   0.2D-11
 8     6  0   0.63397459934207   0.63397459934207   0.1D-11   0.1D-11
 9     6  0   0.63397460438446   0.63397460438446   0.8D-12   0.8D-12
10     6  0   0.63397460491805   0.63397460491805   0.3D-11   0.3D-11
11     6  0   0.63397593459094   0.63397593459094   0.3D-11   0.3D-11
12     6  0   0.63397772606152   0.63397772606152   0.4D-12   0.4D-12
13     6  0   0.63398276779036   0.63398276779036   0.6D-12   0.6D-12
14     6  0   0.63398329929128   0.63398329929128   0.1D-11   0.1D-11
15     6  0   0.63715601382387   0.63715601382387   0.2D-11   0.2D-11
16     6  0   0.63780612130754   0.63780612130754   0.1D-11   0.1D-11
          6          0           0            10          18          180
          7  0.7500000000000000         0.85799999999999916         1.01999
17     4  0   0.76728046769175   0.76728046769175   0.1D-11   0.1D-11
18     4  0   0.98365489920253   0.98365489920253   0.5D-13   0.5D-13
19     4  0   1.00000000396426   1.00000000396426   0.2D-11   0.2D-11
20     4  0   1.00000001103431   1.00000001103431   0.1D-11   0.1D-11
21     4  0   1.00000396859763   1.00000396859763   0.1D-11   0.1D-11
22     4  0   1.00001103528705   1.00001103528705   0.7D-12   0.7D-12
23     4  0   1.00509312702005   1.00509312702005   0.3D-11   0.3D-11
         10          0           0             7          16          112
          6  1.5000000000000000         1.59199999999999986         1.72999
24     4  0   1.65714795272553   1.65714795272553   0.1D-12   0.1D-12
25     4  0   1.69722436754223   1.69722436754223   0.4D-11   0.4D-11
26     4  0   1.69722437466897   1.69722437466897   0.2D-10   0.2D-10
27     4  0   1.69722964174447   1.69722964174447   0.2D-10   0.2D-10
28     4  0   1.69723676343086   1.69723676343086   0.2D-10   0.2D-10
29     4  0   1.70294056116073   1.70294056116073   0.2D-09   0.2D-09
          7          0           0             6          16           96
          1  1.7500000000000000         1.84199999999999964         1.97999
30     2  1   1.94104742546787   1.94104742395532   0.9D-05   0.2D-09
          6          0           0             1          10           10
   BISECT    GINVRY SUBSPACE         LF
          8          6        5       1
                             72
```

14



*RL : レーリー商逆反復の略号

Prob. A4 $eps = 10^{-9}$, $al = 1.5$ ($nc = 23$) (LQM=6)

epsbi	Common		SUBSPACE				TOTAL		LANSSO (L=4)				TOTAL	
	SSTRM	LU	Iter.	$4P:\nu_i/m$	$P:\nu_i/m$	RL*	CPU	USE	Iter.	$4\nu_i/m$	$-\nu_i/m$	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 $eps = 10^{-9}$, $al = 1.5$ ($nc = 29$)

epsbi	SSTRM	LU	Iter.	$4P:\nu_i/m$	$P:\nu_i/m$	RL*	CPU	USE	Iter.	$4\nu_i/m$	$-\nu_i/m$	RL	CPU	USE
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 →							38.75	104.25					20.75	105

Prob. A4 $eps = 10^{-14}$ $al = 1.5$ ($nc = 23$) (LQM=6)

epsbi	SSTRM	LU	Iter.	$4P:\nu_i/m$	$P:\nu_i/m$	RL*	CPU	USE	Iter.	$4\nu_i/m$	$-\nu_i/m$	RL	CPU	USE
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

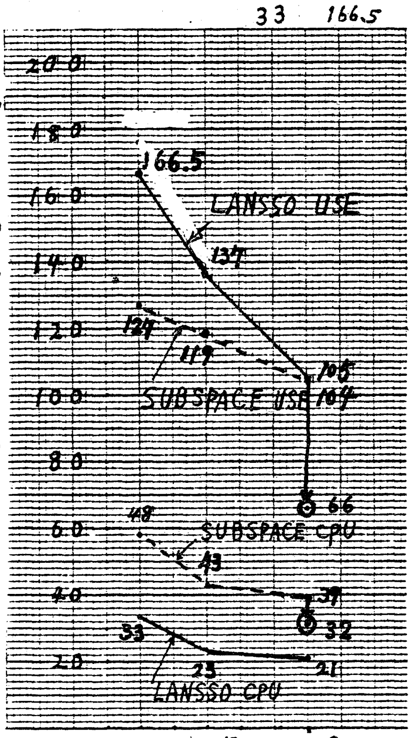
Prob. C4 $eps = 10^{-14}$ $al = 1.5$ ($nc = 29$)

epsbi	SSTRM	LU	Iter.	$4P:\nu_i/m$	$P:\nu_i/m$	RL*	CPU	USE	Iter.	$4\nu_i/m$	$-\nu_i/m$	RL	CPU	USE
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
							47.75	127.25					33	166.5

A4 epsbi = 0.2 $eps = 10^{-12}$ (10^{-14} でも同じ)

out put

```
   EPSO,LQM                  =        0.1000D-12    6
          5  0.0                          0.67173913043478250 5E-01  0.1679347
  1   3   0   0.06958233613252   0.06958233613252   0.2D-15   0.2D-15
  2   3   0   0.03604532610513   0.03604532610513   0.3D-15   0.3D-15
  3   3   1   0.00732362476562   0.00732362476562   0.4D-09   0.4D-15
  4   3   1   0.00450670238372   0.00450670238372   0.4D-08   0.3D-15
  5   3   1   0.00066163028734   0.00066163028734   0.3D-07   0.3D-15
          0          0           0           5          18          90
          1  0.18750000000000000     0.26249999999999997        0.3750000
  6   2   0   0.24135371222785   0.24135371222785   0.2D-14   0.2D-14
          5          0           0           1           8           8
          8  0.56250000000000000     0.62967391304347823 7       0.7304347
  7   5   0   0.63517812091124   0.63517812091124   0.2D-15   0.2D-15
  8   5   0   0.63703731155864   0.63703731155864   0.2D-14   0.2D-14
  9   5   0   0.64151514789898   0.64151514789898   0.1D-15   0.1D-15
 10   5   0   0.64245992387869   0.64245992387869   0.1D-14   0.1D-14
 11   5   0   0.64887717635201   0.64887717635201   0.2D-15   0.2D-15
 12   5   0   0.66502564282909   0.66502564282909   0.7D-15   0.7D-15
 13   5   1   0.70710818723366   0.70710818723365   0.2D-07   0.4D-15
 14   5   1   0.70844914085996   0.70844914085996   0.3D-08   0.2D-15
          1          0           0           8          30         240
          2  0.75000000000000000     0.81717391304347823 7       0.9179347
 15   2   1   0.76823559635370   0.76823559635370   0.1D-09   0.3D-15
 16   2   1   0.76189722384915   0.76189722384893   0.2D-06   0.6D-15
          8          0           0           2          12          24
          6  0.93750000000000000     1.01249999999999996        1.125000
 17   4   0   1.01097989344215   1.01097989344215   0.1D-14   0.1D-14
 18   4   0   1.00398808190031   1.00398808190031   0.1D-14   0.1D-14
 19   4   0   1.04756803514056   1.04756803514056   0.4D-14   0.4D-14
 20   4   0   0.96028138099873   0.96028138099873   0.6D-14   0.6D-14
 21   4   0   1.08512399123548   1.08512399123548   0.2D-14   0.2D-14
 22   4   1   1.11444785532518   1.11444785532518   0.2D-10   0.8D-15
          2          0           0           6          24         144
          1  1.14456521739130435     1.27891304347826074        1.480434
 23   2   0   1.25326002723534   1.25326002723534   0.8D-15   0.8D-15
          6          0           0           1           8           8
   BISECT     GINVRY SUBSPACE        LF
          8          7         6     8
             15        100
```



10^{-14} 10^{-12} 10^{-9}

eps

⊙ EP : チェビシェフ加速付加

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、 $4p\cdot\nu/m$ 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、$m \geqslant 80$ 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：$p = ln[ig]$, $\nu = lq * lp$ のとき，

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

$2mnp\nu/(\frac{1}{2}m^2n) = 4p\nu/m$ 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)	$np\nu$	$\sim p^2\nu$	$np\nu$

$\{2(m+p)n\nu + np\nu + np\nu\}/2mn = \nu + (2p\nu/m)$ 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 $4P_i\nu_i/m$ 項の方が LANSSO の $4\nu_i/m$ 項よりも大中に増す等の事情があらわれて、LANSSO の方が SUBSPAC とくらべ、ここに示した比較より若干有利の方向に変る．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’をして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

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

を離散固有値問題 $A\upsilon = \lambda\upsilon$ 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 $A^c\upsilon^c = \lambda^c\upsilon^c$ の al 以下の固有値 λ_i^c，$0 < \lambda_i^c < al$；$i = 1, \cdots, nc$ 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 $A^f\upsilon^f = \lambda^f\upsilon^f$ 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 $i = ig1, \cdots, igm$ につき初期ベクトル υ_i^{f0} を、υ_i^c からの interpolation によって作る．次いで λ_i^{f0} をレーリ商：

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

によって作り、この λ_i^{f0}, $i = ig1, \cdots, igm$ 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：

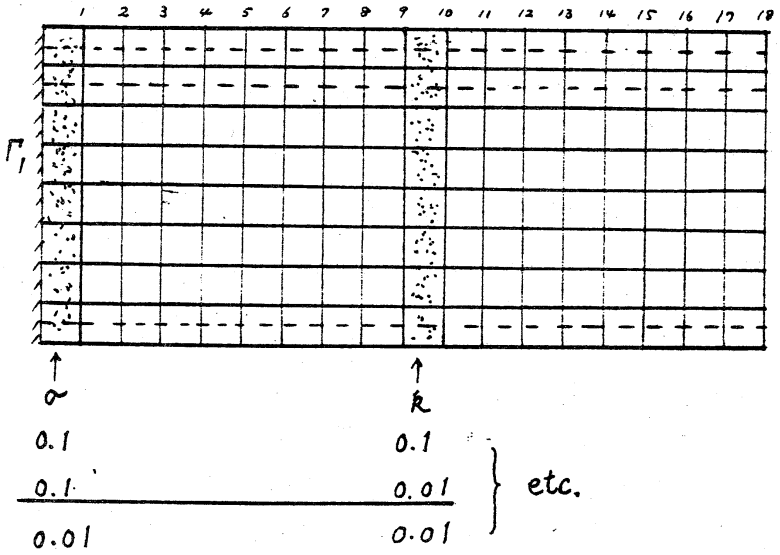
$$(A^f - \alpha_g I)\, X^{(\nu+1)} = \overline{X}^{(\nu)}$$

$$\vdots$$

を行なう．（f は fine の f）

- 最後に、收束し損ねたものを．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 ： $div(-k(x)\nabla u)$ より生成される行列.



粗系は $9\times18 = 162$ 元

精系は $17\times18 = 306$ 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 $0<\lambda<al$ の al を 0.5 前後にとってやるときわどいことが起り易くて面白いわけである。そう考えて al 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.

$\sigma = k = 0.01$, $al = 0.5$ のとき、

$nc = 24$, 従って標準の epsbi は、

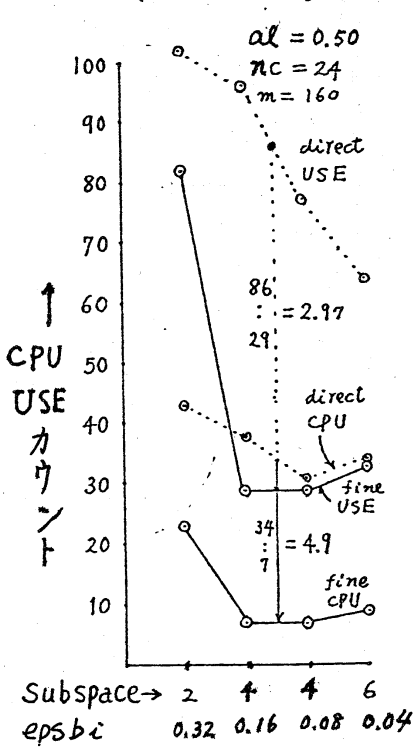
$$(0.5/24) * 5 \doteqdot 0.1$$

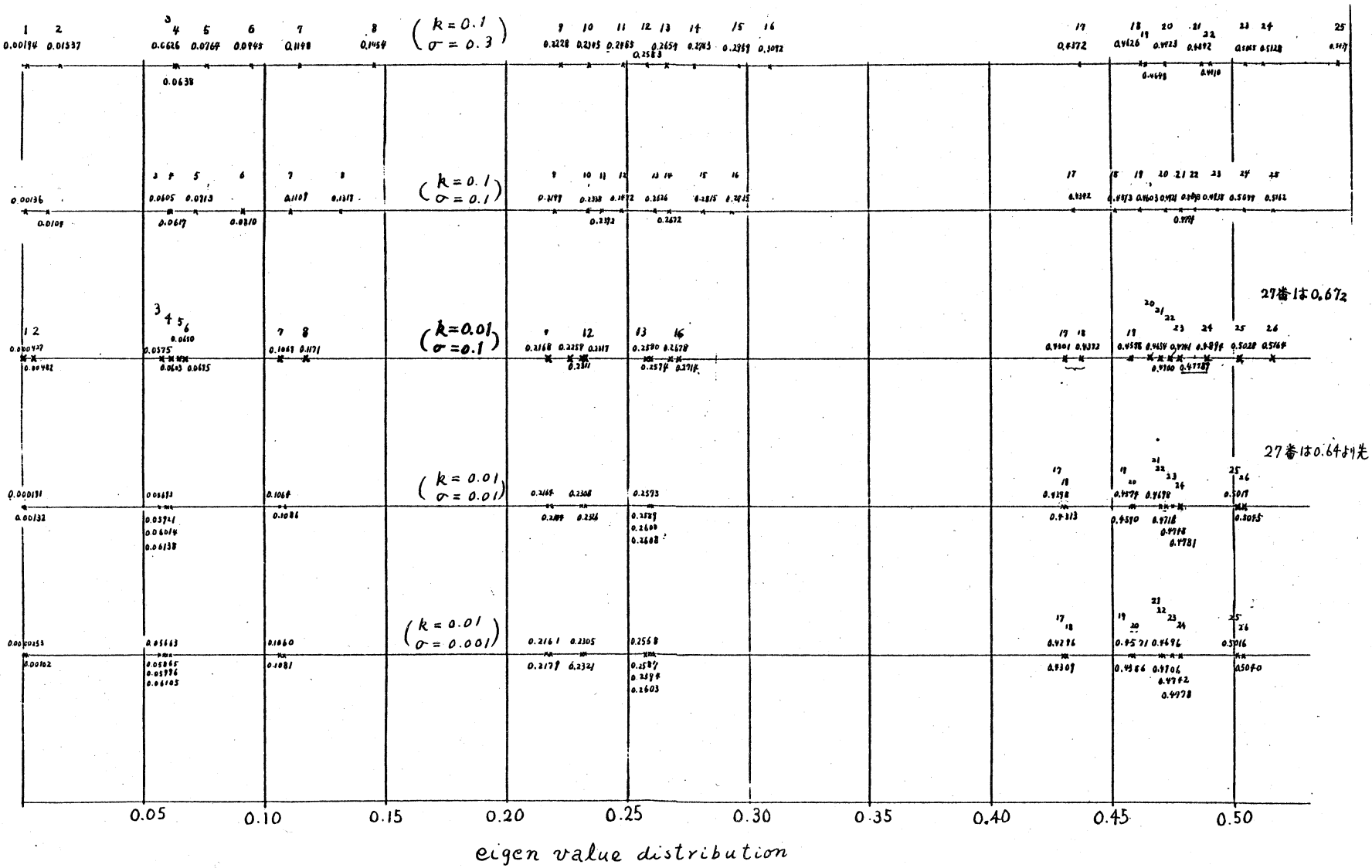
であるが、epsbi を外から与えて

epsbi = 0.32 , 0.16 , 0.08 , 0.04

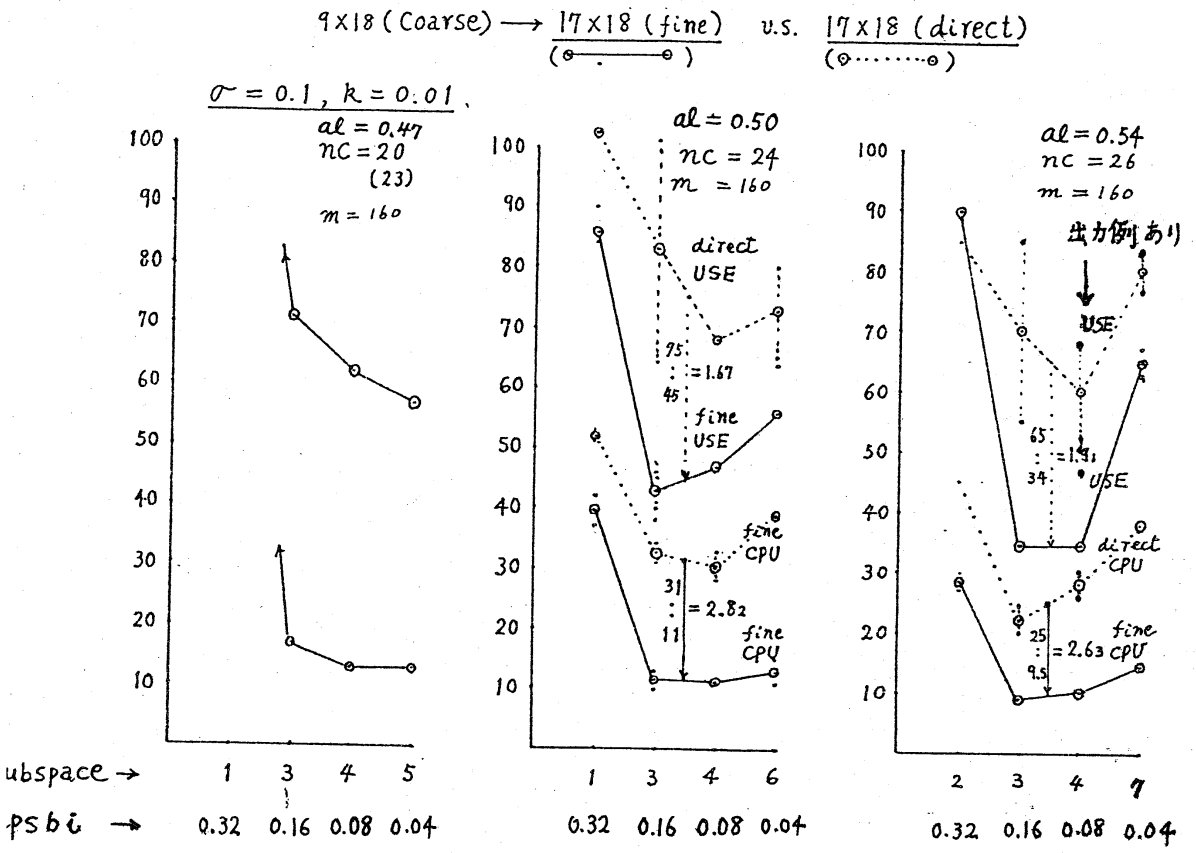
の4ケースについてデータを取った.

(右図) 点線が直接に解いたもの, 実線が二段階メッシュ法によるものである.




(k = 0.1, σ = 0.3)
(k = 0.1, σ = 0.1)
(k = 0.01, σ = 0.1)
(k = 0.01, σ = 0.01)
(k = 0.01, σ = 0.001)
27番は0.672
27番は0.64より先
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
eigen value distribution

$\sigma = 0.1$, $\sigma = 0.01$ 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、つじつまが合う。



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 及至 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をちょっと動かしたとき、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・ランチョスでも同様に起る。) 図の⋮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、$\lambda_{20} = 0.46546\cdots$, $\lambda_{21} = 0.47003\cdots$, $\lambda_{22} = 0.47412\cdots$, $\lambda_{23} = 0.47789\cdots$ 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、$\lambda_{26} = 0.5164\cdots$ 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段階メッシュ法によるものである.

Prob. H 17X18 (direct) al = 0.54, epsbi = 0.08.

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

```
   EPS0.LQM.                 =       0.1000D-06     5
        8  0.0                          0.5399999999999999968E-01  0.1349999999999999995
 1    5    0    0.05750250288748    0.05750250288748    0.2D-14    0.2D-14
 2    5    0    0.06039854049116    0.06039854049116    0.4D-14    0.4D-14
 3    5    0    0.06509907101929    0.06509907101929    0.3D-14    0.3D-14
 4    5    0    0.06751919909665    0.06751919909665    0.2D-14    0.2D-14
 5    5    0    0.00482384434455    0.00482384434455    0.2D-10    0.2D-10
 6    5    0    0.10695153548022    0.10695153548022    0.6D-09    0.6D-09
 7    5    0    0.00042725006469    0.00042725006469    0.3D-08    0.3D-08
 8    5    0    0.11717878372461    0.11717878372461    0.5D-07    0.5D-07
        0           0                0            8          20           160
        8  0.2024999999999999986        0.2315769230769230 55        0.27519230769230 7681
 9    5    0    0.23115836094885    0.23115836094885    0.1D-10    0.1D-10
10    5    0    0.22590909481745    0.22590909481745    0.4D-09    0.4D-09
11    5    0    0.23783240353611    0.23783240353611    0.5D-10    0.5D-10
12    5    0    0.21688821269918    0.21688821269918    0.6D-08    0.6D-08
13    5    0    0.25809199696860    0.25809199696860    0.6D-10    0.6D-10
14    5    0    0.25942077947446    0.25942077947446    0.1D-10    0.1D-10
15    5    0    0.26788081909681    0.26788081909681    0.1D-09    0.1D-09
16    5    0    0.27140637233400    0.27140637233400    0.9D-08    0.9D-08
        8           0                0            8          10            80
       10  0.4049999999999999985        0.4548461538461538 03        0.5296153846153845 99
17    6    0    0.45780835733505    0.45780835733505    0.5D-11    0.5D-11
18    6    0    0.46546082127325    0.46546082127325    0.4D-10    0.4D-10
19    6    0    0.47003307736395    0.47003307736395    0.4D-09    0.4D-09
20    6    0    0.43724449666209    0.43724449666209    0.2D-09    0.2D-09
21    6    0    0.47412804463708    0.47412804463708    0.2D-09    0.2D-09
22    6    0    0.47789865784922    0.47789865784922    0.5D-09    0.5D-09
23    6    0    0.43019819409561    0.43019819409561    0.7D-09    0.7D-09
24    6    0    0.48940703264051    0.48940703264051    0.3D-09    0.3D-09
25    6    0    0.50280573290504    0.50280573290504    0.2D-08    0.2D-08
26    6    0    0.51648349038081    0.51648349038081    0.8D-08    0.8D-08
        8           0                0           10          12           120

BISECT    GINVRY SUBSPACE          LF       TOTALF    TOTALC        360          9          4
     7         8        3           0           18        27
                  42                           45        49
```

Prob. H 9X18 (coarse) → 17X18 (fine)

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

```
   EPS0.LQM                  =       0.1000D-06     5
        8  0.4272565807102407 89E-03  0.4746548583288418 69E-01  0.1180228297111451 11
 1    5    0    0.05750250288748    0.05750250288748    0.5D-10    0.5D-10
 2    5    0    0.06039854049116    0.06039854049116    0.1D-10    0.1D-10
 3    5    0    0.06509907101929    0.06509907101929    0.9D-10    0.9D-10
 4    5    0    0.06751919909665    0.06751919909665    0.3D-08    0.3D-08
 5    5    0    0.00482384434456    0.00482384434456    0.5D-07    0.5D-07
 6    5    0    0.00042725006469    0.00042725006469    0.1D-07    0.1D-07
 7    5    0    0.10695153548024    0.10695153548024    0.7D-07    0.7D-07
 8    5    0    0.11717878372503    0.11717878372503    0.3D-06    0.3D-06
        0           0                0            8           5            40
        8  0.2237317073332962 95        0.2443779679685105 09        0.2753473589213318 58
 9    5    0    0.23783240353611    0.23783240353611    0.9D-14    0.9D-14
10    5    0    0.23115836094885    0.23115836094885    0.3D-13    0.3D-13
11    5    0    0.25809199696860    0.25809199696860    0.3D-13    0.3D-13
12    5    0    0.25942077947446    0.25942077947446    0.5D-14    0.5D-14
13    5    0    0.22590909481745    0.22590909481745    0.1D-13    0.1D-13
14    5    0    0.26788081909681    0.26788081909681    0.6D-12    0.6D-12
15    5    0    0.27140637233400    0.27140637233400    0.4D-11    0.4D-11
16    5    0    0.21688821269918    0.21688821269918    0.1D-10    0.1D-10
        0           0                0            8          10            80
       10  0.4295768056554402 36        0.4706306014378870 56        0.5322112951115573 48
17    6    0    0.47003307736396    0.47003307736396    0.1D-13    0.1D-13
18    6    0    0.47412804463710    0.47412804463710    0.1D-13    0.1D-13
19    6    0    0.46546082127325    0.46546082127325    0.1D-13    0.1D-13
20    6    0    0.47789865784924    0.47789865784924    0.2D-13    0.2D-13
21    6    0    0.45780835733505    0.45780835733505    0.1D-13    0.1D-13
22    6    0    0.48940703264051    0.48940703264051    0.1D-13    0.1D-13
23    6    0    0.50280573290505    0.50280573290505    0.3D-13    0.3D-13
24    6    0    0.43724449666209    0.43724449666209    0.2D-13    0.2D-13
25    6    0    0.43019819409561    0.43019819409561    0.3D-12    0.3D-12
26    6    0    0.51648349038081    0.51648349038081    0.5D-12    0.5D-12
        7           0                0           10          12           120

BISECT    GINVRY SUBSPACE          LF       TOTALF    TOTALC        240          6          3
     0         0        3           0            3         9
                  27                           30        33
```

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．帯半巾 160 で、

CPU カウント： 1固有値当り

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

$eps = 10^{-9}$ 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 カウントは兎も角として、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・同時逆反復法の一応用として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 ～ 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、合理的なシフト実の較正手法などによって同時逆反復 $(A-\alpha I)X^{(\nu)}=X^{(\nu-1)}$ 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' 仝上 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 Rutischauser : 'Simultanious Iteration Methods' (Contribution II/8)

(6) Meijerink & Van der Vorst, Math. Comp. 31 (1977) 148p－162p, 仝上 J. Comptational 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