実対称行列の正定値性判定プログラム

Numerical Verification of Positive-Definiteness for Real Symmetric Matrices

渡部 善隆[†]　　山本 野人[‡]　　中尾 充宏[‡]

実対称行列の正定値性判定プログラム vpdsm を VPP700/56 上でテスト公開します．vpdsm による計算結果は，区間演算 (cf.4章) による丸め誤差の考慮を行なっているため，正定値性が判定された場合，行列が《数学的に厳密な意味で》正定値であることを保証します．また，正定値であることが判定された行列については，最小固有値の下限もあわせて出力します．

本プログラムは，平成 6 年度後期〜平成 9 年度前期のプログラムライブラリ開発課題『精度保証付き数値計算システム』によって得られた成果の一部です．

1　プログラムの概要

この章では，vpdsm の概要を説明します．なお，対象とする行列はすべて正方行列です．

1.1　実対称行列の正定値性

科学の多くの分野において**実対称行列** (*real symmetric matrix*) は頻繁に登場します[1]．そして，それらの行列が**正定値** (*positive definite*) であるならば，理論面 (例えば反復法の収束証明の仮定が満たされる) および実用面 (例えば計算量の少ない行列の分解法を選択できる) 双方でよいことがたくさんあります (例えば [1]，[2]，[6]，[15] を参照)．

正定値の定義

$n \times n$ の実対称行列 A と $x \neq 0$ である任意の $n \times 1$ 実ベクトル x に対し

$$x^T A x > 0$$

となるとき，A は正定値であるといいます．また，行列 A は**正定値性** (*positive-definiteness*) を持つといいます．

定義中の “T” は転置記号です．

実対称行列の正定値[2]性は，すべての実固有値が正であることと同値です (cf.7.1節)．従って，$n \times n$ の実対称行列が正定値であるかどうか判定するには，その n 個の固有値を調べればよいことになります．

ところが，与えられた行列の固有値を定義通り紙と鉛筆 (手計算) で求めようとすると，n 次の代数方程式を解く問題に帰着します．しかし，5 次以上の代数方程式に対する根の公式が存在しないことは，有名な「ガロアの理論」によって証明されています (例えば [9] を参照)．

つまり，仮に無限桁を保証する計算機や有理数演算を用いたとしても，(5 次以上の) 固有値問題を一定回数の操作で完全に解くことは不可能であり[3]，どうしてもある収束条件を課した反復法を計算の一部に取り入れる必要があります．

従って，数値計算によって固有値を評価し，実対称行列の正定値性を判定するためには，丸め誤差の見積りに加えて反復法の打ち切り誤差も考慮する必要があります．

行列の正定値性の判定法については，これまで幾つかの手法が提案されています (例えば [7], [8], [14])．しかし，それらの手法は，限定された行列に対して適用可能であったり，置換行列[4]の存在や反復法の収束を判定条件とするなど，必ずしも汎用性と厳密性を持つアルゴリズムとは言えません．

1.2　正定値性判定プログラム

vpdsm は，利用者が用意した実対称行列データ，または対称行列を含む区間データ読み込み，その正定値性を判定するプログラムです．

具体的には，Householder-bisection 法 (アルゴリズムは例えば [21] を参照) により得られた行列の近似最小固有値の近傍を調べ，最小固有値が真に 0 より大きいかどうかを評価します．また，正定値であることが保証された行列に対しては，最小固有値の下限[5]を出力します．

[†]九州大学大型計算機センター
e-mail:watanabe@cc.kyushu-u.ac.jp
[‡]九州大学大学院数理学研究科
e-mail:{yamamoto, mtnakao}@math.kyushu-u.ac.jp

[1]例えば，自然現象を数学の言葉で記述した支配方程式の “離散化” と “線形化” の結果としてよく得られます．実対称行列とは，要素が実数であり，転置行列と一致するような行列のことです．

[2]「正定値」は，「正値」または「正定符合」と呼ばれることもあります．

[3]数式処理ソフトの *Mathematica* や Maple V は 4×4 以下の行列に対しては根の公式を用いた計算を行ないます．しかし次数が 5 次以上になると，よほど “型にはまった” 行列以外は近似計算によって固有値を計算します．近似計算アルゴリズムに何を用いているのかはマニュアルには書いてありません (ここが数式処理ソフトの不気味なところです)．

[4]行列の特定の行または列を入れ換えるために左右から引っかける行列のこと．1 がどの行もどの列もちょうど 1 個あり，他はすべて 0 の行列です．

[5]これ以下には絶対にないという値．

さらに利用者は，パラメータを自身で調節することにより，最小固有値の下限をより "シャープ" にすることも可能です．

vpdsm の特徴は，計算機の浮動小数点演算において発生する丸め誤差を完全に取り込みながら進むため，もし行列の最小固有値が正であるという結果を得た場合，行列の正定値性が数学的に厳密な意味で保証されるところにあります[6]．

vpdsm のアルゴリズムは 5章で紹介します．

2　利用方法

この章では，vpdsm の利用方法を説明します．

2.1　動作環境

vpdsm (/usr/local/bin/vpdsm) は，VPP700/56 (ホスト名 kyu-vpp，IP アドレス 133.5.9.70) の 1PE[7]で動作します．正定値判定の部分は Fortran 90 で作成された実行可能形式ファイルです．

利用者は，5 種類の形式のどれかを選択し，対応する行列データを用意します．データは kyu-vpp のエディタである emacs または vi で作成するか，ftp(/bin/ftp) などで kyu-vpp に転送します．

また，汎用計算機の UNIX システム (kyu-cc) からの vpdsm の利用はできません．

2.2　入力形式

vpdsm の入力形式は

vpdsm␣*size*␣*type*␣*delta*␣*matrix-file*

です．それぞれ空白で区切ります．順序の入れ換えはできません．

各パラメータ，ファイル名の内容は次の通りです．

size	行列の次数 (整数)
type	行列の種類 (1 〜 5 の整数)
	1 ··· 整数または実数データ
	2 ··· 区間データ
	3 ··· 有理数データ
	4 ··· バイナリデータ
	5 ··· バイナリ区間データ
delta	パラメータ δ ($0 < \delta \ll 1$ の実数)
matrix-file	行列データを格納したファイル名

[6] 大げさに言えば，計算機による定理の証明です．謙虚に言えば，Householder-bisection 法の打ち切り誤差を評価することによる近似計算の信頼性の確認です．

[7] Processor Element の略．詳しくは [22] を参照．

size は行列のサイズ (次数) を指定します．例えば 10× 10 の場合は 10 です．整数以外の値 (例えば 10.0) を指定すると，読み込み時にエラーとなります．また，行列サイズと *matrix-file* のデータとの整合性がとれていない場合にもエラーとなるか誤った結果を返すため注意が必要です．

type は入力する行列の種類を整数で指定します．整数以外の値を指定すると読み込み時にエラーとなります．また，1 〜 5 以外の整数を指定した場合は，1 が指定されたものとして処理が続行されます．

delta は 5章で紹介する微小な値 δ を入力します．例えば，$\delta = 10^{-2}$ の場合，0.01，1.0E-2，1.0D-2 などと指定します．6章の数値実験で見る通り，δ の値は行列の性格 (条件数，サイズなど)，および，どれくらい最小固有値の下限を精度よく求めたいかに応じて適宜使い分けてください．

matrix-file は正定値性を判定する行列データの格納されたファイル名です．形式は *type* で指定した行列の種類に応じて異なります．入力データの形式は 2.5節で説明します．

2.3　対話型での利用

vpdsm のあと，一つ以上のスペースをおいてパラメータとファイル名を入力します．

以下は，次元数 10 の有理数データ，$\delta = 10^{-2}$，ファイル名が "matrix.data" の例です．

```
kyu-vpp% vpdsm 10 3 0.01 matrix.data ↵
```

処理結果は標準出力に書き出されます．結果をファイルに落すにはリダイレクション ">" を用います．例として，ファイル "out" に出力します．

```
kyu-vpp% vpdsm 10 3 0.01 matrix.data > out ↵
```

処理結果の見方は 2.6節を参照してください．

2.4　バッチ処理での利用

行列のサイズが大きい[8]場合はバッチ処理をお勧めします．

利用方法は，VPP700/56 で Fortran, C プログラム，Gaussian などをバッチ処理で実行する手順と同様です[9]．ここでは最低限の手順のみ紹介します．詳しくは [22], [17] を参照してください．

[8] 300 × 300 を超えると分単位の実行時間が必要になります．また，対話型処理の記憶領域の制限値を超えてしまった場合もバッチ処理となります．行列のサイズと計算時間については，3.6節で実測データを紹介します．

[9] ここでの「バッチ処理」とは，コマンドの最後に "&" をつける UNIX の「バックグラウンドでの実行」とは処理形態が異なることにご注意ください．"&" をつけた場合には対話型処理と同様の制限値，演算負担金が適用されます．

まず「バッチリクエスト」と呼ばれるシェルスクリプトを kyu-vpp のファイルとして作成します．例ではファイル名を “a.sh” とします．ファイル a.sh には，kyu-vpp のホームディレクトリから matrix.data のあるディレクトリへの移動 (cd コマンド) に続けて，対話型と同様の処理手順を記述します．

```
#                              ←― csh で記述
cd EXAMPLE                     ←― ディレクトリの移動
vpdsm 100 3 0.001 matrix.data
```

上は，次元数 100 の有理数データ，$\delta = 10^{-3}$，ファイルは EXAMPLE ディレクトリの matrix.data としたバッチリクエストの記述例です．

次に，qsub(/bin/qsub) コマンドでバッチリクエストを投入します．

```
kyu-vpp% qsub a.sh ↵
Request 41739.kyu-vpp submitted to queue: p1.
```

ジョブの処理状況は qps(/usr/local/bin/qps) コマンドで確認できます．

例では，処理結果はジョブを投入したディレクトリに “a.sh.o41739” および “a.sh.e41739” というファイル名で返却されます．返却されるファイルの名前は，バッチリクエストファイル名と qsub コマンドを入力した際に自動的に発行される 5 桁の識別番号で構成されます．

“o” がつくファイルに vpdsm の処理結果が，“e” がつくファイルにエラーが起きた場合のメッセージがそれぞれ出力されます．

2.5 入力データの形式

この節では，行列データの作成方法について説明します．

2.5.1 整数または実数データ (*type*=1)

type=1 の場合，整数または実数データを 10 進のテキスト形式で与えます．

行列を $A = \{A_{ij}\}$，サイズを n とするとき，データは空白を区切りとして

$$A_{11}\ \ A_{21} \ldots A_{n1}\ \ A_{12}\ \ A_{22} \ldots A_{n2} \ldots A_{1n}\ \ A_{1n} \ldots A_{nn}$$

の順番に すべての要素 を記述してください[10]．途中で適当に改行しても構いません．

また，vpdsm は行列データを IEEE 浮動小数点形式 64 ビットデータ[11]に変換して計算を行ないます．よって，指数部 11 ビットで表現できる範囲を超える行列は処理できません．また，仮数部 53 ビット (隠れビットを含む) を超える情報は切捨てられます．

[10]対称性の確認と対称行列を包含する区間行列を作成する理由から，すべての要素が必要となります．

[11]2 進です．VPP700/56 の Fortran コンパイラである Fortran 90/VP では REAL(KIND=8)(倍精度実数型) です．詳しくは [19] を参照してください．

◇ *Example 1*

4 × 4 の整数行列

$$\begin{pmatrix} 4 & 3 & 2 & 1 \\ 3 & 3 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

の行列データです．

```
4  3  2  1            ←― 整数データ
3  3  2  1
2  2  2  1
1  1  1  1
```

◇ *Example 2*

5 × 5 の実数行列

$$\begin{pmatrix} 0.81321 & -0.00013 & 0.00014 & 0.00011 & 0.00021 \\ -0.00013 & 0.93125 & 0.23567 & 0.41235 & 0.41632 \\ 0.00014 & 0.23567 & 0.18765 & 0.50632 & 0.30697 \\ 0.00011 & 0.41235 & 0.50632 & 0.27605 & 0.46322 \\ 0.00021 & 0.41632 & 0.30697 & 0.46322 & 0.41931 \end{pmatrix}$$

の行列データです．

```
 0.81321  -0.00013  0.00014  0.00011  0.00021
-0.00013   0.93125  0.23567  0.41235  0.41632
 0.00014   0.23567  0.18765  0.50632  0.30697
 0.00011   0.41235  0.50632  0.27605  0.46322
 0.00021   0.41632  0.30697  0.46322  0.41931
```

このようなデータを作成する場合は，必ず 3.2節を参照してください．

◇ *Example 3*

4 × 4 の実数行列の例です[12]．

```
0.2173913043478261     0.1739130434782609
4.347826086956522E-02 4.347826086956522E-02

0.1739130434782609     0.2173913043478261
4.347826086956522E-02 4.347826086956522E-02

4.347826086956522E-02 4.347826086956522E-02
0.1739130434782609     8.695652173913043E-02

4.347826086956522E-02 4.347826086956522E-02
8.695652173913043E-02 0.1739130434782609
```

このようなデータを作成する場合は，必ず 3.2節を参照してください．

[12]Fortran プログラムを用いて入力データを作成する簡単な方法として，倍精度または整数行列 A を定義した後に “WRITE(N,*) A” (N は適当な unit 番号) で行列を書き出すことが考えられます．

2.5.2 区間データ ($type$=2)

$type$=2 の場合，データは 10 進のテキスト形式で行列の各要素を区間 (cf.4章) として与えます．

すなわち，行列 A の各要素 A_{ij} は上限と下限を持つ区間 $[\underline{A}_{ij}, \overline{A}_{ij}]$ 内の実数として表現されます．

データは，行列の区間要素の下限，上限を続けて

$$\underline{A}_{11} \quad \overline{A}_{11} \ldots \underline{A}_{n1} \quad \overline{A}_{n1} \ldots \underline{A}_{1n} \quad \overline{A}_{1n} \ldots \underline{A}_{nn} \quad \overline{A}_{nn}$$

の順番にすべての要素を記述してください．区切りは空白です．上限と下限が一致する場合でも，同じ数値を記述してください．

要素に関する条件は $type$=1 と同じです．

◇ *Example 4*

4 × 4 の区間行列

$$\begin{pmatrix} [3.99, 4.01] & [2.99, 3.01] & [1.99, 2.01] & [0.98, 1.02] \\ [2.99, 3.01] & [3.00, 3.00] & [1.99, 2.01] & [0.99, 1.01] \\ [1.99, 2.01] & [1.99, 2.01] & [1.99, 2.01] & [0.99, 1.01] \\ [0.98, 1.02] & [0.99, 1.01] & [0.99, 1.01] & [0.99, 1.01] \end{pmatrix}$$

の行列データです．

```
3.99 4.01  2.99 3.01  1.99 2.01  0.98 1.02
2.99 3.01  3.00 3.00  1.99 2.01  0.99 1.01
1.99 2.01  1.99 2.01  1.99 2.01  0.99 1.01
0.98 1.02  0.99 1.01  0.99 1.01  0.99 1.01
```

行列の要素を区間で与える場合，`vpdsm` は区間行列に含まれる《すべての対称行列》に対する正定値性を判定します．

2.5.3 有理数データ ($type$=3)

$type$=3 の場合，行列を有理数で与えます．

個々の有理数データは空白をおかずに

符合	分子 (整数)	/	分母 (整数)

で与えます．行列の順序は $type$=1 と同じです．

以下は注意事項です．

- 一行に最大 255 文字まで記述できます．それ以上は改行してください．
- 有理数の区切りは空白です．
- 分母，分子は IEEE の 8 バイト整数型で記述できる範囲

  $-9223372036854775808 \sim 9223372036854775807$

  以内で記述してください．
- 整数の場合 “/” は必要ありません．
- 一つの要素を “234/45+1/2” などと記述することはできません．

◇ *Example 5*

4 × 4 の有理数行列

$$\begin{pmatrix} 1 & -\frac{1}{2} & \frac{1}{3} & -\frac{1}{4} \\ -\frac{1}{2} & \frac{2}{3} & -\frac{3}{4} & 1 \\ \frac{1}{3} & -\frac{3}{4} & \frac{11}{3} & -1 \\ -\frac{1}{4} & 1 & -1 & \frac{9}{4} \end{pmatrix}$$

の行列データです．

```
  1   -1/2    1/3  -1/4
-1/2   2/3   -3/4    1
 1/3  -3/4   11/3   -1
-1/4    1     -1    9/4
```

2.5.4 バイナリデータ ($type$=4)

$type$=4 の場合，行列データを Fortran 90/VP の書式なし write 文で出力した IEEE 浮動小数点形式の倍精度実数型データとして与えます．ファイルはバイナリファイルです．

書式なし write 文でのファイルの書き出しは，例えば

```
REAL(KIND=8),DIMENSION(10,10) :: A
                :
WRITE(1) A                    ! 書式なし write 文
```

とします (内容の説明は省略します)．

書式なし write 文での記録方法は処理系 (計算機，コンパイラ) 依存ですが，IEEE 浮動小数点形式をサポートしている処理系の多くに互換性があります[13]．

Fortran 90/VP の書式なし Fortran 記録についての詳細は [19] を参照してください．

2.5.5 バイナリ区間データ ($type$=5)

$type$=5 の場合，$type$=4 のバイナリデータを区間として与えます．データ並びは Fortran 90/VP の書式なし write 文で出力した IEEE 浮動小数点形式の倍精度複素数型データと同じ形式です．

また区間データは，Fortran 90 のモジュール機能 (例えば [4] を参照) を用いることで簡単に定義できます．

以下は Fortran 90/VP での区間行列の定義と出力例です．

[13] センターの wisdom，medics，および東北大学の SX-4/128H4 でデータの互換性を確認しました．kyu-cc，kyu-msp は IBM 形式 (M 形式) のため互換性がありません (変換可能ですが，情報が欠落します)．

```
MODULE IVL_DEF
      TYPE INTERVAL
         SEQUENCE
         DOUBLE PRECISION LOWER, UPPER
      END TYPE INTERVAL
END MODULE IVL_DEF

USE IVL_DEF
TYPE(INTERVAL),DIMENSION(10,10) :: A
               :
WRITE(1) A
```

2.6 処理結果の見方

2.6.1 正定値性が判定された場合

vpdsm によって正定値性が判定された場合にはメッセージと最小固有値の下限が出力されます．

```
*********************************************
*   GIVEN 4 TIMES 4 MATRIX IS
*   POSITIVE DEFINITE RIGOROUSLY.
*   LOWER BOUND OF MINIMUM EIGEN VALUE IS
*     9.573528097924996E-05
*********************************************
```

delta を更に小さくすることで，より最小固有値に近い下限を得ることができます．ただし，判定条件を満たさなくなる可能性も高くなります．

2.6.2 近似最小固有値が負になった場合

近似最小固有値が負になった場合，アルゴリズムの条件が満たされなくなります．従って vpdsm は判定を打ち切り，以下のメッセージを出力します．

```
APPROXIMATE MINIMUM EIGENVALUE = -0.4912715326
----------------------------------------------
APPROXIMATE MINIMUM EIGENVALUE IS NONPOSITIVE.
THE VERIFICATION ALGORITHM CAN NOT BE APPLIED.
**THIS MATRIX MIGHT BE NON-POSITIVE DEFINITE**
----------------------------------------------
```

2.6.3 近似 Cholesky 分解に失敗した場合

5章のアルゴリズムの近似的な Cholesky 分解に失敗した場合は，以下のメッセージを出力し処理を打ち切ります．

```
----------------------------------------------
APPROXIMATE MINIMUM EIGENVALUE = 9.6702234E-05
----------------------------------------------
APPLOXIMATE CHOLESKY DECOMPOSITION FAILED.
PLEASE SET MORE LARGER DELTA THAN  1.00000E-16
----------------------------------------------
```

この場合， *delta* の値を大きくし，再実行してみてください．

2.6.4 検証条件を満たさなかった場合

5章のアルゴリズムの最後の検証条件を満たさなかった場合は，以下のメッセージを出力します．

```
********************************************
*  NUMERICAL VERIFICATION FAILED.
*  PLEASE SET MORE LARGER DELTA THAN  1.0E-08
*  AND TRY AGAIN.
********************************************
```

この場合， *delta* の値を大きくするか，区間データを与えている場合は区間の幅が小さくなるかどうかを検討してください．また，行列の対称性を確認することもお勧めします．

2.6.5 エラーメッセージ

“jwe” で始まるメッセージが標準エラー出力[14]に書き出されている場合は， Fortran 90/VP が何らかのエラーを検出しています．

入出力に関するメッセージの場合，入力パラメータと行列データとの整合性がとれていないことが考えられます．

その他，指数アンダーフローやオーバーフローを起こして実行が打ち切られた旨のメッセージが書き出されている場合は，残念ながら vpdsm が扱える範囲を超えた行列である可能性が高くなります．

3 注意事項

この章では， vpdsm を使用する際の注意事項を述べます．

3.1 数学的な厳密性を得るための条件

行列の正定値性が vpdsm によって判定された場合，その結果が数学的に厳密であるためには，以下の条件が必要です．

- 整数または有理数データ
  結果は厳密に保証されます．
- バイナリ区間データ
  与えた区間行列に欲しい行列が真に含まれていれば，結果は厳密に保証されます．
- 区間データ
  テキスト形式で与える 10 進数データを計算機内で扱うことのできる 2 進数データに変換する時に変換誤差が生じます． vpdsm はこの変換誤差は考慮しません．従って，数学的に厳密であるためには，

[14]標準エラー出力をファイルに出力するには “>&” をつけ，例えば vpdsm 10 4 0.01 test.data >& out とします．

判定したい行列が 2 進数に変換された区間行列に真に含まれていることが条件です[15].

- バイナリデータ
  データが欲しい行列に《完全に》一致していることが条件です.
- 実数データ
  2 進数に変換したデータが欲しい行列に《完全に》一致していることが条件です.

3.2 行列データ自身の持つ誤差

3.1節の条件にあるように，特に行列データが実数の場合は注意が必要です.

例えば `1/3` を `0.3333` で表現することによって発生する誤差，あるいは行列を用意する計算の段階で累積する丸め誤差については利用者が責任を持って対処する必要があります.

従って，実際に厳密な評価を行ないたい場合は，有理数データとして行列を表示するか，区間データによって値を

$$\texttt{[0.333333333333332,0.333333333333334]} \ni \frac{1}{3}$$

などと表現することが必要です[16].

3.3 「正定値でない」ことの判定

`vpdsm` は与えられた行列が正定値であるかそうでないかを判定するプログラムではありません．もちろん，正定値性が判定できた場合，(3.1節の仮定のもと) 数学的に厳密な意味で正定値であることが保証されます．しかし，`vpdsm` によって正定値性が判定されなかった行列が「正定値でない」ことは保証しません[17].

3.4 入力した行列が対称でない場合

`vpdsm` は，入力データが $A_{ij} \neq A_{ji}$ となっている場合，A_{ij}, A_{ji} で表現される区間を再構成し処理を続行します.

例えば，$A_{ij} = 1$, $A_{ji} = 2$ の場合，その両方を含む区間 $A_{ij} = A_{ji} = [1, 2]$ に変換します.

入力が区間の場合も同様です．例えば，$A_{ij} = [-1, 1]$, $A_{ji} = [0, 2]$ の場合，区間 $A_{ij} = A_{ji} = [-1, 2]$ に変換します.

[15] Fortran 90 では 2 進数出力を例えば “`WRITE(6,'(B64)')`” などで簡単に出力できます．変換誤差はこの機能を用いることで確認できます.

[16] ここではわかりやすいように 10 進数で書いていますが，2 進数に変換した後の区間に `1/3` が含まれていることが必要です.

[17] Householder-bisection 法は信頼できる数値解法だといわれています．従って，浮動小数点演算による数値計算の結果，最小固有値が正であることがわかれば，その行列が正定値である可能性は極めて高いといえます．その意味で，`vpdsm` は近似計算の確実な保証を与えるプログラムだともいえます.

3.5 区間の幅

区間の幅が大きい場合は，行列のスペクトル半径 (固有値の絶対値最大) がよほど大きくない限り，正定値性は保証されません.

従って，区間行列でデータを与える場合，区間の幅はできるだけ小さくとるように心がけてください.

区間の幅と正定値性の関係については，[11] で詳しい理論的考察が与えられています.

3.6 演算時間

行列のサイズに応じた `vpdsm` の CPU 時間は次の通りです[18]

$n \times n$	CPU 時間
64×64	0.36 秒
128×128	2.67 秒
256×256	21.1 秒
512×512	212 秒
1024×1024	1,705 秒
2048×2048	10,870 秒
4096×4096	89,070 秒

すべて行列の正定値性の確認に成功したケースです．行列の読み込みに要する時間は含まれていません[19].

4 区間演算の概要

この章では，数値計算結果に含まれる丸め誤差の厳密な評価を得るための基礎となる**区間演算** (*interval arithmetic*) について簡単に説明します．詳細は [10], [13] を参照してください.

4.1 区間演算の定義

区間演算では，数値に含まれる誤差を考慮するため，実数 x を下限 $\underline{x}$ と上限 $\overline{x}$ による区間 (interval)

$$X = [\underline{x}, \overline{x}] = \{x \,|\, \underline{x} \le x \le \overline{x}\}$$

で表現します．さらに，すべての演算を区間に対する演算で置き換えます.

例えば，区間 $X = [\underline{x}, \overline{x}]$, $Y = [\underline{y}, \overline{y}]$ に対する四則演算の定義は次の通りです：

$$\begin{aligned}
X + Y &= [\underline{x} + \underline{y}, \overline{x} + \overline{y}], \\
X - Y &= [\underline{x} - \overline{y}, \overline{x} - \underline{y}], \\
X \cdot Y &= [\min\{\underline{xy}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{xy}\}, \max\{\underline{xy}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{xy}\}], \\
X / Y &= [\underline{x}, \overline{x}] \cdot [1/\overline{y}, 1/\underline{y}].
\end{aligned}$$

[18] 1997 年 11 月測定.

[19] 行列のサイズはバンクコンフリクト (cf.[20]) の起こる可能性のもっとも高い値を選んでいます．n をこの値から 1 ずらすと数 % 早くなります．また，行列の読み込み時間は全体の演算時間に比べてわずかのため，演算時間はどの行列タイプに対しても当てはまります.

ただし除算では $Y \not\ni 0$ とします．

このとき，実数 x は1点から成る区間 $[x, x]$ として考えます．また，区間の絶対値を $|X| \equiv \max\{|\underline{x}|, |\overline{x}|\}$ で，中点を $\mathrm{mid}(X) \equiv (\underline{x} + \overline{x})/2$ で定義します．

4.2 計算機上での実現方法

計算機で表現可能な区間は，その上下限が計算機で表現可能な浮動小数点数に限られます．

計算機上で表現可能な浮動小数点数全体を S として，

$$I(S) \equiv \{\, [a, b] \,|\, a, b \in S \text{ かつ } a \le b \},$$

また "$*$" を四則演算のどれかとします．

$X, Y \in I(S)$ に対して，一般に $X * Y \in I(S)$ とは限りません．しかし，$X * Y \subset Z$ となる $Z \in I(S)$ が存在する場合には，$X * Y$ の $I(S)$ への丸め:$R(X * Y) \in I(S)$ を定めることができます．実際には，$X * Y \subset Z$ となる最も区間幅の狭い Z を $R(X * Y)$ として定めます．

4.3 区間演算ライブラリ

浮動小数点の形式に応じ，四則演算結果の最大誤差が決まります．従って，この最大誤差を用いることで $X * Y$ の $I(S)$ への丸めを定めることができます．

近年，これらの操作を組み込んだ区間演算ライブラリが幾つか公開されるようになりました．特に C++, Fortran 90 の演算子再定義 (operator overloading) 機能を用いるライブラリは，利用者の手持ちの計算機上での区間演算を比較的容易に実現します．

`vpdsm` は，R. Baker Kearfott が中心になって開発した "`INTLIB_90`"([5]) の区間演算に関するモジュールを用いて作成したプログラムです[20]．

区間演算ライブラリに関する詳細は WWW の URL:

`http://interval-computations.home.ml.org/`

を御覧下さい．

5 アルゴリズム

この章では，`vpdsm` のアルゴリズムと証明を述べます．なお，ほぼ同様の手法が [12] で提案されています．

5.1 アルゴリズム

要素を区間として持つ $n \times n$ 行列を X とします．アルゴリズムは，X に含まれるすべての実対称行列の正定値性を判定します．

[20]現在規格化が進行中の次期 (日本では次々期)Fortran 規格 "Fortran 2000" では，区間演算を規格に入れることが話し合われているそうです．もし規格化されると区間演算をより手軽に利用することができるでしょう．

アルゴリズムの 1. と 4. 以外の実際の計算は，区間演算によって丸め誤差を評価しながら行ないます．

また "$\mathrm{mid}(X), \mathrm{mid}(Y)$" は区間行列の各要素の中点[21]によって構成される行列，I は単位行列です．

Algorithm

1. $\mathrm{mid}(X)$ の近似最小固有値 ρ を求める．
2. ある微小な数 $0 < \delta \ll 1$ により $\tilde{\rho} \equiv (1-\delta)\rho$ とする．
3. $Y \equiv X - \tilde{\rho} I$ を計算．
4. $\mathrm{mid}(Y)$ を近似的に Cholesky 分解：$\mathrm{mid}(Y) \approx \tilde{C}\tilde{C}^T$ する．
5. $Z \equiv \tilde{C}\tilde{C}^T - Y$ を計算．
6. $\lambda \equiv \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |Z_{ij}|$ を計算．
7. $\tilde{\rho} - \lambda > 0$ ならば X に含まれるすべての実対称行列は正定値．かつ $\tilde{\rho} - \lambda$ は X の最小固有値の下限．

実際の計算における "$\tilde{\rho} - \lambda > 0$" の判定は，$\tilde{\rho}$ を含む区間の下限が λ を含む区間の上限より大きいことを見ます．

また，4. の近似 Cholesky 分解では，対角成分が 0 でないことを仮定します．

5.2 アルゴリズムの証明

X に含まれる任意の実対称行列を X_0 とおきます．以下，証明の演算は区間ではなく実数の意味で考えます．また λ はアルゴリズムの 6. で得られる区間の上限とします．

$x^T x = 1$ なる任意の n 次元ベクトル x に対し，

$$\tilde{C}\tilde{C}^T - (X_0 - \tilde{\rho} I)$$

は対称行列であることから，2次形式の絶対値

$$|x^T(\tilde{C}\tilde{C}^T - (X_0 - \tilde{\rho} I))x|$$

は行列の2ノルム[22]即ちスペクトル半径で上から押えられます．

さらに，Gerschgorin の定理より，スペクトル半径の上界は行列の l_∞ ノルム[23]で与えられます (例えば [16] を参照)．ここで，$X_0 \in X$ であること，および区間演算による評価方法より，

$$|x^T(\tilde{C}\tilde{C}^T - (X_0 - \tilde{\rho} I))x| \le \lambda$$

[21]浮動小数点で構成される区間の中点が必ずしも浮動小数点で表現できるわけではないので，正しくは「中点の近似」です．実際の計算でもこの部分は近似で構いません．

[22]"スペクトルノルム" とも呼ばれます．
$||A||_2 \equiv \sup\limits_{x^T x = 1} ||Ax||$, ただし $||\cdot||$ はユークリッドノルム．

[23]$||A||_\infty \equiv \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |A_{ij}|$

が成立します[24]．よって，$\tilde{C}\tilde{C}^T$ が正定値であることより，

$$x^T(X_0 - \tilde{\rho}I)x \geq -\lambda$$

となります．従って $x^Tx = 1$ から

$$x^TX_0x \geq \tilde{\rho} - \lambda > 0,$$

即ち結論を得ます．□

6 数値例

この章では，幾つかの数値例をあげ，vpdsm の有効性を明らかにします．

6.1 Hilbert 行列

行列の性格をあらわす指標に**条件数** (*condition number*) があります (cf.[16])．条件数が大きい行列は，その要素が少し変化しただけで問題 (例えば連立 1 次方程式の解) の相対誤差が増大するため「悪条件となる行列」と呼ばれます．

本稿では $n \times n$ 行列 A の固有値 $\lambda_1, \lambda_2, \ldots, \lambda_n$ に対し，条件数を

$$\mathrm{cond}(A) \equiv \frac{\max_{1\leq i\leq n} |\lambda_i|}{\min_{1\leq i\leq n} |\lambda_i|}$$

で定義します[25]．

悪条件となる有名な行列に Hilbert 行列：

$$A_{ij} = \frac{1}{i+j-1} \quad i,j = 1,2,\ldots,n$$

があります． Hilbert 行列は対称な正定値行列ですが，その条件数は指数関数的に増大します[26]．

[3] によると，最小固有値，条件数は次のようになります．条件数は有効数字 6 桁で四捨五入しています．

n	最小固有値	条件数
3	$2.687340355773529 \times 10^{-3}$	5.24057×10^{2}
4	$9.670230402258689 \times 10^{-5}$	1.55137×10^{4}
5	$3.287928772171863 \times 10^{-6}$	4.76607×10^{5}
6	$1.082799484565550 \times 10^{-7}$	1.49511×10^{7}
7	$3.493898605991218 \times 10^{-9}$	4.75367×10^{8}
8	$1.111538966372442 \times 10^{-10}$	1.52576×10^{10}
9	$3.499676402911493 \times 10^{-12}$	4.93155×10^{11}
10	$1.093153819379666 \times 10^{-13}$	1.60263×10^{13}

vpdsm では，$N = 10$ までの正定値性が判定できました．$\delta = 10^{-6}$ です．以下に，最小固有値[27]に対する Householder-bisection 法による近似解の相対誤差，および vpdsm によって得られた最小固有値の下限との相対誤差を示します．それぞれ有効数字 6 桁で四捨五入しています．

n	近似解の相対誤差	下限の相対誤差
3	3.75836×10^{-14}	1.00000×10^{-6}
4	2.63179×10^{-13}	1.00004×10^{-6}
5	1.62260×10^{-11}	1.00139×10^{-6}
6	3.33365×10^{-10}	1.04452×10^{-6}
7	9.50657×10^{-12}	2.40610×10^{-6}
8	5.40777×10^{-8}	4.62505×10^{-5}
9	4.55702×10^{-6}	1.56398×10^{-3}
10	5.29759×10^{-5}	5.07078×10^{-2}

6.2 大規模な問題

次に，比較的性質のよい (条件数がそれほど大きくない) 行列を選んで大規模な問題を解いてみました．

$$A_{ij} = \min\{n-i+1, n-j+1\} \quad i,j = 1,2,\ldots,n$$

の固有値 λ_i は

$$\lambda_i = \frac{1}{2}\left[1 - \cos\frac{(2i-1)\pi}{2n+1}\right]^{-1} \quad i = 1,2,\ldots,n$$

となります ([3])．従って，次数を大きくすると最小固有値は 1/4 に収束します．

数値実験では，$\delta = 10^{-2}$ に対し判定を行ない，$N = 4096$ までの正定値性が判定できました．それ以上のサイズは，演算時間の問題により実行していません．

以下は，条件数と，最小固有値の下限の真の値に対する相対誤差とを表にしたものです．

n	下限の相対誤差	条件数
4	0.0100000000000417	2.92840×10^{1}
16	0.0100000000008186	4.37697×10^{2}
64	0.0100000000303602	6.74067×10^{4}
256	0.0100000016619280	1.06654×10^{5}
1024	0.0100001019772134	1.70154×10^{6}
4096	0.0100064565713022	2.72048×10^{7}

条件数は有効数字 6 桁で四捨五入，相対誤差は小数点 17 桁以降を切捨てています．

数値実験により，条件数がそれほど悪くない行列に対しては，δ が下限の相対誤差とほぼ一致することがわかります．

[24] 区間演算を用いた理由は，2 次形式の上界 λ を (over estimate でも構わないので) 厳密に評価するためです．

[25] 一般的な条件数の定義は，行列ノルム $||\cdot||$ に対し $\mathrm{cond}(A) = ||A||\,||A^{-1}||$ です．実対称行列に対して 2 ノルムを用いると本稿の定義と一致します．

[26] Hilbert 行列が導かれる経緯は [18] を御覧ください．

[27] [3] による値を真の値と見なしています．

6.3 ランダム行列

最後に，条件数と δ の関係を調べるために，100×100 の乱数行列を用いた数値実験を行ないました．

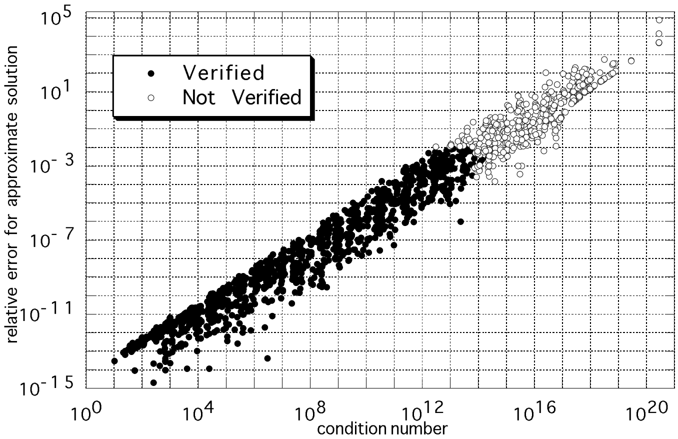
Fortran 90 の乱数発生関数で得られた 100 個の正の数によって構成される対角行列に，同じく乱数によって得られた直交行列を左右からかけることにより，固有値の値のわかる行列を構成しました．行列の作成過程には浮動小数点演算による丸め誤差が入るため，すべての処理を Fortran 90/VP の 4 倍精度 (128 ビット演算) で行なった後，固有値および行列を倍精度 (64 ビット) に変換して vpdsm に渡しています[28]．

次の 3 つの図は，$\delta=10^{-2}, 10^{-6}, 10^{-10}$ の場合の正定値性の判定が成功 (●)，不成功 (○) をプロットしたものです．

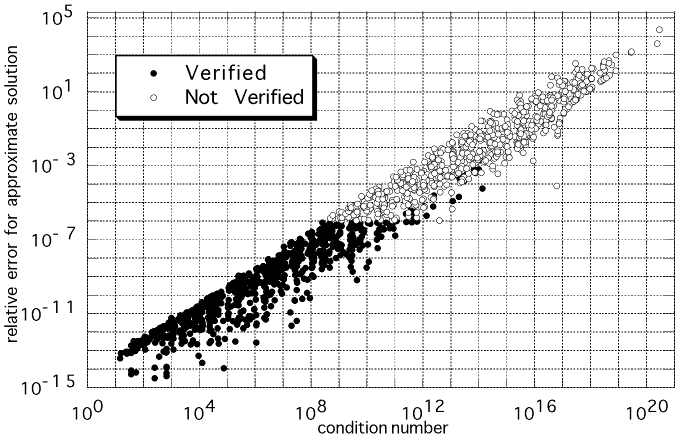
縦軸は近似解と真の解との相対誤差，横軸は条件数です．

この結果から，条件数が大きい行列に対しては精度よい近似解を得ることができず正定値の判定も難しくなること，また，条件数に応じて δ の値を調節することで正定値性が判定できる可能性があることがわかります．

《$\delta=10^{-2}$》

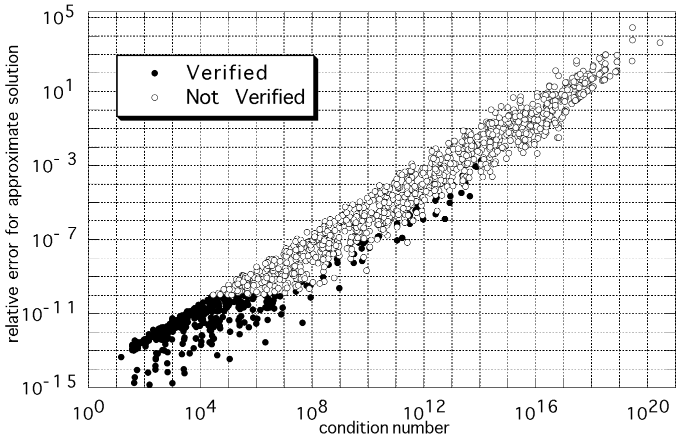


《$\delta=10^{-6}$》



[28]もちろん，厳密には与えた行列と固有値は正確に対応しません．しかし，行列の次元と浮動小数点演算の精度から極めて「近い」関係にあることが期待されます．

《$\delta=10^{-10}$》



7 補足

この章では，これまでの章で前提としたことがらについての補足です．詳細な証明は省略します．

7.1 正定値行列にまつわる性質

正定値性に関して成り立つ基本的な性質を列挙します．

性質 1 正定値の定義中の“$x\neq0$ である任意の $n\times1$ 実ベクトル x に対し”は，“任意の $x^Tx=1$ なる $n\times1$ 実ベクトルに対し”に置き換えることができます．

[理由] 必要十分条件は x のユークリッドノルム $||x||_2=(x^Tx)^{1/2}$ に対して $y=x/||x||_2$ を考えることで確認できます．

性質 2 0 でないベクトル x とスカラー λ が

$$Ax=\lambda x$$

を満たすとき，λ を A の固有値，x を固有ベクトルといいます．A が対称ならば A の固有値はすべて実数です．

[理由] A の固有値 λ と固有ベクトル x の共役成分 $\bar\lambda, \bar x$ に対して $A\bar x=\bar\lambda\bar x$ となることから，$x^TA\bar x=\bar\lambda x^T\bar x$, $\bar x^TAx=\lambda\bar x^Tx$ となります．ここで A の対称性を用いると $\lambda=\bar\lambda$ がいえます．

性質 3 実対称行列が正定値であるための必要十分条件は，すべての固有値が正であることです．

[理由] 固有値 λ と固有ベクトル x に対し $x^TAx=\lambda x^Tx=\lambda||x||_2{}^2$ であることから，A が正定値ならば $\lambda>0$ です．逆は，固有ベクトルが正規直交系をなすようにとれることと，任意のベクトル x を正規直交系で張ることができることを用いて，x の展開形を x^TAx に放り込んで正定値性を導きます．

性質 4 実行列 A が正則 (逆行列を持つ) ならば，AA^T は正定値行列です．

[理由] $x^TAA^Tx=(A^Tx)^T(A^Tx)=||A^Tx||_2{}^2$ より導かれます．

性質 5 実対称行列 A が正定値ならば，対角要素はすべて正です．逆は成立しません．

[理由] 定義の任意の x として単位行列の第 j 列ベクトルを順番にとればわかります．

性質 6　実対称行列 A が正定値ならば，その絶対値最大の要素は対角上にあります．逆は成立しません．

[理由] 絶対値最大の要素を A_{ij} として $i \neq j$ とします．このとき，i 番目の成分が α, j 番目の成分が β, 他の成分は 0 というベクトル e に対し $e^T A e = A_{ii}\alpha^2 + 2A_{ij}\alpha\beta + A_{jj}\beta^2$ が 0 より大きくなるための判別式より矛盾を導きます．

性質 7　A が正定値であれば，

$$A = CC^T$$

となるような下三角行列 C が存在します．このような分解を Cholesky 分解と呼びます．

[理由] Cholesky 分解のアルゴリズム自体が証明になっています．

7.2　正定値性の判定方法

行列の正定値性は，その行列が導かれた経緯から判定できる場合があります．たとえば，2 次形式 $x^T Ax$ が何かのノルムやエネルギーを表すならば，それが 0 以外で正の値をとることがわかります．

また，すべての主小行列式 (principal minor) がすべて正であることがいえれば，もとの行列は正定値です．ただし，次数が高くなると計算量が多くなります．

有効な正定値性の判定法として，「既約かつ対角成分が正である優対角行列は正定値」という定理があります ([16])．

もしこれらの判定方法から正定値性がいえるのならば，もちろん `vpdsm` を用いる必要はありません．

参考文献

[1] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, D. Sorensen : *LAPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, 1992.

[2] 藤野 清次, 張 紹良 : 反復法の数理, 応用数値計算ライブラリ, 朝倉書店, 1996.

[3] Robert T. Gregory, David L. Karney : *A Collection of Matrices for Testing Computational Algorithms*, John Wiley & Sons, New York, 1969.

[4] 東田 幸樹, 山本 芳人, 熊沢 友信 : 入門 Fortran90 実践プログラミング, ソフトバンク, 1994.

[5] R. Baker Kearfott, V. Kreinovich : *Applications of Interval Computations*, Kluwer Academic Publishers, Netherland, 1996.

[6] Thomas H. Kerr : Fallacies in Computational Testing of Matrix Positive Definiteness/Semidefiniteness, *IEEE Transactions on Aerospace and Electronic Systems*, Vol.26, No.2, pp.415–421 (1990).

[7] 李 磊 : 正定値行列の実用的な判別法について, 日本応用数理学会論文誌, Vol.7, No.1, pp.91–96 (1997).

[8] Evan M. Ma, Christopher J. Zarowski : On Lower Bounds for the Smallest Eigenvalue of a Hermitian Positive-Definite Matrix, *IEEE Transactions on Information Theory*, Vol.41, No.2, pp.539–540 (1995).

[9] S. マックレーン (彌永 昌吉監修, 赤尾和男・岡本周一共訳) : 数学 – その形式と機能, 森北出版, 1992.

[10] Arnold Neumaier : *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.

[11] J. Rohn : Positive Definiteness and Stability of Interval Matrices, *SIAM Journal on Matrix Analysis and Applications*, Vol.15, No.1, pp.175–184 (1994).

[12] Siegfried M. Rump : Verification Methods for Dense and Sparse Systems of Equations, in Jürgen Herzberger(Editor), *Topics in Validated Computations* (Proceedings of the IMACS-GAMM International Workshop on Validated Computation, Oldenburg, Germany, August 30th–September 3rd, 1993), Elsevier Science, North Holland, Amsterdam, pp.63–135 (1994).

[13] 杉原 正顕, 室田 一雄 : 数値計算法の数理, 岩波書店, 1994.

[14] T. Szulc : Testing Some Properties of Real Matrices, *Computers and Mathematics with Application*, Vol.31, No.4/5, pp.63–65 (1996).

[15] 山本 野人, 渡部 善隆, 田中 得登, 中尾 充宏 : 一般化固有値問題の精度保証付き解法, 日本応用数理学会 1997 年度年会講演予稿集, pp.10–11 (1997).

[16] Richard S. Varga : *Matrix Iterative Analysis*, Prentice-Hall, 1962.
［邦訳］ 渋谷 政昭, 棚町 芳弘, 金子 正久, 野田 隆 訳 : 計算機による大型行列の反復解法, サイエンスライブラリ 情報電算機 10, サイエンス社, 1972.

[17] 渡部 善隆 : VPP700/56 での対話型処理, 九州大学大型計算機センター広報, Vol.30, No.2, pp.167–185 (1997).

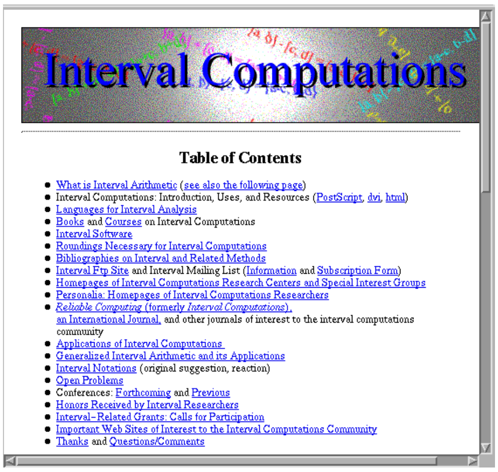
[18] 渡部 善隆 : 連立 1 次方程式の基礎知識〜および Gauss の消去法の安定性について〜, 九州大学大型計算機センター広報, Vol.28, No.4, pp.291–349 (1995).

[19] UXP/V Fortran 90/VP 使用手引書 V10 用, J2U5-0050, 富士通株式会社, 1995.

[20] UXP/V VP プログラミングハンドブック V10 用, J2U5-0070, 富士通株式会社, 1997.

[21] SSL II 使用手引書 (科学用サブルーチンライブラリ), 99SP–4020, 富士通株式会社, 1987.

[22] VPP700/56 利用の手引 (第 1.0 版), 九州大学大型計算機センター・プログラムライブラリ室 (1997).



`http://interval-computations.home.ml.org/`