電 301 数値解析
第 6 回
固有値問題

前回付記 (1)

- Gauss-Seidek 法は, ものの本を見ると, 以下の差分方程式を解け, というふうに書いてある.

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(-\sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} + b_i \right)$$

ただし $x_i^{(k)}$ は第 k 回目の繰り返しにおけるベクトル $\boldsymbol{x}$ の第 i 成分.

前回付記 (2)

- 上述の式は左辺に $x_i^{(k+1)}$, 右辺に $x_j^{(k+1)}$ があるため計算不能に見えるが, $x_j^{(k+1)}$ に関する和が $j < i$ に限定されているため, $i = 1, 2, 3 \cdots$ という順で計算すれば, 矛盾なく計算できる. 順番を変えると計算できないので注意. この種の見づらい記法は数値解析の分野では珍しくない.

前回付記 (3)

- SOR 法についても同様に, 成分ごとの差分方程式で書くと以下のようになる.

$$y_i^{(k+1)} = \frac{1}{a_{ii}} \left(-\sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} + b_i \right)$$

$$x_i^{(k+1)} = x_i^{(k)} + w \left(y_i^{(k+1)} - x_i^{(k)} \right)$$

前回付記 (4)

- 行列の演算を上記の差分方程式で置き換えると, 使用するメモリは減る.
- Jacobi 法も同様.
- 実行速度はどうかというと, 先週の例の Gauss-Seidel 法と SOR 法では, 上記の書き換えにより MATLAB では求解が高速化されたが, Scilab では逆に遅くなった.

前回付記 (5)

- Scilab は MATLAB と比べて全般に低速であり, 特に繰り返し (for 文や while 文など) を用いたときの性能の低下が顕著. MATLAB を使っている分には行列の成分ごとの計算が出てもとくに気にする必要はないが, Scilab では成分ごとの計算をすると処理が遅くなることがある.

はじめに (1)

- 黎明期のプログラミング言語は COBOL (1959–; 事務用) と Fortran (1954–; 科学技術計算用).
- これらは今でも使われている.

年号の典拠:
`http://sunsite.univie.ac.at/Fortran-Guide/ch1-1.html`
`http://americanhistory.si.edu/cobol/introduction`

はじめに (2)

以下の議論の典拠は
Elizabeth Jessup,
Numerical Linear Algebra,
ftp://ftp.mcs.anl.gov/pub/petaflops/summer.study/linear.tex

www.netlib.org/

http://jp.mathworks.com/company/newsletters
/articles/matlab-incorporates-lapack.html

https://www.scilab.org/scilab/history

http://math-atlas.sourceforge.net/

https://software.intel.com/en-us/intel-mkl/

はじめに (3)

- 線形計算のためのサブルーチン集である LINPACK と EISPACK は Fortran で書かれた.
- LINPACK は 1979 年にリリースされた線形方程式を解くためのサブルーチン集.
- EISPACK は 1976 年にリリースされた固有値問題を解くためのサブルーチン集.

はじめに (4)

- MATLAB は, 1970 年代に, LINPACK と EISPACK に基づく対話的な計算ソフトとして出発した (今日では線形計算は MATLAB のごく一部になっている).

- Scilab は 1980 年代に MATLAB と類似したソフトとして開発された (当時の名称は Blaise),

はじめに (5)

- 今日では,LINPACK と EISPACK は LAPACK というパッケージに統合されている.
- ATLAS(Automatically Tuned Linear Algebra Software) を使うと数値計算のライブラリを特定のコンピュータ向きに高効率化できる.
- Intel は, Math Kernel Library というライブラリを公開している.

はじめに (6)

- LINPACK と EISPACK が併存したことからわかるように, 線形方程式を解くことと, 固有値問題を解くことは, 線形計算の二大重要トピック.
- しかし教科書では固有値問題が取り扱われていない.
- よって, 教科書を離れて, 固有値問題の解法について概説する.

今回の講義の参考文献

- 杉原, 室田, 線形計算の数理, 岩波書店, 2009.
- 森, 数値解析, 第 2 版, 共立出版, 2002.
- 山本, 数値解析入門 [増補版], サイエンス社, 2003.
- 斎藤, 数値解析入門, 東京大学出版会, 2012.
- 久保田, 工学基礎 数値解析とその応用, 数理工学社, 2010.
- 伊理, 藤野, 数値計算の常識, 共立出版, 1985.

固有値と固有ベクトル (1)

- n 次の正方行列 $\boldsymbol{A}$ に対し, 複素数 λ と n 次のベクトル $\boldsymbol{x} \neq \boldsymbol{0}$ が $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$ という方程式を満たすとき, λ を $\boldsymbol{A}$ の固有値, $\boldsymbol{x}$ を固有値 λ に対応する固有ベクトルという.
- n 次行列 $\boldsymbol{A}$ の固有値は, 代数方程式 $\det(s\boldsymbol{I} - \boldsymbol{A}) = 0$ の根であり (ただし $\boldsymbol{I}$ は n 次の単位行列で s は変数), 重複度を含めて高々 n 個である.

固有値と固有ベクトル (2)

- 固有値 λ に対応する固有ベクトル全体が張る線形部分空間を λ に対応する固有空間という.
- 行列 $\boldsymbol{A}$ の固有ベクトル全体が張る線形部分空間 (どの固有値に対応するかは問わない) を, $\boldsymbol{A}$ の固有空間という. $\boldsymbol{A}$ の固有空間は必ずしも全空間とは一致しない.

固有値と固有ベクトル (3)

- $\boldsymbol{A}$ の固有空間を拡張して全空間を張るようにしたものを, 一般化固有空間という. これに対応するのが Jordan 標準形である. 一般化固有空間まで広げて考えれば, (一般化) 固有値の数は (重複度を含めて)n に一致する.

固有値・固有ベクトルの応用

- 微分方程式・差分方程式の求解と特性解析
- google によるページのランキング

  http://www.ams.org/samplings/feature-column/fcarc-pagerank
- パターン認識 www.tuat.ac.jp/~s-hotta/SSII/slide_pattern.pdf
- 建造物の共振の解析 (地震や風への応答など)

  http://www.kozosoft.co.jp/gijyutu/s07.html
- 他にも色々

固有値の数値計算 (1)

- 理論的には, $\det(s\boldsymbol{I} - \boldsymbol{A}) = 0$ という代数方程式を s について解けば, 行列 $\boldsymbol{A}$ の固有値をすべて求めることができるが …

- 行列式を求めること自体が数値的に大変な上に, 代数方程式の求解は数値計算の誤差に弱いので, ふつうはそんなことはやらない.

固有値の数値計算 (2)

- Scilab の組み込み関数 `roots()` のマニュアルは次のようになっている.

  x=roots(p) は 多項式 p の x である 複素ベクトルを返す. 100 次以下の実数多項式の場合, 高速な (Jenkins-Traub 法に基づく) RPOLY アルゴリズムが使用される. その他の場合, その根はコンパニオン行列の固有値として計算される.

固有値の数値計算 (3)

- 固有値を求めるために代数方程式を解くのではなく, 代数方程式を解くために固有値を求めることもある.

- 連立一次方程式が理論的には有限回の計算で解けるのに対し, 固有値問題は 5 次以上の行列に対しては有限回の演算では解けない.

固有値の数値計算 (4)

- よって, 固有値問題は数値的には本質的に近似の問題.
- 固有問題の数値解法は技巧的なので, 初学者が自分でプログラムを組むのは危険. この講義でもアルゴリズムの概要の紹介にとどめる.

固有値の数値計算 (5)

- 固有値の数値計算の難易度は, 易しい順に,
  - ▷ 対称行列
  - ▷ 対角化可能行列
  - ▷ 対角化不能行列 (Jordan 標準形を数値的に求めるとき)

固有値の数値計算 (6)

- 有限精度の計算機で Jordan 標準形を求めることには数値的な困難があり (数学的な困難ではないことに注意), これを回避するために様々な工夫がなされている.

典拠:鈴木, 渡邉, 明, Jordan 標準形の数値計算について, 数理解析研究所講究録, Vol. 990, pp. 52–61, 1997.

固有値の数値計算 (7)

- Jordan 標準形に関する議論は初学者には閾が高すぎると思われるが, 一方で対称行列については取り扱う必然性が不明瞭と思われるので, この講義では対角化可能行列な行列 (n 次の行列で, 重複度を含めて n 個の固有値があり, 対応する固有空間が全空間を張るもの) を対象として議論を進める.

冪乗法 (1)

- $\boldsymbol{A}$ の (複素) 固有値を $\lambda_1, \ldots, \lambda_n$ とする．これらは絶対値が大きい順に並べられているものとする．

- $|\lambda_1| > |\lambda_2|$ のときには, λ_1 とそれに対応する固有ベクトルは, 比較的簡単に求められる．

冪乗法 (2)

- 初期値 $\boldsymbol{x}^{(0)}$ を適切な単位ベクトルとし, 以下の差分方程式を解けばよい.

$$\boldsymbol{y}^{k+1} = \boldsymbol{A}\boldsymbol{x}_k,$$
$$\boldsymbol{x}_{k+1} = \boldsymbol{y}^{(k+1)} / \|\boldsymbol{y}^{(k+1)}\|$$

- ただし, 初期値の取り方が悪い場合にはやり直しが必要となることがある.

冪乗法 (3)

- 上記の解法を冪乗法という.
- 冪乗法がうまくいく条件と, その理由は, いずれも簡単.
- $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$ とする.
- λ_i に対応する固有ベクトルを $\boldsymbol{v}_i$ とする.

冪乗法 (4)

- 初期値 $\boldsymbol{x}_0$ を固有ベクトルで展開した結果, $\boldsymbol{x}^{(0)} = c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n$ となったものとする.
- $c_1 \neq 0$ が冪乗法がうまくいくための条件.
- このとき, $k \to \infty$ としたとき, $(\lambda_i/\lambda_1)^k \to 0$ である.

冪乗法 (5)

- $\boldsymbol{A}^k\boldsymbol{x}^{(0)} = c_1\lambda_1^k\boldsymbol{v}_1 + \sum_{j=2}^n c_j\lambda_j^k\boldsymbol{v}_j$ だから,

$$\frac{\boldsymbol{A}^k\boldsymbol{x}^{(0)}}{\|\boldsymbol{A}^k\boldsymbol{x}^{(0)}\|} = \frac{c_1\lambda_1^k\boldsymbol{v}_1}{\|c_1\lambda_1^k\boldsymbol{v}_1 + \sum_{j=2}^n c_j\lambda_j^k\boldsymbol{v}_j\|} + \frac{\sum_{j=2}^n c_j\lambda_j^k\boldsymbol{v}_j}{\|c_1\lambda_1^k\boldsymbol{v}_1 + \sum_{j=2}^n c_j\lambda_j^k\boldsymbol{v}_j\|}q$$

冪乗法 (6)

- 右辺の分母と分子を λ_1^k で割ると

$$\frac{\boldsymbol{A}^k\boldsymbol{x}^{(0)}}{\|\boldsymbol{A}^k\boldsymbol{x}^{(0)}\|} = \frac{c_1\boldsymbol{v}_1}{\|c_1\boldsymbol{v}_1 + \sum_{j=2}^n c_j(\lambda_j/\lambda_1)^k\boldsymbol{v}_j\|} + \frac{\sum_{j=2}^n c_j(\lambda_j/\lambda_1)^k\boldsymbol{v}_j}{\|c_1\boldsymbol{v}_1 + \sum_{j=2}^n c_j(\lambda_j/\lambda_1)^k\boldsymbol{v}_j\|}q$$

冪乗法 (7)

- $k \to \infty$ とすると $j \geq 2$ に対して $(\lambda_j/\lambda_1)^k \to 0$ となるから, $\|v\|_1$ が単位ベクトルに取られていたことを思い出すと, $\displaystyle\lim_{k\to\infty} \frac{\boldsymbol{A}^k\boldsymbol{x}^{(0)}}{\|\boldsymbol{A}^k\boldsymbol{x}^{(0)}\|} = \frac{c_1}{|c_1|}\boldsymbol{v}_1.$
- 固有ベクトルの定数倍はやはり固有ベクトルだから (この場合, 定数として複素数を許容する), $\frac{c_1}{|c_1|}\boldsymbol{v}_1$ は確かに固有ベクトルになっている.

冪乗法 (8)

- 行列 $\boldsymbol{A}$ の固有値の中に絶対値が等しいものがない場合には, (絶対値) 最大固有値とそれに対応する固有ベクトルを使って行列 $\boldsymbol{A}$ を適切に変形することにより, 行列 $\boldsymbol{A}$ のすべての固有値を求めることができる. この手順を減次という. 詳細については杉原, 室田, 線形計算の数理 (岩波書店) などを参照.

QR 法 (1)

- 小規模な行列の固有値の計算に用いられる代表的な方法が QR 法.
- QR 法の説明に入る前に, いくつか言葉の準備をしておく.

QR 法 (2)

- n 次正方行列 $\boldsymbol{Q}$ の列ベクトルが正規直交基底をなすとき, すなわち $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$ となるとき, $\boldsymbol{Q}$ を直交行列という.
- n 次正方行列 $\boldsymbol{Q}$ の列ベクトルが複素内積の意味で正規直交規定をなすとき, すなわち $\boldsymbol{Q}^H\boldsymbol{Q} = \boldsymbol{I}$ となるとき, $\boldsymbol{Q}$ をユニタリ行列という.
- $\boldsymbol{Q}^H$ は Hermite 転置 (行列を転置してから各成分の複素共役を取ったもの) をあらわす.

QR 法 (3)

- QR 法の基礎となるのは, 次の数学的事実である ([斎藤], [杉原, 室田]):

複素行列 $\boldsymbol{A}$ は, 適切なユニタリ行列 $\boldsymbol{U}$ を取ることにより, $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{U}^H$ という形に変換できる. ただし, $\boldsymbol{S}$ は上三角行列である ($\boldsymbol{A}$ は対角化可能でなくてもよい).

QR 法 (4)

- 上記の事実は, 「複素行列 $\boldsymbol{A}$ は少なくともひとつ固有値と対応する固有ベクトルを持つ」という事実を用い, 帰納法によって証明される (この講義では証明は述べない).

- 数値としては $\boldsymbol{S}$ の固有値と $\boldsymbol{A}$ の固有値は一致する.

QR 法 (5)

- $\boldsymbol{A}$ が対角化可能のときには, $\boldsymbol{A}$ の各固有値と対応する $\boldsymbol{S}$ の固有値の重複度は一致する.

- $\boldsymbol{A}$ が対角化不能のときには, $\boldsymbol{A}$ の各固有値と対応する $\boldsymbol{S}$ の固有値の重複度に一致しないものが出る.

QR 法 (6)

- $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{S}\boldsymbol{U}^H$ という形の表現 (ただし $\boldsymbol{Q}$ はユニタリ行列, $\boldsymbol{S}$ は上三角行列) を Schur 分解という.

- QR 法とは, 行列 $\boldsymbol{A}$ の Schur 分解を近似的に求める方法である ([杉原, 室田]).

QR 法 (7)

- Scilab で Schur 分解を求める関数は `schur`.
- $\boldsymbol{A}$ の Schur 分解が $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{T}\boldsymbol{U}^H$ であるとき, `[U,T]=schur(A)` とすると, `U` にユニタリ行列が, `T` に上三角行列が返される (結果を受け取る変数の名称は任意).

QR 法 (8)

- 正方行列の Schur 分解はつねに可能だが, QR 法は行列 $\boldsymbol{A}$ が正則でないと使えない.
- 以下では, $\boldsymbol{A}$ は正則であると仮定する. ($\boldsymbol{A}$ が対角化可能であることも仮定されていたことを思い出すこと).

QR 法 (9)

- QR 法の基礎となるのは, 行列の QR 分解と呼ばれる分解である ([森]).

複素正則行列 $\boldsymbol{A}$ は, 適切なユニタリ行列 $\boldsymbol{Q}$ を取ることにより, $\boldsymbol{A} = \boldsymbol{QR}$ という形に変換でき, この表現は一意的である. ただし, $\boldsymbol{R}$ は対角成分が正の上三角行列である.

QR 法 (10)

- Scilab で QR 分解を求める関数は qr である. `[Q,R]=qr(A)` のようにして使う.

- qr は行列 $\boldsymbol{A}$ が非正則または正方でない場合にも適用可能で, $\boldsymbol{A} = \boldsymbol{QR}$ となるユニタリ行列 $\boldsymbol{Q}$ と上三角行列 $\boldsymbol{R}$(正方行列とは限らない) が計算される.

QR 法 (11)

- QR 分解の手法にはこの講義では立ち入らないが, どのような方法があるかを列挙しておく [杉原, 室田].
- 数学的には, 行列 $\boldsymbol{A}$ の列ベクトルに Gram-Schmidt の直交化を適用すると QR 分解を求められるが, この方法は数値計算の誤差に弱いので実用上は用いられない.

QR 法 (12)

- 実際に使われる主要な方法は, 修正 Gram-Schmidt 法, Householder 変換による方法, Givens 変換による方法である. 詳細については [杉原, 室田] を参照.

- Scilab の関数 `qr` は処理を LAPACK に丸投げ.

QR 法 (13)

- QR 法のアルゴリズムは極めて単純である.
- 次ページにアルゴリズムの形で示す.
- QR 法には停止条件は定められていないので, 利用者が適切に止める必要がある.

QR 法 (14)

QR 法のアルゴリズム

(初期化) $\boldsymbol{A}_1 = \boldsymbol{A}$, $k = 1$ とする.

(ループ) $\boldsymbol{A}_k$ を QR 分解する. $\boldsymbol{A}_k = \boldsymbol{Q}_k \boldsymbol{R}_k$ と分解されているものとし, $\boldsymbol{A}_{k+1} = \boldsymbol{R}_k \boldsymbol{Q}_k$ と定義する. $k = k + 1$ としてループ冒頭に戻る.

QR 法 (15)

- QR 法の収束性については, 次の事実が成り立つ ([森]).

行列 $\boldsymbol{A}$ が正則かつ対角化可能で, その固有値の絶対値がすべて相異なるとき, QR 法によって得られる行列 $\boldsymbol{A}_k$ は上三角行列に収束し, その対角成分には固有値が絶対値の大きい方から順に並ぶ.

QR 法 (16)

- Scilab で固有値を求める関数は spec.
- A を行列としたとき, spec(A) とすると固有値が求められる.

QR 法 (17)

- `[R,D]=spec(A)` とすると (結果を受けとる変数の名前は任意), `D` に `A` を対角化した結果 (対角要素に固有値が並んだ行列), `R` に対角化のための正則行列が返される.

QR 法 (18)

Scilab による数値例 (1)

```
-->A=[1 2;3 4];spec(A)
 ans  =
  - 0.3722813
    5.3722813
```

A の固有値は-0.3722813 と 5.3722813.

QR 法 (19)

Scilab による数値例 (2): A の Schur 分解

```
-->[U,T]=schur(A)
 T  =
  - 0.3722813  - 1.
    0.           5.3722813
 U  =
  - 0.8245648  - 0.5657675
    0.5657675  - 0.8245648
```

QR 法 (20)

Scilab による数値例 (3) QR 法 (1 回目)

```
-->[Q,R]=qr(A);A=R*Q
 A   =
    5.2     1.6
    0.6   - 0.2
```

QR 法 (21)

Scilab による数値例 (4) QR 法 (2 回目)

```
->[Q,R]=qr(A);A=R*Q
 A   =
    5.379562    - 0.9562044
    0.0437956   - 0.3795620
```

対角要素はすでに固有値に近い.

QR 法 (22)

Scilab による数値例 (5) QR 法 (7 回目)

```
->[Q,R]=qr(A);A=R*Q
 A   =
    5.3722813     1.0000001
    6.980D-08   - 0.3722813
```

小数点以下 7 桁まで正しい.

Scilab における一般化固有ベクトル

- Scilab で一般化固有ベクトルを求めるには `bdiag` という関数を使う．名前だけは覚えておくこと．

固有値問題のその他の解法

- QR 法は簡単ではあるが規模が大きい問題には使えない.
- 他の代表的なアルゴリズムの名称のみ列挙すると [杉原, 室田], LR 法, Arnoldi 法, Lanczos 法, Jacobi-Davison 法などといった方法がある.
- 対称行列の固有値を求めるための古典的な解法に Jacobi 法 (1846 年) がある [杉原, 室田]. この手法は今日でも用いられることがある.