第2章　数値線形代数

2.1　目的

数値線形代数の主要テーマである連立1次方程式や行列固有値問題は，科学技術計算から日常生活まで，ありとあらゆるところに現れる基本問題である．例えば（他テーマで扱うであろう）物理現象の数値シミュレーションでは，コアの部分は連立1次方程式の計算であったりする．また大規模データ解析の場面では，しばしば巨大なデータ行列の固有値・固有ベクトルの計算が必要になる．

本章では，そうした数値線形代数の基礎的な知識と技術を身につけることを目的として，連立1次方程式

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2.1}$$

および行列固有値問題

$$A\boldsymbol{x} = \lambda\boldsymbol{x} \tag{2.2}$$

の数値解法を実装し，数値実験によりその性能を調べる．

連立1次方程式 (2.1) の解は（係数行列 A が正則ならば）$\boldsymbol{x} = A^{-1}\boldsymbol{b}$ と表せる．しかし逆行列を計算して解を求める方法は誤差に弱く，計算量にも利点はないため用いられることはめったにない．クラーメルの公式を用いる方法も同様である．

実用的な連立1次方程式の数値解法は，直接法と反復法の2つに大分類される．直接法では有限回の四則演算で（演算が無誤差で行えるならば真の）解を求める．反復法（ヤコビ法，ガウス・ザイデル法，SOR 法，共役勾配法など）では解に収束する近似解の列を逐次生成する．本章では直接法の代表格である消去法と，行列の LU 分解による解法を扱う．

行列 A の固有値は特性多項式 $\varphi(z) = \det(zI - A)$ の根として特徴付けられる．しかし固有値を数値的に求めるために，元の問題の線形構造を無視して，代数方程式 $\varphi(z) = 0$ を直接解くようなことはしない．

実用的な行列固有値問題の解法の多くは，真の固有値・固有ベクトルに収束する近似列をつくる反復法である．そこでは対象とする行列の構造など様々な要請に応じて，多種多様な解法（ヤコビ法，二分法，分割統治法，LR 法，dqds 法など）が開発されている．本章では古典的な解法であるべき乗法と，最も広く利用されている（であろう）QR 法を主に扱う．

数値線形代数を詳しく扱った書籍としては [1, 2] などがある．また線形計算のためのソフトウェアライブラリも数多く公開されている．LAPACK [3] などオープンソースのものもあるので興味のある人は参考にして欲しい．

注意 2.1.1 本資料の作成時に用いた実験環境は次の通りである．

OS	macOS Catalina 10.15.4
CPU	Intel Core i7-8569U 2.8GHz
Memory	16 GB
Compiler	Clang version 11.0.3

2.2　準備

2.2.1　ベクトルと行列のノルム

ベクトルの大きさはノルムで測る．（複素）ベクトル $\boldsymbol{x} = (x_i) \in \mathbb{C}^n$ および実数 $p \geq 1$ に対して

$$\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right) \tag{2.3}$$

を $\boldsymbol{x}$ の p **ノルム**という．特に2ノルム $\|\boldsymbol{x}\|_2$ はユークリッドノルムとも呼ばれる．また

$$\|\boldsymbol{x}\|_\infty = \max_{1 \leq i \leq n} |x_i| \tag{2.4}$$

を ∞ **ノルム**という．数値計算では高速に計算できる $\|\boldsymbol{x}\|_\infty$ が採用されることが多い．

どのノルムでも $\|\boldsymbol{x}\| \geq 0$ である．ただし $\|\boldsymbol{x}\| = 0$ となるのは $\boldsymbol{x} = \boldsymbol{0}$（零ベクトル）の場合に限る．またスカラー倍に関して $\|c\boldsymbol{x}\| = |c|\|\boldsymbol{x}\|$ である．

注意 2.2.1 このようにノルムによらない議論をする場合や，どのノルムを用いてもよい場面では，任意のノルムを表すために記号 $\|\boldsymbol{x}\|$ を用いることにする．特定のノルムを用いる場合は $\|\boldsymbol{x}\|_2$ などノルムの種類を明示する．

行列の大きさを測るためのノルムもある．（複素）行列 $A = (a_{i,j}) \in \mathbb{C}^{n\times n}$ に対して，理論上重要なのは（ベクトルのノルムを用いて定義される）作用素ノルム

$$\|A\| = \max_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|A\boldsymbol{x}\|}{\|\boldsymbol{x}\|} = \max_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\| \tag{2.5}$$

である．実際の数値計算では値を簡単に求められる**最大値ノルム**

$$\|A\|_{\max} = \max_{1\leq i,j\leq n} |a_{i,j}| \tag{2.6}$$

などが重用される．

ベクトルの場合と同様にどの行列ノルムでも $\|A\| \geq 0$ である．ただし $\|A\| = 0$ となるのは $A = O$（零行列）の場合に限る．またスカラー倍に関して $\|cA\| = |c|\|A\|$ である．

2.2.2 内積

ベクトル $\boldsymbol{x} = (x_i),\ \boldsymbol{y} = (y_i) \in \mathbb{C}^n$ に対して

$$(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^*\boldsymbol{y} = \sum_{i=1}^{n} \overline{x_i} y_i \tag{2.7}$$

を**（エルミート）内積**という．ただし（ベクトルを含む）行列 A に対して $A^* = (\overline{A})^{\mathsf{T}}$（共役転置）である．特に 2 ノルムに関して

$$(\boldsymbol{x}, \boldsymbol{x}) = \|\boldsymbol{x}\|_2^2 \tag{2.8}$$

が成り立つ．

実ベクトルの場合

$$(\boldsymbol{x}, \boldsymbol{y}) = 0 \tag{2.9}$$

ならばベクトル $\boldsymbol{x}, \boldsymbol{y}$ は直交する．（つまり $\boldsymbol{x}, \boldsymbol{y}$ のなす角が $\frac{\pi}{2}$ になる．）複素ベクトルの場合にも (2.9) が成り立つとき（形式的に）$\boldsymbol{x}, \boldsymbol{y}$ は**直交する**という．

2.3 連立 1 次方程式の数値解法

行列 $A = (a_{i,j}) \in \mathbb{C}^{n\times n}$ およびベクトル $\boldsymbol{b} = (b_i) \in \mathbb{C}^n$ に対して，連立 1 次方程式

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2.10}$$

の解 $\boldsymbol{x} \in \mathbb{C}^n$ を数値的に求めることを考える．ただし A は正則であるとする．

方程式 (2.10) の数値解 $\tilde{\boldsymbol{x}} \in \mathbb{C}^n$ の精度は，真の解 $\boldsymbol{x}$ との相対誤差

$$\frac{\|\boldsymbol{x} - \tilde{\boldsymbol{x}}\|}{\|\boldsymbol{x}\|} \tag{2.11}$$

で測る．

注意 2.3.1 方程式 (2.10) の解法を実装したプログラムに誤りがないか確認するには，数値解 $\tilde{\boldsymbol{x}}$ の残差ノルム

$$\|\boldsymbol{b} - A\tilde{\boldsymbol{x}}\| \tag{2.12}$$

を調べるのが便利である．この値は $\tilde{\boldsymbol{x}}$ が真の解であれば 0 になる．このため極端に大きな値になる場合はプログラムに誤りがある可能性が高い．

2.3.1 消去法

（ガウスの）消去法では連立 1 次方程式 (2.10) の解を次のように求める．

消去法

次の手続きでベクトル $\boldsymbol{x}$ を求める．

(1) 拡大係数行列 $(A, \boldsymbol{b})$ を行基本変形により行階段形にする：

$$(A', \boldsymbol{b}') = \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} & b'_1 \\ 0 & a'_{2,1} & \cdots & a'_{2,n} & b'_2 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & a'_{n,n} & b'_n \end{pmatrix}. \tag{2.13}$$

(2) 方程式 $A'\boldsymbol{x} = \boldsymbol{b}'$ の解 $\boldsymbol{x}$ を漸化式

$$x_i = \frac{1}{a'_{i,i}}\left(b'_i - \sum_{j=i+1}^{n} a'_{i,j}x_j\right) \qquad (2.14)$$

$(j = n, n-1, \ldots, 1)$ により求める.

消去法において方程式 $A'\boldsymbol{x} = \boldsymbol{b}'$ は元の方程式 (2.10) と同値であることに注意する. (1) で $(A, \boldsymbol{b})$ を行階段形にするには, 例えば次のようにすればよい.

前進消去

次の手続きを $k = 1, 2, \ldots, n-1$ の順に繰り返して拡大係数行列 $(A, \boldsymbol{b})$ を逐次更新する.

(1) 第 k 列の下三角成分 $a_{k,k}, \ldots, a_{n,k}$ の中から $a_{i,k} \neq 0$ をひとつ選ぶ. (この $a_{i,k}$ を**ピボット**という.)

(2) $i \neq k$ ならば第 k 行と第 i 行を入れ替える.

(3) ($a_{k+1,k}, \ldots, a_{n,k}$ がすべて 0 になるように) 各 $k+1 \leq i \leq n$ に対して第 i 行から第 k 行の $\frac{a_{i,k}}{a_{k,k}}$ 倍を引く.

注意 2.3.2 (1) でピボット $a_{i,k}$ として 0 に近いものを選ぶと, 後の演算で桁落ちが起こったり, 情報落ちのため精度が悪化したりする可能性がある. このため通常は絶対値最大の $a_{i,k}$ をピボットとして選ぶ.

単精度 (`float`) で実装した消去法により, 連立 1 次方程式 (2.10) の解を求める数値実験を行った. 100 次の行列 A とベクトル $\boldsymbol{b}$ のペアをランダムに 100 個生成し, 各ペアに対して連立 1 次方程式 (2.10) の数値解を求めた. 計算結果をまとめたものを図 2.1–2.3 に示す.

図 2.1 のグラフは, 消去法で求めた数値解 $\tilde{\boldsymbol{x}}$ の残差ノルム (2.12) を, 各ペア $A, \boldsymbol{b}$ に対して (対数) プロットしたものである. 残差ノルムは概ね小さいので, 実装したプログラムは正しく動作していると思われる.

図 2.2 のグラフは, 真の解との相対誤差 (2.11) を同様にプロットしたものである. (点線はその

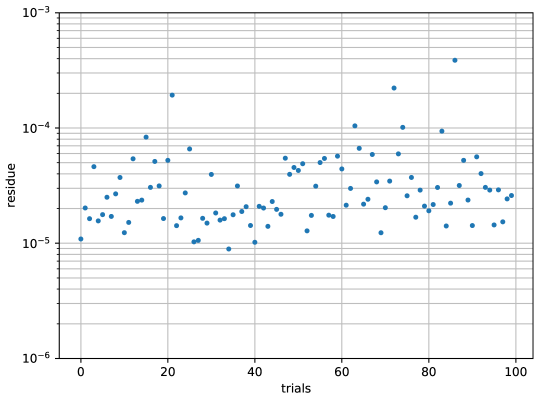

図 2.1: 消去法による残差ノルム,

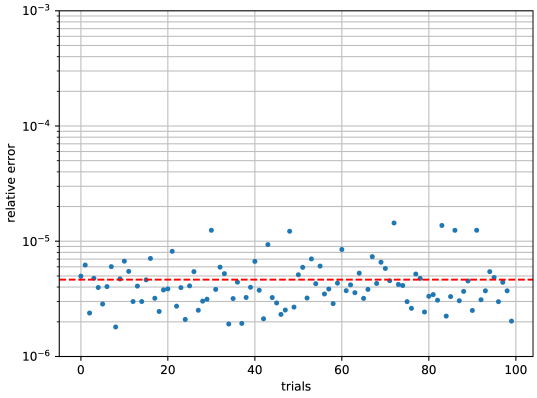

図 2.2: 消去法による相対誤差. (点線は平均値.)

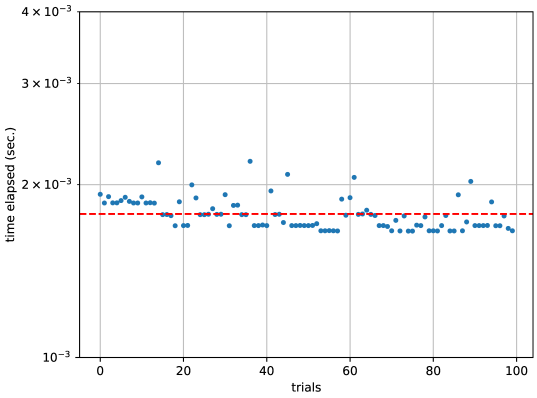

図 2.3: 消去法による計算時間. (点線は平均値.)

平均値である.）ただし不明な真の解の代わりに，倍精度（double）の消去法で求めた数値解 $\boldsymbol{x}$ を用いている．相対誤差はおおよそ 10^{-6}〜10^{-5} 程度なので，求めた数値解には 10 進 5 桁程度の精度があると考えられる．

図 2.3 のグラフは，数値解を求めるのに要した計算時間を同様にプロットしたものである.（点線はその平均値である.）少しばらつきはあるが 2×10^{-3} 秒弱の辺りに固まっている．消去法は直接法であり，解を求めるまでの四則演算の回数は次数 n のみで決まる．このため計算時間が試行によらずほぼ一定になっていると考えられる．

課題 1 消去法により，連立 1 次方程式 (2.10) の解を求める数値実験を行う．$n=100,200,400,800$ のそれぞれに対して，n 次の行列 A とベクトル $\boldsymbol{b}$ のペアをランダムに 100 個生成し，各ペアに対して連立 1 次方程式 (2.10) の数値解を求めよ．

(1) 各 n に対して，数値解 $\tilde{\boldsymbol{x}}$ の残差ノルム (2.12)，相対誤差 (2.11)，計算時間をそれぞれプロットせよ．

(2) 数値解の精度と次数 n の関係について考察せよ．

(3) 計算時間と次数 n の関係について考察せよ．

2.3.2 行列の LU 分解

行列 A を単位下三角行列 L と上三角行列 U を用いて

$$A=LU \tag{2.15}$$

の形に表すことを LU **分解**という．ただし単位下三角行列とは対角成分がすべて 1 の下三角行列のことである．

行列 A が正則ならば $1,\dots,n$ の置換 π が存在して

$$P_\pi A=LU \tag{2.16}$$

の形に LU 分解できる．ただし $P_\pi=(\delta_{i,\pi(i)})$（置換行列）である．この事実は消去法と同じ考え方により示される．

消去法では前進消去の手続き (1)–(3) を繰り返して A を上三角行列 $A'(=U)$ にする．この操作を行列積で表すと

$$L_{n-1}P_{\tau_{n-1}}\cdots L_1P_{\tau_1}A=U \tag{2.17}$$

である．ただし前進消去において k 回目の (1) で選んだピボットを $a_{i_k,k}$ とおくとき

$$\tau_k=(k\ i_k) \tag{2.18}$$

（k と i_k を入れ替える互換）である．ただし $i_k=k$ ならば τ_k は恒等置換である．また k 回目の (2) を終えた直後の $A=(a_{i,j})$ に対して

$$L_k=\begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -\frac{a_{k+1,k}}{a_{k,k}} & 1 & \\ & & \vdots & & \ddots \\ & & -\frac{a_{n,k}}{a_{k,k}} & & 1 \end{pmatrix} \tag{2.19}$$

である．

いま，$i_k\geq k$ より，置換

$$\pi_k=\tau_{n-1}\cdots\tau_k \tag{2.20}$$

は $k,\dots,n$ のみに作用する．これより行列

$$L'_k=P_{\pi_{k+1}}L_kP_{\pi_{k+1}}^{\mathsf{T}} \tag{2.21}$$

は（L_k と同じ形の）下三角行列である．このとき (2.17) より $(L'_{n-1}\cdots L'_1)P_{\pi_1}A=U$ である．従って

$$L=(L'_1)^{-1}\cdots(L'_{n-1})^{-1} \tag{2.22}$$

とおくと

$$P_{\pi_1}A=LU \tag{2.23}$$

である．L は単位下三角行列なので (2.23) は行列 $P_{\pi_1}A$ の LU 分解である．

以上の議論から LU 分解の求め方が分かる．

LU 分解

最初 π を恒等置換とする．次の手続きを $k=$

$1, 2, \ldots, n-1$ の順に繰り返して，行列 A, L および置換 π を逐次更新する．

(1) A の第 k 列の下三角成分 $a_{k,k}, \ldots, a_{n,k}$ の中からピボット $a_{i,k} \neq 0$ をひとつ選ぶ．

(2) $i \neq k$ ならば A, L の第 k 行と第 i 行を入れ替える．また π に左から互換 $(k\ i)$ を掛ける．

(3) 各 $k+1 \leq i \leq n$ に対して A の第 i 行から第 k 行の $\frac{a_{i,k}}{a_{k,k}}$ 倍を引く．

(4) L の第 k 列を

$$\begin{pmatrix} \mathbf{0}_{k-1} \\ 1 \\ \frac{a_{k+1,k}}{a_{k,k}} \\ \vdots \\ \frac{a_{n,k}}{a_{k,k}} \end{pmatrix} \tag{2.24}$$

とする．ただし $\mathbf{0}_d$ は次数 d の零ベクトルである．

最後の A を U とする．（このとき最初の A に対して $P_\pi A = LU$ である．）

2.3.3 LU 分解による解法

LU 分解は連立 1 次方程式 (2.10) の解を求めるのに応用できる．$P_\pi A = LU$ のとき (2.10) は連立方程式

$$L\boldsymbol{y} = P_\pi \boldsymbol{b}, \qquad U\boldsymbol{x} = \boldsymbol{y} \tag{2.25}$$

と同値である．

LU 分解による解法

次の手続きでベクトル $\boldsymbol{x}$ を求める．

(1) LU 分解により $P_\pi A = LU$ を満たす単位下三角行列 $L = (\ell_{i,j})$ と上三角行列 $U = (u_{i,j})$ および置換 π を求める．

(2) 方程式 $L\boldsymbol{y} = P_\pi \boldsymbol{b}$ の解 $\boldsymbol{y} = (y_i)$ を漸化式

$$y_i = b_{\pi(i)} - \sum_{j=1}^{i-1} \ell_{i,j} y_j \tag{2.26}$$

（$i = 1, 2, \ldots, n$）により求める．

(3) 方程式 $U\boldsymbol{x} = \boldsymbol{y}$ の解 $\boldsymbol{x} = (x_i)$ を漸化式

$$x_i = \frac{1}{u_{i,i}} \left(y_i - \sum_{j=i+1}^{n} u_{i,j} x_j \right) \tag{2.27}$$

（$i = n, n-1, \ldots, 1$）により求める．

課題 2 課題 1 と同じ内容を LU 分解による解法に替えて行え．

課題 3 連立 1 次方程式に対する消去法と LU 分解による解法を，数値解の精度と計算速度の観点から比較せよ．その際，理論的な計算量（加減算，乗算，除算それぞれの回数）についても考察せよ．

2.4 行列固有値問題の数値解法

行列 $A = (a_{i,j}) \in \mathbb{C}^{n \times n}$ に対して方程式

$$A\boldsymbol{x} = \lambda \boldsymbol{x} \tag{2.28}$$

を満たすスカラー $\lambda \in \mathbb{C}$ と非零ベクトル $\boldsymbol{x} = (x_i) \in \mathbb{C}^n$ があるとき，λ を A の**固有値**といい，$\boldsymbol{x}$ を（固有値 λ に対する）**固有ベクトル**という．A の固有値と固有ベクトルを数値的に求めることを考える．

A の固有値を $\lambda_1, \ldots, \lambda_n$ とおく．A が正則でない場合や，絶対値が同じ固有値が複数ある場合は議論が煩雑になるため，ここでは

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0 \tag{2.29}$$

を仮定する．特に λ_i を**第 i 固有値**といい，λ_i に対する固有ベクトルを**第 i 固有ベクトル**という．

各 $1 \leq i \leq n$ に対して $\boldsymbol{v}_i$ を第 i 固有ベクトルとする．固有値を並べた対角行列

$$\Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} \tag{2.30}$$

および固有ベクトルを並べた行列

$$V = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n) \tag{2.31}$$

に対して

$$AV = V\Lambda \tag{2.32}$$

である.(これを A の対角化という.)

数値的に求めた第 i 固有値 $\tilde{\lambda}_i$ の精度は，真値 λ_i との相対誤差

$$\frac{|\lambda_i - \tilde{\lambda}_i|}{|\lambda_i|} \tag{2.33}$$

で測る.

数値的に求めた第 i 固有ベクトル $\tilde{\boldsymbol{v}}_i$ の精度は，真の第 i 固有ベクトル $\boldsymbol{v}_i$ との相対誤差

$$\frac{\|\boldsymbol{v}_i - \tilde{\boldsymbol{v}}_i\|}{\|\boldsymbol{v}_i\|} \tag{2.34}$$

で測る．ただし固有ベクトルにはスカラー倍の自由度があるので，次の 2 条件を満たすように $\boldsymbol{v}_i, \tilde{\boldsymbol{v}}_i$ を正規化してから (2.34) を求める.

(1) $\|\boldsymbol{v}_i\| = \|\tilde{\boldsymbol{v}}_i\|$.

(2) ある $1 \le i \le n$ に対して $\boldsymbol{v}_i, \tilde{\boldsymbol{v}}_i$ の第 i 成分はともに正実数である.

注意 2.4.1 行列固有値問題の解法を実装したプログラムに誤りがないか確認するには，数値的に求めた第 i 固有値 $\tilde{\lambda}_i$ および第 i 固有ベクトル $\tilde{\boldsymbol{v}}_i$ に対して，固有方程式 (2.28) の残差ノルム

$$\|\tilde{\lambda}_i \tilde{\boldsymbol{v}}_i - A\tilde{\boldsymbol{v}}_i\| \tag{2.35}$$

を調べるのが便利である．この値は真の固有値・固有ベクトルであれば 0 になる．このため極端に大きな値になる場合はプログラムに誤りがある可能性が高い.

2.4.1 べき乗法

べき乗法は第 1 固有値 λ_1 を求めるための算法であり，第 1 固有ベクトルも同時に求めることができる．べき乗法の考え方は次の観察に基づいている.

> ベクトル $\boldsymbol{x}$ に繰り返し A を掛ける：
>
> $$\boldsymbol{x},\ A\boldsymbol{x},\ A^2\boldsymbol{x},\ A^3\boldsymbol{x},\ \ldots. \tag{2.36}$$
>
> このとき $A^k\boldsymbol{x}$ は第 1 固有ベクトルに近付いていく.

この現象が起こる仕組みを見てみよう．任意のベクトル $\boldsymbol{x} \in \mathbb{C}^n$ に対して $\boldsymbol{x} = V\boldsymbol{c}$ によりベクトル $\boldsymbol{c} = (c_i) \in \mathbb{C}^n$ を定める．このとき (2.32) より

$$A^k\boldsymbol{x} = V\Lambda^k\boldsymbol{c} = \lambda_1^k\left\{c_1\boldsymbol{v}_1 + \sum_{i=2}^{n} c_i\left(\frac{\lambda_i}{\lambda_1}\right)^k \boldsymbol{v}_i\right\} \tag{2.37}$$

である．(2.29) より $\lim_{k\to\infty}(\frac{\lambda_i}{\lambda_1})^k = 0$ ($i \ne 1$) である．従って $c_1 \ne 0$ ならば $A^k\boldsymbol{x}$ は $k \to \infty$ のとき $\boldsymbol{v}_1$ 成分が支配的になる.

$A^k\boldsymbol{x}$ を適当に正規化すれば，$\boldsymbol{v}_1$ 成分を残しつつ，$\boldsymbol{v}_1$ 以外の成分を $\boldsymbol{0}$ に収束させることができる．これがべき乗法の原理である.

> **べき乗法**
>
> 適当な初期ベクトル $\boldsymbol{x}_0$ から次の手続きを $k = 0, 1, 2, \ldots$ の順に反復してベクトル $\boldsymbol{x}_k = (x_{k;i})_i$ およびスカラー μ_k を求める.
>
> (1) $\boldsymbol{y}_k = A\boldsymbol{x}_k$ とする.
>
> (2) $\boldsymbol{x}_k$ の成分の中から $x_{k;i} \ne 0$ をひとつ（例えば絶対値最大のものを）選び $\mu_k = \dfrac{y_{k;i}}{x_{k;i}}$ とする．ただし $\boldsymbol{y}_k = (y_{k;i})_i$ である.
>
> (3) 数列 $\{\mu_k\}$ が十分収束していれば反復を終了する.
>
> (4) $\boldsymbol{x}_{k+1} = \dfrac{\boldsymbol{y}_k}{\|\boldsymbol{y}_k\|}$ とする.
>
> 反復終了時の μ_k を第 1 固有値 λ_1 として採用する．必要ならば $\boldsymbol{x}_k$ を第 1 固有ベクトルとして採用する.

べき乗法では理論上

$$\lim_{k\to\infty} \mu_k = \lambda_1 \tag{2.38}$$

である．しかし実際の計算では真の収束は期待できないため，(3) で近似的な収束判定を行っている．具体的には例えば，予め定めた閾値 $\varepsilon > 0$ に対して，μ_k の相対的な変化量

$$\frac{|\mu_k - \mu_{k-1}|}{|\mu_k|} \tag{2.39}$$

が ε より小さくなった時点で十分収束したと判定する．

べき乗法の収束の速さは $\lim_{k\to\infty}(\frac{\lambda_2}{\lambda_1})^k = 0$ の収束の速さと同じである．つまり収束率 $\frac{|\lambda_2|}{|\lambda_1|}$ の 1 次収束である．これより第 1 固有値と第 2 固有値の絶対値の比 $\frac{|\lambda_2|}{|\lambda_1|}$ が小さいほど収束は速く，1 に近いほど遅い．

単精度（`float`）で実装したべき乗法により，行列 A の第 1 固有値と第 1 固有ベクトルを求める数値実験を行った．100 次行列 A をランダムに 100 個生成し，各 A に対して第 1 固有値と第 1 固有ベクトルを数値的に求めた．ただし収束判定の閾値は $\varepsilon = 1.0 \times 10^{-6}$ とした．計算結果をまとめたものを図 2.4–2.6 に示す．

図 2.4 のグラフは，べき乗法で数値的に求めた第 1 固有値 $\tilde{\lambda}_1$ と第 1 固有ベクトル $\tilde{\boldsymbol{v}}_1$ に対する固有方程式の残差ノルム (2.35) を，各 A に対して（対数）プロットしたものである．残差ノルムは概ね小さいので，実装したプログラムは正しく動作していると思われる．

図 2.5 の 2 つのグラフは，$\tilde{\lambda}_1$ の相対誤差 (2.33) と $\tilde{\boldsymbol{v}}_1$ の相対誤差 (2.34) を同様にプロットしたものである．（点線はそれぞれの平均値である．）た

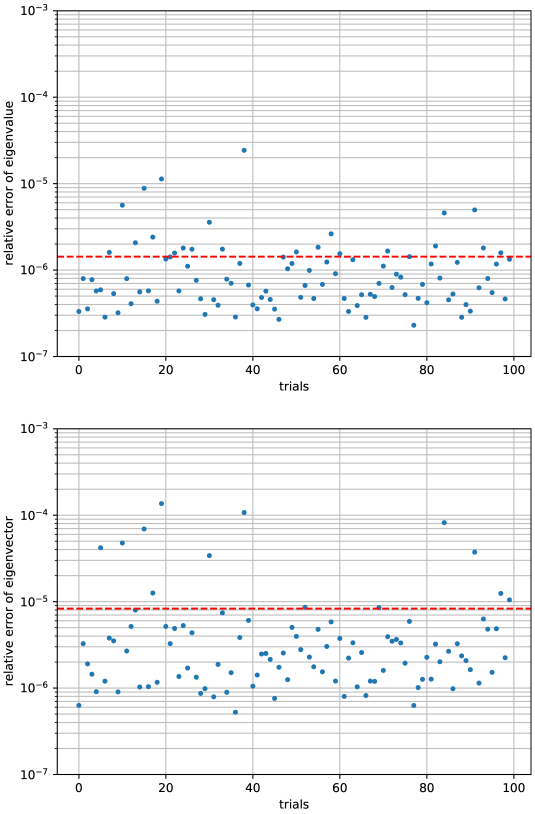

図 2.5: べき乗法による第 1 固有値の相対誤差（上）と第 1 固有ベクトルの相対誤差（下）．（点線は平均値．）

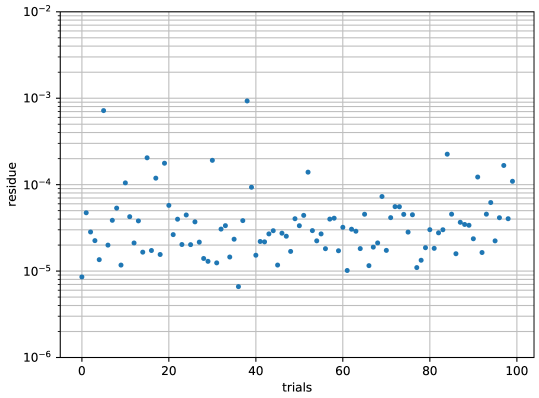

図 2.4: べき乗法による残差ノルム.

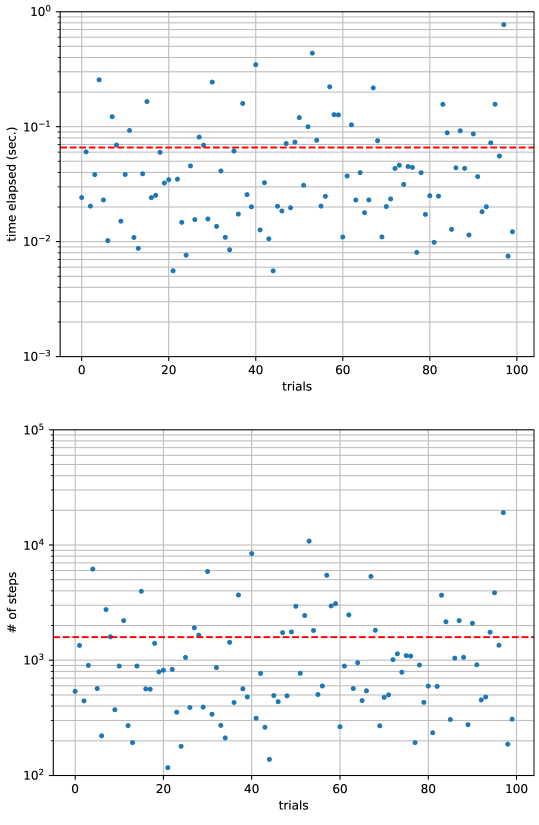

図 2.6: べき乗法による計算時間（上）と反復回数（下）.（点線は平均値.）

だし不明な真の固有値・固有ベクトルの代わりに，倍精度のべき乗法で数値的に求めた第 1 固有値 λ_1 と第 1 固有ベクトル $\boldsymbol{v}_1$ を用いている.

$\tilde{\lambda}_1$ の相対誤差は，極端に大きいものを除けば 10^{-6} 程度である．これより数値的に求めた第 1 固有値には，概ね 10 進 6 桁程度の精度があると考えられる.（収束判定の閾値を 10^{-6} と設定したのだから，これは自然な結果である.）$\tilde{\boldsymbol{v}}_1$ の相対誤差は，極端に大きいものを除けば 10^{-6}〜10^{-5} 程度である．これより数値的に求めた第 1 固有ベクトルには，概ね 10 進 5 桁程度の精度があると考えられる.

図 2.6 の 2 つのグラフは，収束までに要した計算時間と反復回数を同様にプロットしたものである.（点線はそれぞれの平均値である.）2 つのグラフはほとんど同じ形をしている．これは計算時間が反復回数に比例するためである.（べき乗法の 1 ステップに含まれる四則演算の回数は次数 n のみで決まる.）

計算時間は行列 A により大きくばらついている．数 100 回の反復ですぐに収束する例もあれば，収束までに 1 万回以上の反復を要した例もある．これは A の固有値の配置の違いにより，べき乗法の収束率 $\frac{|\lambda_2|}{|\lambda_1|}$ が変化したためと考えられる.

課題 4 べき乗法により，行列 A の第 1 固有値と第 1 固有ベクトルを求める数値実験を行う．$n = 50, 100, 200, 400$ のそれぞれに対して，n 次行列 A をランダムに 100 個生成し，各 A の第 1 固有値と第 1 固有ベクトルを数値的に求めよ.

(1) 各 n に対して，数値的に求めた第 1 固有値 $\tilde{\lambda}_1$ と第 1 固有ベクトル $\tilde{\boldsymbol{v}}_1$ に対する固有方程式の残差ノルム (2.35)，$\tilde{\lambda}_1$ の相対誤差 (2.33)，$\tilde{\boldsymbol{v}}_1$ の相対誤差 (2.34)，計算時間，収束までに要した反復回数をそれぞれプロットせよ.

(2) $\tilde{\lambda}_1, \tilde{\boldsymbol{v}}_1$ の精度と次数 n の関係について考察せよ.

(3) 計算時間と次数 n の関係について考察せよ.

2.4.2 逆反復法

べき乗法の考え方は，実は第 1 固有値・固有ベクトル以外にも適用することができる．スカラー $\sigma \in \mathbb{C}$ は A の固有値でないとする．このとき行列 $(A-\sigma I)^{-1}$ の固有値は

$$\frac{1}{\lambda_1-\sigma}, \quad \dots, \quad \frac{1}{\lambda_n-\sigma} \tag{2.40}$$

である．この中で絶対値最大（つまり $(A-\sigma I)^{-1}$ の第 1 固有値）を実現するのは σ に最も近い λ_i である．これより $(A-\sigma I)^{-1}$ に対してべき乗法を適用すると，A の固有値の中で σ に最も近い λ_i が $\frac{1}{\lambda_i-\sigma}$ の形で求まる．この原理を用いたのが**逆反復法**である．

逆反復法

適当な初期ベクトル $\boldsymbol{x}_0$ から次の手続きを $k=0,1,2,\dots$ の順に反復してベクトル $\boldsymbol{x}_k=(x_{k;i})_i$ およびスカラー μ_k を求める．

(1) 連立 1 次方程式

$$(A-\sigma I)\boldsymbol{y}_k = \boldsymbol{x}_k \tag{2.41}$$

の解 $\boldsymbol{y}_k$ を求める．

(2) $\boldsymbol{x}_k$ の成分の中から $x_{k;i}\neq 0$ をひとつ（例えば絶対値最大のものを）選び $\mu_k = \dfrac{y_{k;i}}{x_{k;i}}$ とする．ただし $\boldsymbol{y}_k=(y_{k;i})_i$ である．

(3) 数列 $\{\mu_k\}$ が十分収束していれば反復を終了する．

(4) $\boldsymbol{x}_{k+1} = \dfrac{\boldsymbol{y}_k}{\|\boldsymbol{y}_k\|}$ とする．

反復終了時の $\frac{1}{\mu_k}+\sigma$ を σ に最も近い固有値 λ_i として採用する．必要ならば $\boldsymbol{x}_k$ を λ_i に対する固有ベクトルとして採用する．

逆反復法では，べき乗法での (1) に当たる

$$\boldsymbol{y}_k = (A-\sigma I)^{-1}\boldsymbol{x}_{k-1} \tag{2.42}$$

を計算する代わりに同値な連立 1 次方程式 (2.41) を解く．その計算には LU 分解を用いるのが効率的である．（係数行列 $A-\sigma I$ は不変なので，LU 分解の計算は最初の 1 回だけでよい！）なお逆反復法の収束判定はべき乗法と同様である．

逆反復法の収束の速さは収束率 $|\frac{\lambda_i-\sigma}{\lambda_j-\sigma}|$ の 1 次収束である．ただし λ_i,λ_j はそれぞれ σ に最も近い固有値と 2 番目に近い固有値である．特に σ が λ_i に近いほど，収束率が 0 に近付くため収束は速くなると期待される．逆反復法の計算途中で σ をより λ_i に近いものに取り替えたらどうなるか考えてみよ．

課題 5 べき乗法と逆反復法を組み合わせて，第 1 固有値と第 1 固有ベクトルを数値的に求めるための算法として，べき乗法より高速なものをつくれ．つくった算法とべき乗法を精度と速度の面から比較せよ．

2.4.3 行列の QR 分解

行列 A をユニタリ行列 Q と上三角行列 R を用いて

$$A = QR \tag{2.43}$$

の形に表すことを QR **分解**という．ただし Q がユニタリ行列であるとは $Q^*Q=I$ が成り立つことをいう．

QR 分解はグラム・シュミット（の直交化）法で求めることができる．$A=(\boldsymbol{a}_1,\dots,\boldsymbol{a}_n)$ の列ベクトルから漸化式

$$\boldsymbol{b}_j = \boldsymbol{a}_j - \sum_{i=1}^{j-1}(\boldsymbol{q}_i,\boldsymbol{a}_j)\boldsymbol{q}_i, \tag{2.44a}$$

$$\boldsymbol{q}_j = \frac{\boldsymbol{b}_j}{\|\boldsymbol{b}_j\|_2} \tag{2.44b}$$

$(j=1,\dots,n)$ によりベクトル $\boldsymbol{q}_1,\dots,\boldsymbol{q}_n$ をつくる．このとき $\boldsymbol{q}_1,\dots,\boldsymbol{q}_n$ は $(\boldsymbol{q}_i,\boldsymbol{q}_j)=\delta_{i,j}$ を満たすので，行列

$$Q = (\boldsymbol{q}_1,\dots,\boldsymbol{q}_n) \tag{2.45}$$

はユニタリ行列である．また (2.44) より $r_{i,j}=(\boldsymbol{q}_i,\boldsymbol{a}_j)$ $(i<j)$ および $r_{j,j}=\|\boldsymbol{b}_j\|_2$ とおくと，上

三角行列

$$R = \begin{pmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ 0 & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & r_{n,n} \end{pmatrix} \tag{2.46}$$

は (2.43) を満たす.

グラム・シュミット法の漸化式 (2.44) は誤差に弱く，数値計算では行列 Q の直交性が悪くなる（つまり Q^*Q と単位行列 I の差が小さくならない）傾向にある．この弱点に対処したのが**修正グラム・シュミット法**である.

修正グラム・シュミット法

次の手続きを $j = 1, \ldots, n$ の順に繰り返してベクトル $\boldsymbol{q}_j$（$1 \le j \le n$）とスカラー $r_{i,j}$（$1 \le i \le j \le n$）を求める.

(1) ベクトル $\boldsymbol{b}_{1,j} = \boldsymbol{a}_j$ から漸化式

$$r_{i,j} = (\boldsymbol{q}_i, \boldsymbol{b}_{i,j}), \tag{2.47a}$$
$$\boldsymbol{b}_{i+1,j} = \boldsymbol{b}_{i,j} - r_{i,j}\boldsymbol{q}_i \tag{2.47b}$$

（$i = 1, \ldots, j-1$）により $r_{i,j}$（$1 \le i < j$）とベクトル $\boldsymbol{b}_{i,j}$（$1 \le i \le j$）を求める.

(2) $r_{j,j} = \|\boldsymbol{b}_{j,j}\|_2$ および $\boldsymbol{q}_j = \dfrac{\boldsymbol{b}_{j,j}}{r_{j,j}}$ とする.

（このとき $Q = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_n)$ はユニタリ行列であり，上三角行列

$$R = \begin{pmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,n} \\ 0 & r_{2,2} & \cdots & r_{2,n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & r_{n,n} \end{pmatrix} \tag{2.48}$$

とともに $A = QR$ を満たす.）

修正グラム・シュミット法は，数学的にはグラム・シュミット法と同値である．しかしペア $\boldsymbol{q}_i, \boldsymbol{q}_j$（$i < j$）の直交性の悪さが $\boldsymbol{q}_i$ 以外のペアに影響しないように漸化式が改良されている.

注意 2.4.2 QR 分解を実装したプログラムに誤りがないか確認するには，行列ノルム

$$\|A - QR\|, \qquad \|I - Q^*Q\| \tag{2.49}$$

を調べるのが便利である.

2.4.4 QR 法

QR 法はすべての固有値を同時に求めることができる算法である．QR 法の考え方は次の観察に基づいている.

同時反復法

適当な初期行列 X_0 から次の手続きを $k = 1, 2, 3, \ldots$ の順に反復して行列 X_k を求める.

(1) $Y_k = AX_k$ とする.

(2) Y_k を QR 分解する：$Y_k = Q_k R_k$.

(3) $X_{k+1} = Q_k$ とする.

このとき行列 $V^{-1}X_k$ は $k \to \infty$ のとき上三角行列に近付く.（つまり X_k の第 j 列ベクトルは $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_j$ のみの 1 次結合で表せるようになっていく.）

この現象が起こる仕組みは，実はべき乗法の原理とよく似ている.（詳細は [1] を見よ.）特に X_k の第 1 列のふるまいはべき乗法のベクトル $\boldsymbol{x}_k$ と全く同じである.

QR 法

初期行列 $A_0 = A$ から次の手続きを $k = 0, 1, 2, \ldots$ の順に反復して行列 A_k を求める.

(1) A_k の狭義下三角部分が十分 0 に収束していれば反復を終了する.

(2) A_k を QR 分解する：$A_k = Q_k R_k$.

(3) $A_{k+1} = R_k Q_k$ とする.

反復終了時の A_k の対角成分を，上から順に固有値 $\lambda_1, \ldots, \lambda_n$ として採用する.

QR 法の反復は

$$A_{k+1} = Q_k^* A_k Q_k \tag{2.50}$$

と表せるので，数学的には（ユニタリ行列による）相似変形の繰り返しである．特に各 A_k は初期行列 $A_0 = A$ と同じ固有値を持つ．従って A_k が $k \to \infty$ のとき上三角行列に近付けば，その対角成分は A の固有値に収束する．（三角行列の固有値はその対角成分である．）

A_k が上三角行列に近付く仕組みを見る．QR 法において

$$X_k = Q_0 Q_1 \cdots Q_{k-1}, \qquad Y_k = AX_k \tag{2.51}$$

とおくと，X_k はユニタリ行列であり

$$Y_k = X_{k+1} R_k \tag{2.52}$$

が成り立つ．これより QR 法は同時反復法を初期行列 $X_0 = I$ から実行するのと同値である．一方 (2.50) より

$$A_k = X_k^* A X_k \tag{2.53}$$

である．さらに (2.32) より

$$A_k = (V^{-1}X_k)^{-1} \Lambda V^{-1} X_k \tag{2.54}$$

である．X_k は同時反復法に従うので $V^{-1}X_k$（および逆行列 $(V^{-1}X_k)^{-1}$）は $k \to \infty$ のとき上三角行列に近付く．従って A_k も上三角行列に近付く．以上が QR 法の原理である．

QR 法では $A_k = (a_{k;i,j})_{i,j}$ の全ての狭義下三角成分 $a_{k;i,j}$（$i > j$）が十分 0 に収束した時点で反復を終了する．具体的には例えば，予め定めた閾値 $\varepsilon > 0$ に対して，同じ行の対角成分と比較した

$$\frac{|a_{k;i,j}|}{|a_{k;i,i}|} \tag{2.55}$$

が ε より小さくなった時点で，第 (i,j) 成分は十分 0 に収束したと見なす．

QR 法では A_k の狭義下三角成分 $a_{k;i,j}$ は，収束率

$$\rho_{i,j} = \frac{|\lambda_i|}{|\lambda_j|} \tag{2.56}$$

で（0 に）1 次収束する．これより QR 法の収束の速さは，収束率

$$\max_{1 \le i < n} \frac{|\lambda_{i+1}|}{|\lambda_i|} \tag{2.57}$$

の 1 次収束である．このため絶対値の近い固有値のペア λ_i, λ_{i+1} が 1 つでもあると収束は遅い．

注意 2.4.3 固有ベクトルまで求めるには，QR 法で数値的に求めた固有値 λ_i を σ として逆反復法を適用すればよい．

課題 6 QR 法により，行列 A の全ての固有値を求める数値実験を行う．$n = 10, 20, 40, 80$ のそれぞれに対して，n 次行列 A をランダムに 100 個生成し，各 A の固有値 $\lambda_1, \ldots, \lambda_n$ を数値的に求めよ．

(1) 各 n に対して，数値的に求めた固有値 $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ に対する固有方程式の残差ノルムの最大値

$$\max_{1 \le i \le n} \|(A - \tilde{\lambda}_i I)\boldsymbol{v}_i\|, \tag{2.58}$$

相対誤差の最大値

$$\max_{1 \le i \le n} \frac{|\lambda_i - \tilde{\lambda}_i|}{|\lambda_i|}, \tag{2.59}$$

計算時間，収束までに要した反復回数をそれぞれプロットせよ．ただし真の固有値 λ_i と固有ベクトル $\boldsymbol{v}_i$ は不明なので，MATLAB や Python における数値計算パッケージで得られるものを真の固有値および固有ベクトルとして用いよ．

(2) $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ の精度と次数 n の関係について考察せよ．

(3) 計算時間と次数 n の関係について考察せよ．

2.4.5 QR 法の高速化

シフト

行列 A の対角成分から定数 σ を引く：

$$A' = A - \sigma I. \tag{2.60}$$

この操作を（**原点**）**シフト**という．シフト前の A の固有値 $\lambda_1, \ldots, \lambda_n$ に対して，シフト後の A' の固有値は $\lambda_1 - \sigma, \ldots, \lambda_n - \sigma$ である．

シフトの考え方は QR 法の収束を加速させるのに役立つ．QR 法の計算途中で，例えば行列 A_k を求めた時点で，第 n 固有値 λ_n の近似値 σ が見つかったとする．(例えば何回か反復した後の A_k の第 (n,n) 成分 $a_{k;n,n}$ など.）このとき A_k を σ だけシフトして（つまり A_k を $A_k - \sigma I$ に取り替えて）QR 法の計算を続ける．

σ が λ_n に十分近ければ，A_k の狭義下三角成分のシフト後の収束率

$$\rho'_{i,j} = \frac{|\lambda_i - \sigma|}{|\lambda_j - \sigma|} \tag{2.61}$$

はシフト前の $\rho_{i,j}$ より小さい．特に A_k の第 n 行成分の収束率 $\rho'_{n,j}$ はとても 0 に近い．このためシフト後の A_k の第 n 行成分は，QR 法の反復計算によりとても速く収束すると期待される．

減次

QR 法の反復計算が進んで，行列 $A_k = (a_{k;i,j})_{i,j}$ の第 n 行成分が（シフトなどの効果もあって）ほとんど収束したとする．つまり収束判定の閾値 $\varepsilon > 0$ に対して

$$\frac{|a_{k;n,j}|}{|a_{k;n,n}|} < \varepsilon \qquad (1 \le j < n) \tag{2.62}$$

が満たされたとする．このとき A_k は行列

$$\begin{pmatrix} & & & * \\ & B_k & & \vdots \\ & & & * \\ 0 & \cdots & 0 & a_{k;n,n} \end{pmatrix} \tag{2.63}$$

に近い．ただし B_k は A_k から第 n 行と第 n 列を除いた小行列である．特に A_k の固有値は B_k の固有値と $a_{k;n,n}$ により近似される．

そこで次のような戦略が考えられる．

減次戦略

(2.62) が満たされた時点で $a_{k;n,n}$ を固有値の 1 つとして採用し，残りの固有値は小行列 B_k から求める．

このように途中で行列の次数を下げて計算量を抑える工夫を**減次**という．

課題 7 シフトと減次を用いて QR 法を高速化せよ．つくった算法と元々の QR 法を精度と速度の面から比較せよ．

参考文献

[1] 杉原正顯・室田一雄，線形計算の数理，岩波書店，2009.

[2] 山本有作・曽我部知広，計算科学のための基本数理アルゴリズム，張紹良編，共立出版，2019.

[3] LAPACK, http://www.netlib.org/lapack/.

[上岡修平]