第 7 章

連立一次方程式の求解

本章では Scilab を用いて連立一次方程式 (線型方程式) を解く手法を学んでいく。連立一次方程式の求解は線型代数を学ぶ一番の目的であると言える。何故なら，後述する固有値問題や各種非線型問題においては連立一次方程式が各所に現れるからである。前章で学んだ Scilab におけるベクトル・行列の扱い方を復習しながら，現実的な規模の連立一次方程式を解く手法を学んで頂きたい。

7.1 連立一次方程式とその解

本書で扱う連立一次方程式は以下の (7.1) 式の形式に限定する。即ち，n 次正方行列 $A \in \mathbb{C}^{n\times n}$，$n$ 次元ベクトル $\mathbf{b} \in \mathbb{C}^n$ が与えられている時，

$$A\mathbf{x} = \mathbf{b} \tag{7.1}$$

を満足する n 次元ベクトル $\mathbf{x} \in \mathbb{C}^n$ を求める問題となる。ここで A を係数行列，$\mathbf{b}$ を定数ベクトル，$\mathbf{x}$ を (7.1) の解 (ベクトル) と呼ぶ。

まず，連立一次方程式が解を持つかどうか，持つ場合は唯一解か複数解か，ということを整理しておこう。

既に述べたように，係数行列 A は正規行列か非正規行列かの二つに分類できる。もし A が正規行列であれば，逆行列 A^{-1} が一つ定まるので，解 $\mathbf{x}$ は

$$\mathbf{x} = A^{-1}\mathbf{b}$$

のように唯一に定まる。つまり $\mathrm{rank}(A) = n$ の場合は必ず唯一解が存在する。

A が非正規の場合は，解が存在しない場合と，無限に解が存在する場合に分類される。

解が存在しない場合とは，例えば $A = O$ である時，$\mathbf{b} \neq 0$ というケースである，この場合，(7.1) 式を満足する解 $\mathbf{x}$ は存在しないことになる。つまり $0 \leq \mathrm{rank}(A) < n$ の場合，解が存在しないことがあり得る。

解が無数に存在する場合とは，例えば $A = O$ の場合，$\mathbf{b} = 0$ であれば，全ての n 次元ベクトルが解となり得る，というケースである。また $A \neq 0$ であっても，$\mathrm{rank}(A) < n$ であれば，無数の解が存在しうる。

以上をまとめると，連立一次方程式 (7.1) の解は次の 3 つのケースに分類できることになる。

CASE 1　$\mathrm{rank}(A) = n$ の時，唯一解 $\mathbf{x} = A^{-1}\mathbf{b}$ が存在。
　$0 \leq \mathrm{rank}(A) < n$ の時
　CASE 2　解が存在しない。
　CASE 3　解が無数に存在。

CASE 2, 3 の場合はランク落ちの係数行列，あるいは，ランク落ちの連立一次方程式と呼ぶこともある。実用的に重要なのは CASE 3 の場合で，これについては次章の固有値問題で詳しく扱う。

なお，本章では特に断らない限り，CASE 1 に当てはまるものだけを扱うことにする。

問題 7.1

$A \in \mathbb{R}^{2\times 2}$, $\mathbf{b} \in \mathbb{R}^2$ が次のように与えられる場合，連立一次方程式 $A\mathbf{x} = \mathbf{b}$ は CASE 1〜CASE 3 のどれに当てはまるかを答えよ。また，CASE 1 の時は唯一解を，CASE 3 の時は解を二つ以上を求めよ。

1. $A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$
2. $A = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} -1 \\ 3 \end{bmatrix}$
3. $A = \begin{bmatrix} -1 & 2 \\ 1 & -2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

7.2　テスト用 Scilab スクリプト (linear_eq.sce) の作成

ここでは Scilab を用いて，真の解 $\mathbf{x}$ があらかじめ分かっている連立一次方程式を導出し，近似解 $\widetilde{\mathbf{x}}$ とそのノルム相対誤差 $rE(\widetilde{\mathbf{x}})$ と成分ごとの相対誤差の最大値を出力するスクリプト，linear_eq.sce を作成していく。Scilab の機能の詳細は既に前章で示したので，不明な点はそちらを参照されたい。

7.2.1　次元数，係数行列，真の解，定数ベクトルの導出

ここでは連立一次方程式 (7.1) における $A \in \mathbb{R}^{n\times n}$, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$ を

$$A = \begin{bmatrix} n & n-1 & \cdots & 1 \\ n-1 & n-1 & \cdots & 1 \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} = [n - \max(i,j) + 1]_{i,j=1}^{n},\ \mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ \vdots \\ n \end{bmatrix},\ \mathbf{b} = A\mathbf{x}$$

として与える部分を作っていく。真の解は true_x という変数に格納されている。

次元数 n は標準入力から読み込み，自動的に $A, \mathbf{x}$ を与え, $\mathbf{b}$ を自動計算して表示する。次元数が大きい場合は表示を抑制するため，disp 関数の部分をコメントにしておくと良い。なお下記の左の数字「xx:」は行番号なのでスクリプトには記述しないこと！

```
1: // 次元数: n
2: disp("次元数 n を入力してください：");
```

```
 3: n = scanf("%d");
 4: mprintf("次元数 (n) = %d", n);
 5:
 6: // 係数行列 A
 7: a = [];
 8: for i = 1:n
 9:     for j = 1:n
10:         a(i, j) = n - max(i, j) + 1;
11:     end;
12: end;
13: disp("係数行列 a = "); disp(a);
14:
15: // 真の解 true_x
16: true_x = [];
17: for i = 1:n
18:     true_x(i) = i;
19: end;
20: disp("真の解ベクトル true_x = "); disp(true_x);
21:
22: // 定数ベクトル b
23: b = a * true_x;
24: disp("定数ベクトル b = "); disp(b);
```

$n = 5$ とした場合の実行結果は以下のようになる。

```
 次元数 n を入力してください：
-->5
次元数 (n) = 5
 係数行列 a =

    5.    4.    3.    2.    1.
    4.    4.    3.    2.    1.
    3.    3.    3.    2.    1.
    2.    2.    2.    2.    1.
    1.    1.    1.    1.    1.

 真の解ベクトル true_x =

    1.
```

```
    2.
    3.
    4.
    5.
```

定数ベクトル b =

```
    35.
    34.
    31.
    25.
    15.
```

7.2.2　近似解の導出と計算時間の計測

上記のスクリプトに，下記のように近似解を導出し，計算時間を計測する部分を追加する。計算時間の計測は，計算開始直前に tic 関数 (チック) を入れ，計算終了直後に toc 関数 (タック）を入れ，toc 関数の返り値で求められる。

下記の場合，近似解は x という変数に格納され，
という演算子を用いて計算されている。

```
26: // 計算時間計測開始
27: disp("準備完了！  START!")
28: tic();
29:
30: // a
31: x = a \ b;
32:
33: // 計算時間計測終了
34: time_invsolve = toc();
35: mprintf( "逆行列 * 係数行列 (秒) = %f", time_invsolve);
36:
37: // 解ベクトル x 出力
38: disp("x = A^(-1)*b = "); disp(x);
```

上記スクリプト部分に対応する実行結果は次のようになる。

```
 準備完了！  START!
逆行列 * 係数行列 (秒) = 0.000000
```

```
 x = A^(-1)*b =


    1.
    2.
    3.
    4.
    5.
```

次元数が小さい場合は計測不可能なほど短時間で計算されていることが分かる。係数行列，真の解，定数ベクトル，近似解の出力をコメントにして抑制し，$n = 100$ とすると

```
次元数 n を入力してください：
-->100
次元数（n） = 100
 準備完了！  START!
逆行列 * 係数行列（秒） = 0.040000
```

となり，計算時間を要するようになることが分かる。

問題 7.2
linear_eq.sce スクリプトを用いて，$n = 50, 100, 200, 500, 1000$ の時の計算時間をそれぞれ求めよ。

7.2.3 誤差の計測による検算

これで連立一次方程式を解くための一通りの作業は終わったが，実際に計算した近似解が正しいものかどうかのチェックを行うため，残差ベクトルと相対誤差の計算を行う習慣をつけておくと良い。

■残差ベクトル 残差 (ベクトル)(residual) とは，次のようにして計算されるベクトル $\mathbf{r}$ である。

$$\mathbf{r} = \mathbf{b} - A\widetilde{\mathbf{x}} \tag{7.2}$$

近似解 $\widetilde{\mathbf{x}}$ が真の解 $\mathbf{x}$ と一致していれば $\mathbf{r} = 0$ となることが期待できる。しかし，近似解に誤差 $\mathbf{e}(\neq 0)$ が混入し，$\widetilde{\mathbf{x}} = \mathbf{x} + \mathbf{e}$ であれば

$$\mathbf{r} = \mathbf{b} - A(\mathbf{x} + \mathbf{e}) = -A\mathbf{e}$$

となり，ゼロにはならない。

この残差を求め，相対残差ノルム $\|\mathbf{r}\|/\|\mathbf{b}\|$ を求めるスクリプトを追加してみよう。

```
40: // 検算（1）残差ベクトルの計算
41: r = b - a * x;
42:
43: disp("||b - a * x||_2 / ||b||_2 = ");
44: if(norm(b) ~= 0)
```

```
45:     disp(norm(r) / norm(b));
46: else
47:     disp(norm(r));
48: end;
```

$n = 5$ の場合は次のように表示され，相対残差ノルムはゼロになっている。

```
 次元数 n を入力してください：
-->5
次元数 (n) = 5

(省略)

 ||b - a * x||_2 / ||b||_2 =

    0.
```

次元数を増やし，$n = 100$ とすると，次のように 4.5×10^{-13} にまで拡大する。つまり近似解の誤差が大きくなっていることが分かる。

```
 次元数 n を入力してください：
-->100
次元数 (n) = 100

(省略)

 ||b - a * x||_2 / ||b||_2 =

    4.577D-13
```

残差の計算は，真の解 **x** が未知であっても可能なので，実用上の問題ではよく検算のために使用される。

問題 7.3
linear_eq.sce スクリプトを用いて，$n = 50, 100, 200, 500, 1000$ の時の相対残差ノルムをそれぞれ求めよ。

■ノルム相対誤差と，要素ごとの相対誤差とその最大値の計算　真の解が既知の場合は，ノルム相対誤差と要素ごとの相対誤差の計算が次のように行える。要素ごとの相対誤差はその最大値のみ表示するようにしてある。

```
50: // 検算 (2) ノルム単位の相対誤差の計算
51: disp("||x - true_x||_2 / ||true_x||_2 = ");
```

```
52: if(norm(true_x) ~= 0)
53:     disp(norm(x - true_x) / norm(true_x));
54: else
55:     disp(norm(x - true_x));
56: end;
57:
58: // 検算 (3) 成分単位の相対誤差の計算
59: disp("max(|x_i - true_x_i| / |true_i|) = ")
60: res_err = [];
61: for i = 1:n
62:     if(true_x(i) ~= 0)
63:         res_err(i) = abs(x(i) - true_x(i)) / abs(true_x(i));
64:     else
65:         res_err(i) = abs(x(i) - true_x(i))
66:     end;
67: end;
68: disp(max(res_err));
```

$n = 5$ の場合は次のようになる。

```
 次元数 n を入力してください：
-->5
次元数 (n) = 5

(省略)

 ||x - true_x||_2 / ||true_x||_2 =

    1.051D-15

 max(|x_i - true_x_i| / |true_i|) =

    2.442D-15
```

$n = 100$ の場合は次のようになり，やはり相対誤差が増大していることが分かる。

```
 次元数 n を入力してください：
-->100
次元数 (n) = 100
```

（省略）

```
 ||x - true_x||_2 / ||true_x||_2 =

    4.215D-13

 max(|x_i - true_x_i| / |true_i|) =

    3.675D-12
```

成分ごとの相対誤差の最大値を見る限り，有効桁数が 12 桁程度まで落ちていることになる。

問題 7.4
linear_eq.sce スクリプトを用いて，$n = 50, 100, 200, 500, 1000$ の時のノルム相対誤差，成分ごとの相対誤差の最大値をそれぞれ求めよ。

7.3　条件数と誤差の関係

連立一次方程式 (7.1) を実際にコンピュータで計算しようとすると，一般的には行列，ベクトル成分は丸められて誤差が混入していると考えられる。また，解法の過程でも丸め誤差が混入するのが普通である。よって，これらを全て，行列あるいは定数ベクトルに忍び込んだ初期誤差として解釈することにすれば，解こうとする (7.1) は

$$\widetilde{A}\,\widetilde{\mathbf{x}} = \widetilde{\mathbf{b}}$$

という形に化けているということが出来る。では最終的に得られる $\widetilde{\mathbf{x}}$ に含まれる誤差 $E(\widetilde{\mathbf{x}})$ はどうなっているのだろうか。

それを解析する基になる補題，定理を示すことにする。

補題 7.1 (定数項 b に誤差がある場合)
連立一次方程式

$$A(\mathbf{x} + E(\widetilde{\mathbf{x}})) = \mathbf{b} + E(\widetilde{\mathbf{b}})$$

において

$$rE(\widetilde{\mathbf{x}}) \le \kappa(A) \cdot rE(\widetilde{\mathbf{b}})$$

である。

(証明)　$E(\widetilde{\mathbf{x}}) = A^{-1}E(\widetilde{\mathbf{b}})$ から

$$\|E(\widetilde{\mathbf{x}})\| \le \|A^{-1}\|\,\|E(\widetilde{\mathbf{b}})\|$$

を得る。更に，$\|\mathbf{b}\| \le \|A\|\,\|\mathbf{x}\|$ から与式を得る。　(証明終)

補題 7.2 (係数行列 A に誤差がある場合)

$$(A + E(\widetilde{A}))(\mathbf{x} + E(\widetilde{\mathbf{x}})) = \mathbf{b}$$

において

$$\frac{\|E(\widetilde{\mathbf{x}})\|}{\|\mathbf{x} + E(\widetilde{\mathbf{x}})\|} \leq \kappa(A) \cdot rE(\widetilde{A})$$

である。

(証明)

$$\begin{aligned}\mathbf{x} &= A^{-1}\mathbf{b} \\ &= A^{-1}(A + E(\widetilde{A}))(\mathbf{x} + E(\widetilde{\mathbf{x}})) \\ &= \mathbf{x} + E(\widetilde{\mathbf{x}}) + A^{-1}E(\widetilde{A})(\mathbf{x} + E(\widetilde{\mathbf{x}}))\end{aligned}$$

から

$$-E(\widetilde{\mathbf{x}}) = A^{-1}E(\widetilde{A})(\mathbf{x} + E(\widetilde{\mathbf{x}}))。$$

従って,

$$\begin{aligned}\|E(\widetilde{\mathbf{x}})\| &\leq \|A^{-1}\|\,\|E(\widetilde{A})\| \cdot \|\mathbf{x} + E(\widetilde{\mathbf{x}})\| \\ &= \|A^{-1}\|\,\|A^{-1}\| \cdot \frac{\|E(\widetilde{A})\|}{\|A\|} \cdot \|\mathbf{x} + E(\widetilde{\mathbf{x}})\|\end{aligned}$$

より，与式を得る。 (証明終)

定理 7.1 (係数行列，定数項共に誤差を含んでいるとき)

$$(A + E(\widetilde{A}))(\mathbf{x} + E(\widetilde{\mathbf{x}})) = \mathbf{b} + E(\widetilde{\mathbf{b}})$$

なるとき $\|A^{-1}E(\widetilde{A})\| < 1$ ならば,

$$rE(\widetilde{\mathbf{x}}) \leq \frac{\kappa(A)}{1 - \|A^{-1}E(\widetilde{A})\|}\left(rE(\widetilde{\mathbf{b}}) + rE(\widetilde{A})\right)$$

である。

(証明) $I + A^{-1}E(\widetilde{A})$ の固有値は $1 + \lambda(A^{-1}E(\widetilde{A}))$ だから，$\lambda(A^{-1}E(\widetilde{A}))$ によらず，正則になる。
従って,

$$(I + A^{-1}E(\widetilde{A}))^{-1} = I - A^{-1}E(\widetilde{A})(I + A^{-1}E(\widetilde{A}))^{-1}$$

が成立するから,

$$\|(I + A^{-1}E(\widetilde{A}))^{-1}\| \leq 1 + \|A^{-1}E(\widetilde{A})\|\,\|(I + A^{-1}E(\widetilde{A}))^{-1}\|$$

よって,

$$\|(I + A^{-1}E(\widetilde{A}))^{-1}\|\,(1 - \|A^{-1}E(\widetilde{A})\|) \leq 1$$

から

$$\|(I + A^{-1}E(\widetilde{A}))^{-1}\| \leq \frac{1}{1 - \|A^{-1}E(\widetilde{A})\|}$$

を得る。

ここで $(A + E(\widetilde{A}))(\mathbf{x} + E(\widetilde{\mathbf{x}})) = \mathbf{b} + E(\widetilde{\mathbf{b}})$ と $A\mathbf{x} = \mathbf{b}$ より

$$E(\widetilde{A})\mathbf{x} + (A + E(\widetilde{A}))E(\widetilde{\mathbf{x}}) = E(\widetilde{\mathbf{b}})。$$

これに，左から A^{-1} をかけると

$$A^{-1}E(\widetilde{A})\mathbf{x} + (I + A^{-1}E(\widetilde{A}))E(\widetilde{\mathbf{x}}) = A^{-1}E(\widetilde{\mathbf{b}})$$

これを $E(\widetilde{\mathbf{x}})$ について解くと，

$$E(\widetilde{\mathbf{x}}) = (I + A^{-1}E(\widetilde{A}))^{-1}A^{-1}(E(\widetilde{A})\mathbf{x} - E(\widetilde{\mathbf{b}}))。$$

よって，

$$\begin{aligned}\frac{\|E(\widetilde{\mathbf{x}})\|}{\|\mathbf{x}\|} &\leq \|(I + A^{-1}E(\widetilde{A}))^{-1}\|\|A^{-1}\|\left(\|E(\widetilde{A})\| + \frac{\|E(\widetilde{\mathbf{b}})\|}{\|\mathbf{x}\|}\right) \\ &\leq \frac{\|A^{-1}\|}{1 - \|A^{-1}E(\widetilde{A})\|}\left(\|E(\widetilde{A})\| + \frac{\|E(\widetilde{\mathbf{b}})\|}{\|\mathbf{b}\|}\right)。\end{aligned}$$

ここで $\|\mathbf{x}\| \leq \|A\|\|\mathbf{b}\|$ より与式を得る。　(証明終)

この不等式は $\|A^{-1}\| \geq \frac{1}{\|A\|}$ を用いて，

$$rE(\widetilde{\mathbf{x}}) \leq \frac{\kappa(A)}{1 - \dfrac{\|E(\widetilde{A})\|}{\|A\|}} \cdot \left(rE(\widetilde{A}) + rE(\widetilde{\mathbf{b}})\right) \tag{7.3}$$

という形にしたものが主に用いられる。

今まで見てきた不等式がいずれも条件数をファクターとして持つ不等式になっていることが分かった。以上をまとめると

1. 定数項のみに誤差がある場合 —
$$rE(\widetilde{\mathbf{x}}) \leq \kappa(A) \cdot rE(\widetilde{\mathbf{b}})$$
2. 係数行列のみに誤差がある場合 —
$$\frac{\|E(\widetilde{\mathbf{x}})\|}{\|\mathbf{x} + E(\widetilde{\mathbf{x}})\|} \leq \kappa(A) \cdot rE(\widetilde{A})$$
3. 双方に誤差がある場合 —
$$rE(\widetilde{\mathbf{x}}) \leq \frac{\kappa(A)}{1 - \|A^{-1}E(\widetilde{A})\|}\left(rE(\widetilde{\mathbf{b}}) + rE(\widetilde{A})\right)$$

となる。

問題 7.5

linear_eq.sce に条件数 $\kappa(A) = \|A\|\|A^{-1}\|$ を求める処理を追加し，次元数と条件数，そしてノルム相対誤差と要素ごとの最大相対誤差を調べ，下記の表を埋めよ。

n	$\kappa(A)$	$rE(\widetilde{\mathbf{x}})$	$\max_i rE(\widetilde{x_i})$
50			
100			
200			
500			
1000			

7.4 LU 分解法による連立一次方程式の求解

LU 分解法とは，正則行列 $A \in \mathbb{C}^{n\times n}$ を，下三角行列 $L \in \mathbb{C}^{n\times n}$

$$L = \begin{bmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & \cdots & l_{n,n-1} & 1 \end{bmatrix} \quad (l_{ij} \neq 0\ (i > j))$$

と上三角行列 $U \in \mathbb{C}^{n\times n}$

$$U = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \ddots & \vdots \\ & & & u_{nn} \end{bmatrix} \quad (u_{ij} \neq 0\ (i \leq j))$$

に LU 分解し，

$$A = LU \tag{7.4}$$

と表現できることを利用して連立一次方程式 (7.1) を解く手法のことである。

LU 分解法のあらすじは次のようになる。

まず行列 A を LU 分解 (7.4) する。具体的な方法は後述する。

次に，

$$(LU)\mathbf{x} = \mathbf{b} \Rightarrow L(U\mathbf{x}) = \mathbf{b}$$

から，まず

$$L\mathbf{y} = \mathbf{b} \tag{7.5}$$

を $\mathbf{y} \in \mathbb{C}^n$ について解いて $\mathbf{y}$ を求め（前進代入），これを利用して

$$U\mathbf{x} = \mathbf{y} \tag{7.6}$$

を $\mathbf{x}$ について解く (後退代入)。

LU 分解法の利点は，逆行列 A^{-1} を直接求めるよりも計算量が少なくて済むことである。また，係数行列 A が変化せず，定数ベクトル $\mathbf{b}$ だけが変化する場合は，その LU 分解 (7.4) はそのまま流用でき，計算時間の短縮が可能となる。

7.4.1　LU 分解のアルゴリズム

まず，LU 分解法の考え方を説明する。LU 分解法の L と U はそれぞれ下三角行列 (lower triangular matrix)，上三角行列 (upper triangular matrix) を意味する。具体的には

$$L = \begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & & & \\ \vdots & \ddots & & \\ l_{n1} & \cdots & l_{nn} \end{bmatrix} = [l_{ij}],\ l_{ij} = \begin{cases} 0 & (i < j) \\ l_{ij} & (i \geq j, l_{ii} \neq 0) \end{cases} \tag{7.7}$$

$$U = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & u_{nn} \end{bmatrix} = \begin{bmatrix} u_{11} & \cdots & u_{1n} \\ & \ddots & \vdots \\ & & u_{nn} \end{bmatrix} = [u_{ij}],\ u_{ij} = \begin{cases} u_{ij} & (i \leq j, u_{ii} \neq 0) \\ 0 & (i > j) \end{cases} \tag{7.8}$$

という形の行列である。

もし，行列 A が L と U の積，つまり

$$A = LU$$

と分解できたとする。これを A の LU 分解と呼ぶことにする。さすれば (7.1) は

$$(LU)\mathbf{x} = \mathbf{b} \tag{7.9}$$

となる。これを 2 段階に分けて計算する。まず,

$$L\mathbf{y} = \mathbf{b} \quad \Leftrightarrow \quad \begin{bmatrix} l_{11} & & \\ \vdots & \ddots & \\ l_{n1} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \downarrow = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} \tag{7.10}$$

を y_1 から y_n の順に $\mathbf{y}(= U\mathbf{x})$ について解き，次にこの $\mathbf{y}$ を定数ベクトルとする連立一次方程式

$$U\mathbf{x} = \mathbf{y} \quad \Leftrightarrow \quad \begin{bmatrix} u_{11} & \cdots & u_{1n} \\ & \ddots & \vdots \\ & & u_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \uparrow = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \tag{7.11}$$

を x_n から x_1 の順に，$\mathbf{x}$ について解く。(7.10) を解く過程を前進代入 (forward substitution), (7.11) を解く過程を後退代入 (backward substitution) と呼ぶ。よって，もし A が LU 分解できればたやすく (7.1) は解けることになる。

ではどのようにして LU 分解を行うのか。これは A を行列の基本変形によって上三角行列にする (上三角化)，Gauss の消去法の考え方を使うことで得ることが出来る。簡単のために，$A \in \mathbb{C}^{3\times 3}$, $\mathbf{x}, \mathbf{b} \in \mathbb{C}^3$ の場合に限ってこの方法を説明する。この時，(7.1) を書き下すと

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \Leftrightarrow \quad \begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & = & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & = & b_3 \end{cases}$$

となる。この方程式を，解 **x** が変化しないように，最終的には

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a_{22}^{(1)} & a_{23}^{(1)} \\ & & a_{33}^{(2)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(2)} \end{bmatrix} \Leftrightarrow \begin{cases} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & = & b_1 \\ & & a_{22}^{(1)}x_2 & + & a_{23}^{(1)}x_3 & = & b_2^{(1)} \\ & & & & a_{33}^{(2)}x_3 & = & b_3^{(2)} \end{cases}$$

という形に変形する方法が，Gauss の消去法であり，これによって行列 A は上三角化される。

1. まず，第 1 列目を a_{11} で割り

$$\begin{bmatrix} 1 & \frac{1}{a_{11}} \cdot a_{12} & \frac{1}{a_{11}} \cdot a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{a_{11}} \cdot b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} x_1 & + & \frac{1}{a_{11}} \cdot a_{12}x_2 & + & \frac{1}{a_{11}} \cdot a_{13}x_3 & = & \frac{1}{a_{11}} \cdot b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & = & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & = & b_3 \end{cases}$$

と「変形したと仮定する」。この第 1 列目に a_{21} を乗じて第 2 列目から引けば

$$\begin{bmatrix} 1 & \frac{1}{a_{11}}a_{12} & \frac{1}{a_{11}}a_{13} \\ a_{21} - a_{21} \cdot 1 & a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12} & a_{23} - \frac{a_{21}}{a_{11}} \cdot a_{13} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{a_{11}} \cdot b_1 \\ b_2 - \frac{a_{21}}{a_{11}} \cdot b_1 \\ b_3 \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} x_1 & + & \frac{a_{12}}{a_{11}}x_2 & + & \frac{a_{13}}{a_{11}}x_3 & = & \frac{b_1}{a_{11}} \\ (a_{21} - a_{21} \cdot 1)x_1 & + & \left(a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12}\right)x_2 & + & \left(a_{23} - \frac{a_{21}}{a_{11}} \cdot a_{13}\right)x_3 & = & b_2 - \frac{a_{21}}{a_{11}} \cdot b_1 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & = & b_3 \end{cases}$$

となる。

2. 同様にして，第 1 列目に a_{31} を乗じて第 3 列目から引くと，

$$\begin{bmatrix} 1 & \frac{1}{a_{11}}a_{12} & \frac{1}{a_{11}}a_{13} \\ a_{21}-a_{21}\cdot 1 & a_{22}-\frac{a_{21}}{a_{11}}\cdot a_{12} & a_{23}-\frac{a_{21}}{a_{11}}\cdot a_{13} \\ a_{31}-a_{31}\cdot 1 & a_{32}-\frac{a_{31}}{a_{11}}\cdot a_{12} & a_{33}-\frac{a_{31}}{a_{11}}\cdot a_{13} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{1}{a_{11}}\cdot b_1 \\ b_2-\frac{a_{21}}{a_{11}}\cdot b_1 \\ b_3-\frac{a_{31}}{a_{11}}\cdot b_1 \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} x_1 + \frac{a_{12}}{a_{11}}x_2 + \frac{a_{13}}{a_{11}}x_3 = \frac{b_1}{a_{11}} \\ (a_{21}-a_{21}\cdot 1)x_1 + \left(a_{22}-\frac{a_{21}}{a_{11}}\cdot a_{12}\right)x_2 + \left(a_{23}-\frac{a_{21}}{a_{11}}\cdot a_{13}\right)x_3 = b_2-\frac{a_{21}}{a_{11}}\cdot b_1 \\ (a_{31}-a_{31}\cdot 1)x_1 + \left(a_{32}-\frac{a_{31}}{a_{11}}\cdot a_{12}\right)x_2 + \left(a_{33}-\frac{a_{31}}{a_{11}}\cdot a_{13}\right)x_3 = b_3-\frac{a_{13}}{a_{11}}\cdot b_1 \end{cases}$$

となる。煩雑なのでこれをまとめて

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a_{22}^{(1)} & a_{23}^{(1)} \\ & a_{32}^{(1)} & a_{33}^{(1)} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(1)} \end{bmatrix} \Leftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = b_2^{(1)} \\ a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 = b_3^{(1)} \end{cases} \tag{7.12}$$

と書くことにする。第 1 列目は元に戻しても差し支えない。

3. 同様にして，今度は第 2 列目を $a_{22}^{(1)}$ で割り，

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & 1 & \frac{1}{a_{22}^{(1)}}\cdot a_{23}^{(1)} \\ & a_{32}^{(1)} & a_{33}^{(1)} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ \frac{1}{a_{22}^{(1)}}\cdot b_2^{(1)} \\ b_3^{(1)} \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ x_2 + \frac{1}{a_{22}^{(1)}}\cdot a_{23}^{(1)}x_3 = \frac{1}{a_{22}^{(1)}}\cdot b_2^{(1)} \\ a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 = b_3^{(1)} \end{cases}$$

と「変形したと仮定する」。この第 2 列目に $a_{32}^{(1)}$ を乗じて第 3 列目から引けば

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & 1 & \frac{1}{a_{22}^{(1)}}\cdot a_{23}^{(1)} \\ & a_{32}^{(1)}-a_{32}^{(1)}\cdot 1 & a_{33}^{(1)}-\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\cdot a_{23}^{(1)} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ \frac{1}{a_{22}^{(1)}}\cdot b_2^{(1)} \\ b_3^{(1)}-\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\cdot b_2^{(1)} \end{bmatrix}$$

$$\Leftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ x_2 + \frac{1}{a_{22}^{(1)}}\cdot a_{23}^{(1)}x_3 = \frac{1}{a_{22}^{(1)}}\cdot b_2^{(1)} \\ (a_{32}^{(1)}-a_{32}^{(1)}\cdot 1)x_2 + \left(a_{33}^{(1)}-\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\right)x_3 = b_3^{(1)}-\frac{a_{32}^{(1)}}{a_{22}^{(1)}}b_2^{(1)} \end{cases}$$

となる。煩雑なのでこれをまとめ，第 2 列目を元に戻して，目的の

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a_{22}^{(1)} & a_{23}^{(1)} \\ & & a_{33}^{(2)} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2^{(1)} \\ b_3^{(2)} \end{bmatrix} \Leftrightarrow \begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = b_2^{(1)} \\ a_{33}^{(2)}x_3 = b_3^{(2)} \end{cases} \tag{7.13}$$

を得る。

さて，この 1〜3 までの手順のうち，行列に対して行われる基本変形を行列積の形で表現してみる。

まず 1 の部分では A に対して，1 行目に $-a_{21}/a_{11}$ を掛けて 2 行目に加えているので，$R(2,1;-a_{21}/a_{11})$ を左から乗じていることになる。次に，2 の部分では同様に 1 行目に $-a_{31}/a_{11}$ を掛けて 3 行目に加えているので，$R(3,1;-a_{31}/a_{11})$ を左から乗じていることになる。最後の 3 でも同様に，$R(3,2;-a_{32}^{(1)}/a_{22}^{(1)})$ を左から乗じていることになる。よって

$$R\left(3,2;-\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\right)R\left(3,1;-\frac{a_{31}}{a_{11}}\right)R\left(2,1;-\frac{a_{21}}{a_{11}}\right)A = U$$

という操作をして U を得ていることになる。

従って L は

$$A = R\left(2,1;\frac{a_{21}}{a_{11}}\right)R\left(3,1;\frac{a_{31}}{a_{11}}\right)R\left(3,2;\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\right)U$$

より

$$\begin{aligned} L &= R\left(2,1;\frac{a_{21}}{a_{11}}\right)R\left(3,1;\frac{a_{31}}{a_{11}}\right)R\left(3,2;\frac{a_{32}^{(1)}}{a_{22}^{(1)}}\right) \\ &= \begin{bmatrix} 1 & & \\ \frac{a_{21}}{a_{11}} & 1 & \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & & \\ 0 & 1 & \\ \frac{a_{31}}{a_{11}} & 0 & \end{bmatrix}\begin{bmatrix} 1 & & \\ 0 & 1 & \\ 0 & \frac{a_{32}^{(1)}}{a_{22}^{(1)}} & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & & \\ \frac{a_{21}}{a_{11}} & 1 & \\ \frac{a_{31}}{a_{11}} & \frac{a_{32}^{(1)}}{a_{22}^{(1)}} & 1 \end{bmatrix} \\ U &= \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a_{22}^{(1)} & a_{23}^{(1)} \\ & & a_{33}^{(2)} \end{bmatrix} \end{aligned}$$

となる。

なお，ここで U を更に対角行列 D を用いて

$$U = DU' = \begin{bmatrix} a_{11} & & \\ & a_{22}^{(1)} & \\ & & a_{33}^{(2)} \end{bmatrix}\begin{bmatrix} 1 & \frac{a_{12}}{a_{11}} & \frac{a_{13}}{a_{11}} \\ & 1 & \frac{a_{23}^{(1)}}{a_{22}^{(1)}} \\ & & 1 \end{bmatrix}$$

と表現したものを A の LDU 分解（LDU decomposition) と呼ぶ。

7.4.2　LU 分解法の Scilab スクリプト

以上，LU 分解法を Scilab で実行するには次のように行う。実行する際には，linear_eq.sce スクリプトの 30 行目〜35 行目を以下のように置き換えればよい。なお，ベンチマークテストの際には以下のスクリプトの disp 関数をすべてコメント化して無効にしておくこと。

まず, lu 関数を用いて係数行列 A を LU 分解し, L, U および枢軸選択に伴う並び替え行列 P ($PLU = A$) を求める。

```
// A -> LU = A
[l, u, p] = lu(a);
disp("L = "); disp(l)
disp("U = "); disp(u)
disp("P = "); disp(p)
```

$PLU = A$ であることを相対残差ノルム $||PLU - A||/||A||$ を求めることで確認する。

```
// P * L * U = A ?
disp("||PLU - A|| / ||A|| "); disp(norm(p * l * u - a)); // norm(a)
```

以下，$P\mathbf{z} = \mathbf{b}$ を $\mathbf{z}$ について解き，$L\mathbf{y} = \mathbf{z}$ を $\mathbf{y}$ について解き，$U\mathbf{x} = \mathbf{y}$ を $\mathbf{x}$ について解き，トータルの計算時間を求める。。

```
// P * z = b
z = p \ b;
disp("P * z = b -> z = "); disp(z);

// L * y = z
y = l \ b;
disp("L * y = z -> y = "); disp(y);

// U * x = y
x = u \ y;
disp("U * x = y -> x = "); disp(x);

// 計算時間計測終了
time_invsolve = toc();
mprintf( "LU 分解法 (秒) = %f", time_invsolve);
```

$n = 5$ の時は次のような結果を得る。

```
 次元数 n を入力してください：
-->5
次元数 (n) = 5
 準備完了！  START!

 L =

    1.      0.       0.           0.      0.
    0.8     1.       0.           0.      0.
    0.6     0.75     1.           0.      0.
    0.4     0.5      0.6666667    1.      0.
    0.2     0.25     0.3333333    0.5     1.

 U =

    5.     4.      3.       2.           1.
    0.     0.8     0.6      0.4          0.2
    0.     0.      0.75     0.5          0.25
    0.     0.      0.       0.6666667    0.3333333
    0.     0.      0.       0.           0.5

 P =

    1.     0.     0.     0.     0.
    0.     1.     0.     0.     0.
    0.     0.     1.     0.     0.
    0.     0.     0.     1.     0.
    0.     0.     0.     0.     1.

 ||PLU - A|| / ||A||

    4.441D-16

LU 分解法 (秒) = 0.002000
 ||b - a * x||_2 / ||x||_2 =

    0.
```

```
 ||x - true_x||_2 / ||true_x||_2 =

    1.051D-15

 max(|x_i - true_x_i| / |true_i|) =

    2.442D-15
```

7.5　QR 分解法による連立一次方程式の求解

QR 分解法とは，正則行列 $A \in \mathbb{C}^{n\times n}$ を，ユニタリ行列 $Q \in \mathbb{C}^{n\times n}$

$$Q = [\mathbf{q}_1\ \mathbf{q}_2\ \cdots\ \mathbf{q}_n] = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1n} \\ q_{21} & q_{22} & \cdots & q_{2n} \\ \vdots & \vdots & & \vdots \\ q_{n1} & q_{n2} & \cdots & q_{nn} \end{bmatrix} ((\mathbf{q}_i, \mathbf{q}_j) = 0\ (i \neq j), (\mathbf{q}_i, \mathbf{q}_i) = 1) = 1\ (i = 1, 2, ..., n))$$

と上三角行列 $R \in \mathbb{C}^{n\times n}$

$$R = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \vdots \\ & & & r_{nn} \end{bmatrix} (r_{ij} \neq 0)$$

に QR 分解し，

$$A = QR \tag{7.14}$$

であることを利用して連立一次方程式 (7.1) を求める手法である。分解の方法が異なるだけで，アルゴリズムは LU 分解同様，まず A の QR 分解 (7.14) を求め，次に

$$(QR)\mathbf{x} = \mathbf{b} \Rightarrow Q(R\mathbf{x}) = \mathbf{b}$$

より

$$\mathbf{y} = \bar{Q}^T\mathbf{b} \tag{7.15}$$

として $\mathbf{y} \in \mathbb{C}^n$ を求める。最後はこの $\mathbf{y}$ を用いて

$$R\mathbf{x} = \mathbf{y} \tag{7.16}$$

を解いて $\mathbf{x}$ を得る。

QR 分解法のメリットは，枢軸 (pivot) が 0 になることが比較的多い LU 分解に比べ，安定してアルゴリズムを構築できることにある。また，成分ごとに順序良く計算する必要のある LU 分解に比べ，行列の列ベクトル単位の計算で構築されていることも，現在の並列計算が容易な計算機環境では有効である。

7.5.1 QR 分解のアルゴリズム: Gram-Schmit 法

行列 A が正則であれば，その列ベクトル $A = [\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n]$ は n 個の一次独立なベクトルの組と見なせる。

複数の一次独立なベクトルの組を，正規直交基底に変換するアルゴリズムとして，Gram-Schmidt の直交化法がある。$\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_n$ へ適用し，正規直交基底 $\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_n$ を作るアルゴリズムは次のようになる。

■Gram-Schmidt の直交化法

for $i = 1, 2, ..., n$
(a) $\mathbf{u}_i := \mathbf{a}_i - \sum_{j=1}^{i-1}(\mathbf{q}_j, \mathbf{a}_i)\mathbf{q}_j$
(b) $\mathbf{q}_i := \mathbf{u}_i / \|\mathbf{u}_i\|_2$

このアルゴリズムを行列の形で表わすと，

$$[\mathbf{a}_1\ \mathbf{a}_2\ \cdots\ \mathbf{a}_n] = [\mathbf{q}_1\ \mathbf{q}_2\ \cdots\ \mathbf{q}_n] \begin{bmatrix} \|\mathbf{u}_1\|_2 & (\mathbf{q}_1, \mathbf{a}_2) & (\mathbf{q}_1, \mathbf{a}_3) & \cdots & (\mathbf{q}_1, \mathbf{a}_n) \\ & \|\mathbf{u}_2\|_2 & (\mathbf{q}_2, \mathbf{a}_3) & \cdots & (\mathbf{q}_2, \mathbf{a}_n) \\ & & \ddots & \ddots & \vdots \\ & & & \|\mathbf{u}_{n-1}\|_2 & (\mathbf{q}_{n-1}, \mathbf{a}_n) \\ & & & & \|\mathbf{u}_n\|_2 \end{bmatrix}$$

となる。これを

$$A = QR$$

と書くと，Q はユニタリ行列 $Q\overline{Q}^T = I$ に，R は上三角行列になる。これを行列 A の QR 分解と呼ぶ。

7.5.2 QR 分解法の Scilab スクリプト

QR 分解法を実行するには，LU 分解法における linear_eq.sce スクリプトの改良と同様の変更を加えればよい。

まず，qr 関数を用いて係数行列 A を QR 分解し，相対残差ノルム $\|QR - A\|/\|A\|$ を求めて確認する。

```
// A -> QR = A
[q, r] = qr(a);
disp("Q = "); disp(q);
disp("R = "); disp(r);


// P * L * U = A ?
disp("||QR - A|| / ||A|| "); disp(norm(q * r - a)); // norm(a)
```

次に，$Q\mathbf{y} = \mathbf{b}$ を $\mathbf{y}$ について解き，$R\mathbf{x} = \mathbf{b}$ として前述の LU 分解同様に解く。

```
// Q * y = b
```

```
y = q \ b;
disp("Q * y = b -> y = "); disp(y);

// U * x = y
x = r \ y;
disp("R * x = y -> x = "); disp(x);

// 計算時間計測終了
time_invsolve = toc();
mprintf( "QR 分解法 (秒) = %f", time_invsolve);
```

前述の行列 A を用いると次のような結果を得る。

```
 次元数 n を入力してください：
-->5
次元数 (n) = 5
 準備完了！   START!

 Q =

  - 0.6741999    0.7385489  - 5.053D-16    2.846D-16    0.
  - 0.5393599  - 0.4923660    0.6831301  - 3.645D-16    2.401D-17
  - 0.4045199  - 0.3692745  - 0.5855400    0.5976143  - 9.242D-17
  - 0.2696799  - 0.2461830  - 0.3903600  - 0.7171372  - 0.4472136
  - 0.1348400  - 0.1230915  - 0.1951800  - 0.3585686    0.8944272

 R =

  - 7.4161985  - 6.7419986  - 5.5284389  - 3.9103592  - 2.0225996
    0.         - 0.7385489  - 0.9847319  - 0.8616404  - 0.4923660
    0.           0.         - 0.6831301  - 0.7807201  - 0.4879500
    0.           0.           0.         - 0.5976143  - 0.4780914
    0.           0.           0.           0.           0.4472136

 ||QR - A|| / ||A||

    4.868D-15

 Q * y = b -> y =
```

```
  - 63.239947
  - 10.339685
  - 7.6120206
  - 4.7809144
    2.236068

 R * x = y -> x =

    1.
    2.
    3.
    4.
    5.
QR 分解法（秒） = 0.004000
 ||b - a * x||_2 / ||x||_2 =

    3.741D-15

 ||x - true_x||_2 / ||true_x||_2 =

    1.877D-15

 max(|x_i - true_x_i| / |true_i|) =

    1.088D-14
```

演習問題

1. 真の解 $\mathbf{x} \in \mathbb{R}^n$ が

$$\mathbf{x} = \begin{bmatrix} -1 \\ 2 \\ \vdots \\ (-1)^n n \end{bmatrix}$$

，係数行列 $A \in \mathbb{R}^{n\times n}$ が三重対角行列

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}$$

である時，定数ベクトル $\mathbf{b} = A\mathbf{x}$ を求め，$n = 100, 200, 500, 1000, 2000$ の時の計算時間，相対残差ノルム，相対誤差ノルム，成分ごとの相対誤差の最大値をそれぞれ求めよ。

2. 真の解 $\mathbf{x} \in \mathbb{C}^n$ が

$$\mathbf{x} = \begin{bmatrix} 1 - \mathrm{i} \\ 2 - 2\mathrm{i} \\ \vdots \\ n - n\mathrm{i} \end{bmatrix}$$

，係数行列 $A \in \mathbb{C}^{n\times n}$ が

$$A = \begin{bmatrix} \frac{1}{1+\mathrm{i}} & \frac{1}{2+2\mathrm{i}} & \cdots & \frac{1}{n+n\mathrm{i}} \\ \frac{1}{2+2\mathrm{i}} & \frac{1}{3+3\mathrm{i}} & \cdots & \frac{1}{n+1+(n+1)\mathrm{i}} \\ \vdots & \vdots & & \vdots \\ \frac{1}{n+n\mathrm{i}} & \frac{1}{n+1+(n+1)\mathrm{i}} & \cdots & \frac{1}{2n+2n\mathrm{i}} \end{bmatrix}$$

である時，定数ベクトル $\mathbf{b} = A\mathbf{x}$ を求め，$n = 10, 20, 50, 100, 200, 500$ の時の計算時間，相対残差ノルム，相対誤差ノルム，成分ごとの相対誤差の最大値をそれぞれ求めよ。