第3学年　電気電子工学実験実習報告書

6　数値計算（2）

実験日 令和 4 年 11 月 17 日 (木)

班	学生番号	氏名
2	3322	高橋広旭

共同実験者名

提出日			備考	評価
予定日　11/24				
提出日				

東京都立産業技術高等専門学校

電気電子工学コース

1　目的

この実験では

- 回路図より連立方程式が立てられる。
- MATLAB を用いて、行列とベクトルの基本演算ができる。

ことを目的とする。

2　原理

2.1　MATLAB

C 言語とは異なり、MATLAB はプログラムをその場で翻訳しながら実行していくインタプリタ型プログラミング言語である。その特徴は

- ベクトルや行列の全要素を 1 つの変数に代入でき、簡単な表記で行列・ベクトル演算ができる。
- 豊富な数値計算の Toolbox（関数群）が用意されており、複雑な処理を関数一つで実行できる。
- 実行結果をその場でグラフ表示する機能を備える。

などがある。

2.2　代入と表示

MATLAB では、変数の型や配列の大きさなどの宣言をせず、変数への代入を行える。ここでは、変数をベクトルや行列と見なし（ベクトル値が代入された変数をベクトル変数、行列が代入された変数を行列変数とする）、その内容を表示させることができる。

(i) ベクトル変数への代入
　`a = [1 2 3]` のように、スペースで区切った数字を、角括弧 `[ ]` でくくると行ベクトルになる。この操作は、式 (1) の左辺 a に右辺を代入していることと同じである。

$$a = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \tag{1}$$

また、`b = [4;5;6]` のように、区切りをスペースからセミコロン「`;`」にすることで変数に列ベクトルを代入する式 (2) の操作を行うことができる。

$$b = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \tag{2}$$

(ii) ベクトル変数の表示
　変数名を入力するとベクトルの全要素が表示される。また変数名の後に丸括弧を使って「`a(2)`」と入力すると、ベクトルの指定された要素のみが表示される。さらに「`b([1 3])`」のように丸括弧内に要素番号をスペースで区切って角括弧「`[ ]`」でくくり指定すると、複数の要素を表示できる。

(iii) 行列要素への代入

`c = [1 2; 3 4]` のようにスペース区切りの行ベクトルを複数用意し、これらをセミコロンで区切って並べ、全体を角括弧でくくると行列となり、変数 c に対して式 (3) の操作を行うことが出来る。

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

(iv) 行列変数の表示

ベクトル変数と同じように、変数名を打つと行列の全要素が表示される。また変数名の後に丸括弧を使って `c(2, 1)` とすると 2 行 1 列目の値が表示される。その他に、コロン「`:`」を行要素や列要素に使用すると、全部の行（列）要素を表示できる。

2.3 基本演算

(i) 加減算

式 (4) に示すような、要素数の等しい N 行 M 列の変数 A、B について、

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{21} & a_{22} & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NM} \end{pmatrix}, \quad B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1M} \\ b_{21} & b_{22} & \cdots & b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ b_{N1} & b_{N2} & \cdots & b_{NM} \end{pmatrix} \tag{4}$$

「`C=A+B`」とすることで式 (5) の演算結果 C を得ることができる。

$$C = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1M} \\ c_{21} & c_{22} & \cdots & c_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ c_{N1} & c_{N2} & \cdots & c_{NM} \end{pmatrix} = A \pm B = \begin{pmatrix} a_{11} \pm b_{11} & a_{12} \pm b_{12} & \cdots & a_{1M} \pm b_{1M} \\ a_{21} \pm a_{21} & a_{22} \pm b_{22} & \cdots & a_{2M} \pm b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} \pm b_{N1} & a_{N2} \pm b_{N2} & \cdots & a_{NM} \pm b_{NM} \end{pmatrix} \tag{5}$$

スカラー変数（行列数のない値）との加減算を除き、行数や列数が異なる変数同士の加減算は行うことができない。

(ii) 2 つの行列の積

式 (6) で定義される 2 つの変数を

$$D = \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1L} \\ d_{21} & d_{22} & \cdots & d_{2L} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N1} & d_{N2} & \cdots & b_{NL} \end{pmatrix}, \quad E = \begin{pmatrix} e_{11} & e_{12} & \cdots & e_{1M} \\ e_{21} & e_{22} & \cdots & e_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ e_{L1} & e_{L2} & \cdots & e_{LM} \end{pmatrix} \tag{6}$$

「`F=D*E`」とすることで式 (7) に示す行列同士の乗算を行うことができる。

$$F = \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1M} \\ f_{21} & f_{22} & \cdots & f_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ f_{N1} & f_{N2} & \cdots & f_{NM} \end{pmatrix} = DE = \begin{pmatrix} \sum_{i=1}^{L} d_{1i}e_{i1} & \sum_{i=1}^{L} d_{1i}e_{i2} & \cdots & \sum_{i=1}^{L} d_{1i}e_{iM} \\ \sum_{i=1}^{L} d_{2i}e_{i1} & \sum_{i=1}^{L} d_{2i}e_{i2} & \cdots & \sum_{i=1}^{L} d_{2i}e_{iM} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{L} d_{Ni}e_{i1} & \sum_{i=1}^{L} d_{Ni}e_{i2} & \cdots & \sum_{i=1}^{L} d_{Ni}e_{iM} \end{pmatrix} \tag{7}$$

ただし、スカラー変数との乗算を除き、この演算では左行列（*の左側）の列数と、右行列の行数が一致していなくてはならない。

(iii) アダマール積

「A.*B」と入力することで、式 (8) のような、

$$G = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1M} \\ g_{21} & g_{22} & \cdots & g_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ g_{N1} & g_{N2} & \cdots & g_{NM} \end{pmatrix} = A \circ B = \begin{pmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1M}b_{1M} \\ a_{21}b_{21} & a_{22}b_{22} & \cdots & a_{2M}b_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1}b_{N1} & a_{N2}b_{N2} & \cdots & a_{NM}b_{NM} \end{pmatrix} \tag{8}$$

行列数が等しい二つの行列の要素同士の乗算（アダマール積）を行うことができる。加減算の場合の制約と同じように、アダマール積を行う 2 つの行列の行数・列数が一致していなくてはならない。

(iv) 転置行列、逆行列

行列演算において、i 行 j 列目の要素を j 行 i 列目の要素と置き換える

$$A^T = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{N1} \\ a_{12} & a_{22} & \cdots & a_{N2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1M} & a_{2M} & \cdots & a_{NM} \end{pmatrix} \tag{9}$$

の演算を行うには、「A'」と入力する。

また、行列 A との間に次式の関係が成り立つ行列 B を A の逆行列（また同時に A は B の逆行列）と呼び、A^{-1} と表す。

$$AB = I = BA \tag{10}$$

ただし、I は単位行列であり、式 (11) に示すように、対角要素が 1、その他の要素が 0 である行列である。

$$I = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \tag{11}$$

MATLAB においてある行列変数の逆行列を得るには「inv(A)」とすれば良い。

(v) 数学関数と行列

sin、cos のような三角関数や exp 関数あるいは 2 乗根などの数学関数の引数に n 行 m 列の行列変数を用いる（「sin(A)」と入力する）と、返り値も同じ行列数の行列変数となる。

$$\begin{pmatrix} \sin(a_{11}) & \sin(a_{12}) & \cdots & \sin(a_{1M}) \\ \sin(a_{21}) & \sin(a_{22}) & \cdots & \sin(a_{2M}) \\ \vdots & \vdots & \ddots & \vdots \\ \sin(a_{N1}) & \sin(a_{N2}) & \cdots & \sin(a_{NM}) \end{pmatrix} \tag{12}$$

返り値の行列要素は、式 (12) のように引数の要素をそれぞれスカラー値として引数にした値となる。

3　方法

3.1　使用器具

今回の実験で使用した器具を表 1 に示す。

表 1: 使用器具

使用器具名	製造元	型番
パソコン	Apple	iMac 21.5-inch

3.2　行列演算

MATLAB を用いて以下の行列の演算を行った。

1) ベクトル [1 5 4 6] とベクトル [7 -3 4 -5] の加算

2) ベクトル [1 5 4 6] からベクトル [7 -3 4 -5] の減算

3) ベクトル [1 5 4 6] とベクトル [7 -3 4 -5] の内積

4) ベクトル [1 5 4 6] とベクトル [7 -3 4 -5] の要素どうしの乗算

5) 以下の行列の転置行列

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

6) 以下の行列の加算

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{pmatrix} 2 & 9 & 4 \\ 7 & 5 & 3 \\ 6 & 1 & 8 \end{pmatrix}$$

7) 上の行列の乗算 (左の行列を左から掛ける)

8) 以下の行列の逆行列

$$\begin{pmatrix} 1 & 2 & 0 \\ 0 & -1 & 1 \\ 2 & 0 & 3 \end{pmatrix}$$

3.3　グラフ作成

以下の関数のグラフを書いた。

1) e^t $(0 \leq t \leq 5)$ 0.01 刻み

2) $e^t \sin 8\pi t$ $(0 \leq t \leq 5)$ 0.01 刻み

3) $\sin 2\pi t$ と「この正弦波と 60°位相が遅れた信号」(同時に表示) $(0 \leq t \leq 5)$ 0.01 刻み

4) $e^{-(x^2+y^2)} \cos 2\pi(x^2+y^2)$ $(-2 \leq x \leq 2,\ -2 \leq y \leq 2)$ 0.1 刻み

4 結果

4.1 代入と表示

原理 2.2 に記載されている内容も実習中に行ったためその実行結果プログラム 1 に示す。

プログラム 1: 代入と表示

```
>> a=[1 2 3]

a =

                  1 2 3

>> b=[4;5;6]

b =

                  4
                  5
                  6

>> c=[5:2:9]

c =

                  5 7 9

>> d=[7:9]

d =

                  7 8 9

>> a

a =

                  1 2 3

>> b(2)

ans =

                  5

>> c([1 3])

ans =

                  5 9

>> d([1:3])

ans =

                  7 8 9

>> e=[1 2;3 4]

e =

                  1 2
                  3 4

>> f=[3:-1:1;5:7]

f =

                  3 2 1
```

```
                    5 6 7

>> g=[a;c]

g =

                    1 2 3
                    5 7 9

>> ee=[1 2 3;4 5]
エラー: vertcat
連結する配列の次元が一致しません。
```

4.2 基本演算

原理 2.3 に記載されている内容も実習中に行ったためその実行結果をプログラム 2 に示す。また出力したグラフを図 1 と図 2 に示す。

プログラム 2: 基本演算

```
>> e

e =

      1 2
      3 4

>> f(2,1)

ans =

      5

>> g(2,:)

ans =

      5 7 9

>> g(:,[1 3])

ans =

                   1 3
                   5 9

>> a+c

ans =

                   6 9 12

>> f-g

ans =

                   2 0 -2
                   0 -1 -2

>> b-a

ans =

                   3 2 1
                   4 3 2
                   5 4 3

```

```
>> 1+e

ans =

        2 3
        4 5

>> a*b

ans =

    32

>> f*b

ans =

    28
    92

>> e*f

ans =

    13 14 15
    29 30 31

>> 2*e

ans =

        2 4
        6 8

>> f.*g

ans =

        3 4 3
    25 42 63

>> a'

ans =

        1
        2
        3

>> inv(e)

ans =

   -2.0000 1.0000
    1.5000 -0.5000

>> sqrt(b)

ans =

    2.0000
    2.2361
    2.4495

>> sin(e)

ans =

    0.8415 0.9093
    0.1411 -0.7568

```

```
>> exp(b)

ans =

      54.5982
      148.4132
      403.4288

>> u=[11 12 13 14];
>> length(u)

ans =

               4

>> max(u)

ans =

              14

>> min(u)

ans =

              11

>> for i = [1 2 3 4 5]
v(i) = i*i;
end
>> v

v =

               1 4 9 16 25

>> clear z
>> x = [-1 : 0.05 : 1];
>> y = [-1 : 0.05 : 1]';
>> for i = [1 : length(y)]
z(i,:) = cos(2*pi*( x.*x + y(i)*y(i)));
end
>> mesh(x, y, z)
```

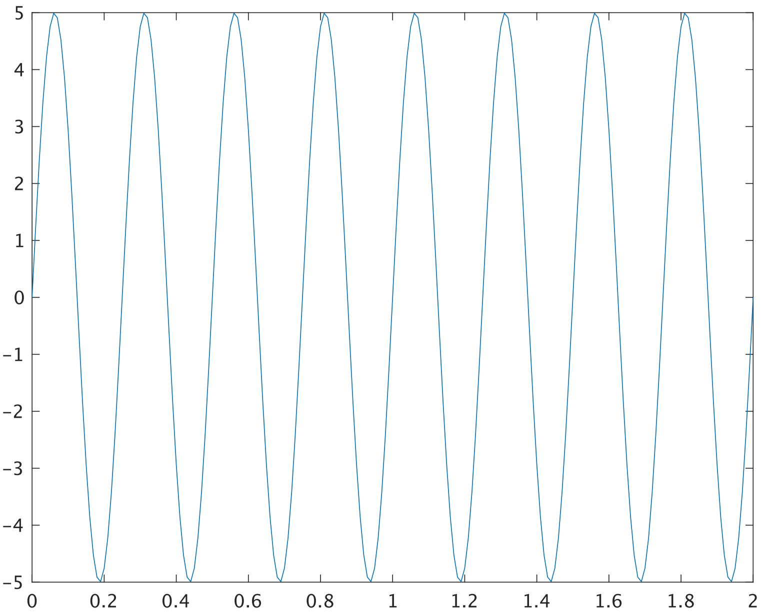

図 1: 正弦波信号のグラフ

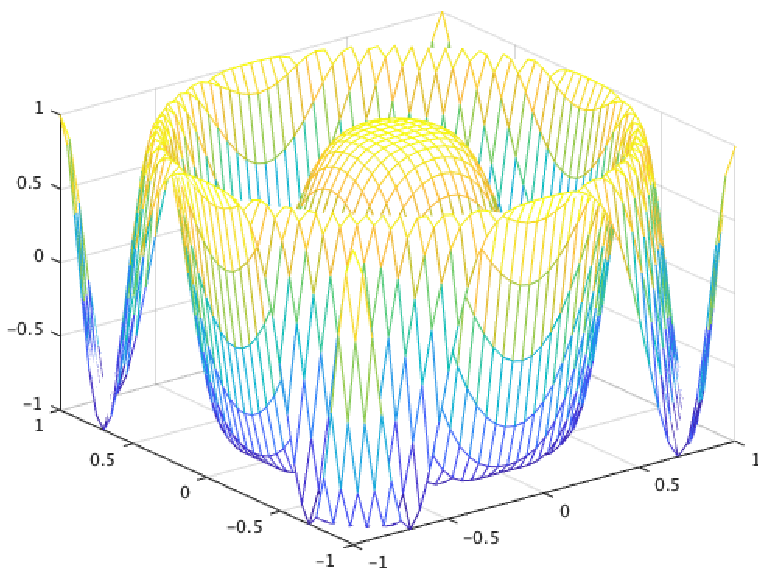

図 2: 2 次元信号のグラフ

4.3 行列演算

プログラム 3 に行列演算の実行結果を示す。

プログラム 3: 実行結果

```
>> a=[1 5 4 6]

a =

                    1 5 4 6

```

```
>> b=[7 -3 4 -5]

b =

        7 -3 4 -5

>> a+b

ans =

        8 2 8 1

>> a-b

ans =

       -6 8 0 11

>> dot(a,b)

ans =

   -22

>> a.*b

ans =

        7 -15 16 -30

>> c=[1 2 3;4 5 6;7 8 9]

c =

        1 2 3
        4 5 6
        7 8 9

>> c'

ans =

        1  4  7
        2  5  8
        3  6  9

>> d=[2 9 4;7 5 3 ;6 1 8]

d =

        2  9  4
        7  5  3
        6  1  8

>> c+d

ans =

        3  11  7
       11  10  9
       13  9  17

>> c.*d

ans =

        2  18  12
       28  25  18
       42  8  72

>> c*d
```

```

ans =

              34  22  34
              79  67  79
              124  112  124

>> e=[1 2 0;0 -1 1;2 0 3]

e =

               1  2  0
               0  -1  1
               2  0  3

>> inv(e)

ans =

              -3  -6  2
               2  3  -1
               2  4  -1
```

4.4 グラフ作成

プログラム 4 からプログラム 7 にグラフ作成に用いた MATLAB の実行結果を示す。また、図 3 から図 6 にグラフの出力結果を示す。

プログラム 4: グラフ 1

```
>> t=[0 : 0.01 :5];
>> et=exp(t)

et = 省略

>> plot(t,et)
```

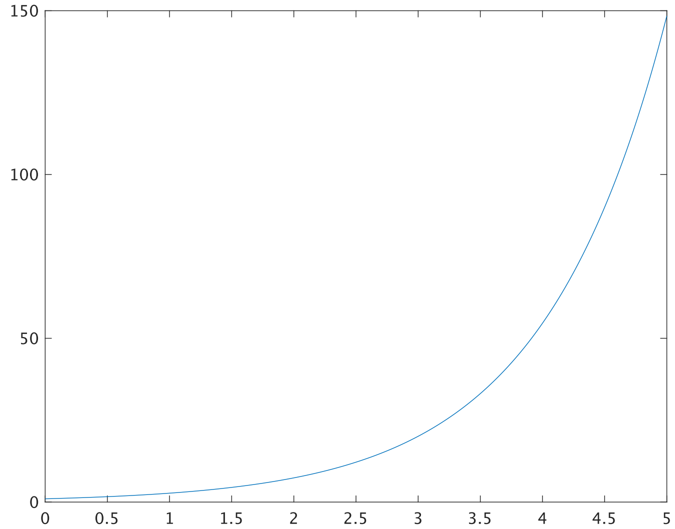

図 3: e^t のグラフ

プログラム 5: グラフ 2

```
>> etsin=et.*sin(8*pi*t)
```

```
etsin = 省略

>> plot(t,etsin)
```

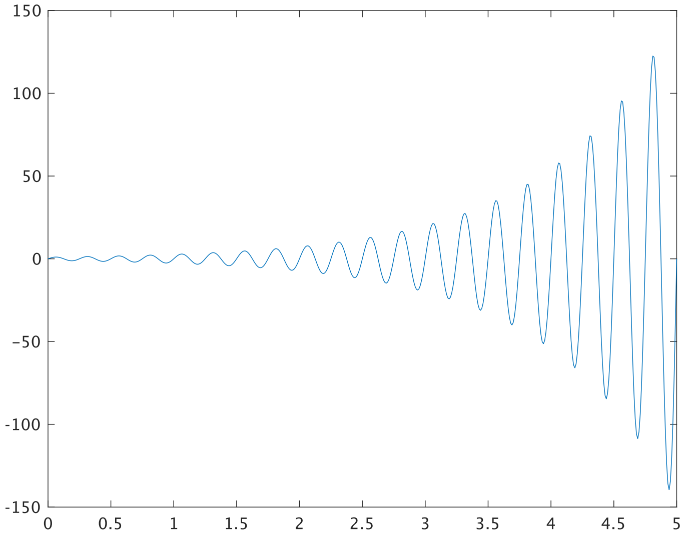

図 4: $e^t \sin 8\pi t$ のグラフ

プログラム 6: グラフ 3

```
>> sin2pt=sin(2*pi*t)

sin2pt = 省略

>> sin2pt60=sin(2*pi*t-pi/3)

sin2pt60 = 省略

>> plot(t,sin2pt,t,sin2pt60)
```

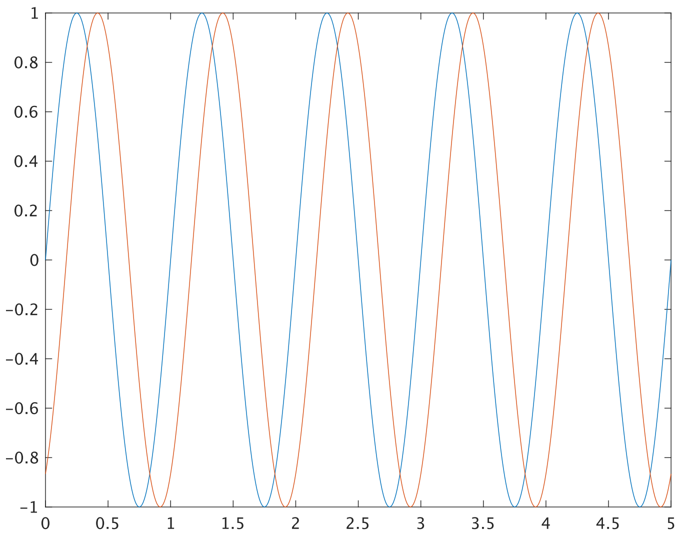

図 5: $\sin 2\pi t$, $\sin(2\pi t - \frac{\pi}{3})$ のグラフ

プログラム 7: グラフ 4

```
>> x=[-2:0.1:2]

x = 省略

>> y=[-2:0.1:2]

y = 省略

>> for i=[1:length(y)]
z(i,:)=exp(-(x.^2+y(i).^2)).*cos(2*pi*(x.^2+y(i).^2));
end
>> mesh(x,y,z)
```

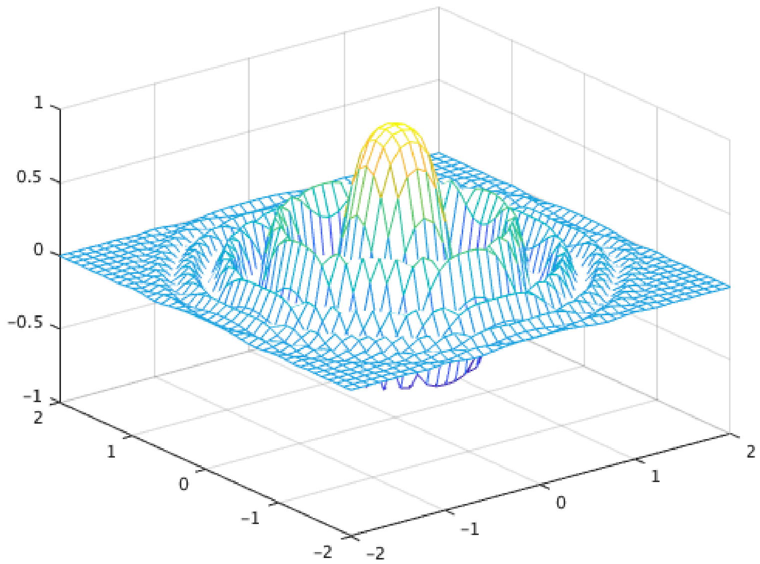

図 6: $e^{-(x^2+y^2)}\cos 2\pi(x^2+y^2)$ のグラフ

5　考察

5.1　C 言語と比較して MATLAB の利点、欠点

MATLAB は C 言語と比較してベクトルや行列の値を容易に入力できる点が利点である。また、MATLAB 単体でグラフの作成ができる点も利点といえる。しかし、MATLAB ではベクトルや行列の中の値を設定すると、その値がすべて画面上に出力するため、過去に入力したものが見づらくなってしまうことが欠点である。

5.2　この実習で得られた知見

C 言語より MATLAB のほうがベクトルや行列の値を容易に入力でき、グラフへの出力も容易にできる。従って、ベクトルや行列計算は C 言語より MATLAB のほうが適している。

5.3　実習から浮かんだ技術的な疑問点

MATLAB は行列やベクトルの計算以外にどのようなことができるか。

5.4 上の疑問点を調査し解決せよ

MATLAB は外部ソースからのデータ収集・解析やアルゴリズム開発、組み込みコードの生成ができ、ディープラーニングや IoT、データサイエンスなど幅広い分野で用いられている [1]。

6 結論

今回の実験を通して MATLAB を用いた行列とベクトルの基本演算ができた。

参考文献

[1] 中川大輝, “MATLAB でできることは？活用事例とあわせて紹介”, 株式会社 SAMURAI, `https://www.sejuku.net/blog/108396`, 2022 年 11 月 23 日

C 言語を用いて逆関数の計算をする

C 言語を用いて逆関数の計算をするプログラムをプログラム 8 に示す。また、前回実験分と併せてテストした結果をプログラム 9 に示す。

プログラム 8: 逆関数を計算するプログラム

```
int inverseMatrix(double anspp[N][N], double inpp[N][N], int n) {
int i, j, n2;
n2 = n * 2;
for (i = 0; i < n; i++) {
	for (j = 0; j < n; j++) {
		anspp[i][j] = inpp[i][j];
		anspp[i][j + n] = i == j ? 1.0 : 0.0;
	}
}
for (j = 0; j < n - 1; j++) {
	int p = j;
	double vmax = fabs(anspp[j][j]);
	for (i = j + 1; i < n; i++) {
		double v = fabs(anspp[i][j]);
		if (v > vmax) {
			p = i;
			vmax = v;
		}
	}
	if (vmax < 1.0e-12) {
		printf("too^^e2^^90^^a3small^^e2^^90^^a3pivot!^^e2^^90^^a3\n");
		return 1;
	}
	if (p != j) {
		rowConvertMatrix(anspp, n, n2, j, p);
	}
	for (i = j + 1; i < n; i++) {
		double w = -anspp[i][j] / anspp[j][j];
		rowCmulAddMatrix(anspp, n, n2, j, w, i);
	}
}
for (j = n - 1; j >= 0; j--) {
	rowCmulMatrix(anspp, n, n2, j, 1.0 / anspp[j][j]);
	for (i = j - 1; i >= 0; i--) {
		rowCmulAddMatrix(anspp, n, n2, j, -anspp[i][j], i);
	}
}
for (j = 0; j < n; j++) {
	colConvertMatrix(anspp, n, n2, j, j + n);
}
for (i = 0; i < n; i++) {
	for (j = 0; j < n; j++) {
		if (!isfinite(anspp[i][j])) {
			return 1;
		}
	}
}
return 0;
}
```

プログラム 9: 逆関数を計算するプログラムを含めたテスト結果

```
	== Test testSetValues ==
	== Test addMatrix ==
	== Test mulMatrix ==
	== Test rowCmulMatrix ==
	== Test rowConvertMatrix ==
	== Test colConvertMatrix ==
	== Test rowCmulAddMatrix ==
	== Test inverseMatrix ==
```

```
All tests are Ok. [# of Tests = 86, # of pass = 86 (100%)]
```