

レポート課題 2

奥屋 直 己

2018 年 7 月 31 日

1 課題 1

作成したガウス消去法を行うプログラムをソースコード 1 に示す。

ソースコード 1 ガウス消去法

```
1 #include<stdio.h>
2
3 #define N 10
4 #define M 100
5 #define L 100
6
7 //ガウス消去法の関数
8 void s_gauss(int n ,double x[] ,double a[][N] ,double b[]){
9     int i,j,k;
10    double mik;
11
12    //ガウスの前進消去
13    for(k=0 ;k<n-1 ;k++){
14        for(i=k+1 ;i<n ;i++){
15            mik=a[i][k]/a[k][k];
16            for(j=k+1 ;j<n ;j++){
17                a[i][j]-=mik*a[k][j];
18            }
19            b[i]-=mik*b[k];
20        }
21    }
22
23    //ガウスの後退代入
24    for(k=n-1 ;k>=0 ;k--){
25        x[k]=b[k];
26        for(i=k+1 ;i<n ;i++){
27            x[k]-=a[k][i]*x[i];
28        }
29        x[k]/=a[k][k];
30    }
31 }
32
33 int main(int argc ,char *argv[]){
34     int i,j,k;
35     int x_n=0;
36     int sum;
37     double a_b[N][N];
```

```

38 double a[N][N];
39 double x[N];
40 double b[N];
41 char f_name[M];
42 char temp[L]={};
43 FILE *fr;
44
45 //引数から取ったファイル名をf_nameに代入
46 //ファイルには行列Aとbを複合した物を書き込んでおく
47 sprintf(f_name,"%s",argv[1]);
48
49 if((fr=fopen(f_name,"r")) == NULL)
50     printf("ファイルが開けません\n");
51 else{
52     fgets(temp,L,fr); //ファイルの1行目だけ読み込み、数字の数を数える
53     for(i=0 ;i<L ;i++){
54         if(temp[i]>='0' && temp[i]<='9'){
55             x_n++;
56         }
57     }
58
59     rewind(fr); //もう一度1行目から読み込む
60
61     i=0;
62
63     //ファイルから値を読み込みそれぞれを配列a_bに代入
64     while(!feof(fr)){
65         for(j=0 ;j<x_n ;j++){
66             if(j < x_n-1)
67                 fscanf(fr,"%lf",&a_b[i][j]);
68             else
69                 fscanf(fr,"%lf\n",&a_b[i][j]);
70         }
71         i++;
72     }
73     fclose(fr);
74 }
75
76 //iの値は行数となるのでそれを変数sumに代入
77 sum=i;
78
79 //ファイルから読み込んだ値を行列Aとbに分け、それぞれ配列a,bに代入
80 for(i=0 ;i<sum ;i++){
81     for(j=0 ;j<x_n ;j++){
82         if(j < x_n-1){
83             a[i][j]=a_b[i][j];
84         }
85         else
86             b[i]=a_b[i][j];
87     }
88 }
89
90 printf("A=\n");

```

```

91     for(i=0 ;i<sum ;i++){
92         for(j=0 ;j<x_n-1 ;j++){
93             printf(" %5.2f ",a[i][j]);
94         }
95         printf("\n");
96     }
97
98     printf("\nb=\n");
99     for(i=0 ;i<sum ;i++){
100         printf(" %5.2f\n",b[i]);
101     }
102
103     s_gauss(sum,x,a,b);
104
105     printf("\nx=\n");
106     for(i=0 ;i<sum ;i++){
107         printf(" %5.2f\n",x[i]);
108     }
109
110     return 0;
111 }

```

今回、行列の入力を容易にするため、あらかじめ入力すべき行列を保存したファイルを用意し、そのファイル読み込む形式とした。

1.1 実行例

以下にガウス消去法の実行例をソースコード 2 に示す。

ソースコード 2 実行結果

```

1  lesser [java] $ cat 1-1.dat
2  1 1 1 2
3  2 3 4 4
4  2 1 1 3
5  lesser [java] $ ./gauss 1-1.dat
6  A=
7  1.00 1.00 1.00
8  2.00 3.00 4.00
9  2.00 1.00 1.00
10
11 b=
12 2.00
13 4.00
14 3.00
15
16 x=
17 1.00
18 2.00
19 -1.00
20 lesser [java] $ ./gauss 1-2.dat
21 A=
22 1.00 2.00 1.00 2.00
23 -1.00 -1.00 0.00 -1.00
24 -2.00 -1.00 2.00 0.00

```

```

25    2.00 6.00 1.00 4.00
26
27    b=
28    0.00
29    -1.00
30    -6.00
31    5.00
32
33    x=
34    1.00
35    2.00
36    -1.00
37    -2.00
38    lesser [java] $

```

1.2 解説

まず、main 関数内で作成しておいたデータが格納されたファイルを読み込む。ファイル名は引数でして、ファイル内は行列 **A** と行列 **b** を複合した値を保存しておく。これはソースコード 2 の 1 行目で、cat 関数で表示している。ソースコード 1 の 47 行目で char 型の変数 f_name に引数で入力したファイル名を代入し、49 行目でファイルを開く。ファイルが開けた場合、まず、ファイルの 1 行目のみ読み込み、何列で構成されているかを数字の数を読み込むことにより判定し、変数 x_n に代入する。59 行目の rewind 関数を使いもう一度 1 行目から読み込むようにリセットし、64 行目からデータを読み込む。fscanf 関数でそれぞれ読み込むが、1 行目の終端を読み込む時のみ、改行 \n も読み込む。この読み込んだ値は行列 **A** と **b** が混ざっているの、これらを 2 次元配列 a_b に格納する。80 行目より、2 次元配列 a_b を 2 次元配列 a と配列 b に分ける。次にガウス消去法の関数を解説する。このプログラムではガウスの消去法は 8 行目からの関数で行っている。引数として、行数 n、答えを格納する配列 x、行列 **A** を格納する配列 a、右辺の行列 **b** を格納する行列 b をとる。まず 12 行目からガウスの前進消去を行う。1 回目のループで 1 行目以降の 1 列目を 0 にするように計算する。2 行目が 1 行目の何倍となるかを計算した値を変数 mik に代入する。これを使い mik 倍した 1 行目引く 2 行目をし、1 列目を 0 にする。これを右辺の行列 b にも同じことをする。3 行目も同様に行う。2 回目のループで 2 行目以降の 2 列目を 0 にするように計算する。これを繰り返すことで、以下のような形にする。

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22} & a_{23} & \dots & a_{2n} \\ 0 & 0 & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}$$

次に 23 行目からの交代代入について説明する。後退代入では n 行目から走査を始める。n 行目では $a_{nn}x_n = b_n$ となっているので、 $x_n = \frac{b_n}{a_{nn}}$ が答えとなる。n-1 行の時、 $a_{n-1,n-1}x_{n-1} = b_{n-1} - a_{n-1,n}x_n$ となる。これを関数内では x[n-1] に b[n-1] を代入し、それから $a[n][n-1]*x[n]$ を引いた値を x[n-1] に代入。最後に x[n-1] を $a[n-1][n-1]$ で割った値を x[n-1] に代入し、求める。これを繰り返すことで行列 **x** を求める。

2 LU 分解

LU 分解を行うプログラムをソースコード 3 に示す。

ソースコード 3 LU 分解

```

1 #include<stdio.h>
2
3 #define N 10

```

```

4 #define M 100
5 #define L 100
6
7 void intlz_lu(int n ,double a[] [N] ,double l[] [N] ,double u[] [N]){
8     int i,j;
9
10    for(i=0 ;i<n ;i++){
11        for(j=0 ;j<n ;j++){
12            u[i] [j]=a[i] [j];
13        }
14    }
15
16    for(i=0 ;i<n ;i++){
17        for(j=0 ;j<n ;j++){
18            if(i == j)
19                l[i] [j]=1.0;
20            else
21                l[i] [j]=0.0;
22        }
23    }
24 }
25
26 void lu(int n ,double l[] [N] ,double u[] [N]){
27     int i,j,k;
28     double mik;
29
30     for(k=0 ;k<n-1 ;k++){
31         for(i=k+1 ;i<n ;i++){
32             mik=u[i] [k]/u[k] [k];
33             l[i] [k]=mik;
34             u[i] [k]=0.0;
35             for(j=k+1 ;j<n ;j++){
36                 u[i] [j]-=mik*u[k] [j];
37             }
38         }
39     }
40 }
41
42 int main(int argc ,char *argv[]){
43     int i,j;
44     int sum=0;
45     int x_n=0;
46     double a[N] [N];
47     double l[N] [N],u[N] [N];
48     char f_name[M];
49     char temp[L]={};
50     FILE *fr;
51
52     sprintf(f_name,"%s",argv[1]);
53
54     if((fr=fopen(f_name,"r")) == NULL)
55         printf("ファイルが開けません\n");
56     else{

```

```

57     fgets(temp,L,fr);
58     for(i=0 ;i<L ;i++){
59         if(temp[i]>='0' && temp[i]<='9'){
60             x_n++;
61         }
62     }
63
64     rewind(fr);
65
66     i=0;
67
68     while(!feof(fr)){
69         for(j=0 ;j<x_n ;j++){
70             if(j < x_n-1)
71                 fscanf(fr,"%lf",&a[i][j]);
72             else
73                 fscanf(fr,"%lf\n",&a[i][j]);
74         }
75         i++;
76     }
77     fclose(fr);
78 }
79
80 sum=i;
81 printf("A=\n");
82 for(i=0 ;i<sum ;i++){
83     for(j=0 ;j<x_n ;j++){
84         printf(" %5.2f ",a[i][j]);
85     }
86     printf("\n");
87 }
88
89 intlz_lu(sum,a,l,u);
90 lu(sum,l,u);
91
92 printf("\nL=\n");
93 for(i=0 ;i<sum ;i++){
94     for(j=0 ;j<x_n ;j++){
95         printf(" %5.2f ",l[i][j]);
96     }
97     printf("\n");
98 }
99
100 printf("\nU=\n");
101 for(i=0 ;i<sum ;i++){
102     for(j=0 ;j<x_n ;j++){
103         printf(" %5.2f ",u[i][j]);
104     }
105     printf("\n");
106 }
107
108 return 0;
109 }

```

このプログラムもガウス消去法と同様にファイルからデータを読み込んでいる。ただし、今回は行列 A の部分だけとなっている。

2.1 実行例

以下に実行例をソースコード 4 に示す。

ソースコード 4 LU 分解の実行結果

```
1 lesser [java] $ cat 2-1.dat
2 1 1 1
3 2 3 4
4 2 1 1
5 lesser [java] $ ./LU 2-1.dat
6 A=
7 1.00 1.00 1.00
8 2.00 3.00 4.00
9 2.00 1.00 1.00
10
11 L=
12 1.00 0.00 0.00
13 2.00 1.00 0.00
14 2.00 -1.00 1.00
15
16 U=
17 1.00 1.00 1.00
18 0.00 1.00 2.00
19 0.00 0.00 1.00
20 lesser [java] $ cat 2-2.dat
21 1 2 1 2
22 -1 -1 0 -1
23 -2 -1 2 0
24 2 6 1 4
25 lesser [java] $ ./LU 2-2.dat
26 A=
27 1.00 2.00 1.00 2.00
28 -1.00 -1.00 0.00 -1.00
29 -2.00 -1.00 2.00 0.00
30 2.00 6.00 1.00 4.00
31
32 L=
33 1.00 0.00 0.00 0.00
34 -1.00 1.00 0.00 0.00
35 -2.00 3.00 1.00 0.00
36 2.00 2.00 -3.00 1.00
37
38 U=
39 1.00 2.00 1.00 2.00
40 0.00 1.00 1.00 1.00
41 0.00 0.00 1.00 1.00
42 0.00 0.00 0.00 1.00
43 lesser [java] $
```

2.2 解説

main 関数はガウス消去法とほぼ同じとなっている。この問題では配列 l と u をそれぞれ初期化する関数と実際に LU 分解を行う関数に分けた。7 行目の `intlz_lu` 関数では配列 l と u をそれぞれ初期化している。引数として、配列の行数 n と行列 A、L、U をそれぞれ格納した配列 a、l、u となっている。配列 u は配列 a をそのまま代入する。配列 l は単位行列で初期化する。すなわち、配列の行番号と列番号が等しい時のみ 1.0 を代入し、それ以外は 0.0 を代入する。次に 26 行目で LU 分解を行う `lu` 関数を定義する。引数として、配列の行数 n と配列 l、u を取っている。LU 分解の計算方法は 3×3 の時だと

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{23} & a_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{23} & 1 \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix}$$

となればいいので、行列 U の 1 行目は

$$u_{11} = a_{11}$$

$$u_{12} = a_{12}$$

$$u_{13} = a_{13}$$

となる、次に L の 1 列目を計算する

$$l_{21} = \frac{a_{21}}{u_{11}}$$

$$l_{31} = \frac{a_{31}}{u_{11}}$$

となる。さらに U の 2 行目を計算すると

$$u_{22} = a_{22} - l_{21}u_{12}$$

$$u_{23} = a_{23} - l_{21}u_{13}$$

同様に 3 行目も行うことで計算できる。ソースコード 3 の 26 行目はこれと同様のことを行っている。配列 u はすでに配列 a で初期化しているため。u の 1 行目は計算されている。上記の計算では $\frac{a_{21}}{u_{11}}$ だが、配列 u は配列 a で初期化しているので、`u[2][1]/u[1][1]` で行う。この値を変数 `mik` に代入し、これを `l[1][0]` に代入し、同じ行と列の `u[1][0]` に 0 を代入する。このループ内のまま次の行の配列 u を求める。これを繰り返し LU 分解を行う。

3 ドゥーリトル法

作成したプログラムをソースコード 5 に示す。

ソースコード 5 ドゥーリトル法

```
1 //LU 分解から x を求める
2
3 #include<stdio.h>
4
5 #define N 10
6 #define M 100
7 #define L 100
8
9 //LU 分解をするの関数
10 void lu(int n ,double a[][N] ,double l[][N] ,double u[][N]){
11     int i,j,k;
12     double mik;
13
```



```

14 //U 行列に A 行列の値を代入
15 for(i=0 ;i<n ;i++){
16     for(j=0 ;j<n ;j++){
17         u[i][j]=a[i][j];
18     }
19 }
20
21 //L 行列を単位行列で初期化
22 for(i=0 ;i<n ;i++){
23     for(j=0 ;j<n ;j++){
24         if(i == j)
25             l[i][j]=1.0;
26         else
27             l[i][j]=0.0;
28     }
29 }
30
31 //LU 分解
32 for(k=0 ;k<n-1 ;k++){
33     for(i=k+1 ;i<n ;i++){
34         mik=u[i][k]/u[k][k];
35         l[i][k]=mik;
36         u[i][k]=0.0;
37         for(j=k+1 ;j<n ;j++){
38             u[i][j]-=mik*u[k][j];
39         }
40     }
41 }
42 }
43
44 //LU 分解後の L と行列 b から行列 y を求める
45 void f_gauss(int n ,double y[] ,double a[][N] ,double b[]){
46     int i,j;
47
48     //ガウスの前進代入
49     for(i=0 ;i<n ;i++){
50         y[i]=b[i];
51         for(j=0 ;j<i ;j++){
52             y[i]-=a[i][j]*y[j];
53         }
54         y[i]/=a[i][i];
55     }
56 }
57
58 //LU 分解後の行列 U と行列 y から答えの行列 x をもとめる
59 void r_gauss(int n ,double x[] ,double a[][N] ,double b[]){
60     int i,j;
61
62     //ガウスの後退代入
63     for(i=n-1 ;i>=0 ;i--){
64         x[i]=b[i];
65         for(j=i+1 ;j<n ;j++){
66             x[i]-=a[i][j]*x[j];

```

```

67     }
68     x[i]/=a[i][i];
69 }
70 }
71
72 int main(int argc ,char *argv[]){
73     int i,j;
74     int sum=0;
75     int x_n=0;
76     double a_b[N][N];
77     double a[N][N],b[N];
78     double l[N][N],u[N][N];
79     double x[N],y[N];
80     char f_name[M];
81     char temp[L]={};
82     FILE *fr;
83
84     sprintf(f_name,"%s",argv[1]);
85
86     if((fr=fopen(f_name,"r")) == NULL)
87         printf("ファイルが開けません\n");
88     else{
89         fgets(temp,L,fr); //一列目だけ読み込み、数字の数を数える
90         for(i=0 ;i<L ;i++){
91             if(temp[i]>='0' && temp[i]<='9'){
92                 x_n++; //行列A と b を並べた横の数字の数
93             }
94         }
95
96         rewind(fr); //ファイルをもう一度最初から読み込む
97
98         i=0;
99
100        while(!feof(fr)){
101            for(j=0 ;j<x_n ;j++){
102                if(j < x_n-1) //一列目の最後の値を読み込むときだけ改行も読み込む
103                    fscanf(fr,"%lf",&a_b[i][j]);
104                else
105                    fscanf(fr,"%lf\n",&a_b[i][j]);
106            }
107            i++;
108        }
109        fclose(fr);
110    }
111
112    sum=i; //列数を格納
113
114    //読み込んだ行列A と b が複合された数列を a と b に分ける
115    for(i=0 ;i<sum ;i++){
116        for(j=0 ;j<x_n ;j++){
117            if(j < x_n-1){
118                a[i][j]=a_b[i][j];
119            }

```

```

120         else
121             b[i]=a_b[i][j];
122     }
123 }
124
125 printf("A=\n");
126 for(i=0 ;i<sum ;i++){
127     for(j=0 ;j<x_n-1 ;j++){
128         printf(" %5.2f ",a[i][j]);
129     }
130     printf("\n");
131 }
132
133 printf("\nb=\n");
134 for(i=0 ;i<sum ;i++){
135     printf(" %5.2f \n",b[i]);
136 }
137
138 lu(sum,a,l,u);
139
140 f_gauss(sum,y,l,b);
141
142 r_gauss(sum,x,u,y);
143
144 printf("\nx=\n");
145 for(i=0 ;i<sum ;i++){
146     printf(" %5.2f \n",x[i]);
147 }
148
149 return 0;
150 }

```

3.1 実行結果

以下に実行例をソースコード 6 に示す。

ソースコード 6 ドゥーリトル法の実行結果

```

1 lesser [java] $ ./LU-2 1-1.dat
2 A=
3   1.00 1.00 1.00
4   2.00 3.00 4.00
5   2.00 1.00 1.00
6
7 b=
8   2.00
9   4.00
10  3.00
11
12 x=
13   1.00
14   2.00
15  -1.00
16 lesser [java] $ ./LU-2 1-2.dat

```

```

17 A=
18   1.00 2.00 1.00 2.00
19  -1.00 -1.00 0.00 -1.00
20  -2.00 -1.00 2.00 0.00
21   2.00 6.00 1.00 4.00
22
23 b=
24   0.00
25  -1.00
26  -6.00
27   5.00
28
29 x=
30   1.00
31   2.00
32  -1.00
33  -2.00
34 lesser [java] $

```

実行結果よりガウス消去法と値が一致していることがわかる。

3.2 解説

このプログラムの main 関数はガウス消去法とほぼ同じとなっている。また、10 行目の lu 関数も LU 分解で説明したものとなっている。このプログラムは LU 分解したもののから答えとなる行列 x を求めるものとなっている。LU 分解を行なった結果をそれぞれ配列 l, u に格納している。これらとガウス消去法で行なった、後退代入を使い、答えを求める。LU 分解を行なったあと、以下のように式を書き換えれる。

$$Ax = b$$

$$LUx = b$$

$$Ux = y$$

とおくと、

$$Ly = b$$

となる。これより、 $Ly = b$ を使い y を求め、 $Ux = y$ を使い x を求める。ソースコード 3 の 45 行目より、行列 y を求める。行列 L の形はガウスの後退代入を行うときの形の逆なのでループする向きを逆にすれば良い。また、58 行目で x を求める時は、ガウスの後退代入がそのまま適応できるので、これから x を求めるところで、解を求めた。

4 逆行列のプログラム

行列 A の逆行列は

$$A^{-1} = L^{-1}U^{-1}$$

で求めることができる。3×3 の場合、 L^{-1} と U^{-1} は以下のように求めることができる。

$$\begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{23} & 1 \end{pmatrix} \begin{pmatrix} p_{11} & 0 & 0 \\ p_{21} & p_{22} & 0 \\ p_{31} & p_{23} & p_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ 0 & p_{22} & p_{23} \\ 0 & 0 & p_{33} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

これらはそれぞれ

```

1 for(i=0 ;i<n ;i++){
2   for(j=0 ;j<i-1 ;i++){
3     for(k=j ;k<i-1 ;k++){
4       p[i][j]+=l[i][k]*p[k][j];
5     }
6     p[i][j]/=-l[i][i];
7   }
8 }

```

```

1 for(i=n-1 ;i>=0 ;i--){
2   p[i][i]=1/u[i][i];
3   for(j=i+1 ;j<n ;j++){
4     for(k=i+1 ;k<j ;k++){
5       p[i][j]+=u[i][k]*p[k][j];
6     }
7     p[i][j]/=-u[i][i];
8   }
9 }

```

で求めることができる。これより行列 A の逆数を LU を使い求めることができる。

5 行列式の計算

行列 A の行列式は以下のようにして求めることができる。

$$|A| = |L||U|$$

また、 L と U の行列式はそれぞれ

$$|L| = l_{11} * l_{22} * \dots * l_{nn}$$

$$|U| = u_{11} * u_{22} * \dots * u_{nn}$$

あとはこの二つを掛け合わせると行列 A の行列式が求まる。