２０２２年度前期　数値解析　第８回授業ノート

２０２２年６月９日　三浦憲二郎

スライド１　講義アウトライン

- 連立１次方程式の直接解法
  - 「　　」消去法

スライド２　連立１次方程式　p.38

$$Ax = b$$

- 連立１次方程式の重要性
  - 「　　」の問題は基本的には解けない.
  - 非線形問題を「　　」化して解く.
  - 複雑な構造物,「　　」な要素に分解
  - 各要素に対して「　　」方程式を立てる.
  - 「　　」の原理により統合
  - ただし，変数の数は「　　」
  - コンピュータにより「　　」に解く.

スライド３　ガウス消去法　p.38

連立１次方程式の解法

x－y=1 (1),　x+2y=4 (2)

１．代入法

式(1)より y=x－1, 式(2)に代入 x+2(x-1)=4, したがって 3x=6, よって x=2, 式(1)より y=1

２．加減法

式(2)より式(1)を引く：3y=3, したがって y=1, 式(1)より x=2

ガウス消去法は,「　　」をコンピュータに適した方法で行う.

スライド４　ガウス消去法 p.38

$$\begin{aligned} 3x_0 + x_1 + 2x_2 &= 13 \\ 5x_0 + x_1 + 3x_2 &= 20 \\ 4x_0 + 2x_1 + x_2 &= 13 \end{aligned}$$

$$\begin{bmatrix} 3 & 1 & 2 \\ 5 & 1 & 3 \\ 4 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 13 \\ 20 \\ 13 \end{bmatrix}$$

(1) 第１列の２行目と３行目を０にする.

「第１行×(－5/3)+第２行目」

「第１行×(－4/3)+第３行目」

$$\begin{bmatrix} 3 & 1 & 2 \\ 0 & -\frac{2}{3} & -\frac{1}{3} \\ 0 & \frac{2}{3} & -\frac{5}{3} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 13 \\ -\frac{5}{3} \\ -\frac{13}{3} \end{bmatrix}$$

(2) 第２列の３行目を０にする.

「第２行×1+第３行」

(1), (2)：「　　」消去

$$\begin{bmatrix} 3 & 1 & 2 \\ 0 & -\frac{2}{3} & -\frac{1}{3} \\ 0 & 0 & -2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 13 \\ -\frac{5}{3} \\ -6 \end{bmatrix}$$

(3) x3, x2, x1 の順に代入して答えを求める.

(3)：「　　」代入

スライド５　ガウスの消去法：ｎ元　p.39

$$\begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0,n-1} \\ a_{10} & a_{11} & \cdots & a_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1,0} & a_{n-1,2} & \cdots & a_{n-1,n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \end{bmatrix}$$

（「　　」）

If $a_{00} \neq 0$

$$\alpha_{i0} = -\frac{a_{i0}}{a_{00}}, \quad i = 1, 2, \ldots, n$$

$$\begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0,n-1} \\ 0 & a_{11}^{(1)} & \cdots & a_{1,n-1}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n-1,2} & \cdots & a_{n-1,n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1^{(1)} \\ \vdots \\ b_{n-1}^{(1)} \end{bmatrix}$$

$$a_{ij}^{(1)} = a_{ij} + \alpha_{i0} a_{0j}, b_i^{(1)} = b_i + \alpha_{i0} b_0, i = 1,2,\cdots,n-1$$

スライド６　ガウス消去法：ｎ元　p.40

（「　　」）

$$x_{n-1} = \frac{b_{n-1}^{(n-1)}}{a_{n-1,n-1}^{(n-1)}}$$

$$x_{n-2} = \frac{b_{n-2}^{(n-2)} - a_{n-2,n-1}^{(n-2)} x_{n-1}}{a_{n-1,n-1}^{(n-1)}}$$

$$x_k = \frac{b_k^{(k)} - \sum_{j=k+1}^{n-1} a_{kj}^{(k)} x_j}{a_{kk}^{(k)}}$$

スライド７　ガウス消去法：アルゴリズム　p.41

行列 A とベクトル b の入力

/* 前進消去 */

for $k = 0, 1, \cdots, n-2$
  for $i = k+1, k+2, \cdots, n-1$
    $\alpha \leftarrow -\frac{a_{ik}}{a_{kk}}$
    for $j = k+1, k+2, \cdots, n-1$
      $a_{ij} \leftarrow a_{ij} + \alpha a_{kj}$
    end for
    $b_i = b_i + \alpha b_k$
  end for
end for

/* 後退代入 */

for $k = n-1, n-2, \cdots, 0$
  $b_k \leftarrow \frac{b_k - \sum_{j=k+1}^{n-1} a_{kj} b_j}{a_{kk}}$
end for

b を出力　/* 答えは b に上書き*/

スライド８　ガウス消去法：プログラム　p.41

```
#include <stdio.h>
#include <stdlib.h>
#define N 4 /* N 次正方行列 */
void input_matrix(double a[N][N],char c,FILE* fin, FILE* fout);
void input_vector(double b[N],char c,FILE* fin,FILE* fout);
void simple_gauss(double a[N][N],double b[N]);
int main(void){
      FILE *fin, *fout;
      double a[N][N], b[N];
      int i;
      if((fin=fopen("input.dat","r"))==NULL) exit(1);
      if((fout=fopen("output.dat","w"))==NULL) exit(1);
      input_matrix(a,'A',fin,fout); input_vector(b,'b',fin,fout);
      simple_gauss(a,b);
      fprintf(fout,"Ax=b の解は次の通りです¥n");
      for(i=0;i<N;i++){ fprintf(fout,"%f¥n",b[i]);}
      fclose(fin); fclose(fout);
      return 0;
}
void simple_gauss(double a[N][N],double b[N]){
      int i,j,k;
      double alpha, tmp;
      for(k=0;k<N-1;k++){ /* 前進消去 */
            for(i=k+1;i<N;i++){
                  alpha=-a[i][k]/a[k][k];
                  for(j=k+1;j<N;j++){
                        a[i][j]=a[i][j]+alpha*a[k][j];
                  }
                  b[i]=b[i]+alpha*b[k];
            }
      }
      b[N-1]=b[N-1]/a[N-1][N-1]; /* 後退代入 */
      for(k=N-2;k>=0;k--){
            tmp=b[k];
            for(j=k+1;j<N;j++){
                  tmp=tmp-a[k][j]*b[j];
            }
            b[k]=tmp/a[k][k];
      }
}
```

input.dat の内容

```
1.0     2.0     1.0     1.0
4.0     5.0     -2.0    4.0
4.0     3.0     -3.0    1.0
2.0     1.0     1.0     3.0
-1.0    -7.0    -12.0   2.0
```

output.dat の内容

```
行列 A は次の通りです
 1.00    2.00    1.00    1.00
 4.00    5.00   -2.00    4.00
 4.00    3.00   -3.00    1.00
 2.00    1.00    1.00    3.00
ベクトル b は次の通りです
-1.00
-7.00
-12.00
 2.00
Ax=b の解は次の通りです
-2.000000
-1.000000
1.000000
2.000000
```

スライド９　データ入出力　p.19 (「C 言語入門」p.282)

プログラム 2.3 改

```
#include <stdio.h>
#include <stdlib.h>
#define N 4
void input_matrix(double a[N][N],char c,FILE* fin,FILE* fout);
void input_vector(double b[N],char c,FILE* fin,FILE* fout);
int main(void)
{
    FILE *fin,*fout;
    double a[N][N],b[N];
    if((fin=fopen("input.dat","r"))==NULL){
        printf("ファイルは見つかりません:input.dat ¥n");
        exit(1);
    }
    if((fout=fopen("output.dat","w"))==NULL){
        printf("ファイルは見つかりません:output.dat ¥n");
        exit(1);
    }
input_matrix(a,'A',fin,fout); /* 行列 A の入出力 */
    input_vector(b,'b',fin,fout); /* ベクトル b の入出力 */
    fclose(fin); fclose(fout);
}
void input_matrix(double a[N][N],char c,FILE* fin,FILE* fout){
    int i,j;
    fprintf(fout,"行列%c は次の通りです¥n",c);
    for(i=0;i<N;++i){
        for(j=0;j<N;++j){
            fscanf(fin,"%lf",&(a[i][j]));
            fprintf(fout,"%5.2f¥t",a[i][j]);
        }
        fprintf(fout,"¥n");
    }
}
void input_vector(double b[N],char c,FILE* fin,FILE* fout){
    int i,j;
    fprintf(fout,"ベクトル%c は次の通りです¥n",c);
    for(i=0;i<N;++i){
        fscanf(fin,"%lf",&(b[i]));
        fprintf(fout,"%5.2f¥t",b[i]);
        fprintf(fout,"¥n");
    }
}
```

input.dat の内容

```
1.0   2.0   1.0   1.0
4.0   5.0 -2.0   4.0
4.0   3.0 -3.0   1.0
2.0   1.0   1.0   3.0
-1.0 -7.0 -12.0 2.0
```

output.dat の内容

```
行列 A は次の通りです.
1.0   2.0   1.0   1.0
4.0   5.0 -2.0   4.0
4.0   3.0 -3.0   1.0
2.0   1.0   1.0   3.0
ベクトル b は次の通りです.
-1.0
-7.0
-12.0
 2.0
```