

# 物理学情報処理演習

## 12. データ処理

2017年7月5日  
VER20170516

本日の推奨作業directory

lesson12

12.1 連立一次方程式

12.2 最小二乗法

### 参考文献

- やさしいC++ 第4版 高橋 麻奈 (著)  
ソフトバンククリエイティブ
- プログラミング言語C++第4版  
ビャーネ・ストラウストラップ, Bjarne Stroustrup, 柴田 望洋
- Numerical Recipes: The Art of Scientific Computing, Third Edition in C++

身内賢太郎

レポート提出: [fsci-phys-jouhou@edu.kobe-u.ac.jp](mailto:fsci-phys-jouhou@edu.kobe-u.ac.jp)

## 12.1 連立1次方程式の解法

連立方程式の解法として以下のものがある。

- ・ Gauss-Jordan法 (ガウス・ジョルダン法)
- ・ Gaussの消去法 (ガウスの消去法)
- ・ Gauss-Seidel法 (ガウス・ザイデル法)
- ・ 共役傾斜法
- ・ Cholesky法 (コレスキー法)
- ・ Crout法 (クラウト法)

$N$  個の未知数  $x_j$  ( $j=1, 2, \dots, N$ ) に対して、 $M$  個の方程式

$$a_{11} \cdot x_1 + a_{12} \cdot x_2 + \dots + a_{1N} \cdot x_N = b_1$$

$$a_{21} \cdot x_1 + a_{22} \cdot x_2 + \dots + a_{2N} \cdot x_N = b_2$$

.....

$$a_{M1} \cdot x_1 + a_{M2} \cdot x_2 + \dots + a_{MN} \cdot x_N = b_M$$

があるとき、その解を求めることを考えよう。

ここで、 $a_{ij}$ ,  $b_j$  は既知の数である。

これらの方程式は、行列・ベクトルを用いると

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$$

と書き表すことができる。

これら  $M$  個の方程式の内、 $N$  個が線形独立なら (変数の数  $N$  と同じか、それ以下ならば)、解が一意に定まる。

行列 $\mathbf{A}$ の逆行列 $\mathbf{A}^{-1}$ は

$$\mathbf{A} \cdot \mathbf{A}^{-1} = \mathbf{A}^{-1} \cdot \mathbf{A} = \mathbf{I}$$

( $\mathbf{I}$ は単位行列) となる。これを用いると解は

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

と書き表される。

また、逆行列 $\mathbf{A}^{-1}$ の $i$ 列目のベクトルを $\mathbf{e}^{(i)}$ とすると、 $\mathbf{e}^{(i)}$ は方程式

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{e}^{(i)}$$

$$\mathbf{e}_j^{(i)} = \delta_{ij}$$

の解となる。

以下では、簡単のため $N=3$ として話を進めていく。

# 連立1次方程式の解法: Gauss-Jordan法

方程式を行列・ベクトルを用いてGauss-Jordan法で解く際に、行列式は以下の操作に普遍である性質を使う;

1. 任意の2行を入れ替える。
2. 任意の行をその行と別の行の線形接合で置き換える。

# 連立1次方程式の解法: Gauss-Jordan法

Gauss-Jordan法は、いわゆる消去法による連立1次方程式の解法。

次の連立1次方程式を考える;

$$a_{11} \cdot x_1 + a_{12} \cdot x_2 + a_{13} \cdot x_3 = b_1 \cdots \cdots \cdots (1)$$

$$a_{21} \cdot x_1 + a_{22} \cdot x_2 + a_{23} \cdot x_3 = b_2 \cdots \cdots \cdots (2)$$

$$a_{31} \cdot x_1 + a_{32} \cdot x_2 + a_{33} \cdot x_3 = b_3 \cdots \cdots \cdots (3)$$

(1)式を $a_{11}$ で割り、 $x_1$ の係数を1にする((1)')。

(2)式から(1)' 式を $a_{21}$ 倍したものを引く。

(3)式から(1)' 式を $a_{31}$ 倍したものを引く。

すると次のように(2)式、(3)式の $x_1$ の係数は0になる。

$$x_1 + a'_{12} \cdot x_2 + a'_{13} \cdot x_3 = b'_1 \cdots \cdots \cdots (1)'$$

$$a'_{22} \cdot x_2 + a'_{23} \cdot x_3 = b'_2 \cdots \cdots \cdots (2)'$$

$$a'_{32} \cdot x_2 + a'_{33} \cdot x_3 = b'_3 \cdots \cdots \cdots (3)'$$

# 連立1次方程式の解法: Gauss-Jordan法

同様に

(2)'を $a_{22}'$ で割る((2)'' )。

$a_{12}'$ を(2)''にかけて(1)'から引く

$a_{32}'$ を(2)''にかけて(3)'から引く

$$x_1 + a_{13}'' \cdot x_3 = b_1'' \cdots \cdots (1)''$$

$$x_2 + a_{23}'' \cdot x_3 = b_2'' \cdots \cdots (2)''$$

$$a_{33}'' \cdot x_3 = b_3'' \cdots \cdots (3)''$$

さらに

(3)''を $a_{33}''$ で割る((3)''' )。

$a_{13}''$ を(3)'''にかけて(1)''から引く

$a_{23}''$ を(3)'''にかけて(2)''から引く

$$x_1 = b_1''' \cdots \cdots (1)'''$$

$$x_2 = b_2''' \cdots \cdots (2)'''$$

$$x_3 = b_3''' \cdots \cdots (3)'''$$

# 連立1次方程式の解法: Gauss-Jordan法

これで、

$$x_1 = b_1''' \quad x_2 = b_2''' \quad x_3 = b_3'''$$

と解が求められた。

上記の連立方程式は、以下のようにも表せる；

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \cdot \begin{bmatrix} y_{11} & y_{12} & y_{13} & x_1 \\ y_{21} & y_{22} & y_{23} & x_2 \\ y_{31} & y_{32} & y_{33} & x_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & b_1 \\ 0 & 1 & 0 & b_2 \\ 0 & 0 & 1 & b_3 \end{bmatrix}$$

ここで、

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

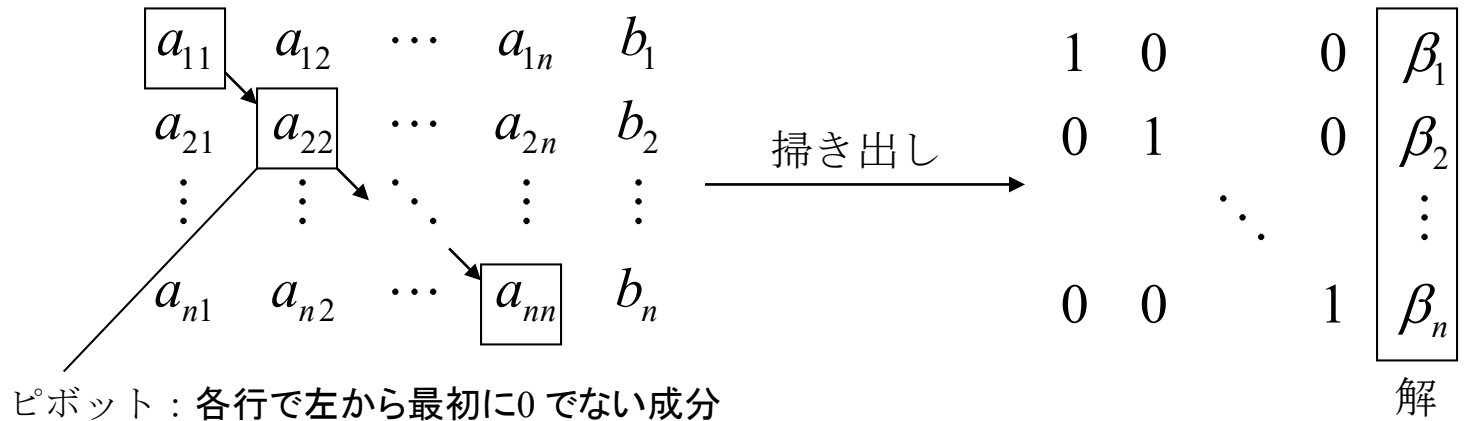
$\mathbf{x} (=x_j)$  は、 $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$  の解

$\mathbf{Y} (=y_{ij})$  は、 $\mathbf{A}$  の逆行列 である。



# 連立1次方程式の解法: Gauss-Jordan法

実際のプログラムは、次のような係数行列を作り、これが単位行列になるように掃き出し演算を行う。



アルゴリズムは以下の通り

ピボットを1行1列から $n$ 行 $n$ 列に移しながら以下を繰り返す。

1. ピボットのある行の要素 $(a_{kk}, a_{k,k+1}, \dots, a_{kn}, b_k)$ をピボット係数 $(a_{kk})$ で割る。結果としてピボットは1となる。なおピボット以前の要素 $(a_{k1}, a_{k2}, \dots, a_{k,k-1})$ はすでに0。
2. ピボット行以外の各行について以下を繰り返す。  
(各行) - (ピボット行)  $\times$  (係数)。なお、この操作についてもピボット以前の列要素についてはすでに0になっている。

# Gauss-Jordan法の実例

次の連立1次方程式を解く実例

$$2x_1 + 3x_2 + 1x_3 = 4$$

$$4x_1 + x_2 - 3x_3 = -2$$

$$-x_1 + 2x_2 + 2x_3 = 2$$

これを先の行列で表すと

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 4 \\ 4 & 1 & -3 & -2 \\ -1 & 2 & 2 & 2 \end{bmatrix}$$

```
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
#define N 3 // 3x3 matrix
int main(int argc, char *argv[]){
    double a[N][N+1] = {{2.0, 3.0, 1.0, 4.0}, {4.0, 1.0, -3.0, -2.0}, {-1.0, 2.0, 2.0, 2.0}};
    double p, a;
    int i, j, k;
    for (k=0; k<N; k++){
        p = a[k][k]; // pivot coefficient
        for (i=k; i<N+1; i++){
            a[k][i]=a[k][i]/p; /* subtract pivot gyou by p */
        }
        for (i=0; i<N; i++){ /* pivot sweep out */
            if (i!=k){
                d = a[i][k];
                for (j=k; j<N+1; j++){
                    a[i][j]=a[i][j]-d*a[k][j];
                }
            }
        }
    }
    for (k=0; k<N; k++){
        cout << "x" << k+1 << "=" << a[k][N] << endl;
    }
}
return 0;
}
```

4ren\_1.cxx

係数の定義

ピボットで割る

## 12.2 最小二乗法

データ点を与えられているとき、データ点を最も再現する理論式 $f(x)$ を求める方法。

理論式:  $f(x) = a_0 + a_1x + a_2x^2 + \dots + a_mx^m$

とデータ点 $(x_i, y_i)$ との距離の2乗は、  
 $(y_i - f(x_i))^2$

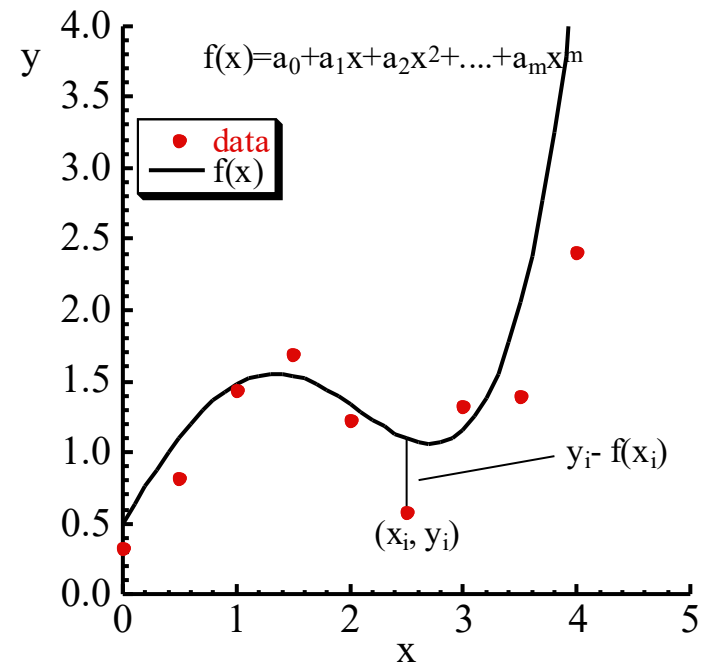
である。したがって各点でのそれらの総和 $I$ は、

$$I(a_0, a_1, \dots, a_m) = \sum (y_i - f(x_i))^2 = \sum \left\{ y_i - \left( a_0 + a_1x_i + a_2x_i^2 + \dots + a_mx_i^m \right) \right\}^2$$

この $I(a_0, a_1, \dots, a_m)$ を最小にする係数の組 $(a_0, a_1, \dots, a_m)$ を求めることは、 $I$ を $(a_0, a_1, \dots, a_m)$ の関数として見て、この関数の最小値を求める問題である。関数の最小値は変数の微分がゼロになる点であるから、

$$\frac{\partial I(a_0, a_1, \dots, a_m)}{\partial a_i} = 0$$

を満足すればよい。したがって、係数の組を変数とした $m$ 個の連立方程式の解を求める問題に帰着する。



$m$ 個のデータ点 $(x_i, y_i)$ を一次式の理論式 $f(x)=a_0+a_1x$ でフィットすることを考えよう。

理論式 $f(x)=a_0+a_1x$ とデータ点 $(x_i, y_i)$ との距離の2乗の総和は

$$\begin{aligned} I(a_0, a_1) &= \sum_i (y_i - f(x_i))^2 = \sum_i \{y_i - (a_0 + a_1x_i)\}^2 \\ &= \sum_{i=1}^m \{(y_i - a_0)^2 - 2a_1(y_i - a_0)x_i + a_1^2x_i^2\} \end{aligned}$$

この $I(a_0, a_1)$ は係数 $(a_0, a_1)$ の2次式である。この $I$ の最小値は2次式の極値であるから各係数による微分がゼロである。よって、

$$\frac{\partial I(a_0, a_1)}{\partial a_0} = 0 \qquad \frac{\partial I(a_0, a_1)}{\partial a_1} = 0$$

を満たす $a_0, a_1$ を求めればよい。

$$\frac{\partial I(a_0, a_1)}{\partial a_0} = ma_0 + \left( \sum_i x_i \right) a_1 - \sum_i y_i = 0$$

$$\frac{\partial I(a_0, a_1)}{\partial a_1} = a_0 \sum_i x_i + a_1 \sum_i x_i^2 - \sum_i x_i y_i = 0$$

より理論式  $f(x) = a_0 + a_1 x$  の  $a_0, a_1$  は

$$a_0 = \frac{\sum_i y_i \sum_i x_i^2 - \sum_i x_i \sum_i x_i y_i}{m \sum_i x_i^2 - \left( \sum_i x_i \right)^2}$$

$$a_1 = \frac{m \sum_i x_i y_i - \sum_i x_i \sum_i y_i}{m \sum_i x_i^2 - \left( \sum_i x_i \right)^2}$$

となる。

課題12: 物理実験データに関して、以下の解析を行え。

① data.datを最小二乗法を用いて、直線でフィットせよ。読み込みに関しては input.cxxと第7講を参照のこと。

( $f(x) = 3.3 + 2.2x$  程度となるはず)

② 上記データには、測定値 $y_i$ について測定誤差 $\sigma_i$ があることが分かっている。このデータを含んだデータdata\_e.datについてカイ2乗を計算、切片および傾きを1%ステップで $\pm 100\%$ 程度スキャンし(2重+1重ループとする必要あり)、カイ2乗が最小になる直線を求めよ。ただしカイ2乗は以下のように定義される。

$$\chi^2 = \sum \left( \frac{f(x_i) - y_i}{\sigma_i} \right)^2$$

③ data\_e.dat、①で求めた直線、②で求めた直線を重ね書きせよ。data\_e.datに関しては、plot “data\_e.dat” using 1:2:3 with errorbars でエラー付き表示が可能。