データの統計処理（基礎）

重田 出

学習の目標

さまざまな確率分布，乱数の発生方法，最小値と最大値，度数分布，最小二乗法を学習し，理解する。

1　さまざまな確立分布

ここでは，代表的な 3 つの分布を示しておく。確率分布は，これ以外にもカイ 2 乗分布，ライス分布など，さまざまなものがある。

1.1　二項分布

ある集団において，特性 A を持つものの割合が p であり，持たないものの割合が q であるとする（$p+q=1$）。このように特性 A の“ある・ない”の 2 つで集団を考えたとき，集団から無作為に n 個を抽出したとき，特性 A を持つものが x 個となっている確率 $f(x)$ を考えてみると，

- n 個のうち x 個を選び出す組合せの数は ${}_n\mathrm{C}_x$ 通り。
- 各組み合わせに対して，x 個が特性 A を持つ確率は p^x，残り $n-x$ 個が特性を持たない確率は q^{n-x}。
- 両者が共に起こるのは，‘ 組み合わせの数 ’ ×‘ 特性 A を持つ確立 ’ ×‘ 特性 A を持たない確立 ’ であり，
  $f(x) = {}_n\mathrm{C}_x \cdot p^x \cdot q^{n-x},\ n = 0, 1, \cdots, n,\ p > 0, q > 0, p + q = 1$
  となる。この分布を二項分布と呼ぶ。

例えば，図 1 に示すように，特性 A を持つものをコインの表，特性 A を持たないものをコインの裏というようにした場合などが相当し，$p = q = 0.5$，$p + q = 1$ である。このとき，10 回のトスを行って半分が表，半分が裏になる確率は，$n = 10$，$x = 5$ として，

$$f(x) = {}_{10}\mathrm{C}_5 \times 0.5^5 \times 0.5^{10-5} = \frac{10!}{(10-5)!5!} \times 0.03125 \times 0.03125 = 0.24. \qquad (1)$$

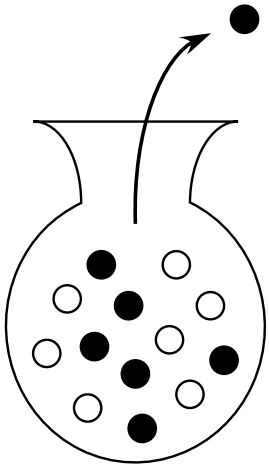

図 1: 二項分布の例
袋の中に黒い玉（特性 A を持つもの）が a 個，白い玉（特性 A を持たないもの）が b 個あるとき，割合 $p = a/(a+b)$, $q = b/(a+b)$。

10 回のうち，すべてが表になる確率は，

$$f(x) = {}_{10}\mathrm{C}_0 \times 0.5^0 \times 0.5^{10-0} = \frac{10!}{(10-0)!0!} \times 1.0 \times 0.000976562 = 0.00097. \quad (2)$$

すなわち，1000 分の 1 以下である。

図 2 は，$p = 0.25$ である集団から n 個を取り出したとき（$n = 10$）および 70 個（$n = 70$）を取り出したときに，特性 A を持つものが x 個である確率を示したものである。

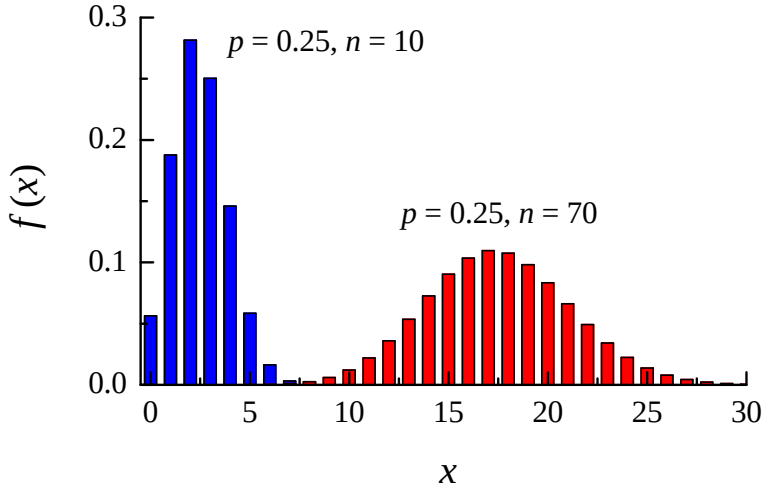

図 2: 二項分布の確率分布関数 $f(x)$。

プログラム 1　階乗と二項係数
順列や組合せの計算に使われる

$$n! = 1 \times 2 \times 3 \times \cdots \times (n-1) \times n \tag{3}$$

$$_n\mathrm{C}_r = \left(\begin{array}{c} n \\ r \end{array} \right) = \frac{n!}{r!(n-r)!} \tag{4}$$

を計算するプログラムを関数の形を用いて作成する。

```
#include<stdio.h>

int main(void)
{
    int int n, r;
    int ncr(int, int);
    printf("n=")
    scanf("%d",&n);
    printf("r=");
    scanf("%d",&r);
    printf("nCr(%d,%d)=%d",n,r,ncr(n,r));
    return 0;
}

int ncr(int n, int r)
{
    int m;
    int kaijou(int);
    m=kaijou(n)/(kaijou(r)*kaijou(n-r));
    return m;
}

int kaijou(int n)
{
    int i,k=1;
    for (i=1; i<=n; ++i)
        k*=i;
    return k;
}
```

演習 1　受講者全員でコイントスを 30 回行い，表の数が 0 回の人から 30 回の人まで，人数を数え，確率分布関数 $f(x)$ をエクセルで描いてみる。
演習 2　$p = 0.5$，$q = 0.5$ として，$n = 50$ のときの確率分布関数 $f(x)$ を求めるプログラムを作成せよ。結果を csv 形式でファイルに書き出し，エクセルで表示せよ。
演習 3　$p = 0.1$，$q = 0.9$ として，演習 2 と同様に確率分布関数 $f(x)$ を求め，演習 1 の結果とどう違うか比較せよ。

1.2 ポアソン分布

時間的あるいは空間的に離散的な自然現象（0 回，1 回，2 回，3 回 … と発生する現象，あるいは 0 個，1 個，2 個，3 個…と数えられる分布）を扱うときに発生する分布である。現象が発生する確率は，時間ないし空間内において一定である。該当する現象としては，

- 1 時間に特定の交差点を通過する車両の台数，あるいは特定の交差点を通過する車両の通過間隔。
- 1 日に受け取る電子メールの件数，あるいはメールとメールとの時間間隔。
- 1 時間に窓口を訪れる訪問者の数，あるいは訪問者と訪問者の時間間隔。
- 1 立方光年あたりの恒星の個数。
- 1 分間に単位面積に飛来する宇宙線の数，あるいは飛来する宇宙線の時間間隔。
- 単位時間当たりに放射性同位元素が崩壊する数，あるいは放射線が放射される時間間隔。

などがある。これらの例は，1 次元ポアソン過程と呼ばれる。1 次元ポアソン過程には，各時間内で事象が発生する回数を確率変数とする離散ポアソン分布と，待機時間を確率変数とする連続アーラン分布の両方がある。現象が単位時間に発生する平均回数を λ とすると，現象が x 回発生する確率 $f(x)$ は，

$$f(x) = \frac{e^{-\lambda}\lambda^x}{x!} \quad \text{ただし}, x = 0, 1, \cdots, \ \lambda > 0 \tag{5}$$

で与えられる。λ が大きくなると正規分布に近づく。

1.3 正規分布

自然現象の多くは，正規確率分布（正規分布）で表されることが多い。2 つのパラメータ，母平均 μ，母分散 σ^2 を持つ正規分布は，$N(\mu, \sigma^2)$ と表記される。

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{-(x-\mu)^2}{2\sigma^2}\right\}, \quad -\infty < x < \infty, \ \sigma^2 > 0. \tag{6}$$

平均 $E(x)$，分散 $V(x)$ は $E(x) = \mu$，$V(x) = \sigma^2$ である。変数変換 $z = (x - \mu)/\sigma$ をほどこしたとき（この変数変換のことを標準化と呼ぶ），確率変数 z は，平均値 0，分散 1 の正規分布に従い，$N(0, 1^2)$ と表される。これを特に，標準正規分布と呼ぶ。

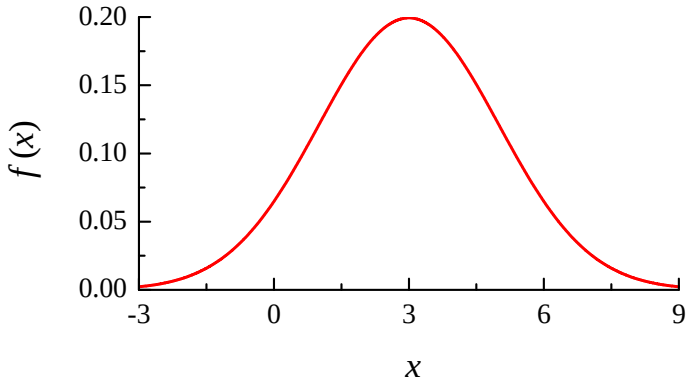

図 3: 正規分布 $N(3, 2^2)$ の概形。

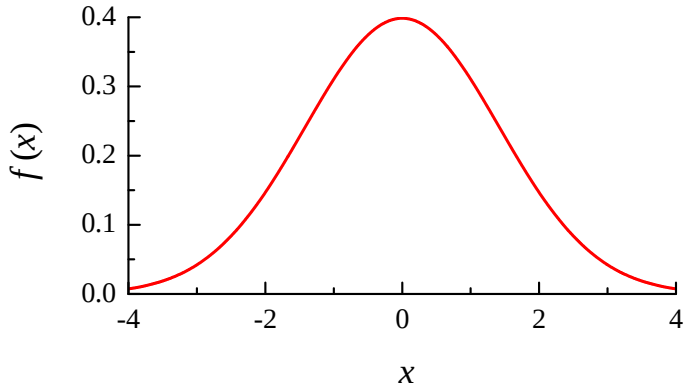

図 4: 標準正規分布 $N(0, 1^2)$ の概形。

2　乱数を発生

確率・統計に関する計算をするときやシミュレーションなどでは，乱数がよく使われる。C 言語には，乱数を発生させる関数として rand() が良く知られている。これは，0 から最大値 RAND_MAX までの範囲の整数値をランダムに返してくれる 関数である。この範囲で発生確率は一様である。RAND_MAX は，あらかじめ C 言語の翻訳システムに組み込まれている値があるが，各自で再設定することもできる。Visual Studio では RAND_MAX の初期値として 32767 が設定されている。プログラム 1 は，1 から 6 までの数をランダムに発生するプログラム，すなわちサイコロのプログラムである。サイコロは 10 回振られる。

プログラム 2　一様乱数を発生させる

```
#include<stdio.h>
#include<stdlib.h>                    /* rand を使うときのヘッダ */
```

```
int main()
{
    int sai, i, j;                /* 整数変数の定義 */
    for(i=0; i<10; i++)
    {
        j=rand();                 /* 乱数を発生させる */
        sai=j%6+1;                /* 1から6までの数にする */
        printf("出た目は = %d\n",sai);    /* 結果を表示 */
    }

    printf("RAND_MAX = %d\n",RAND_MAX);  /* RAND_MAXの表示 */
    return 0;
}
```

このプログラムを走らせると，いつも同じサイコロの目がでる。これは，乱数を発生させるときの初期値がいつも同じであるためである。このことから，逆に乱数はある種の計算により作られていることがわかる。このような乱数を擬似乱数と呼ぶ。初期値を変更するには，srand(初期値) を使う。具体的には，プログラム2のように，符号なしの整数をsrandの値として入れる。この値を変更すれば，異なる乱数が作れる。

演習4　プログラム1を変更し，0から9までの一様乱数を発生させるプログラムを作成せよ。

プログラム3　乱数の初期値を変える

```
#include<stdio.h>
#include<stdlib.h>              /* randを使うときのヘッダ */

int main()
{
    int sai, i, j;              /* 整数変数の定義 */
    srand((unsigned)100);       /* 初期値を変える */
    for(i=0; i<10; i++)
    {
        j=rand();               /* 乱数を発生させる */
        sai=j%6+1;              /* 1から6までの数にする */
        printf("出た目は = %d\n",sai);    /* 結果を表示 */
    }

    printf("RAND_MAX = %d\n",RAND_MAX);  /* RAND_MAXの表示 */
    return 0;
}
```

3　頻度分布

プログラム1で，1から6の発生回数を数えるプログラムを考えてみる。もっとも単純な方法は，各数値に対応したカウンタを用意し，乱数を発生させるごとに

その数値を調べ，その数値に対応したカウンタの数値を 1 増やす，という方法である。プログラム 1 では，カウンタは 1 が発生したときに 1 増やすもの，2 が発生したとき 1 増やすもの，･･･，6 が発生したときに 1 増やすもの，まで 6 つ用意すればよい。数字の判定には，`if` 文を用いればよい（複数の数を判定するときには，`switch` 文 +`case` 文を使う方法もある）。

演習 5　プログラム 1 で 100 回乱数を発生させ，頻度分布を求めるプログラムを作成せよ。

4　平均値，標準偏差

データが N 個あり，各データの値が x_i $(i = 1, 2, 3, \cdots, N)$ であるとすると，平均値 $\bar{x}$ は，

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{7}$$

により求められる。

正規分布では，データのばらつき具合を表す量として，標準偏差が用いられる。標準偏差 σ は，

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2 - \bar{x}^2} \tag{8}$$

により与えられる。$(x_i - \bar{x})^2$ はデータの離れ具合（距離）と考えることができるので，$\dfrac{1}{N}\displaystyle\sum_{i=1}^{N}(x_i - \bar{x})^2$ は，全体としてデータが平均からどの程度ばらついているか（平均距離）を意味している。

演習 6　プログラム 1 で乱数を発生させ，平均値を求めるプログラムを作成せよ。次に，繰り返し回数を 50 回，100 回と変えて，値がどうなるか調べよ。

（ヒント）繰り返し回数が 50 回のときの平均の求め方は，

```
mean=0.0;
for (i=0; i<50; i++)
    mean+=(double)sai;
mean=mean/(double)50;
```

なお，これは一例であり，別の書き方もある。繰り返し回数が 100 回のときも同じプログラムが使えるように，回数の変更を容易にする工夫をせよ。`math.h` のインクルードを忘れないこと。

また，`mean+=(double)sai;` の部分を `mean=+(double)sai;` とすると，うまく動かない。なぜか？

5　データの直線近似

図 5 のような $x-y$ の二次元座標上に，N 個の点があったとし，x と y の関係を傾き a，切片 b の直線で近似するとする。このとき，近似直線から各点がどれだけ離れているかは，

$$E^2 = \{y(x_i) - y_i\}^2 = \{(a \cdot x_i + b) - y_i\}^2 \tag{9}$$

という量で評価することができる。

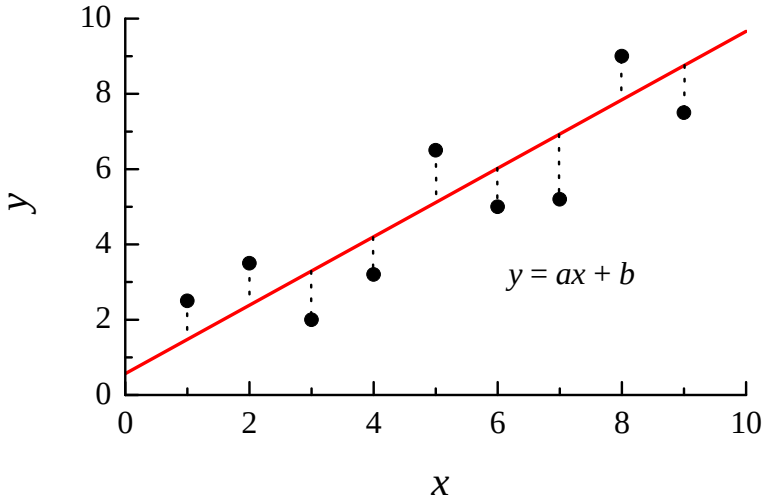

図 5: 2 次元のデータとそれを近似する直線。

演習 7　下記のデータを横軸を x，縦軸を y としてグラフ用紙に書き込み，この点を近似する直線を適当に引く。このときに，図 5 のようにして直線と各点との距離をグラフから読み取り，E^2 の値を求める。次に，切片 b を一定値（ここでは，0.57）として，傾き a を変えた直線を引き，E^2 の値を求める。この操作を様々な傾き a に対して行い，E^2 が a に対してどのように変化しているか調べよ。
（コメント）: 1 人でこの作業を行うのは大変なので，切片 b を同じにしたままクラスの各自で異なる傾きに対する E^2 を分担して計算する。

x	1	2	3	4	5	6	7	8	9	10
y	2.0	2.7	2.4	3.5	5.8	6.0	7.2	7.4	8.8	9.9

最適値 （傾き）$= 0.91$,（切片）$= 0.57$

演習 8　上記の結果から，傾き a の最適値を求めるにはどうしたら良いか考えよ。
演習 9　以上の考察に基づき，データを最もよく近似する直線を求めるプログラムを作成せよ。