6　乱数を利用した実験

6.1　初めに

未来のことは分からない。十年後の経済情勢の話でなくても、今作っている料理でさえどんな味に仕上がるか食べてみるまで分からない。同じ材料を用いて、同じ手順で調理しても、いつも同じ味になるとは限らない。まあそんな料理の下手な人ばかりでもあるまいが。ところで、「料理はする度に味が違う」という現象を科学的な目で見るとどうなるだろう。全く同じ条件の材料を使い、まったく同じ手順で調理するならば、同じ味のものができるはずなのである。作るたびに味が違うというのは、結局は用いた食材の持ち味 (構成栄養素の混合割合) や熟成度合いの違いがあるために、完成品の出来具合が変化するのである。とはいえ、まったく同じ条件をそろえるのは難しい。雨の日が続けば野菜は味が薄くて柔らかくなるだろうし、部屋の温度が違えば、下ごしらえの時の味のしみ方も違ってくるだろう。我々の実際の生活は完全に制御することが難しい不確定なことが満ち溢れているのである。

一方、計算機の世界は単純である。決められた手順を正確に繰り返すだけである。[1] ここには次にどんな数字が出てくるか予測のできない、サイコロを振った時に出る目のような不確定さは一切ない。しかしながら、計算機は短い時間に大量のデータを扱うことができるので、我々の身のまわりにありふれた不確定さを計算機で再現できたら便利に違いない。実は計算機にも擬似乱数とよばれる概念があって、サイコロもどきの機能を実現することができる。厳密に言えば「完璧なランダム」ではないのだけれど、気をつけて用いさえすればいろいろなことに利用できる。この擬似乱数の発生方法について論じようとすると分厚い本が書けてしまうほど奥の深い問題である。こうした話題には深入りしないで、ここでは誰かが作ってくれたプログラム (ライブラリプログラムなどと呼ばれる) を部品として自分のプログラムに組み込んで用いることで、二三の 擬似実験(シミュレーション)をしてみよう。

6.2　乱数の発生

今回の演習で用いているコンパイラには、区間 [0,1] の一様乱数 (読んで字の如く 0 ～ 1 の間の数が等しい確率でランダムに出て来る乱数) を発生する組込み関数 “rand” が用意してある。関数 “rand” は引数を取らずに呼び出すと、0 と 1 の間の実数をランダムに返す。例えば、

```
...
      do i = 1, nr
        r = rand()
        print *, r
      end do
...
```

のように用いる。“r”, “i” などの変数を適切に宣言して、変数 “nr” に発生したい乱数の個数を入れておくと、その数だけ乱数列が画面に表示される。

うまく乱数列が得られたら、この数列が本当にランダムな数字の列になっているか調べよう。「本当に」のところは実はとてつもなく深い。これを追求していくと再び本が一冊書けてしまう。ここでは妥協して乱数列の分布を調べるに止めておこう。区間 [0, 1] の範囲に 20 個の一様乱数が得られたとすると、0 から 0.2 の間には平均して 4 個、0.2 から 0.4 の間にも平均すれば 4 個…の数字が出現するはずである。

問題: 本当に一様に分布するか、実際に手を動かしてヒストグラム (図 1 のようなグラフ) を描いてみよ。

[1] 時としてこの正確さは腹立たしいものとなる。プログラムをミスタイピングしたときでさえプログラムを書いた人のこうして欲しいんだという意図を一切 慮(おもんばか)ってはくれず、指示されたとおりの手順を繰り返して大量のゴミのような出力を出す。

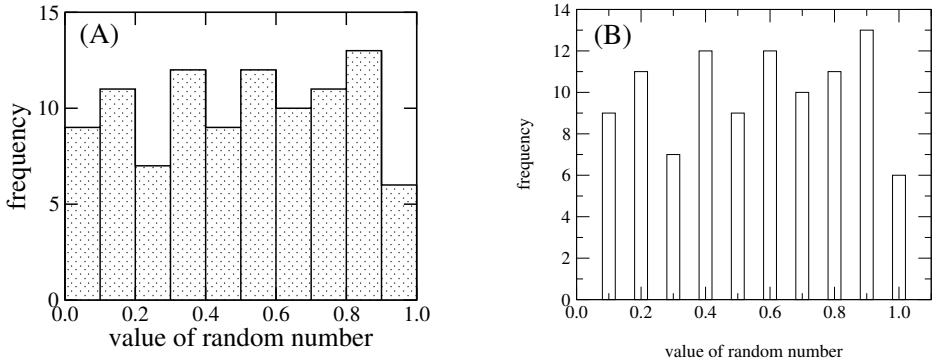

図 1: 発生した乱数の分布を示すヒストグラム。関数 rand を 100 回呼んで得られた数値を 0–0.1, 0.1–0.2, ..., 0.9–1.0 の 10 階級に分けて、それぞれの階級に属する数字の出現回数 (度数) を示した。(A) が良い例で、(B) は悪い例である。(A) では区間 [0.2, 0.3] の数字の出現回数が 7 回という具合に読み取ることができるが、(B) ではあたかも 0.2 という数字が 7 回現れたように見えてしまう。このように数値のラベルの上に棒グラフを描かないこと。

得られたヒストグラムは、苦労した割には平らでない期待はずれのものであったであろう。平らな分布が得られない理由について、少し詳しい説明は TA がしてくれることになっているので難しそうなことはそちらに任せて、ここでは直感的な説明をしよう。乱数がたとえ一様な分布を持っていたとしても、区間が 5 つで乱数の数が 4 つしかなければ必ずどこかの階級は度数が 0 になってしまう。この状態で 5 つ目の乱数を発生させたら、今まで度数が 0 であった階級に入るであろうか。乱数というものの性質を考えるとそれはまずい。次にくる数がここからここまでの範囲の中に必ず入ると分かっているようでは、それは乱数とは言い難い。乱数列には規則性があってはいけないので、5 つある階級を端から順に埋めるようなことをせずに同じ階級の数値が続けて出現するかも知れない。しかし沢山の乱数を発生させるならば、特定の階級ばかりが多く出現する確率は十分に低いものになる。一様に分布していないように見えたのは発生させた乱数の数が少ないためだったのである。

そこで今度は発生する乱数の数を 2000 に増やして同じ実験を繰り返してみよう。20 個でも結構面倒だったので 2000 個なんて多分やりたくないだろうから、勘定するのにも計算機を使ってみることを考えよう。簡単に思いつくのがこんなイメージのプログラムであろう。

```
class1 = 0
class2 = 0
 ...
do i = 1, 2000
  r = rand()
  if (r < 0.2d+0) then
    class1 = class1 + 1
  else if (r < 0.4d+0) then
    class2 = class2 + 1
  ...
  else
    class5 = class5 + 1
  end if
end do
print *, class1, class2, class3, class4, class5
```

「do i = 1, 2000」のループの先頭で一つずつ乱数を発生して、次に続くブロック if 文で各階級に篩分け、各階級のカウンターを一ずつ増して数を勘定する。これで十分なのだけれど、階級の数が固定

されているのが不満である。上のプログラムを見るとclass1～class5について同じ手続きを繰り返している。class1、class2 等の1、2、3、...のところを配列の添え字にして、doループの中に入れれば、階級の数も5に固定されず10でも20でも増やせるし、プログラムもスマートになるだろう。篩分けの所だけ示すと

```
...
      do i = 1, 2000
        r = rand()
        do j = 1, 5
          if ((j - 1)*0.2d+0 <= r .and. r < j*0.2d+0) then
            class(j) = class(j) + 1
          end if
        end do
      end do
...
```

こんな感じである。

> 練習問題 13: このプログラムの断片を参考に度数を数えるプログラムを完成せよ。
> ヒント: 初期化を忘れずに。

例えば、

```
~$ ./a.out
 input number
2000
    4.68777E-05|----+----1----+----2----+----3----+----4----+----5
0.00-0.20:  396:******************
0.20-0.40:  380:******************
0.40-0.60:  435:**********************
0.60-0.80:  393:******************
0.80-1.00:  396:******************
    0.99999    |----+----1----+----2----+----3----+----4----+----5
```

こんな感じに、度数を数えるだけでなくグラフも作ってくれたらクールだよね。ヒストグラムを上のような形で出力する関数“histgram()”をダウンロードできるようにしておいた。この関数は、引数としてデータの個数、データ、階級の数を与えると内部で篩い分けをして、アスキーアート(?)の形で出力する。上の例ではデータの個数は2000、階級数は5としてプログラムを実行した。2000個のデータの内、最小値が4.68777×10^{-5}、最大値が0.99999で、この二つの値の間を5等分して階級を作る。「0.20-0.40」のように表示された範囲は、最小値と最大値からなる区間の20%から40%を意味している。この関数“histgram()”を用いたプログラムの例を示すと、

```
      integer :: histgram, rc, nx, ncls
      double precision :: x(10000)
...
      rc = histgram(nx, x, ncls)
      if (rc /= 0) then
        print *, 'error !!'
        stop
      end if
...
```

関数“histgram”は正常にデータ処理ができた場合には 0 を返し、与えられたデータが不適切で正常に処理ができない場合には0以外の数を返す。第一番目の引数(“nx”)と第三番目の引数(“ncls”)は整数型で、それぞれデータの数とヒストグラムの階級数を与え、第二引数はデータを入れる倍精度実数の配列である。配列の大きさはデータが入る大きさが確保してあれば任意で良い。

関数 "histgram" の内部では、この演習で扱わない文法も用いているのだが、そんなことは意識せずに利用することができる。また、この例で示したように、FORTRAN 言語の外部関数は単に三角関数を計算すると言った単純な作業をするだけでなく、複雑なことをさせることもできることが分る。

> 練習問題 14: 参考プログラムをダウンロードして、この関数を呼ぶメインプログラムを書け。
> ヒント: 関数 "histgram" の「end 文」の後ろに自分のプログラムを書けばよい。

6.3 中心極限の定理と誤差

さて、乱数は使えるようになっただろうか。ここで応用問題として誤差について考えよう。実験レポートなどで「不純物」と並んで考察の双璧をなすあの「誤差」である。実験誤差について統計学的に考える時に重要なのは中心極限定理と呼ばれる数学の定理である。その定理の主張するところによれば、無数の誤差の原因がそれぞれ独立に働いて測定結果が変動するときには、測定値の分布は正規分布 (Gauss 分布とも言う) に従う。無数の誤差の原因が積み重なるというのは正に我々の身のまわりで起きていることで、それ故に何かの量の測定を行なうと、測定値の分布は多くの場合正規分布になる。

恐らく一番簡単な例は酔歩の問題と呼ばれるものである。へべレケに酔っ払ったオッサン (勿論オバハンでも良い) の歩みを想像してみよう。ご当人は真っ直ぐ歩いている積りなんだろうけれど、それはなにしろ酔っ払いのこと。右にフラフラ左にフラフラ。一歩進んでは止まり、次の一歩をどちらの方向に踏み出すか見当がつかない。50 歩進んだそのあとで、果してやっこさん何処にいるのかというのが酔歩の問題である。測定値が大きくなるような変動が右への一歩、小さくなるのが左への一歩と問題を読み替えれば、無数の無秩序な変動が積み重なって生じる誤差と、酔っ払いの運命は同じなのである。実はこの酔歩の問題は原子の拡散現象[2] のモデルでもあり、答えは知られている。中心極限の定理が主張する通り、酔歩の結果は正規分布をしているのである。[3]

さて、この正規分布をしている誤差を定量的に扱えるようにしよう。式を使って表現すれば、平均値が μ になるような測定で $[x, x+dx]$ の区間の測定結果が得られる確率は、

$$P(x)dx = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) dx$$

となる。ここで、σ は標準偏差と呼ばれる量であり、測定結果のバラツキの度合いを示す量になっている。この分布を $N(\mu, \sigma^2)$ と表わすこともある。

問題: 上の式で与えられる分布の平均と分散がそれぞれ μ、σ^2 で与えれらることを示せ。

ヒント: 積分公式

$$\int_0^\infty \exp(-x^2)dx = \frac{\sqrt{\pi}}{2}$$

を用いると良い。分布の平均 M と分散 V は、それぞれ

$$M = \int_{-\infty}^{+\infty} xP(x)dx、\quad V = \int_{-\infty}^{+\infty} (x - \langle x \rangle)^2 P(x)dx$$

で計算される。ここで $\langle x \rangle$ は確率変数 x の平均値を表わす。ちなみに標準偏差は分散の平方根で与えられる。

多数の独立な原因が積み重なってできる誤差をシミュレートしよう。もともとがある大きさ μ だった量に、無数の小さくて無秩序な変動を加える。初めの大きさ μ に、小さな数に一様乱数を掛けたもの

[2] 原子の拡散現象については、学生実験 I のテキストの「拡散現象」の項目の Appendix B に解説を書いた。テキストは http://www.eng.hokudai.ac.jp/labo/lmsm/jikken/text/diffusion/text.pdf に置いてある。

[3] 本当は二項分布。歩数の大きな極限で正規分布に近づく。

を幾つも加え合せる。再びプログラムのイメージを示せばこんな感じ。

```
...
      delta = 1.0d+0
      x(i) = 200.0d+0
      do j = 1, 30
        r = delta * (2.0d+0 * rand() - 1.0d+0)
        x(i) = x(i) + r
      end do
...
```

厚さ 0.2mm(200 ミクロン) の板材を製造しているのだけれど、ローラーの歪や回転速度の不均一、材料の温度分布、更には工場の隣りで道路工事が始まり振動が伝わる、暑い夏の盛りには電力消費が逼迫して電圧降下が起きた (北海道なら送電線への着雪で瞬電が起きたという方が現実的か) 等など、制御しがたい様々な不確定な要因があって、製品の厚みは一様にならない。誤差の原因が 30 個あって、それぞれは最大 ±1 ミクロンのランダムな変動を与える。こんな状況をシミュレートしたものである。

> 練習問題 15: 上のプログラムの断片を完成して、こうした製品を 10000 個作った時の厚さの分布の平均と分散を計算せよ。厚さの平均は予想通りに $\mu = 200$ ミクロンになっているか。分散は幾らになるか。

平均値と分散を計算するだけでなく、練習問題 14 のプログラムを用いてヒストグラムを描くことができれば面白い。得られた分布は Gauss 分布に近いものであるか。

6.4 サンプルからの分布の推定

計算機のなかで実験するだけなら 1000 個であろうと 10000 個であろうと簡単にできてしまうが、現実の実験は大変である (Vickers 硬さ試験を 1000 回繰り返すことを想像してみよう!)。それでも、現実の生産現場では出荷する製品の品質管理をしないわけにはいかない。そこで普通は抜き取り検査を行なう。ここでは少数の抜き取られたサンプルの検査 (標本検査) から、全体の分布 (平均と分散) を調べる方法を体験してみよう。

まず、前節の手続きで、正規分布を持った 10000 個の測定値を準備する。その中から 5 個をランダムに選んで (というか、元々がランダムな集合なのだから最初から 5 つを取れば良い)、その平均を全体の平均値と比べることを考える。余程運が良くなければ、たまたま選ばれた 5 個のデータからなる標本の平均値は、丁度 $m = 200$ ミクロン とはならないであろう。サンプルの平均と全体の平均値 (真の値) との関係はつかないものだろうか。実は一回だけサンプルを取ってその平均値を求めるのではなく、次の 5 個の正規乱数の平均値、更に次の 5 個の平均値という具合にサンプルを次々に選び、それぞれのサンプルの平均値をデータとする集合を考えてその分布を調べると、真の値を平均に持つ正規分布をしていることが知られている。サンプルの「平均値の平均」は真の値 (母集団の平均) と等しいのである。ここで重要なのは、サンプルの平均値の分散はもとのデータの分散より小さくなることである。定量的に表現すると、実験データが正規分布 $N(\mu, \sigma^2)$ に従うなら、n 個のデータの平均値は $N(\mu, \sigma^2/n)$ に従う正規分布になる。サンプルの平均値は個々の測定値よりも真の値により近い所に存在する確率が高いということである。測定を繰り返して幾つもデータを得たら、とりあえずその平均を取るという統計学的な理由はここにある。

データの分散 (及び標準偏差) についてはどうなるだろう。こちらも 5 個のサンプルの分散

$$V = \frac{(x_1 - \bar{x})^2 + (x_2 - \bar{x})^2 + (x_3 - \bar{x})^2 + (x_4 - \bar{x})^2 + (x_5 - \bar{x})^2}{5} \quad (\bar{x} \text{ は } x_i \text{ の平均})$$

の期待値が母集団の分散と等しくなっているだろうか。実は標本を取って全体の分散を推定するためには、除数は標本の大きさ n ではなく、$n-1$ でなければならないことが知られている。標本の大きさから 1 を引いた数で残差の二乗和を割って得られる量

$$s^2 \equiv \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

のことを不偏分散と呼ぶ。上の 5 個のサンプルの例では、5 で割っているところを 4 で割るように改めると、得られる値の分布の平均値は母集団の分散に一致する。因みに標本の不偏分散は χ^2 分布 (上の例では自由度 4 の χ^2 分布) と呼ばれる分布を持つことが知られている。

練習問題 16: 練習問題 15 で作ったシミュレーションのデータを用いて、5 個のサンプルの平均値を 1000 個用意して平均と分散を計算せよ。標本平均の平均値は $\langle x \rangle = 200.0$ になっているか。平均値 (の分布) の分散は母集団の分散より小さいか。

練習問題 17: 練習問題 16 で用意した標本の分散の分布を調べよ。標本の分散の平均を母集団と比較し、「普通に」計算した標本の分散と不偏分散のどちらが母集団の分散に近いか調べよ。

6.5 Monte Carlo シミュレーション

まだスペースがあるので課題をもう一つ。区間 $[0, 1]$ の一様乱数を二つずつ組にして座標平面上にプロットすることを想像してみよう。乱数列を $r_1, r_2, r_3, \ldots$ とした時に、点 $(x, y) = (r_1, r_2), (r_3, r_4), \ldots$ を考えるのである。良くできた乱数を用いれば、プロットされた点は一辺の長さが 1 の正方形の中に均一に分布するであろう (図 2 参照)。分布が均一であれば、これらの点のうちで原点からの距離が 1 未満の点 (図中の影をつけた領域にある点) の数の割合は四分円の面積 $\pi/4$ になると期待できる。図 2 では 14 点のうち 11 点が影をつけた領域にあるので、$\pi/4 \sim 11/14$。故に円周率の近似値として $\pi \sim 11/14 \times 4 = 3.1428$。これを利用すると円周率 π の近似値を実験的に求めることができる。プログラムの一部を示すと、

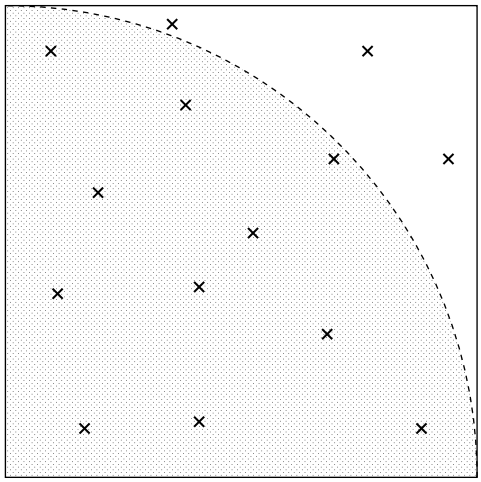

図 2: 乱数の組の二次元プロット。破線は原点からの距離が 1 になる円弧。

```
...
      read *, n
      do i = 1, n
        x = rand()
        y = rand()
        if (x * x + y * y < 1.0d+0) then
          in = in + 1
        end if
      end do
      print *, 'pi = ', ...
```

こんなイメージ。計算機上で乱数を用いた数値実験を、一般に Monte Carlo シミュレーションと呼ぶ。この計算も Monte Carlo シミュレーションの一種である。

練習問題 18: 上述のアイディアに基づき円周率の近似値を求めよ。試行回数が増加すると計算値は真の値に近づくか実験せよ。