環境科学計算機実験

担当：高須 夫悟

takasu@es.nara-wu.ac.jp

- 擬似乱数の生成と応用
- 5月11, 18, 25日の 3回 —> 集中講義8月中
- レポートで成績を評価する

1

高須担当分では擬似乱数の生成に取り組みます。5月の3回分を実施する機会を逸してしまったので、皆さんはこの資料を自分で読み、問題に取り組んで下さい。レポート提出により成績をつけます。

現象のモデル化

- 多くの自然現象は、数理モデル、として記述できる
- 物体の自由落下、電気回路、放射性物質の崩壊、化学反応、生物集団の増減などは微分方程式で記述できる（初期値を決めれば振る舞いが一意に決まる決定論的モデル）
- 自然界には確率的に起こる現象がある（確率論的モデル）
- 確率論的モデルを実装するために乱数の生成が必要
- 乱数にもいろいろある。最も基本的なものが一様乱数

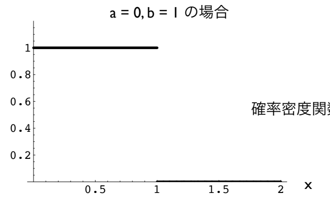
2

自然現象の多くは数理モデルとして記述されます。ニュートンの運動方程式、電気回路、理想的な化学反応、生物集団の増減などは微分方程式（物体の位置、電流、化学物質濃度、集団サイズに関する微分方程式）で記述されます。微分方程式は初期値が決まれば解が一意に決まる「決定論的モデル」と呼ばれます。一方で自然界には確率的に起こる出来事があります。このような現象を表すには確率論的モデルが必要になります。確率論的モデルを実際に実行するためには「乱数」の生成が必要となります。

一様乱数

- 区間 [a, b) で一様な乱数 X を考える。X は a から b の値を同じ確からしさで取る確率変数

$$\mathrm{Prob}[x < X < x + dx] = \begin{cases} \frac{1}{b-a}dx & a \leq x < b \\ 0 & \text{Otherwise} \end{cases}$$



3

最も基本的な乱数に、一様乱数、があります。連続量 X がある区画のどの値も同じ確からしさで取る確率変数です。ここではaからbの間の値を取る一様乱数の確率密度関数を記しています。

一様乱数

- 区間 [0, 1) で一様な乱数 U[0, 1) から任意の一様乱数を生成可能

$$U[0,1) \times a = U[0,a)$$

$$U[0,1) + b = U[b, 1+b)$$

- どうやって U[0, 1) を生成するか？

4

一様乱数の中でも0以上1未満の値を取る一様乱数（記号では U[0, 1)と記す。Uは一様 uniformのU）が最も大切です。というのも U[0, 1) を生成できれば、これにある定数を掛けたり足したりすることで、任意の区画の一様乱数を生成できるからです。また、適当な変数変換によって指数分布や正規分布など様々な分に従う乱数を生成できます。それではどうやってU[0, 1)を生成するか？について古くから研究がなされています。

擬似乱数の生成

- 擬似乱数をソフトウェア的に生成するアルゴリズムには幾つかある

線形合同法

$$X_{i+1} = (A \times X_i + B) \mod M$$

A mod B は整数 A を整数 B で割った余り

適当な整数値定数 A, B, M ($M > A, M > B, A > 0, B > 0$) を用いて、
X_0 を乱数の種 seed として逐次乱数列 X_i を生成する方法

最大値は $M - 1$、周期は最大で M

5

乱数といってもここではアルゴリズム的に生成する乱数を指します。本当の乱数ではないので擬似乱数と呼びます（本当の乱数をアルゴリズム的に生成できるか？という哲学的問題はここでは取り扱わない）。ある区画の擬似乱数（整数値）を生成するアルゴリズムとして、線形合同法、があります。この漸化式により整数値Xを次々と生成します（単なる差分式）。

線形合同法

- 初期値 X_0 を与える漸化式に他ならない
- A, B, M をうまく選ぶとそれなりの質の擬似乱数が生成出来る？

sample-1.c

```c
#include <stdio.h>
#define IA   3877
#define IB   29573
#define IM   139968 // 周期！
int main (void) {

  int i, randomInt;
  double randomDouble;

  randomInt = 10;

  for(i=0; i<3000; i++){
   randomInt = (randomInt*IA + IB) % IM;
   randomDouble = (double)randomInt/IM;
   printf("%f\n", randomDouble);
  }

  return 0;
}
```


6

経験的にこのパラメータを用いるとそれなりの一様乱数が生成されることが知られています。ここで生成する擬似乱数は整数値を取りますが、最大の整数値で割ってやることで0以上1未満の擬似乱数となります。簡単なC言語プログラミングなので、各自実行してみること。

7

疑似乱数の質

- 線形合同法で生成された乱数 [0, 1) はどの程度良質か？
- 擬似乱数列 $\{X_0, X_1, X_2, \ldots\}$ を生成してファイルに書き出す
- 生成した疑似乱数列を gnuplot/Mathematica 等を用いて可視化する

```
% cc sample-1.c
```
線形合同法によるプログラムをコンパイルして実行

```
% ./a.out > data-1d
```
リダイレクションを用いて出力をファイルに保存

線形合同法で生成した擬似乱数の質（0から1の間のどの値も同じ確からしさになっているか？など）を調べるためにgnuplotやMathematicaなどの可視化ツールを用いて視覚化します。まずは先ほどのCソースプログラムをコンパイルして、出力をファイルに書き出して下さい。

2020 奈良女子大学理学部 環境科学コース 環境科学計算機実験

8

データの可視化

gnuplot でデータを読み込み視覚化する

```
% gnuplot
```
<— ターミナルからgnuplotを起動

```
    G N U P L O T
    Version 5.2 patchlevel 8    last modified 2019-12-01

    Copyright (C) 1986-1993, 1998, 2004, 2007-2019
    Thomas Williams, Colin Kelley and many others

    gnuplot home:     http://www.gnuplot.info
    faq, bugs, etc:   type "help FAQ"
    immediate help:   type "help"  (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> plot "data-1d"
```
<— ファイル "data-1d" を読み込み視覚化

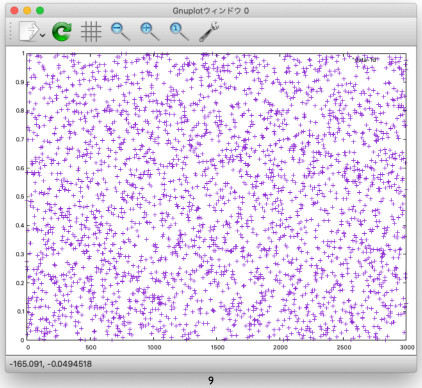
```
qt.qpa.fonts: Populating font family aliases took 271 ms. Replace uses of missing
font family "Sans" with one that exists to avoid this cost.
gnuplot>
```


ここでは、gnuplotを用いた視覚化をします。gnuplotは macOS/Windows/LinuxどのOSでも使えるフリーのツールです。手持ちPCにインストールして先ほど生成した擬似乱数列を視覚化してみる。

9

データの可視化

gnuplot でデータを読み込み視覚化した結果。ランダムに見えるか？



ここでは、3000個の擬似乱数を視覚化しました（横軸が生成した乱数列の長さ）。これを見てランダムに見えるかどうか？

10

データの可視化

疑似乱数を2つ生成して2次元空間上の点として表現する（x, y座標を空白で区切る）

sample-2.c

```
#include <stdio.h>
#define IA   3877
#define IB   29573
#define IM   139968 // 周期！
int main (int argc, const char * argv[]) {

   int i, randomInt;
   double randomDouble;

   randomInt = 10;

   for(i=0; i<3000; i++){
    randomInt = (randomInt*IA + IB) % IM;
    randomDouble = (double)randomInt/IM;
    printf(“%f ”, randomDouble);        <— 出力
    randomInt = (randomInt*IA + IB) % IM;
    randomDouble = (double)randomInt/IM;
    printf(“%f\n”, randomDouble);       <— 出力
   }

   return 0;
}
```


次に、同じ線形合同法で0から1の間の擬似乱数を生成しますが、今度は、2つの擬似乱数を2次元平面上の点 (x, y) 座標として表現します。空白で区切って出力します。

疑似乱数の質

- 擬似乱数列を用いて2次元空間上の点として $\{X_0, X_1\}, \{X_2, X_3\}, \{X_4, X_5\}, \ldots$ をファイルに書き出す
- 生成した疑似乱数列を gnuplot/Mathematica 等を用いて可視化する

```
% cc sample-2.c
```
線形合同法によるプログラムをコンパイルして実行

```
% ./a.out > data-2d
```
リダイレクションを用いて出力をファイルに保存

11

2次元平面上の点として生成した多数の点をファイルに書き出します。そして、gnuplotで2次元平面上の点として視覚化します。

2020 奈良女子大学理学部　環境科学コース　環境科学計算機実験

データの可視化

gnuplot でデータを読み込み視覚化する

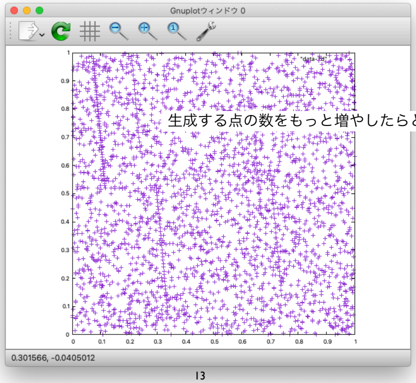
```
% gnuplot        <— ターミナルからgnuplotを起動

  G N U P L O T
  Version 5.2 patchlevel 8    last modified 2019-12-01

  Copyright (C) 1986-1993, 1998, 2004, 2007-2019
  Thomas Williams, Colin Kelley and many others

  gnuplot home:     http://www.gnuplot.info
  faq, bugs, etc:   type "help FAQ"
  immediate help:   type "help"  (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> set size square   <— 図の縦横比を1:1に設定
gnuplot> plot "data-2d"    <— ファイル "data-2d" を読み込み視覚化
gnuplot>
```


12

gnuplotで視覚化。ここでgnuplotの設定として縦横比を1:1に設定しておきます。x–y平面上で長さ1の正方形の領域の中に点が打たれることになります。

データの可視化

gnuplot でデータを読み込み視覚化した結果。ランダムに見えるか？



生成する点の数をもっと増やしたらどうなるか？

13

ここでは3000個の点を表示しました。何か線上の模様が見えています。点の数を増やすとどうなるか？今回用いた線形合同法が本当に0から1の擬似乱数を生成するなら、2次元空間中の点の並びはランダムになるはず。そうなっているかどうか？自分で確認すること。

データの可視化

sample-3.c

```c
#include <stdio.h>
#define IA   3877
#define IB   29573
#define IM   139968 // 周期！

int main (int argc, const char * argv[]) {

   int i, randomInt;
   double randomDouble;

   randomInt = 10;

   for(i=0; i<3000; i++){
    randomInt = (randomInt*IA + IB) % IM;
    randomDouble = (double)randomInt/IM;
    printf("%f ", randomDouble);         <— 出力
    randomInt = (randomInt*IA + IB) % IM;
    randomDouble = (double)randomInt/IM;
    printf("%f ", randomDouble);         <— 出力
    randomInt = (randomInt*IA + IB) % IM;
    randomDouble = (double)randomInt/IM;
    printf("%f\n", randomDouble);        <— 出力
   }

   return 0;
}
```

疑似乱数を3つ生成して
3次元空間上の点として表現する
x, y, z 座標を空白で区切って出力

14

今度は、生成した0から1の擬似乱数を3つの組として3次元空間中の点、 (x, y, z)座標、として表現します。3次元空間中の辺の長さが1の立方体の内部に点を打つことになります。

疑似乱数の質

- 擬似乱数列を用いて3次元空間上の点として $\{X_0, X_1, X_2\}, \{X_3, X_4, X_5\}, \{X_6, X_7, X_8\}, \ldots$ をファイルに書き出す
- 生成した疑似乱数列を gnuplot/Mathematica 等を用いて可視化する

```
% cc sample-3.c
```
線形合同法によるプログラムをコンパイルして実行

```
% ./a.out > data-3d
```
リダイレクションを用いて出力をファイルに保存

15　出力結果をファイルに書き出し、gnuplotで視覚化。

2020 奈良女子大学理学部　環境科学コース　環境科学計算機実験

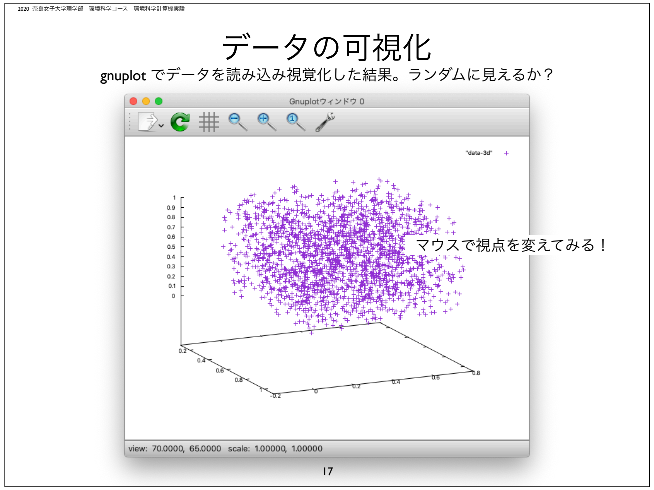
データの可視化

gnuplot でデータを読み込み視覚化する

```
% gnuplot
```
<— ターミナルからgnuplotを起動

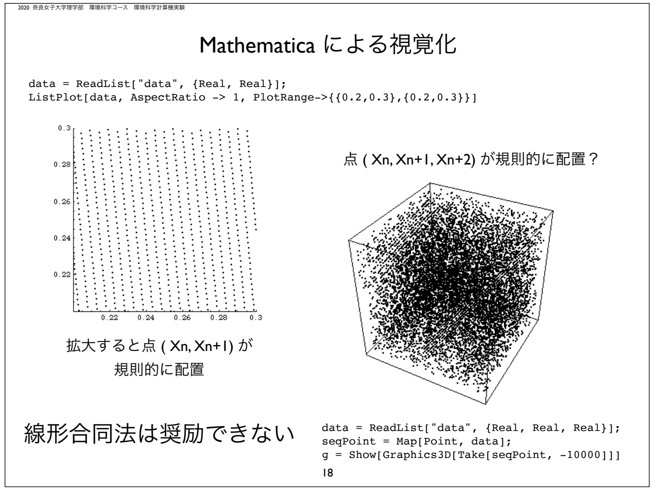
```
  G N U P L O T
  Version 5.2 patchlevel 8    last modified 2019-12-01

  Copyright (C) 1986-1993, 1998, 2004, 2007-2019
  Thomas Williams, Colin Kelley and many others

  gnuplot home:     http://www.gnuplot.info
  faq, bugs, etc:   type "help FAQ"
  immediate help:   type "help"  (plot window: hit 'h')

Terminal type is now 'qt'
gnuplot> splot "data-3d"
gnuplot>
```
<— ファイル“data-3d” を読み込み splot で視覚化

16　3次元グラフを splot で描きます。どうなったか？



17

gnuplotの3次元のグラフはマウスを使って視点を変えることが出来ます。今回用いた線形合同法が本当に0から1の擬似乱数を生成するなら、3次元空間中の点の並びはランダムになるはず。そうなっているかどうか？自分で確認すること。



18

ここでは、線形合同法で生成した十分長い乱数列を “data”というファイルに書き出し、Mathematicaを使って可視化しています。左側は、乱数列を2つ組で読み込み2次元空間上の点として表したもの（描画するx, y範囲を拡大してある）。右側は、乱数列を3つ組で読み込み3次元空間上の点として表したもの。線形合同法が生成する乱数が規則的に並んでいることがわかる。

19

rand 関数

- 多くの C 言語処理系で実装される rand 関数は線形合同法により擬似乱数を生成

sample-rand.c

void srand(unsigned)
乱数の種（初期値）を指定

int rand(void)
擬似乱数を生成

rand 関数は奨励できない

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void)
{
  unsigned seed;
  int randomInt, i;
  double randomDouble;

  seed = (unsigned)time(NULL);

  srand(seed);
  printf("RAND_MAX is %d\n", RAND_MAX);
  printf("Seed is %d\n", seed);

  for(i=0; i<3000; i++){
    randomInt = rand();
    randomDouble = randomInt/(RAND_MAX + 1.0);
    printf("%d, %f\n", randomInt, randomDouble);
  }

  return 0;
}
```


C言語処理系の多くで実装されている rand() 関数は、線形合同法により整数値の擬似乱数を生成する。このとき、初期値X0に相当する値を乱数の種 seed と呼ぶ。seedが同じだと、常に同一の乱数列が生成され、同じ結果になってしまいます。そのため、プログラムを実行するたびに異なるseed値を設定する必要がある。ここでは time.h をインクルードしてプログラム実行時の時刻（何年何月何日何時何分何秒）を取得し、long型に型変換した値をseedとしている。これにより、毎回異なる乱数列が生成される。ただし、rand() 関数が生成する乱数列はサイエンスの分野では使用に耐えないので、別のアルゴリズムを用いて乱数を生成します。

20

メルセンヌ・ツイスタ

- より高品質な擬似乱数を生成する Mersenne Twister

sample-MT.c

void init_genrand(long)
乱数の種（初期値）を指定

double genrand_real2(void)
擬似乱数 [0, 1) を生成

両関数とも外部ファイル
ran-MT.c で定義されている

乱数の種を決めて初期化することに注意！

```c
#include <stdio.h>
#include <time.h>

extern void init_genrand(long);
extern double genrand_real2(void);

int main(void)
{
  long seed;
  int i;
  double rand;

  seed = (long)time(NULL); // seed の設定

  init_genrand(seed); // seed で初期化

  for(i=0; i<3000; i++){
    rand = genrand_real2();
    printf("%.20f\n", rand);
  }

  return 0;
}
```


線形合同法よりも高品質な擬似乱数生成アルゴリズムがあるか？有る。中でもメルセンヌ・ツイスタと呼ばれるアルゴリズムが考案されている。実体は別ファイル（ここでは ran-MT.c として配布する）に記述されている。使い方は、1）long型のseedを設定して初期化 init_genrand(seed)、2）genrand_real2() で U[0, 1) が double型の値として生成される。

メルセンヌ・ツイスタ

メルセンヌ・ツイスタの本体 ran-MT.c は以下から取得可能（8月31日まで有効）

https://pisa.ics.nara-wu.ac.jp/nextcloud/index.php/s/f2LFMdWzFDYw5Aa

`% cc sample-MT.c ran-MT.c`　2つのファイルをコンパイル＋リンクする

`% ./a.out > data-1d-MT`　実行結果をファイルに書き出す

`% gnuplot`　gnuplot で視覚化

メルセンヌ・ツイスタについて詳しく知りたい場合は本家本元を参照

http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/mt.html

21

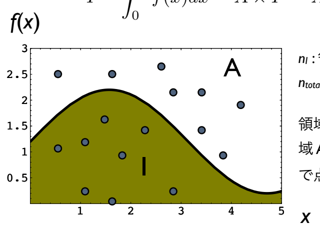
このURLからメルセンヌ・ツイスタ本体プログラムをダウンロード可能（リンクは8月31日まで有効）。その後、メインプログラムと一緒にコンパイル・リンクして実行する。線形合同法でやったのと同じ方法で、メルセンヌ・ツイスタによる擬似乱数の質を視覚的に確認せよ。

モンテカルロ積分

- 乱数を用いた $f(x)$ の積分の計算

領域 A ($0 \le x \le 5, 0 \le y \le 3$) でランダムな点をとる: x = 5 U[0, 1), y = 3 U[0, 1)
積分 I は、ランダムな点が曲線 $f(x)$ の下側に落ちる割合 P と領域 A の面積 5 x 3 の積で与えられる（と予想される）

$$I = \int_0^5 f(x)dx = A \times P = A \times \frac{n_I}{n_{total}}$$



n_I : 領域 I に落ちた点の数
n_{total} : 点の総数（十分大きく取る）

領域 A でランダムな点をとるとは、領域 A 内のどの場所にも同じ確からしさで点を打つこと。

22

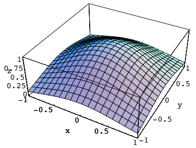
擬似乱数を用いて積分値を近似する方法にモンテカルロ積分があります。アルゴリズムは語句単純で、求めたい積分範囲を囲む領域（ここでは長方形）にランダムに多数の点を打ち、積分範囲に落ちた点の割合に長方形の面積を掛けることで求める積分の近似値とするという方法です。この図の場合、点の(x, y)座標を x = 5 U[0, 1)、y = 3 U[0, 1) とすることで、長方形（横の長さ5、縦の長さ3、面積15）の中にランダムに点を打ったことになります。

モンテカルロ積分の用途

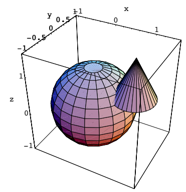
- 重積分など被積分関数 f の評価や積分範囲の数式表現が困難な場合に用いられるモンテカルロ積分
- 擬似乱数を用いて積分値の近似値を求めることができる

$$V = \iint_{\Omega} f(x,y)\,dx\,dy$$

$$\Omega : -1.5 \le x, y \le 1.5$$



被積分関数と積分領域の解析表示は困難



23

被積分関数が簡単な場合は、解析的に（手計算で）積分値を求めることが出来ますが、関数が複雑な場合や、積分範囲が数式では表現しにくい積分（左図のように球体に三角錐がめり込んだ物体の体積）の近似値を求める際にモンテカルロ積分が用いられます。

問題 I

- メルセンヌ・ツイスタを用いて、区間 [0, 1) の一様擬似乱数を N 個生成せよ
- [0, 1) を区間幅 0.1 で 10 個の区間に区切り、各区間に落ちた擬似乱数の数を数えてファイルに書き出せ
- 乱数が一様であれば、各区間に落ちる数の平均は N/10 と予想される。生成した擬似乱数が統計的に一様であるか、どのようにして判定したら良いか考えよ
- メルセンヌ・ツイスタにより生成した乱数列 $\{X_1, X_2, \ldots,\}$ を 2次元もしくは 3次元上に描いて、視覚的に乱数の質を調べよ

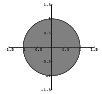
24

問題に取り組んで下さい。メルセンヌ・ツイスタ本体のソースコード ran-MT.c を入手して、乱数の種 seed を設定して乱数生成ルーチンを初期化し、N >>1（多数）の擬似乱数を生成します。

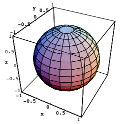
問題 2

- モンテカルロ法により以下の積分の近似値を求め、真の解と比較せよ

$$2\int_{-1}^{1}\sqrt{1-x^2}dx = \iint_{x^2+y^2\leq 1} dxdy$$

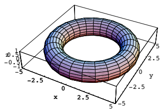


$$\iint_{x^2+y^2\leq 1}\sqrt{1-x^2-y^2}dxdy = \iiint_{x^2+y^2+z^2\leq 1} dxdydz$$



半径 1 の円を z 軸の周りで半径 4 で回転させて出来るトーラスの体積

$$\iiint_{(\sqrt{x^2+y^2}-4)^2+z^2\leq 1} dxdydz$$



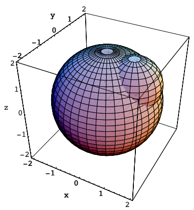
25

モンテカルロ積分を用いて、1) 半径1の円の面積、2) 半径1の球の体積、3）トーラス（ドーナツ状の物体）の体積、の近似値を求めること。トーラスの体積は解析的に求めることが出来ます（google!）。1) の場合、円を囲む1辺の長さが2の正方形の中にランダムに点を打ち、円の内部にあるかどうかを判定します。2) は、球を囲む辺の長さ2の立方体の内部にランダムに点を打ちます。3)は、トーラスを囲む体積 10x10x2 の箱形物体を考えます。Cでプログラムを組みます。

問題 2 続き

- モンテカルロ法により以下の積分の近似値を求めよ

原点を中心とする半径 2 の球と、(1, 0, 1) を中心とする半径 1 の球の合成部分の体積



26

二つの球がめり込んだ形状の体積の近似値をモンテカルロ積分で求めて下さい。