純真学園大学雑誌
第8号　平成 31 年3月

Journal of Junshin Gakuen University,
Faculty of Health Sciences Vol.8, March 2019

資料

コンピュータシミュレーションにおける擬似乱数の生成法と統計的評価

小林 龍徳・王丸 愛子・椎山 謙一

純真学園大学 保健医療学部 放射線技術科学科

The generation method of pseudorandom numbers and statistical evaluation of computer simulation

Tatsunori KOBAYASHI, Aiko OHMARU, Kenichi SHIIYAMA

Department of Radiological Science, Faculty of Health Sciences, JUNSHIN GAKUEN UNIVERSITY

要旨： 疑似乱数は，モンテカルロ（Monte Calro: MC）法や数値計算で利用される．擬似乱数は，乱数の出現頻度の一様性（等確率性）と不規則性が必要である．本論文の目的は，代表的な擬似乱数の生成法と，その擬似乱数の一様性と不規則性の統計的な評価である．本論文では，1）線形合同法（不適切な定数），2）線形合同法（推奨されている定数），3）メルセンヌ・ツイスター法で生成した3種類の疑似乱数を用いた．全ての疑似乱数の出現頻度は，一様性（等確率性）の検定の結果，等確率であった．しかし，不規則性の検定では，1）の疑似乱数には不規則性が認められなかった．プログラミング言語は C 言語と Python を用いた．Python は MT 法が標準関数として採用されており，C 言語よりも短い行数でプログラムを作成できる．また，簡単な MC 法の例として，2次関数の曲線下面積を MC 法で求める計算プログラムを C 言語と python で作成した．計算精度は疑似乱数の個数に比例して向上した．乱数の個数が，10^6個以上では，理論値との誤差は10^{-3}以下となった．

キーワード： 擬似乱数，一様性，不規則性，統計的評価，モンテカルロ法，Python

Abstract: The pseudorandom number is used by the Monte Carlo method and numerical computation. The Uniform（equiprobability）and irregularity of the appearance frequency of the random number are necessary for the pseudorandom number. The purpose of this article is the generation method of a representative pseudorandom number and a statistical evaluation of uniform（equiprobability）and irregularity of the pseudorandom number. In this study generate three type pseudorandom numbers 1）linear congruential generators pseudo（the inappropriate fixed number）, 2）linear congruential generators（the recommended fixed number）, 3）Messene Twister（MT）method. All of the pseudorandom number appearance frequency was an equal probability as the result of the statistical test of uniform（equiprobability）. As the result of equiprobability test, all pseudorandom numbers were equiprobability. However, the result of the irregularity test showed pseudorandom number by linear congruential generators（the inappropriate fixed number）was no an irregularity. The programming language used C language and Python. The MT method is adopted for a standard function in Python, the program is shorter than C language. Finally, we showed a Monte Carlo simulation code for calculation of an area in quadratic function by C language and python. The calculation precision improved in proportion to the number of the pseudorandom number. The number of the pseudorandom number was more than 10^6 and became less than the error with the theoretical value in 10^{-3}.

Keyword: pseudorandom numbers, uniformity, irregularity, statistical evaluation, Monte Carlo method, Python

1. はじめに

乱数（乱数列）は，不規則な数字の並びである．乱数は，統計学における無作為標本抽出や，乱数を用いてシミュレーションを行うモンテカルロ（Monte Carlo: MC）法などで用いられる．乱数は，真の乱数と擬似乱数に分けられる．真の乱数は，規則性がなく，次に出現する数値の予測ができない．一方，擬似乱数は，数式を用いて生成した乱数であり，周期性や規則性を持つ．一般的にコンピュータで用いられている乱数は，擬似乱数であり，使用する乱数の周期性と規則性に注意が必要である．

平成31年3月29日
純真学園大学　保健医療学部　放射線技術科学科　講師

擬似乱数の生成法には，線形合同（Linear congruential generators: LCGs）法，一様乱数，正規乱数，平方採中法などがある[1)]．LCGs 法は，代表的な疑似乱数の生成法であり簡便に乱数を作成できるが，疑似乱数の規則性と周期性，生成式の定数の設定値に注意が必要である[2)]．近年，優れた擬似乱数の生成法としてメルセンヌ・ツイスター（Mersenne Twister: MT）法が報告されている[3, 4)]．MT 法は，疑似乱数の規則性と周期性が，実用上ほぼ問題がない程度に改善されている．

MC 法は，乱数を用いて物理現象などのシミュレーションを行う．MC 法で，大規模または多数回のシミュレーションでは，シミュレーションの条件などをプログラミング言語で記述したプログラムが用いられる．多くのプログラミング言語があるが，Fortran，C 言語，Python（パイソン）などが代表的である．プログラミング言語は，プログラムの目的や用途に応じて選択される．本論文では，代表的なプログラミング言語のひとつである C 言語と，近年，深層学習など様々な分野で利用されている Python を用いた[5)]．C 言語で疑似乱数を生成する rand 関数には LCGs 法が採用されている．Python で擬似乱数を生成する random モジュールには，MT 法が採用されている[6, 7)]．

われわれは，放射線物理学，放射線計測学，核医学検査技術学などの放射線領域において MT 法を用いた MC 法の実施を目指している．本論文では，その初期的な検討として，まず，擬似乱数の代表的な生成法である LCGs 法と MT 法で生成した疑似乱数について擬似乱数の規則性と周期性について統計的に比較する．次に，代表的なプログラミング言語である C 言語と Python による疑似乱数の生成法を比較し，Python によるプログラミングの簡便性について述べる．最後に，MC 法の具体例として，2次関数の曲線下面積を C 言語の rand 関数と，Python の random モジュールで生成した疑似乱数を用いて計算し，計算精度の比較する．

本論文を読んだプログラミング初学者や学生が，本論文の内容を各自のプログラミング環境で実行できるように C 言語と Python のソースコードを示した．加えて，C 言語，Python，モンテカルロシミュレーションに関する参考書籍を多く示した．

2. 疑似乱数の生成法

2.1. C 言語による疑似乱数の生成法

C 言語の rand 関数には LCGs 法が用いられている．LCGs 法は，式（1）に基づいて，擬似乱数を生成する．ここで，n は疑似乱数の個数，Xn は n 番目の乱数，A，B，M は定数であり，$M>A$，$M>B$，$A>0$，$B \geq 0$の条件を満たす．

$$X_{n+1}=(A \times X_n+B) \bmod M \cdots (1)$$

source code 1は，C 言語の rand 関数を用いて擬似乱数を生成するプログラムである．このプログラムでは，区間 [0, 32767] の疑似乱数が生成される．MC 法で擬似乱数を確率変数として用いる場合，疑似乱数が区間 [0.0, 1.0] であれば確率変数として利用しやすい．C 言語の rand 関数を用いて区間 [0.0, 1.0] の擬似乱数を生成するプログラムを source code 2に示す．C 言語で [0.0,1.0] の疑似乱数は、rand 関数で出力された乱数値を C 言語の最大の整数値を返す RAND_MAX（32767）で除算すればよい．rand 関数は，擬似乱数の生成で利用する定数が固定されているため，毎回同じ擬似乱数が生成されてしまう．そこで，source code 3のように rand 関数をコンピュータの時刻で初期化すれば，毎回，異なった擬似乱数を生成できる．

C 言語はプログラミング言語として歴史があり，多くの参考書籍やインターネットに情報が記載されている．C 言語のプログラミングの参考書籍としては，参考文献8～14が参考になる．

2.2. Python による疑似乱数の生成法

Python の random モジュールには，MT 法が採用されている．Python の擬似乱数は，半開区間 [0.0, 1.0）の範囲の一様乱数となる．Python で擬似乱数を生成するプログラムを source code 4に示す．また，Fig.1は，source code 4で生成した乱数を，3次元空間の座標（x, y, z）として扱い，（x_0, x_1, x_2），（x_3, x_4, x_5）…．（x_{n-2}, x_{n-1}, x_n）を組としてプロットしたものである．本論文では，3次元空間の座標（x, y, z）を10002組作るために，30006個の擬似乱数を生成した．Fig.1に示すように，MT 法で生成した乱数は，3次元空間に分布している様子が確認できる．source code 5は，Python で作成した LCGs 法のプログラムである．source code

5では，式(1)の A = 21，B = 11，M = 256とした．以降，この定数で生成した乱数を，LCGs 乱数1とする．Fig.2は，LCG 乱数1を Fig.1と同様に3次元空間にプロットしたものである．Fig.2からわかるように，LCGs 乱数1の擬似乱数には，規則性がみられた．

そこで，*A*，*B*，*M* の定数を，先行研究で推奨される組み合わせとして挙げられている A = 16807，B = 0，M = 2147483647で擬似乱数を生成した[2)]．この定数で生成した乱数を，以降 LCGs 乱数2とした．Fig.3は，LCGs 乱数2を Fig.1と同

source code 1

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   printf("%d\n",rand());
}
```

source code 2

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   printf("%f\n",rand()/(double)RAND_MAX);
}
```

source code 3

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main(void)
{
   srand((unsigned short)time(NULL));
   printf("%f\n",rand()/(double)RAND_MAX);
}
```

source code 4

```
import random
print(random.random())
```

source code 5

```
N =10
x0 = 10
A = 16807
B = 0
M = 2147483647
rnd = (A * x0 + B) % M
for i in range(0, N, 1):
   rnd = (A * rnd + B) % M
   r = (rnd + 1.0)/(M + 1.0)
   print(r)
```

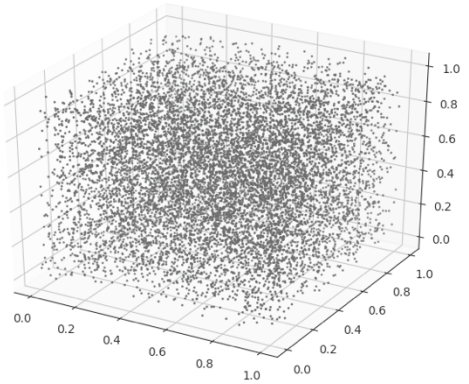

Fig.1　MT 法によって生成した乱数の分布

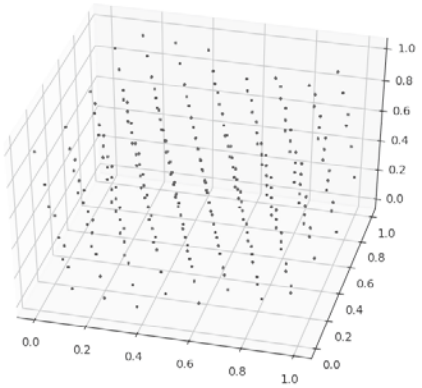

Fig.2　LCGs 乱数1の分布

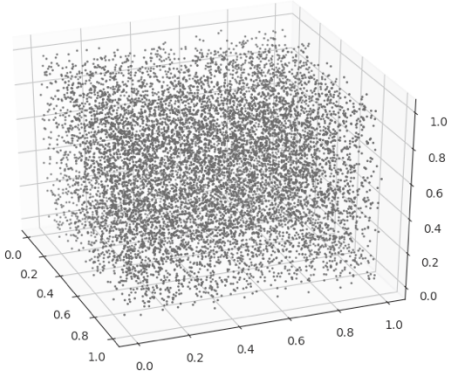

Fig.3　LCGs 乱数2の分布

様に3次元空間にプロットしたものである．LCGs乱数2は，LCGs 乱数1で認められた規則性が改善され，3次元空間に分布している．これらの結果から，LCGs 法は，A，B，M の定数を適切な値にしなければならない．

Python のプログラミングに関しては参考文献15~17が参考になる．また，参考文献7に示した公式のオンラインドキュメントが日本語でも公開されている．

3. 擬似乱数の統計的評価

本研究の擬似乱数は，特別な確率分布を用いていないため，理想的には一様乱数になる．また，真の乱数は，数値の出現が不規則であるが，擬似乱数では，規則性を完全に除外はできない．そこで，本論文のプログラムを使って生成した擬似乱数について，一様性と不規則性を統計的に評価した．

3.1.1. 乱数の等確率性の検定

Fig.4-6は，それぞれ LCGs 乱数1，LCGs 乱数2，MT 法で生成した疑似乱数のヒストグラムである．横軸は乱数の半開区間 [0.0, 1.0）を0.1刻みで区切った値，縦軸は頻度である．乱数の個数は，30006個である

これらのヒストグラムについて，等確率性の検定を行った[1)]．この検定の帰無仮説は，実測度数と理論度数（期待値）は同じであるとした．等確率性の検定では，式（2）より χ^2 値を求め χ^2 検定をする．ここで，b は区間の個数，fi はヒストグラムより得られる実測度数，Fi は乱数の個数を b で除算して得られる理想度数（期待値）である．

$$S = \chi^2 = \sum_{i=1}^{b} \frac{(f_i - F_i)^2}{F_i} \quad \cdots (2)$$

Fig.4-6に示したヒストグラムの χ^2 値は，それぞれ，LCGs 法1が11.09，LCGs 法2が2.98，MT 法が11.05であった．10区間に分けたため，χ^2 分布（自由度9, 有意水準95%）の値 A は，16.919である．本論文で得た各擬似乱数の値 S は，値 A よりも小さくなり，3つの擬似乱数には，統計的に等確率性が認められた．

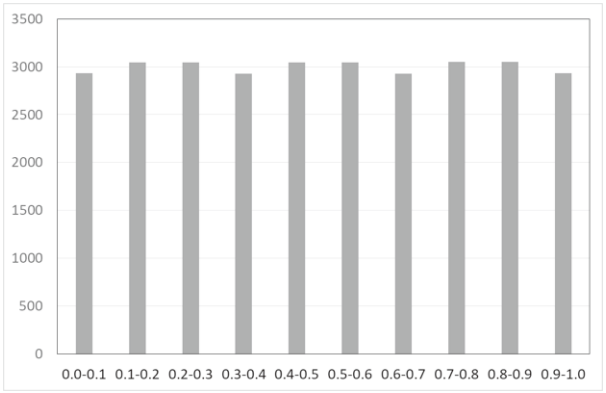

Fig.4 LCGs 乱数1のヒストグラム

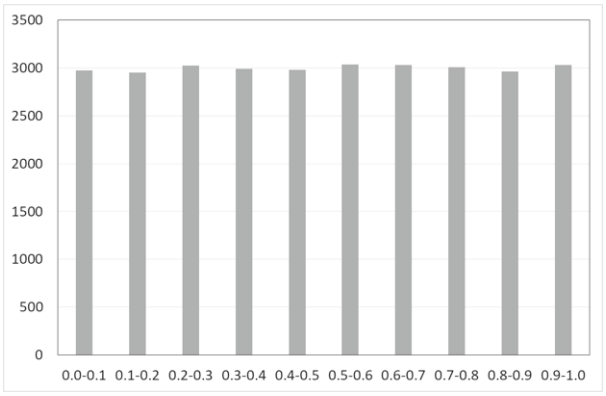

Fig.5 LCGs 乱数2のヒストグラム

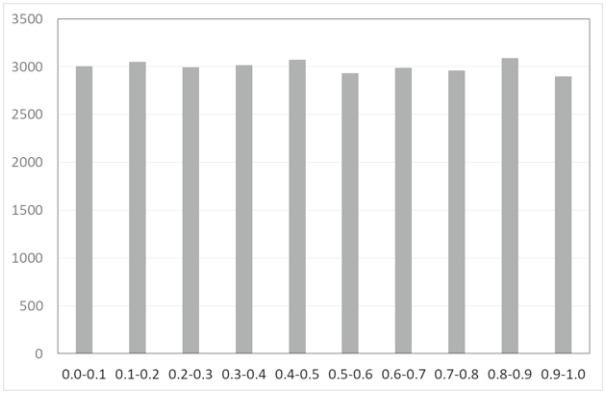

Fig.6 MT 法で生成した乱数

3.2. 乱数の不規則性の検定

3.1で生成した擬似乱数に対して，不規則性を，おくれ k の系列相関検定により統計的に評価した[1)]．おくれ k の系列相関検定は，$x_1, x_2, x_3, \cdots, x_n$ の乱数について，k 個ずらした乱数の組み（x_1, x_{1+k}），（x_2, x_2+k），…（x_{n-1}, x_{k-1}），（x_n, x_k）の相関係数 $r_{xy}^{(k)}$ 求める．$r_{xy}^{(k)}$ は，式（3）から計算できる．

$$r_{xy^{(k)}}=\frac{12}{n}\sum_{i=1}^{n}x_i\,x_{i+k}-3 \cdots (3)$$

おくれkの系列相関検定の帰無仮説は，疑似乱数は不規則であるとし，$r_{xy}{}^{(k)}$を標準正規分布と比較する[1)]．本論文では，k=1，有意水準95% とした．つまり，$r_{xy}{}^{(k)}$が，$-1.96 < r_{xy}{}^{(k)} < 1.96$の範囲にあれば，不規則性のある擬似乱数であると統計的に評価できる．3.1で生成したLCGs乱数1，LCGs乱数2，MT法で生成した擬似乱数の$r_{xy}{}^{(k)}$は，それぞれ，2.76，0.31，−1.23であった．この検定結果から，LCGs乱数2とMT法で生成した擬似乱数には，不規則性が統計的に認められた．

次に，LCGs乱数1，LCGs乱数2，MT法について，30006個の擬似乱数を10000回生成させ不規則性の検定を行った．その結果，10000回の試行のうち，不規則性があると統計的に評価された回数は，LCGs乱数1は0回，LCGs乱数2は9170回，MT法は9465回であった．

この結果は，MT法で生成した疑似乱数は，LCGs法に比べて，不規則性のある擬似乱数を生成できる可能性が高い．また，LCGs法は，定数の設定が不適切な場合，不規則性のある乱数を生成できない可能性がある．Pythonは，source code 4に示したようにMT法による擬似乱数を数行で生成できるため，簡便に優れた乱数を利用できる．

4. 擬似乱数を用いたMC法の例

MC法の簡単な例として，式（4）の2次関数について$0 \le x \le 1$の曲線下の面積を求める．

$$y=x^2 \cdots (4)$$

式（4）の$0 \le x \le 1$の面積は，定積分すると0.33…となる．source code 6はC言語，source code 7はPythonによる式(4)の解をMC法で求めるプログラムである．それぞれのプログラムは，10^M個（M=0〜8）の擬似乱数を生成できる．Table 1は，乱数の個数による面積の計算結果と理論値との差である．面積は，MC法を5回行い，その平均値を利用した．この結果から，簡単な数値計算であれば，擬似乱数の個数が10^6個以上では，理論値と誤差は，10^{-3}以下となった．

Table 1　MC法の結果と理論値の比較

乱数の個数	C言語	誤差	python	誤差
1	0.400000	−0.066667	0.400000	−0.066667
10	0.420000	−0.086667	0.120000	0.213333
100	0.322000	0.011333	0.342000	−0.008667
1000	0.341200	−0.007867	0.336600	−0.003267
10000	0.330660	0.002673	0.332500	0.000833
100000	0.334360	−0.001027	0.332206	0.001127
1000000	0.333018	0.000316	0.333383	−0.000050
10000000	0.333311	0.000022	0.333425	−0.000091
100000000	0.333346	−0.000013	0.333341	−0.000008

MC法は，source code 6, 7に示したように，擬似乱数と条件判定という比較的簡単なプログラムで実現できる．ただし，より複雑なMC法を行う場合，物理現象を表す数式や，複雑な条件判定をプログラミングするためのスキルが必要になる．コンピュータを用いたMC法については，参考文献12，14，18，19が参考になる．

5. まとめ

本論文では，コンピュータシミュレーションで利用される擬似乱数の生成法と，その乱数の一様性と不規則性について統計的に評価した．擬似乱数は，プログラミング言語を用いて簡便に生成できるが，疑似乱数の生成式と特徴の理解が重要である．Pythonのrandomモジュールで採用されているMT法は，現在，最も優れた擬似乱数の生成法といわれている．本論文の結果においても，MT法で生成した擬似乱数は一様性と不規則が統計的に認められた．また，Pythonは，C言語よりで短い行数でプログラムを作成でき，MT法を用いた高精度なMC法の実施が可能である．

source code 6

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
int main(void)
{
   int i, j, count, N, M;
   double x, y;
   srand((unsigned)time(NULL));
   M = 8;
   for(j=0; j<M; j++){

      x = y = 0.0;
      N = pow(10.0, j);
      count = 0;
      for(i=0; i<N; i++){
         x = rand()/(double)RAND_MAX;
         y = rand()/(double)RAND_MAX;
         if(y < x*x){
            count++;
            }
      }
      printf("area = %f\n ", count/(double)N);
   }
}
```

source code 7

```
import random
M = 8
for j in range（0, M+1, 1）:
   count = 0
   area = 0
   N = 10**j
   for i in range（0, N, 1）:
      x = random.random（）
      y = random.random（）
      if（y < x*x）:
         count = count + 1
   print（"area: ",count/float（N））
```

6. 参考文献

1）脇本 和昌，乱数の知識，森北出版株式会社，統計科学のための電子図書システム http://ebsa.ism.ac.jp/ebooks/ebook/1223?page=0%2C2

2）SK Park, KW Miller, Random number generators: good ones are hard to find, Communications of the ACM, 31（10）, 1192-1201, 1988.

3）Mersenne Twister Home Page, http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/mt.html

4）M. Matsumoto and T. Nishimura, "Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator", ACM Trans. on Modeling and Computer Simulation, 8（1）, 3-30, 1998.

5）ZDNet japan, プログラミング言語の人気ランキング：Python が C++ を上回る日も近いのか？, https://japan.zdnet.com/article/35123875/（2018）

6）Welcome to Python.org, https://www.python.org/

7）Python Documentation contents, 9.6. random ― Generate pseudo-random numbers, https://docs.python.jp/3/library/random.html

8）B.W. カーニハン，プログラミング言語 C 第2版 ANSI 規格準拠，共立出版，東京，1989.

9）レス ハンコック，C 言語入門（ASCII SOFTWARE SCIENCE Language），アスキー，東京，1992.

10）MMGames，苦しんで覚える C 言語，秀和システム，東京，2011.

11）株式会社アンク，C の絵本 第2版 C 言語が好きになる新しい9つの扉，株式会社アンク，2016.

12）河西 朝雄，改訂第4版 C 言語によるはじめてのアルゴリズム入門，技術評論社，2017.

13）粂井 康孝，猫でもわかる C 言語プログラミング 第3版（猫でもわかるプログラミング），SB クリエイティブ，東京，2013.

14）小高 知宏，C による数値計算とシミュレーション，オーム社．東京，2009.

15）柴田 淳，みんなの Python 第4版，SB クリエイティブ，東京，2016.

16）小田切 篤，露木 誠，15時間でわかる Python 集中講座，技術評論社，東京，2016.

17）中久喜 健司，科学技術計算のための Python 入門―開発基礎，必須ライブラリ，高速化，技術評論社，東京，2016.

18）宮武修，モンテカルロ法　増補版，日刊工業株式会社，東京，1969

19）四辻 哲章，計算機シミュレーションのための確率分布乱数生成法，プレアデス出版，長野，2010.