

乱数の品質

樋口さぶろお

龍谷大学理工学部数理情報学科

計算科学☆実習 B L15(2021-07-15 Thu)

最終更新: Time-stamp: "2021-07-05 Mon 21:02 JST hig"

今日の目標

- 線形合同法による擬似乱数生成を説明できる
- 擬似乱数列の品質の評価方法を説明できる



L14-Q1

Quiz 解答: 自己回帰モデル AR(1)

- ① $\phi_1 = 1$ のとき, $X_1 = 115, X_2 = 107$.
- ② $\phi_1 = 0.9$ のとき, $X_1 = 105, X_2 = 86.5$.
- ③ $\phi_1 = -0.9$ のとき, $X_1 = -75, X_2 = 59.5$.

ここまで来たよ

13 AR(m) モデル・時系列解析

13 乱数の品質

- 擬似乱数生成の仕組みとその品質

疑似乱数列生成の仕組みとシード

疑似乱数列 = ‘ほぼ’ ランダムな数列

2 回続けて `int rand()` を呼んで得られる 2 個の数の分布は独立であるかのように考えてる (けど… そこが疑似).

(標本抽出のたびに別々の)seed を指定することが必要

- seed に応じて, 毎回異なる乱数列が得られる.
- 特定の乱数列に対する動作を再現できる. デバッグでは必須.
- seed を適切に設定すると, 複数回の実行で, 別々の (独立な) 標本抽出が行える.

L15-Q1

Quiz(rand() の振る舞い)

次のプログラムで、seed を無作為に入力するとき、A が出力される確率は？

```
1  int getRandom(double y){
2      if( y<1/3.0 ){
3          return 0;
4      }else{
5          return 1;
6      }
7  }
8
9  int main(){
10     int seed;
11     scanf("%d",&seed);
12     srand(seed);
13     if( getRandom(getuniform())==getrandom(getuniform()) ){
14         printf("A\n");
15     }
16     return 0;
17 }
```

- ① 0
- ② 1/2
- ③ 5/9 に近い
- ④ 6/9 くらい
- ⑤ 1

double getuniform() を疑おう

これまで使っていた `double getuniform()` は本当に独立同分布 $U(0, 1)$ にしているのだろうか? → No.

```
1 #include <stdlib.h>
2 double getuniform() {
3     return rand() / (RAND_MAX + 1.0);
4 }
```

見るからにだめ

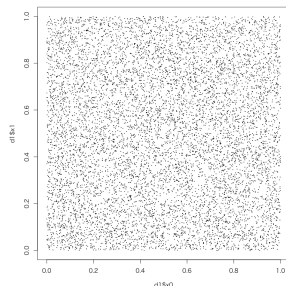
- 実数って言うけど, $1.0 / (\text{RAND_MAX} + 1.0)$ 幅の格子点.
- 乱数表の長さだけ呼び出したら, その後は繰り返しになる.

疑うべき点

- ヒストグラムは $U(0, 1)$ に似てる? $P(\text{getuniform()} < y) = y$?
- $E[\text{getuniform()}] = 0.5$? $V[\text{getuniform()}] = 1/12.0$?
- `getuniform()` と次の `getuniform()` は独立同分布にしたがう?

擬似乱数列の独立性

$R(0), R(1)$ = getuniform() の散布図

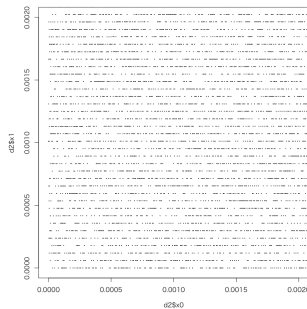


一様, 独立っぽい.

$N = 10000$ で 標本共分散

$S_{R(0)R(1)} = 0.0002553512$.

しかし, 独立なら $\text{Cov}=0$ だけど,
 $\text{Cov}=0$ だから独立とは言えない.
 もっとサンプルサイズを大きくして,
 $[0, 0.02) \times [0, 0.002)$ 部分を拡大.



$R(0)$ と $R(1)$ は独立でないっぽい.

実は `int rand()` は乱数表を見ているわけではない.

```
1 srand (seed);  
2 for (t=0;t<100;t++){  
3     print ("%d\n",rand ());  
4 }
```

は次と同じようなことをやってる.

線形合同法

```
1 int a=214013,b=2531011,m=32768; /* Visual C++*/  
2 x=seed;  
3 for (t=0;t<100;t++){  
4     x = (a*x + b)%m; /*  $ax + b \bmod m$  */  
5     printf ("%d\n",x);  
6 }
```

前の項から次の項が決まる. これ自身, 時系列
周期は最大でも $m \leq \text{RAND_MAX}$.

整数論 (離散数学)

今後、乱数を使うときのアドバイス

言い訳 `int rand()` は `stdlib.h` に入ってるから C では OS やコンパイラによらず使える.. そのため、この授業では `int rand()` から `double getuniform()` を作って使ってきた. しかし, `rand()` は低品質.

一般的なおすすめ

各処理系に一樣分布や主要な確率分布にしたがう乱数生成関数が備わっている. 品質は様々. Linux ではまず `drand48`.

- 近江崇宏先生 (東京大学) C 言語による乱数生成 http://www.sat.t.u-tokyo.ac.jp/~omi/random_variables_generation.html

MT=Mersenne Twister という数学的理論を使った高品質な乱数があれば/信頼できる 3rd party が提供していれば, それを使おう. 整数論 (離散数学)

- 松本真先生 (広島大学) Mersenne Twister
<http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/mt.html>