ランダムウォークと離散型擬似乱数

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 L01(2021-04-08 Thu)

最終更新: Time-stamp: "2021-04-09 Fri 10:24 JST hig"

今日の目標

- この科目ののりを説明できる
- ランダムウォークとは何か説明できる
- 離散型確率変数 確率統計☆演習 I(2020)L06 に対応する擬似乱数列を生成するプログラムを書ける

ここまで来たよ

科目の目標

もう少し正確にはシラバスを見てね.

- 確率過程 = 確率 + 時間変化
- 現象の確率モデル 分野 P
  - 学期末のプロジェクトの例: この宝くじ 1000 枚買って, 収支が+1 万円以上になる確率は? 台風の多い年, 夏の終わりまでに琵琶湖で水害が起きる確率は? このすごろく, 勝負つくまでの手数の母期待値は?
- 確率シミュレーション = 確率 + プログラミング. 3 年前期も C 続けよう. モンテカルロ法
- データサイエンス のツール R, RStudio

計算科学及び実習 A とは中身ものりも別です. 確率統計☆演習 I の続きです.

どんな人のための科目?

計算科学☆実習 B を履修した方がいい人

- 確率過程 (=時間に依存する確率的現象) を知りたい人
- 微分方程式 (決定論的モデル) が見ていない, 残り半分の世界を確率論的モデルで見たい人
- モデル駆動の研究が見ていない, データ駆動の研究の世界を見たい人
- 偶然性のあるゲームを仕組みからわかって作りたい人
- 確率を, プログラム作成の中で実感したい人
- ランダムアルゴリズムが使えるようになりたい人
- コンピュータでデータの解析ができるようになりたい人

計算科学☆実習 B を履修しない方がいい人

- (単位をとっているかどうかに関わらず) 確率統計☆演習 I, 数値計算法及び実習がぜんぜんわかってない感がある人, この機会にわかろうという決意のない人

科目ののり (2021)

難しくありませんが, 注文が多くめんどくさい科目です…
成績計算 科目の成績 100 ピーナッツは

- 25 ピーナッツ:平常点．毎回授業での quiz, 授業時間外の予習復習．
  - だいたい 10 講義の Quiz ほか
  - だいたい 15 実習時間内の課題提出 TA の現場チェックでなく教員の提出プログラムチェック．TA は間違いの発見に努めますが,「それで OK」とは言いません．
- 40 ピーナッツ:プチテスト群
  - 20 紙のプチテスト
  - 20=5+15 プログラミング実技の非参照非相談プチテスト
- 15 ピーナッツ:プロジェクトとプレゼンテーション (2 回 5+10)
- 20 ピーナッツ:紙のファイナルトライアル (外部記憶あり)．参加必須．
- その他追加ピーナッツ．その時に説明．

ファイナルトライアル時点で 40 点未満の方も, (平均点を上げるために) 本試験に参加をおすすめしますが, 追試験は実施しません．

教科書 理工系の数理 確率・統計 確率統計☆演習 I(2020)L00 と同じ. 岩薩林 確率・統計 で言及.

担当者ののり

- なまえ: 樋口さぶろお
- Teams Chat a00010
- `hig@math.ryukoku.ac.jp`
- へや: 1-507
- オフィスアワー: 木 5(この後), , Teams chat もどうぞ.
  - 他の相談先: Math ラウンジ (1 号館 5 階, 昼休みは大学院生常駐), Math ラウンジ ch on Teams
- Web ページ: `https://hig3.net`

LMS この授業の連絡はすべて hig3Moodle `https://moodle.hig3.net` と Teams の **科目 計算科学及び実習 B 樋口** で.

実習ののりと週間タイムライン (2021) 木 4 月 3

実習ののり

- 実習時間内の課題提出 TA の現場チェックでなく, プログラム提出後の教員によるチェック. TA は間違いの発見に努めますが,「それでOK」とは言いません.
- 個人課題 (最初のころ) とチーム課題があります

実習室でやるときイヤフォン必須. 講義実習でのノート PC 持参推奨.

1. 木 4 講義的, Quiz(相談参照あり)
2. このころ実習の課題公開
3. 土 20:00(?) 先週の実習課題の一部の提出締切
4. 月 15:15 予習復習問題 (e ラーニング) の一部の解答締切. 何度でも. 最高点.
5. 月 3 実習的 (1-609),
6. 月 23:59 今週の課題の一部の提出締切

ここまで来たよ

- はじめに
  - この授業どんなのり/取る価値ある?

1. ランダムウォークと離散型擬似乱数
   - ランダムウォーク
   - 確率変数
   - 擬似乱数列
   - 離散型確率分布にしたがう擬似乱数

ふつうの数列とランダムウォーク (確率過程の例)

数列 $\{X(t)\}$, 階差数列 $\{R(t)\}$.

$$\text{漸化式 } X(t) = X(t-1) + R(t) \quad (t = 1, 2, \ldots), \quad \text{初項 } X(0) = a.$$

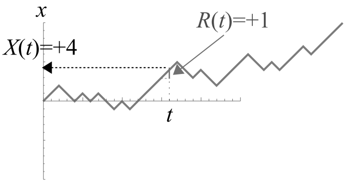
ランダムウォーク (確率過程の例)

確率変数 $X(t)$ がランダムウォークの座標 (数列 $\{X(t)\}$), 確率変数 $R(t)$ が独立同分布にしたがう (階差数列 $\{R(t)\}$)

現象の数理 A

$$\text{漸化式 } X(t) = X(t-1) + R(t) \quad (t = 1, 2, \ldots), \quad \text{初項 } X(0) = a.$$

https://youtu.be/yGNIzkQzp7g



例.

$R(t)$	確率
$+1$	p
-1	$q(=1-p)$

岩薩林 確率・統計ベルヌーイ分布, $n = 1$ 二項分布 (p.66)

どうやってコンピュータで?

ここまで来たよ

- はじめに
  - この授業どんなのり/取る価値ある?

1. ランダムウォークと離散型擬似乱数
   - ランダムウォーク
   - 確率変数
   - 擬似乱数列
   - 離散型確率分布にしたがう擬似乱数

連続型確率変数の復習

確率統計☆演習 I(2019)L07　岩薩林 確率・統計 §4.1

X: 連続型確率変数の確率分布は, 確率密度関数 $f(x) \geq 0$ で指定される.

確率密度関数から事象の確率を求める

$$
\begin{aligned}
P(\text{事象}) = P(\text{条件}) &= \mathrm{E}[\mathrm{I}_{[\text{条件}]}(X)] \\
P(a \leq X < b) &= \mathrm{E}[\mathrm{I}_{[a \leq X < b]}(X)] \\
&= \int_{-\infty}^{+\infty} f(x)\mathrm{I}_{[a \leq X < b]}(x)\,\mathrm{d}x = \int_a^b f(x)\,\mathrm{d}x
\end{aligned}
$$

面積

連続一様分布 $\mathrm{U}(0,1)$

岩薩林 確率・統計 p.78　確率統計☆演習 I(2018)L08

確率密度関数 $f(x) = \begin{cases} 1 & (0 \leq x < 1) \\ 0 & (\text{他}) \end{cases}$.

L01-Q1

Quiz(連続的な確率変数の母平均値・母分散・母標準偏差・確率 (一様分布))

連続型確率変数 X は次の確率密度関数 $f(x)$ に従う.

$$f(x) = \begin{cases} 2 & (\frac{5}{2} \leq x < 3) \\ 0 & (\text{他}) \end{cases}$$

1. 母期待値 $\mathrm{E}[\cos(\pi X)]$ を求めよう.
2. 確率 $P(\frac{22}{8} < X < \frac{23}{8})$ を求めよう.

ここまで来たよ

離散型確率分布にしたがう擬似乱数列の生成

モンテカルロ法

確率的/決定的な量を計算するのに, 確率変数の標本抽出を実際にコンピュータで (擬似) 乱数 ((pseudo) random number) を使って行う方法

だからモンテカルロ法には擬似乱数列が必要.

(擬似) 乱数列

$R(1), R(2), R(3), R(4), \cdots$ 独立同分布にしたがう確率変数としたとき,
サイコロ (やコンピュータや乱数表) を使って作る, そのサンプル (標本).

擬似乱数列を C 言語で生成しよう.

確率統計☆演習 I(2019)L05　岩薩林 確率・統計 §3.1　確率統計☆演習 I(2019)L07

C 言語での乱数の使い方

学習用で ‘低品質’. ゲームくらいならいいけど, 正確な科学技術計算に使う方法はまた後で Linux, macOS では高品質な drand48() や mt=メルセンヌツイスター

include <stdlib.h> すると使えるライブラリ関数

```
int rand(); /* 0以上RAND_MAX以下の整数を
               同確率 1/(1+RAND_MAX) で返す関数 */
void srand(unsigned seed); /* その初期化. まて次回以降. */
```

RAND_MAX は M_PI みたいな定数. 値はコンパイラ依存. 例 $2^{31}-1$.

一様離散分布 0,1,...,RAND_MAX にしたがう擬似乱数列

ソースコード 1: 擬似乱数

```
/*
rand0.c -- 0,1,2,,, RANDMAX の離散一様分布
Time-stamp: "2020-04-08 Wed 07:48 JST hig"
*/
#define _CRT_SECURE_NO_WARNINGS // Visual C++用おまじない
#include <stdio.h>
#include <stdlib.h>   /* srand(), rand() を使うのに必要 */


int main(){
  int seed;        /* 疑似乱数のシード */
  int r;
  int n;
  int nmax=10;

  scanf("%d",&seed);
  srand(seed);          /* シードの設定 */
  for(n=0;n<nmax;n++){
    r=rand();
    printf("%d\n",r);
  }
  return 0;
}
```

連続一様分布 $\mathrm{U}(0,1)$ にしたがう擬似乱数列

ソースコード 2: 擬似乱数

```
/*
randa.c ―― [0,1)一様乱数を返す関数
Time-stamp: "2020-04-08 Wed 10:04 JST hig"
*/
#define _CRT_SECURE_NO_WARNINGS // Visual C++用おまじない
#include <stdio.h>
#include <stdlib.h>    /* srand(), rand() を使うのに必要 */

/* 関数プロトタイプ宣言 */
double getuniform();

int main(){
  int seed;          /* 疑似乱数のシード */
  int n;             /* カウンタ 標本内通し番号*/
  int nmax=10;       /* 疑似乱数を得る回数=サンプルサイズN */
  double r;

  scanf("%d",&seed);
  srand(seed);           /* シードの設定 */
  for(n=0;n<nmax;n++){
    r=getuniform();
    printf("%f\n",r);
  }
  return 0;
}

/** [0,1) 一様疑似乱数を返す */
double getuniform(){
  return rand()/(RAND_MAX+1.0);
}
```

ここまで来たよ

離散型確率分布にしたがう擬似乱数列

ソースコード 3: 擬似乱数

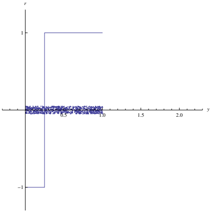
```
/*
rand1.c --  -1 or +1 を確率1/4, 3/4で選ぶ乱数
Time-stamp: "2020-04-08 Wed 09:08 JST hig"
*/
#define _CRT_SECURE_NO_WARNINGS // Visual C++用おまじない
#include <stdio.h>
#include <stdlib.h>   /* srand(), rand() を使うのに必要 */

/* 関数プロトタイプ宣言 */
double getuniform();
int getrandom(double y);

int main(){
  int seed;         /* 疑似乱数のシード */
  int n;            /* カウンタ 標本内通し番号*/
  int nmax=100;     /* 疑似乱数を得る回数=サンプルサイズN */
  int r;

  scanf("%d",&seed);
  srand(seed);          /* シードの設定 */
  for(n=0;n<nmax;n++){
    /* srand(seed);  */ /*ここに置くと? */
    r=getrandom(getuniform());
    printf("%d\n",r);
  }
  return 0;
}
/** [0,1) 一様疑似乱数を返す */
double getuniform(){
  return rand()/(RAND_MAX+1.0);
}

/** -1 or +1 を確率1/4, 3/4 で返す乱数 */
int getrandom(double y){
  if( y < 0.25 ){
    return -1;
  } else {
    return +1;
  }
}
```

しくみ

離散型確率変数 X.

確率関数 $f(x) = \begin{cases} \frac{1}{4} & (x = -1) \\ \frac{3}{4} & (x = +1) \end{cases}$

```
/* 引数yが[0,1)一様乱数なら,
   getrandom の返り値は
   確率1/4で−1, 確率3/4で+1*/
int getrandom(double y){
  if( y< 0.25){
    return -1;
  } else {
    return +1;
  }
}
```




```
r=getrandom(getuniform());
```

マイいかさまコイン関数を書こう

L01-Q2

Quiz(擬似乱数の使いかた)

引数 y として $[0,1)$ 一様乱数が与えられたとき, 下の確率で値を返す `int getrandom(double y)` を, サンプルプログラムを参考に書こう.

返り値	確率
0.6	0.7
0.4	0.3

マイいかさま三角エンピツ関数を書こう

L01-Q3

Quiz(離散的な乱数の生成)

離散的確率変数 R の確率分布は次であたえられる.

$$f(r) = \begin{cases} \frac{2}{8} & (r = 1) \\ \frac{1}{8} & (r = 2) \\ \frac{5}{8} & (r = 3) \\ 0 & (\text{他}) \end{cases}.$$

引数 y として $[0, 1)$ 一様乱数を与えるとき, 上の確率分布に従う乱数 r を返す関数
`int getrandom(double y)` を定義しよう.

動画解説
https://youtu.be/Yc6bzcrfLeo

$a \leq y < b$ のとき, 1 を返すとすると, 1 が返される確率は $\int_a^b 1 \, \mathrm{d}x$.
$r = 1, 2, 3$ について a, b をうまく調整していけばいい.

L01-Q4

Quiz(期待値)

離散型確率変数 R は, 値 $R=0$ を確率 $2/13$ で, 値 $R=3$ を確率 $4/13$ で, 値 $R=4$ を確率 $7/13$ でとる.
引数 y として $[0,1)$ 一様乱数を与えるとき, 上の確率分布に従う乱数 r を返す関数 `int getrandom(double y)` を定義しよう.