**Генератор случайных чисел**
**Датчик случайных чисел**
**Random Number Generators (RNG)**

Генератор случайных чисел создает последовательность чисел, которую мы не можем отличить от случайной.

**История вопроса** (пунктиром)

1. Древнейшие игральные кости имеют возраст около 5200 лет,
метки на гранях костей практически не отличаются от современных

2. Comte de Buffon 18 век
вычислял число $\pi$ (так можно интерпретировать его решение)
случайное событие - пересекает иголка длиной 1 сетку с шагом 1 или нет
P{пересечения} = 2/ $\pi$

3. Интервалы между сигналами счетчика Гейгера

**Генератор псевдо-случайных чисел**

Далее слово псевдо буду опускать
Random Number Generators (**RNG**)

**Схема алгоритма генерации случайных чисел**

Генератор случайных чисел состоит из двух функций

S текущее зерно датчика
$f : S \to S$ функция, меняющая текущее зерно датчика
$g : S \to [0,1]$ функция, вычисляющее очередное случайное число (цифру)

**Пример RNG**

Джон фон Нейман (1946)

S - k-значное число

$f : S \rightarrow S$

число возводится в квадрат, из результата берется центральное k-значное число

$g : S \rightarrow [0,1, 2, \dots , 8, 9]$

число возводится в квадрат, из результата берется центральная цифра

**Пример**

S = 98765 первое зерно датчика сл. чисел

$98765^2 = 9754525225$

g(98765) = ~~9754~~525~~225~~ = 5 первая случайная цифра

f(98765) = ~~97~~54525~~225~~ = 54525 новое зерно датчика сл. чисел

S = 54525 новое зерно датчика сл. чисел

$54525^2 = 2972975625$

g(54525) = ~~2972~~975~~625~~ = 9 вторая случайная цифра

f(98765) = ~~29~~72975~~625~~ = 72975 новое зерно датчика сл. чисел

S = 72975

Дональд Кнут:

метод генерирует качественные последовательности,

метод отвергнут, так как период может быть малым,

не умеем предсказывать длину периода для конкретного зерна

**Зерно генератора случайных чисел**

Обеспечивает воспроизводимость последовательности случайных чисел

**Период генератора случайных чисел**

Всегда конечный
У RNG Неймана период зависит от зерна
У Mersenne Twister (R, Python) период не зависит от зерна

**RNG для произвольного распределения**

**Если** умеем генерировать случайные цифры,
то умеем генерировать наблюдения равномерно распределенной случайной величины

**Если** умеем генерировать наблюдения равномерно распределенной случайной величины
то умеем генерировать наблюдения случайной величины с любым распределением

**Задача.**
Выписать функцию распределения и плотность равномерно распределенной случайной величины

**Утверждение**
Пусть U – равномерно распределенная с.в.
F(t) – строго возрастающая функция, $F(-\infty)=0$, $F(+\infty) = 1$
Тогда функция распределения случайной величины $X = F^{-1}(U)$ равна F(t)

**Комментарий**
Методом пользуются редко: обычно сложно и долго вычислять $F^{-1}$

**Утверждение**
Пусть случайные величины $U_1 \sim U(0,1)$ и $U_2 \sim U(0,1)$
Обозначим
$\theta = 2 \cdot \pi \cdot U_2$ ,
$\rho = \sqrt{-2 \cdot \log U_1}$ .
Тогда $Z_1 = \rho \cdot \cos(\theta)$ нормально распределенная с.в.

**Комментарий**
Методом пользуются редко: долго вычислять cos, log и sqrt

**Комментарий**
Мы убедились, что преобразование возможно, в выборе конкретного варианта доверимся специалистам.

**Линейный RNG**

$x_n+1 \equiv (a * x_{n-1} + c) \pmod m$

**Обозначение** $LCG(m, a, c, x_0)$

**Напоминание** про остаток от деления

(14) mod 10 = 4
(24) mod 7 = 3

**Примеры RNG**

ANSIC LCG ($2^{31}$ , 1103515245, 12345, 12345)
MINSTD LCG ($2^{31-1}$, $7^5$ , 0, 1)
RANDU LCG ($2^{31}$ , $2^{16}$, 0, 1)
APPLE LCG ($2^{35}$ , $5^{13}$ , 0, 1)
Super-duper LCG ($2^{32}$, 69069, 0, 1)
NAG LCG ($2^{59}$ , $13^{13}$, 0 , $2^{32} + 1$)
DRAND48 LCG ($2^{48}$ , 25214903917, 11, 0)

**Как добиться постоянного периода последовательности?**

теорема Hull-Dobell

**неточный пересказ теоремы**
**m** желательно выбирать

- простым (чтобы период был равен **m-1**)
- побольше (чтобы период был побольше)

Простые числа можно выбирать среди ч**исел Мерсена** $2^{к}$ -1

На август 2020 известно 51 **число Мерсена**.
Наибольшее из них $2^{82,589,933} - 1$

Great Internet Mersenne Prime Search
( https://en.wikipedia.org/wiki/Great_Internet_Mersenne_Prime_Search)

**Два примера линейных RNG**

1 Явно плохой RNG - видны Marsaglia planes

$x_n = 12 \cdot x_{n-1} \bmod 101$

$u_n = x_n / 101$

***смотрим скрипт***

2 Вариант получше (но тоже плохой)

$x_n = 4809922 x_{n-1} \bmod 60466169$

$u_n = x_n / 60466169$

***смотрим скрипт***

**Кто гарантирует, что распределение равномерное?**

Статистические тесты

Гипотеза согласия

**Несколько батарей статистических тестов**

**diehard** (Marsaglia, 1996)

diehard - «Крепкий орешек», «Умри тяжело, но достойно»

**TestU01** (L'Ecuyer, Simard, 2007), включает большинство тестов, выбранных Knuth-ом (защищено авторскими правами)

**STS** Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptography (National Institute of Standards and Technology, 2001/2010). Состоит из 16 тестов.

**DieHarder** (Brown, продолжает пополняться) Все тесты из diehard, STS, Кнута и какие-то еще. Лицензия GPL.

**Катастрофа**

Marsaglia
*Random Numbers fall mainly in the plane (*1968*)*

Marsaglia planes Marsaglia effect

трем последовательно расположенным числам сопоставим точку

точки расположены на плоскостях

Ferrenberg, Landau, Wong

Monte Carlo simulations: Hidden errors from ‘‘good’’ random number generators

1992 Phys. Rev. Lett. 69, 3382

**В прошлом были найдены дефекты** в разных процедурах

C-library rand(), random() и drand48()
Python random()
standard Perl rand
Matlab rand
Mathematica SWB generator
Java.util.Random
книга Numerical recipes ran0() and ran1()

**Свойства хорошего генератора  случайных чисел**

Надежное теоретическое обоснование

Большой период

Проверен статистическими критериями

Эффективный

Воспроизводимый

Автономный (не нужны специальные устройства)

**Криптографически не стойкие**

Когда-то было легче
XOR'ом смешать с белым шумом, получим белый шум
a – ключ, использовать один раз

Почему XOR?
(a XOR b) XOR b = a

Кругосветный поход двух атомных подводных лодок Северного флота К-116 и К-133 в феврале 1966 года

**Mersenne Twister**
Исходная версия  Matsumoto, Nishimura (1998)
"Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator," ACM Trans. Model. Comput. Simul. Vol. 8, No. 1 (1998)

Период  $2^{19937} - 1$.

RNG по умолчанию в R и Python

Звучат голоса против доминирования  Mersenne Twister
RNG Mersenne Twister не прошел 2+ теста...

Ждем новостей о новом фаворите

**RNG в  Python**

Модуль  random в Python 3.9.0 использует Mersenne Twister.
53-bit precision floats
период  2**19937-1.
Обертка кода, написанного на C.

**RNG в R**

**The default is "Mersenne-Twister".**

"Wichmann-Hill"

> The seed, .Random.seed[-1] == r[1:3] is an integer vector of length 3, where each r[i] is in 1:(p[i] - 1), where p is the length 3 vector of primes, p = (30269, 30307, 30323). The Wichmann–Hill generator has a cycle length of *6.9536e12* (= prod(p-1)/4, see Applied Statistics (1984) **33**, 123 which corrects the original article).

"Marsaglia-Multicarry":

> A multiply-with-carry RNG is used, as recommended by George Marsaglia in his post to the mailing list ‘`sci.stat.math`’. It has a period of more than *2^60* and has passed all tests (according to Marsaglia). The seed is two integers (all values allowed).

"Super-Duper":

> Marsaglia's famous Super-Duper from the 70's. This is the original version which does not pass the MTUPLE test of the Diehard battery. It has a period of *about 4.6*10^18* for most initial seeds. The seed is two integers (all values allowed for the first seed: the second must be odd).
>
> We use the implementation by Reeds et al (1982–84).
>
> The two seeds are the Tausworthe and congruence long integers, respectively. A one-to-one mapping to S's .Random.seed[1:12] is possible but we will not publish one, not least as this generator is **not** exactly the same as that in recent versions of S-PLUS.

"Mersenne-Twister":

From Matsumoto and Nishimura (1998); code updated in 2002. A twisted GFSR with period *2^19937 - 1* and equidistribution in 623 consecutive dimensions (over the whole period). The ‘seed’ is a 624-dimensional set of 32-bit integers plus a current position in that set.

R uses its own initialization method due to B. D. Ripley and is not affected by the initialization issue in the 1998 code of Matsumoto and Nishimura addressed in a 2002 update.

"Knuth-TAOCP-2002":

A 32-bit integer GFSR using lagged Fibonacci sequences with subtraction. That is, the recurrence used is

$$X[j] = (X[j-100] - X[j-37]) \bmod 2^{30}$$

and the ‘seed’ is the set of the 100 last numbers (actually recorded as 101 numbers, the last being a cyclic shift of the buffer). The period is around *2^129*.

"Knuth-TAOCP":

An earlier version from Knuth (1997).

The 2002 version was not backwards compatible with the earlier version: the initialization of the GFSR from the seed was altered. **R** did not allow you to choose consecutive seeds, the reported ‘weakness’, and already scrambled the seeds.

Initialization of this generator is done in interpreted **R** code and so takes a short but noticeable time.

"L'Ecuyer-CMRG":

A ‘combined multiple-recursive generator’ from L'Ecuyer (1999), each element of which is a feedback multiplicative generator with three integer elements: thus the seed is a (signed) integer vector of length 6. The period is around *2^191*.

The 6 elements of the seed are internally regarded as 32-bit unsigned integers. Neither the first three nor the last three should be all zero, and they are limited to less than 4294967087 and 4294944443 respectively.

This is not particularly interesting of itself, but provides the basis for the multiple streams used in package **parallel**.

"user-supplied": Use a user-supplied generator.

**Домашнее задание**

***Задача 1***. Парадокс Монти Холла

*Представьте, что вы стали участником игры, в которой вам нужно выбрать одну из трёх дверей. За одной из дверей находится автомобиль, за двумя другими дверями—козы. Вы выбираете одну из дверей, например, номер 1, после этого ведущий, который знает, где находится автомобиль, а где— козы, открывает одну из оставшихся дверей, например, номер 3, за которой находится коза. После этого он спрашивает вас — не желаете ли вы изменить свой выбор и выбрать дверь номер 2?*
*Какая стратегия дает Вам наибольшие шансы выиграть автомобиль?*
*А) Принять предложение ведущего и изменить свой выбор.*
*Б) Отвергнуть предложение ведущего и сохранить свой выбор.*
*В) Все равно, что делать. Стратегии А) и Б) дают Вам одинаковые шансы.*

***Задача 2***.

*Есть 1000 рублей.*
*Срочно нужно еще 1000 рублей.*

*Выход есть: играем в казино.*
*Вероятность выигрыша равна 18/37, ничьих не бывает.*

*Две стратегии*
*а) один раз ставим 1000 рублей*
*б) делаем ставки по одному рублю каждая, пока не выиграем 1000 рублей либо не проиграем все деньги.*

*При какой стратегии вероятность выигрыша выше?*