Минобрнауки РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Тульский государственный университет»

Кафедра информационной безопасности

Теория систем и системный анализ

Практическая работа № 5

ГЕНЕРАТОРЫ СЛУЧАЙНЫХ ЧИСЕЛ

ТУЛА 2023

Цель работы: знакомство с принципами генерации случайных чисел.

Краткие теоретические положения

В основе метода Монте-Карло лежит генерация случайных чисел, которые должны быть равномерно распределены в интервале (0; 1).

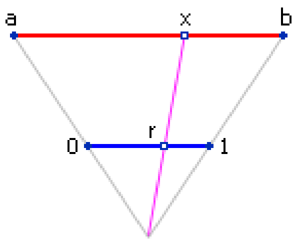
Если генератор выдает числа, смещенные в какую-то часть интервала (одни числа выпадают чаще других), то результат решения задачи, решаемой статистическим методом, может оказаться неверным. Поэтому проблема использования хорошего генератора действительно случайных и действительно равномерно распределенных чисел стоит очень остро.

Математическое ожидание m_r и дисперсия D_r такой последовательности, состоящей из n случайных чисел r_i, должны быть следующими (если это действительно равномерно распределенные случайные числа в интервале от 0 до 1):

$$m_r = \frac{\sum_{i=1}^{n} r_i}{n} = 0.5$$

$$D_r = \frac{\sum_{i=1}^{n} (r_i - m_r)^2}{n} = \frac{1}{12}$$

Если пользователю потребуется, чтобы случайное число x находилось в интервале $(a; b)$, отличном от (0; 1), нужно воспользоваться формулой $x = a + (b - a) \cdot r$, где r — случайное число из интервала (0; 1). Законность данного преобразования демонстрируется на **рис. 1**.



$$\frac{r-0}{1-0} = \frac{x-a}{b-a} \Leftrightarrow r = \frac{x-a}{b-a} \Leftrightarrow x = a + (b-a) \cdot r$$

Рис. 1. Схема перевода числа из интервала (0; 1) в интервал (a; b)

Теперь x — случайное число, равномерно распределенное в диапазоне от a до b.

За **эталон генератора случайных чисел** (ГСЧ) принят такой генератор, который порождает **последовательность** случайных чисел с *равномерным* законом распределения в интервале (0; 1). За одно обращение данный генератор возвращает одно случайное число. Если наблюдать такой ГСЧ достаточно длительное время, то окажется, что, например, в каждый из десяти интервалов (0; 0.1), (0.1; 0.2), (0.2; 0.3), …, (0.9; 1) попадет практически одинаковое количество случайных чисел — то есть они будут распределены равномерно по всему интервалу (0; 1). Если изобразить на графике $k = 10$ интервалов и частоты N_i попаданий в них, то получится экспериментальная кривая плотности распределения случайных чисел (см. **рис. 2**).

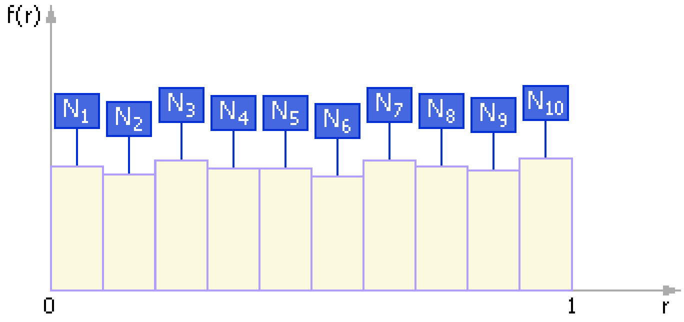


Рис. 2. Частотная диаграмма выпадения случайных чисел, порождаемых реальным генератором

Заметим, что в идеале кривая плотности распределения случайных чисел выглядела бы так, как показано на **рис. 3**. То есть в идеальном случае в каждый интервал попадает одинаковое число точек: $N_i = N/k$, где N — общее число точек, k — количество интервалов, $i = 1, \ldots, k$.

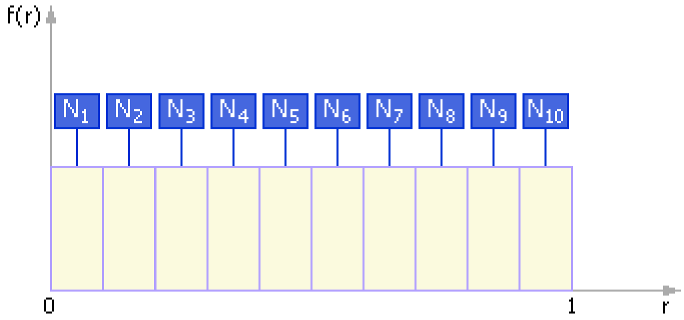


Рис. 3. Частотная диаграмма выпадения случайных чисел, порождаемых идеальным генератором теоретически

Следует помнить, что генерация произвольного случайного числа состоит из двух этапов:

- генерация нормализованного случайного числа (то есть равномерно распределенного от 0 до 1);
- преобразование нормализованных случайных чисел r_i в случайные числа x_i, которые распределены по необходимому пользователю (произвольному) закону распределения или в необходимом интервале.

Генераторы случайных чисел по способу получения чисел делятся на:

- физические;
- табличные;
- алгоритмические.

Физические ГСЧ

Примером физических ГСЧ могут служить: монета («орел» — 1, «решка» — 0); игральные кости; поделенный на секторы с цифрами барабан со стрелкой; аппаратурный генератор шума (ГШ), в качестве которого используют шумящее тепловое устройство, например, транзистор (**рис. 4–5**).

Рис. 4. Схема аппаратного метода генерации случайных чисел

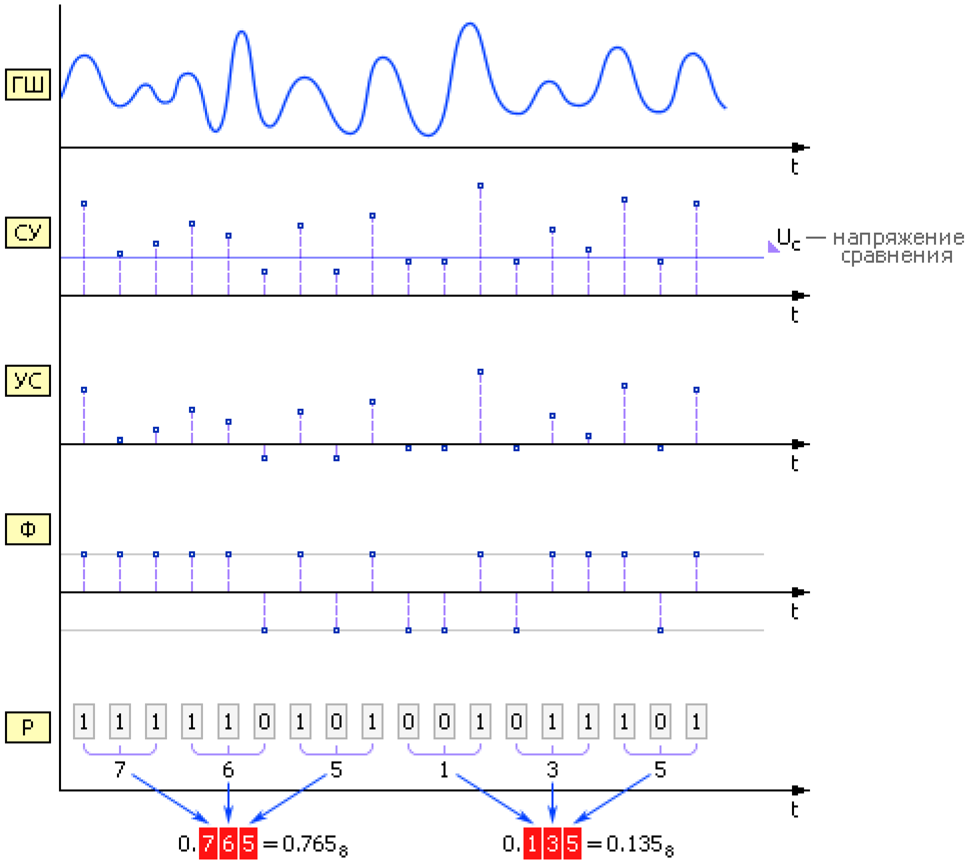


Рис. 5. Диаграмма получения случайных чисел аппаратным методом

Задача «Генерация случайных чисел при помощи монеты»

Сгенерируйте случайное трехразрядное число, распределенное по равномерному закону в интервале от 0 до 1, с помощью монеты. Точность — три знака после запятой.

Первый способ решения задачи Подбросьте монету 9 раз, и если монета упала решкой, то запишите «0», если орлом, то «1». Итак, допустим, что в результате эксперимента получили случайную последовательность 100110100.

Начертите интервал от 0 до 1. Считывая числа в последовательности слева направо, разбивайте интервал пополам и выбирайте каждый раз одну из частей очередного интервала (если выпал 0, то левую, если выпала 1, то правую). Таким образом, можно добраться до любой точки интервала, сколь угодно точно.

Итак, **1**: интервал [0; 1] делится пополам — [0; 0.5] и [0.5; 1], — выбирается правая половина, интервал сужается: [0.5; 1]. Следующее число, **0**: интервал [0.5; 1] делится пополам — [0.5; 0.75] и [0.75; 1], — выбирается левая половина [0.5; 0.75], интервал сужается: [0.5; 0.75]. Следующее число, **0**: интервал [0.5; 0.75] делится пополам — [0.5; 0.625] и [0.625; 0.75], — выбирается левая половина [0.5; 0.625], интервал сужается: [0.5; 0.625]. Следующее число, **1**: интервал [0.5; 0.625] делится пополам — [0.5; 0.5625] и [0.5625; 0.625], — выбирается правая половина [0.5625; 0.6250], интервал сужается: [0.5625; 0.6250].

По условию точности задачи решение найдено: им является любое число из интервала [0.5625; 0.6250], например, 0.625.

В принципе, если подходить строго, то деление интервалов нужно продолжить до тех пор, пока левая и правая границы найденного интервала не СОВПАДУТ между собой с точностью до третьего знака после запятой. То есть с позиций точности сгенерированное число уже не будет отличимо от любого числа из интервала, в котором оно находится.

Второй способ решения задачи
Разобьем полученную двоичную последовательность 100110100 на триады: 100, 110, 100. После перевода этих двоичных чисел в десятичные получаем: 4, 6, 4. Подставив спереди «0.», получим: 0.464. Таким методом могут получаться только числа от 0.000 до 0.777 (так как максимум, что можно «выжать» из трех двоичных разрядов — это $111_2 = 7_8$) — то есть, по сути, эти числа представлены в восьмеричной системе счисления. Для перевода *восьмеричного* числа в *десятичное* представление выполним: $0.464_8 = 4 \cdot 8^{-1} + 6 \cdot 8^{-2} + 4 \cdot 8^{-3} = 0.6015625_{10} = 0.602_{10}$.
Итак, искомое число равно: 0.602.

Табличные ГСЧ

Табличные ГСЧ в качестве источника случайных чисел используют специальным образом составленные таблицы, содержащие проверенные некоррелированные, то есть никак не зависящие друг от друга, цифры. В табл. 1 приведен небольшой фрагмент такой таблицы. Обходя таблицу слева направо сверху вниз, можно получать равномерно распределенные от 0 до 1 случайные числа с нужным числом знаков после запятой (в нашем примере мы используем для каждого числа по три знака). Так как цифры в таблице не зависят друг от друга, то таблицу можно обходить разными способами, например, сверху вниз, или справа налево, или, скажем, можно выбирать цифры, находящиеся на четных позициях.

Случайное число в заданном интервале (например, от 0 до 1) формируется из случайных цифр, записанных в таблице. Например,

таблица	число
3 5 2	0.352
7 4 1	0.741

Таблица 1. Случайные цифры. Равномерно распределенные от 0 до 1 случайные числа

Случайные цифры								Равномерно распределенные от 0 до 1 случайные числа
9	2	9	2	0	4	2	6	0.929
9	5	7	3	4	9	0	3	0.204
5	9	1	6	6	5	7	6	0.269
...								...

Достоинство данного метода в том, что он дает действительно случайные числа, так как таблица содержит проверенные некоррелированные цифры. Недостатки метода: для хранения большого количества цифр требуется много памяти; большие трудности порождения и проверки такого рода таблиц, повторы при использовании таблицы уже не гарантируют случайности числовой последовательности, а значит, и надежности результата.

Алгоритмические ГСЧ

Числа, генерируемые с помощью этих ГСЧ, всегда являются псевдослучайными (или квазислучайными), то есть каждое последующее сгенерированное число зависит от предыдущего:

$r_{i+1} = f(r_i)$.

Последовательности, составленные из таких чисел, образуют петли, то есть обязательно существует цикл, повторяющийся бесконечное число раз. Повторяющиеся циклы называются **периодами**.

Достоинством данных ГСЧ является быстродействие; генераторы практически не требуют ресурсов памяти, компактны. Недостатки: числа нельзя в полной мере назвать случайными, поскольку между ними имеется зависимость, а также наличие периодов в последовательности квазислучайных чисел.

Рассмотрим несколько алгоритмических методов получения ГСЧ:

- метод серединных квадратов;
- метод серединных произведений;
- метод перемешивания;
- линейный конгруэнтный метод.

Метод серединных квадратов

Имеется некоторое четырехзначное число $R0$. Это число возводится в квадрат и заносится в $R1$. Далее из $R1$ берется середина (четыре средних цифры) — новое случайное число — и записывается в $R0$. Затем процедура повторяется (см. **рис. 6**). Отметим, что на самом деле в качестве случайного числа необходимо брать не **ghij**, а **0.ghij** — с приписанным слева нулем и десятичной точкой. Этот факт отражен как на **рис. 6**, так и на последующих подобных рисунках.

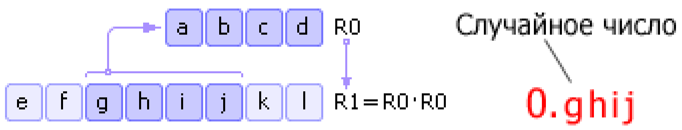


Рис. 6. Схема метода серединных квадратов

Недостатки метода: 1) если на некоторой итерации число $R0$ станет равным нулю, то генератор вырождается, поэтому важен правильный выбор начального значения $R0$; 2) генератор будет повторять последовательность через M^n шагов (в лучшем случае), где n — разрядность числа $R0$, M — основание системы счисления.

Для примера на **рис. 6**: если число $R0$ будет представлено в двоичной системе счисления, то последовательность псевдослучайных чисел повторится через $2^4 = 16$ шагов. Заметим, что повторение последовательности может произойти и раньше, если начальное число будет выбрано неудачно.

Описанный выше способ был предложен Джоном фон Нейманом и относится к 1946 году. Поскольку этот способ оказался ненадежным, от него очень быстро отказались.

Метод серединных произведений

Число $R0$ умножается на $R1$, из полученного результата $R2$ извлекается середина $R2^*$ (это очередное случайное число) и умножается на $R1$. По этой схеме вычисляются все последующие случайные числа (см. **рис. 7**).

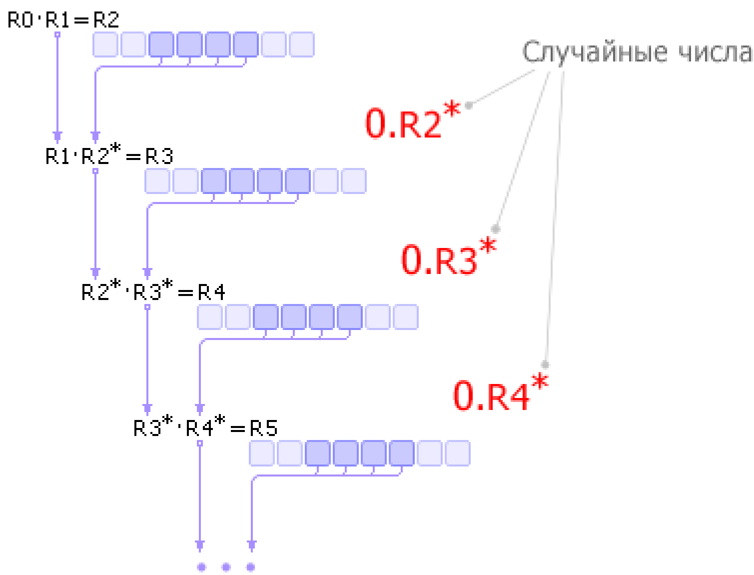


Рис. 7. Схема метода серединных произведений

Метод перемешивания

В методе перемешивания используются операции циклического сдвига содержимого ячейки влево и вправо. Идея метода состоит в следующем. Пусть в ячейке хранится начальное число $R0$. Циклически сдвигая содержимое ячейки влево на 1/4 длины ячейки, получаем новое число $R0^*$. Точно так же, циклически сдвигая содержимое ячейки $R0$ вправо на 1/4 длины ячейки, получаем второе число $R0^{**}$. Сумма чисел $R0^{*}$ и $R0^{**}$ дает новое случайное число $R1$. Далее $R1$ заносится в $R0$, и вся последовательность операций повторяется (см. **рис. 8**).

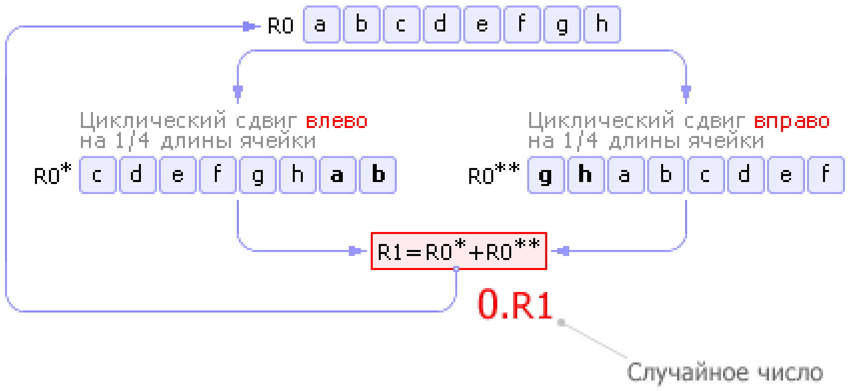


Рис. 8. Схема метода перемешивания

Обратите внимание, что число, полученное в результате суммирования $R0^{*}$ и $R0^{**}$, может не уместиться полностью в ячейке $R1$. В этом случае от полученного числа должны быть отброшены лишние разряды. Поясним это для **рис. 8**, где все ячейки представлены восемью двоичными разрядами. Пусть $R0^{*} = 10010001_2 = 145_{10}$, $R0^{**} = 10100001_2 = 161_{10}$,
тогда $R0^{*} + R0^{**} = 100110010_2 = 306_{10}$. Как видим, число 306 занимает 9 разрядов (в двоичной системе счисления), а ячейка $R1$ (как и $R0$) может вместить в себя максимум 8 разрядов. Поэтому перед занесением значения в $R1$ необходимо убрать один «лишний», крайний левый бит из числа 306, в результате чего в $R1$ пойдет уже не 306, а $00110010_2 = 50_{10}$. Также заметим, что в таких языках, как Паскаль, «урезание» лишних битов при переполнении ячейки производится автоматически в соответствии с заданным типом переменной.

Линейный конгруэнтный метод

Линейный конгруэнтный метод является одной из простейших и наиболее употребительных в настоящее время процедур, имитирующих случайные числа. В этом методе используется операция $\mathrm{mod}(x, y)$, возвращающая остаток от деления первого аргумента на второй. Каждое последующее случайное число рассчитывается на основе предыдущего случайного числа по следующей формуле:

$$r_{i+1} = \mathrm{mod}(k \cdot r_i + b, M).$$

M — модуль $(0 < M)$;
k — множитель $(0 \le k < M)$;
b — приращение $(0 \le b < M)$;
r_0 — начальное значение $(0 \le r_0 < M)$.

Последовательность случайных чисел, полученных с помощью данной формулы, называется **линейной конгруэнтной последовательностью**. Многие авторы называют линейную конгруэнтную последовательность при $b = 0$ **мультипликативным конгруэнтным методом**, а при $b \ne 0$ — **смешанным конгруэнтным методом**.

Для качественного генератора требуется подобрать подходящие коэффициенты. Необходимо, чтобы число M было довольно большим, так как период не может иметь больше M элементов. С другой стороны, деление, использующееся в этом методе, является довольно медленной операцией, поэтому для двоичной вычислительной машины логичным будет выбор $M = 2^N$, поскольку в этом случае нахождение остатка от деления сводится внутри ЭВМ к двоичной логической операции «AND». Также широко распространен выбор наибольшего простого числа M, меньшего, чем 2^N: в специальной литературе доказывается, что в этом случае младшие разряды получаемого случайного числа r_{i+1} ведут себя так же случайно, как и старшие, что положительно сказывается на всей последовательности случайных чисел в целом. В качестве примера можно привести одно из *чисел Мерсенна*, равное $2^{31} – 1$, и таким образом, $M = 2^{31} – 1$.

Одним из требований к линейным конгруэнтным последовательностям является как можно большая длина периода. Длина периода зависит от значений M, k и b. Теорема, которую мы приведем ниже, позволяет определить, возможно ли достижение периода максимальной длины для конкретных значений M, k и b.

Теорема. Линейная конгруэнтная последовательность, определенная числами M, k, b и r_0, имеет период длиной M тогда и только тогда, когда:

- числа b и M взаимно простые;
- $k – 1$ кратно p для каждого простого p, являющегося делителем M;
- $k – 1$ кратно 4, если M кратно 4.

Наконец, в заключение рассмотрим пару примеров использования линейного конгруэнтного метода для генерации случайных чисел.

Пример 1

$M = 2^N$
$k = 3 + 8 \cdot q$ (или $k = 5 + 8 \cdot q$)
$b = 0$
r_0 — нечетно

Было установлено, что ряд псевдослучайных чисел, генерируемых на основе данных из примера 1, будет повторяться через каждые $M/4$ чисел. Число q задается произвольно перед началом вычислений, однако при этом следует иметь в виду, что ряд производит впечатление случайного при больших k (а значит, и q). Результат можно несколько улучшить, если b нечетно и $k = 1 + 4 \cdot q$ — в этом случае ряд будет повторяться через каждые M чисел. После долгих поисков k исследователи остановились на значениях 69069 и 71365.

Пример 2

$M = 2^{31} – 1$
$k = 1\,220\,703\,125$
$b = 7$
$r_0 = 7$

Генератор случайных чисел, использующий данные из примера 2, будет выдавать случайные неповторяющиеся числа с периодом, равным 7 миллионам.

Мультипликативный метод генерации псевдослучайных чисел был предложен Д. Г. Лехмером (D. H. Lehmer) в 1949 году.

Проверка качества работы генератора

От качества работы ГСЧ зависит качество работы всей системы и точность результатов. Поэтому случайная последовательность, порождаемая ГСЧ, должна удовлетворять целому ряду критериев.

Осуществляемые проверки бывают двух типов:

- проверки на равномерность распределения;
- проверки на статистическую независимость.

Проверки на равномерность распределения

1) ГСЧ должен выдавать близкие к следующим значения статистических параметров, характерных для равномерного случайного закона:

$$m_r = \frac{\sum_{i=1}^{n} r_i}{n} \approx 0.5$$ — математическое ожидание;

$$D_r = \frac{\sum_{i=1}^{n} (r_i - m_r)^2}{n} \approx$$ — дисперсия;

$$\sigma_r = \sqrt{D_r} \approx 0.2887$$ — среднеквадратичное отклонение.

2) Частотный тест

Частотный тест позволяет выяснить, сколько чисел попало в интервал ($m_r - \sigma_r$; $m_r + \sigma_r$), то есть (0.5 – 0.2887; 0.5 + 0.2887) или, в конечном итоге, (0.2113; 0.7887). Так как 0.7887 – 0.2113 = 0.5774, заключаем, что в хорошем ГСЧ в этот интервал должно попадать около 57.7% из всех выпавших случайных чисел (см. **рис. 9**).

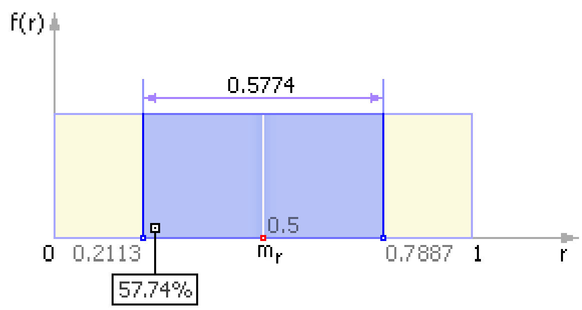


Рис. 9. Частотная диаграмма идеального ГСЧ в случае проверки его на частотный тест

Также необходимо учитывать, что количество чисел, попавших в интервал (0; 0.5), должно быть примерно равно количеству чисел, попавших в интервал (0.5; 1).

3) Проверка по критерию «хи-квадрат»

Критерий «хи-квадрат» (χ^2-критерий) — это один из самых известных статистических критериев; он является основным методом, используемым в сочетании с другими критериями. Критерий «хи-квадрат» был предложен в 1900 году Карлом Пирсоном. Его замечательная работа рассматривается как фундамент современной математической статистики.

Для нашего случая проверка по критерию «хи-квадрат» позволит узнать, насколько созданный нами *реальный* ГСЧ близок к эталону ГСЧ, то есть удовлетворяет ли он требованию равномерного распределения или нет.

Частотная диаграмма *эталонного* ГСЧ представлена на **рис. 10**. Так как закон распределения эталонного ГСЧ равномерный, то (теоретическая) вероятность p_i попадания чисел в i-ый интервал (всего этих интервалов k) равна $p_i = 1/k$. И, таким образом, в каждый из k интервалов попадет *ровно* по $p_i \cdot N$ чисел (N — общее количество сгенерированных чисел).

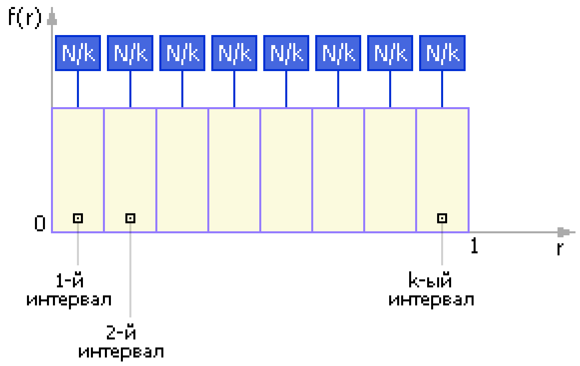


Рис. 10. Частотная диаграмма эталонного ГСЧ

Реальный ГСЧ будет выдавать числа, распределенные (причем, не обязательно равномерно!) по k интервалам и в каждый интервал попадет по n_i чисел (в сумме $n_1 + n_2 + \ldots + n_k = N$). Как же нам определить, насколько испытываемый ГСЧ хорош и близок к эталонному? Вполне логично рассмотреть квадраты разностей между полученным количеством чисел n_i и «эталонным» $p_i \cdot N$. Сложим их, и в результате получим:

$$\chi^2_{\text{эксп.}} = (n_1 - p_1 \cdot N)^2 + (n_2 - p_2 \cdot N)^2 + \ldots + (n_k - p_k \cdot N)^2.$$

Из этой формулы следует, что чем меньше разность в каждом из слагаемых (а значит, и чем меньше значение $\chi^2_{\text{эксп.}}$), тем сильнее закон распределения случайных чисел, генерируемых реальным ГСЧ, тяготеет к равномерному.

В предыдущем выражении каждому из слагаемых приписывается одинаковый вес (равный 1), что на самом деле может не соответствовать действительности; поэтому для статистики «хи-квадрат» необходимо провести нормировку каждого i-го слагаемого, поделив его на $p_i \cdot N$:

$$\chi^2_{\text{эксп.}} = \frac{(n_1 - p_1 \cdot N)^2}{p_1 \cdot N} + \frac{(n_2 - p_2 \cdot N)^2}{p_2 \cdot N} + \ldots + \frac{(n_k - p_k \cdot N)^2}{p_k \cdot N}$$

Наконец, запишем полученное выражение более компактно и упростим его:

$$\chi^2_{\text{эксп.}} = \sum_{i=1}^{k} \frac{(n_i - p_i \cdot N)^2}{p_i \cdot N} = \frac{1}{N} \sum_{i=1}^{k} \left(\frac{n_i^2}{p_i} \right) - N$$

Мы получили значение критерия «хи-квадрат» для *экспериментальных* данных.

В табл. 22.2 приведены *теоретические* значения «хи-квадрат» ($\chi^2_{теор.}$), где $\nu = N - 1$ — это число степеней свободы, **p** — это доверительная вероятность, задаваемая пользователем, который указывает, насколько ГСЧ должен удовлетворять требованиям равномерного распределения, или **p** — *это вероятность того, что экспериментальное значение* $\chi^2_{эксп.}$ *будет меньше табулированного (теоретического)* $\chi^2_{теор.}$ *или равно ему*.

Таблица 2. Некоторые процентные точки χ^2-распределения

	p = 1%	**p = 5%**	**p = 25%**	**p = 50%**	**p = 75%**	**p = 95%**	**p = 99%**
***ν* = 1**	0.00016	0.00393	0.1015	0.4549	1.323	3.841	6.635
***ν* = 2**	0.02010	0.1026	0.5754	1.386	2.773	5.991	9.210
***ν* = 3**	0.1148	0.3518	1.213	2.366	4.108	7.815	11.34
***ν* = 4**	0.2971	0.7107	1.923	3.357	5.385	9.488	13.28
***ν* = 5**	0.5543	1.1455	2.675	4.351	6.626	11.07	15.09
***ν* = 6**	0.8721	1.635	3.455	5.348	7.841	12.59	16.81
***ν* = 7**	1.239	2.167	4.255	6.346	9.037	14.07	18.48
***ν* = 8**	1.646	2.733	5.071	7.344	10.22	15.51	20.09
***ν* = 9**	2.088	3.325	5.899	8.343	11.39	16.92	21.67
***ν* = 10**	2.558	3.940	6.737	9.342	12.55	18.31	23.21
***ν* = 11**	3.053	4.575	7.584	10.34	13.70	19.68	24.72
***ν* = 12**	3.571	5.226	8.438	11.34	14.85	21.03	26.22
***ν* = 15**	5.229	7.261	11.04	14.34	18.25	25.00	30.58
***ν* = 20**	8.260	10.85	15.45	19.34	23.83	31.41	37.57
***ν* = 30**	14.95	18.49	24.48	29.34	34.80	43.77	50.89
***ν* = 50**	29.71	34.76	42.94	49.33	56.33	67.50	76.15
***ν* > 30**	ν + sqrt(2ν) · x_p + 2/3 · x^2_p – 2/3 + O(1/sqrt(ν))						
x_p =	–2.33	–1.64	–0.674	0.00	0.674	1.64	2.33

Приемлемым считают **p** *от 10% до 90%.*

Если $\chi^2_{эксп.}$ много больше $\chi^2_{теор.}$ (то есть **p** — велико), то генератор **не удовлетворяет** требованию равномерного распределения, так как наблюдаемые значения n_i слишком далеко уходят от теоретических $p_i \cdot N$ и не могут рассматриваться как случайные. Другими словами, устанавливается такой большой доверительный интервал, что ограничения на числа становятся очень нежесткими, требования к числам — слабыми. При этом будет наблюдаться очень большая абсолютная погрешность.

Еще Д. Кнут в своей книге «Искусство программирования» заметил, что иметь $\chi^2_{эксп.}$ маленьким тоже, в общем-то, нехорошо, хотя это и кажется, на первый взгляд, замечательно с точки зрения равномерности. Действительно, возьмите ряд чисел 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, … — они идеальны с точки зрения равномерности, и $\chi^2_{\text{эксп.}}$ будет практически нулевым, но вряд ли вы их признаете случайными.

Если $\chi^2_{\text{эксп.}}$ много меньше $\chi^2_{\text{теор.}}$ (то есть **p** — мало), то генератор **не удовлетворяет** требованию случайного равномерного распределения, так как наблюдаемые значения n_i слишком близки к теоретическим $p_i \cdot N$ и не могут рассматриваться как случайные.

А вот если $\chi^2_{\text{эксп.}}$ лежит в некотором диапазоне, между двумя значениями $\chi^2_{\text{теор.}}$, которые соответствуют, например, **p** = 25% и **p** = 50%, то можно считать, что значения случайных чисел, порождаемые датчиком, вполне являются случайными.

При этом дополнительно надо иметь в виду, что все значения $p_i \cdot N$ должны быть достаточно большими, например больше 5 (выяснено эмпирическим путем). Только тогда (при достаточно большой статистической выборке) условия проведения эксперимента можно считать удовлетворительными.

Итак, процедура проверки имеет следующий вид.

1. Диапазон от 0 до 1 разбивается на k равных интервалов.
2. Запускается ГСЧ N раз (N должно быть велико, например, $N/k > 5$).
3. Определяется количество случайных чисел, попавших в каждый интервал: n_i, $i = 1, \ldots, k$.
4. Вычисляется экспериментальное значение $\chi^2_{\text{эксп.}}$ по следующей формуле:
$$\chi^2_{\text{эксп.}} = \sum_{i=1}^{k} \frac{(n_i - p_i \cdot N)^2}{p_i \cdot N} = \frac{1}{N} \sum_{i=1}^{k} \left(\frac{n_i^2}{p_i} \right) - N$$
где $p_i = 1/k$ — теоретическая вероятность попадания чисел в k-ый интервал.
5. Путем сравнения экспериментально полученного значения $\chi^2_{\text{эксп.}}$ с теоретическим $\chi^2_{\text{теор.}}$ (из табл. 2) делается вывод о пригодности генератора для использования. Для этого: а) входим в табл. 2 (**строка = количество экспериментов – 1**); б) сравниваем вычисленное $\chi^2_{\text{эксп.}}$ с $\chi^2_{\text{теор.}}$, встречающимися в строке. При этом возможно три случая.

Первый случай: $\chi^2_{\text{эксп.}}$ много больше любого $\chi^2_{\text{теор.}}$ в строке — гипотеза о случайности равномерного генератора не выполняется (разброс чисел слишком велик, чтобы быть случайным).

Второй случай: $\chi^2_{\text{эксп.}}$ много меньше любого $\chi^2_{\text{теор.}}$ в строке — гипотеза о случайности равномерного генератора не выполняется (разброс чисел слишком мал, чтобы быть случайным).

Третий случай: $\chi^2_{\text{эксп.}}$ лежит между значениями $\chi^2_{\text{теор.}}$ двух рядом стоящих столбцов — гипотеза о случайности равномерного генератора выполняется с вероятностью **p** (то есть в **p** случаях из 100).

Заметим, что чем ближе получается **p** к значению 50%, тем лучше.

Проверки на статистическую независимость

1) Проверка на частоту появления цифры в последовательности

Рассмотрим пример. Случайное число 0.2463389991 состоит из цифр 2463389991, а число 0.5467766618 состоит из цифр 5467766618. Соединяя последовательности цифр, имеем: 24633899915467766618.

Понятно, что теоретическая вероятность p_i выпадения i-ой цифры (от 0 до 9) равна 0.1.

Далее следует вычислить частоту появления каждой цифры в выпавшей экспериментальной последовательности. Например, цифра 1 выпала 2 раза из 20, а цифра 6 выпала 5 раз из 20.

Далее считают оценку и принимают решение по критерию «хи-квадрат».

2) Проверка появления серий из одинаковых цифр

Обозначим через n_L число серий одинаковых подряд цифр длины L. Проверять надо все L от 1 до m, где m — это заданное пользователем число: максимально встречающееся число одинаковых цифр в серии.

В примере «24633899915467766618» обнаружены 2 серии длиной в 2 (33 и 77), то есть $n_2 = 2$ и 2 серии длиной в 3 (999 и 666), то есть $n_3 = 2$.

Вероятность появления серии длиной в L равна: $p_L = 9 \cdot 10^{-L}$ (теоретическая). То есть вероятность появления серии длиной в один символ равна: $p_1 = 0.9$ (теоретическая). Вероятность появления серии длиной в два символа равна: $p_2 = 0.09$ (теоретическая). Вероятность появления серии длиной в три символа равна: $p_3 = 0.009$ (теоретическая).

Например, вероятность появления серии длиной в один символ равна $p_L = 0.9$, так как всего может встретиться один символ из 10, а всего символов 9 (ноль не считается). А вероятность того, что подряд встретится два одинаковых символа «XX» равна $0.1 \cdot 0.1 \cdot 9$, то есть вероятность 0.1 того, что в первой позиции появится символ «X», умножается на вероятность 0.1 того, что во второй позиции появится такой же символ «X» и умножается на количество таких комбинаций 9.

Частость появления серий подсчитывается по ранее разобранной нами формуле «хи-квадрат» с использованием значений p_L.

Порядок выполнения работы

1. Используя вид генератора случайных чисел согласно варианту, сформировать последовательность 10 случайных чисел с 5-ю знаками после запятой в интервале [0,1]. Варианты

1	Физический метод	13	Физический метод
2	Табличный метод	14	Табличный метод
3	метод серединных квадратов	15	метод серединных квадратов
4	метод серединных произведений	16	метод серединных произведений
5	метод перемешивания	17	метод перемешивания
6	линейный конгруэнтный метод	18	линейный конгруэнтный метод
7	Физический метод	19	Физический метод

8	Табличный метод	20	Табличный метод
9	метод серединных квадратов	21	метод серединных квадратов
10	метод серединных произведений	22	метод серединных произведений
11	метод перемешивания	23	метод перемешивания
12	линейный конгруэнтный метод	24	линейный конгруэнтный метод

2. Проверить качество работы генератора всеми представленными в данных методических указаниях методами.