Федеральное агентство связи (Россвязь)
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра ВС

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к нулевой лабораторной работе
«Оценка качества генераторов псевдослучайных
чисел» по дисциплине «Моделирование»

Выполнил:
студент гр.ИВ-621 _____ /В.В.Симонова/
подпись

Проверил:
ассистент кафедры ВС
_____/Я.В. Петухова/
ОЦЕНКА, подпись

Новосибирск, 2020

Содержание

Постановка задачи

Подобрать три независимых друг от друга генератора случайных чисел, провести сравнительную оценку равномерности с помощью критерия согласия Пирсона («Хи-квадрат»), оценить автокорреляцию и сделать вывод, какой генератор равномернее и о чём это говорит.

Теоретические сведения

«Хи-квадрат» или критерий согласия Пирсона

С помощью критерия согласия определяют, удовлетворяет ли ГСЧ требования равномерного распределения или нет.

p_i – теоретическая вероятность попадания чисел в i-ый интервал (всего этих интервалов k) равна $p_i = 1/k$

N – общее количество сгенерированных чисел

n_i – попадание чисел в каждый интервал

$\chi 2$ – критерий, который позволяет определить, удовлетворяет ли генератор случайных чисел требованиям равномерного распределения или нет

Процедура проверки имеет следующий вид:

1. Диапазон от 0 до 1 разбивается на k равных интервалов
2. Запускается генератор случайных чисел N раз (N должно быть велико, например, $N/k > 5$)
3. Определяется количество случайных чисел, попавших в каждый интервал
4. Вычисляется экспериментальное значение $\chi 2$ по следующей формуле:

$$x^2 = \frac{1}{N}\sum_{i=1}^{k}\left(\frac{n_i^2}{p_i}\right) - N$$

Автокорреляция

Автокорреляция — это статистическая взаимосвязь между последовательностями величин одного ряда, взятыми со сдвигом. Определяется по формуле:

$$\hat{a}(\tau) = \frac{\sum_{i=1}^{N-\tau}(x_i - E_x)(x_{i+\tau} - E_x)}{(N-\tau) * S^2}$$

$$E_x = \sum_{i=1}^{N} \frac{x_i}{N}$$

$$S^2 = \sum \frac{x_i^2}{N} - (E_x)^2$$

E_x– математическое ожидание

S^2 – выборочная дисперсия

$\hat{a}(\tau)$ – автокорреляция

x_i– множество псевдослучайных чисел

τ – смещение

Генераторы

1. Вихрь Мерсе́нна (англ. Mersenne twister, MT) — генератор псевдослучайных чисел (ГПСЧ), который основывается на свойствах простых чисел Мерсенна (отсюда название) и обеспечивает быструю генерацию высококачественных по критерию случайности псевдослучайных чисел.

2. Генератор случайных чисел Xorshift, также называемый генератором регистра сдвига, представляет собой класс генераторов псевдослучайных чисел, которые были обнаружены Джорджем Марсаглией.

3. Линейно конгруэнтный метод (linear congruential) – применяется в простых случаях и не обладает криптографической стойкостью. Его использует *get_rand()* – функция из стандартной библиотеки C, которая генерирует псевдослучайные числа в диапазоне от 0 до RAND_MAX.

Результаты экспериментов

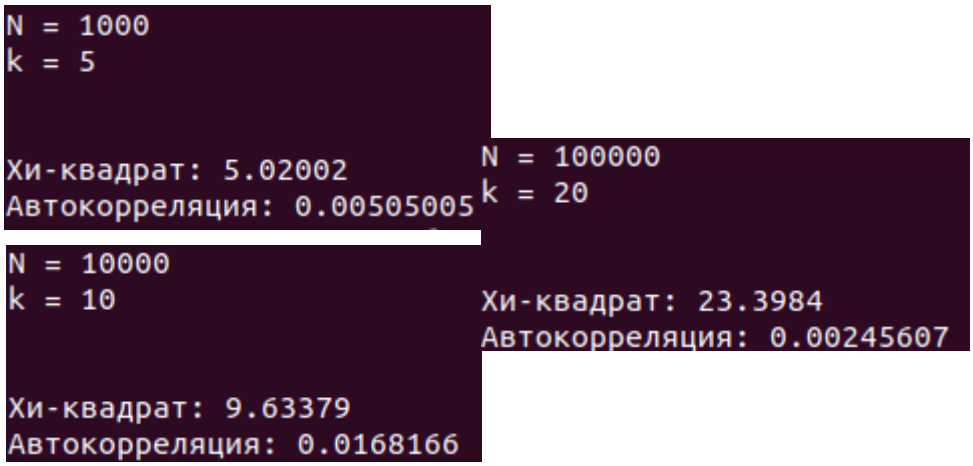

Рис.1. Результаты работы линейно-конгруэнтного генератора

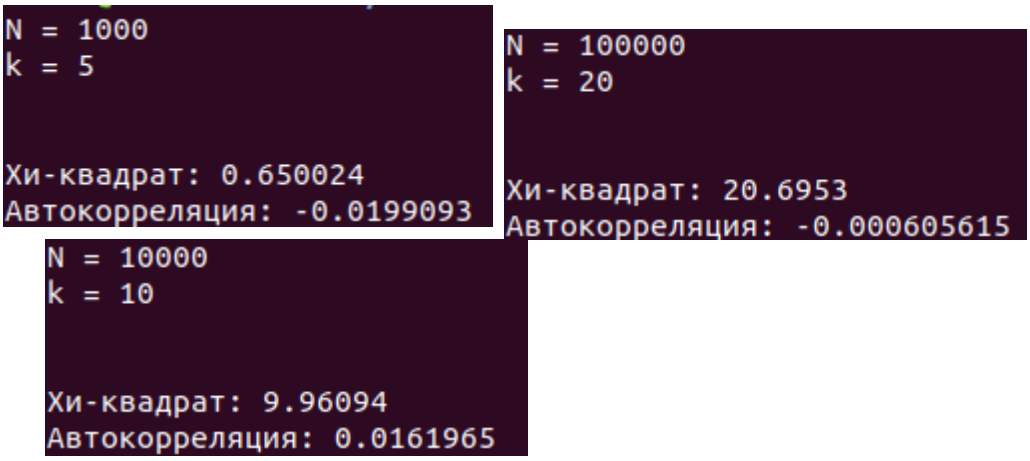

Рис.2. Результаты работы Xorshift

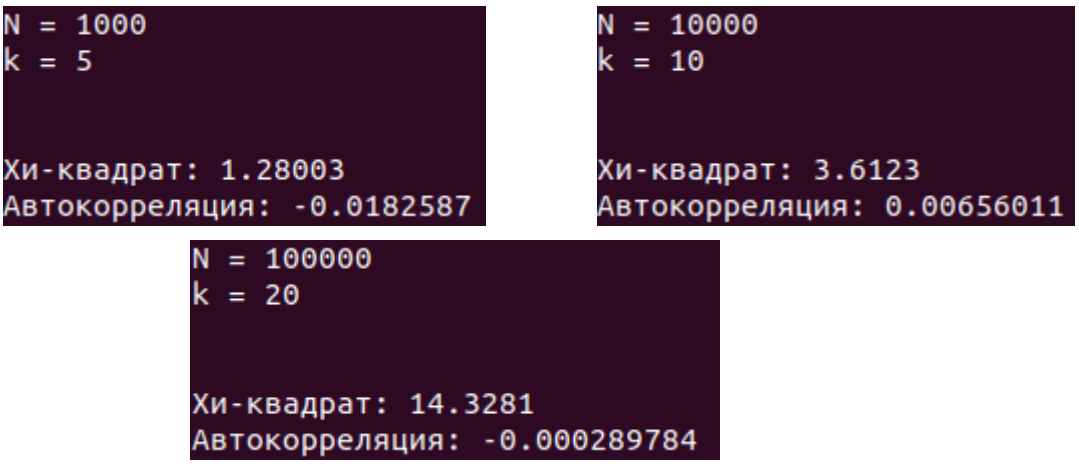

Рис.3. Результаты работы Вихря Мерсонна

Выводы

По критерию Пирсона:

$k = m - r - 1$ — степень свободы

m — количество интервалов (20 интервалов)

r — число параметров предполагаемого распределения (2 параметра у равномерного распределения)

$k = 20 - 2 - 1 = 17$ $k = 10 - 2 - 1 = 7$ $k = 5 - 2 - 1 = 2$

$\alpha = 0.05$

$\chi = 27.6$ $\chi = 14$ $\chi = 6$

Ближе всего к равномерному распределению – результаты, полученные с помощью генератора вихря Мерсенна, так как отклонение экспериментальных значений от теоретических больше, чем при других запусках. Нулевая гипотеза подтверждена для всех ГСЧ.

По автокорреляции:

Автокорреляционная функция для всех трех ГСЧ колеблется около 0, принимая положительные и отрицательные значения, близкие к нулю. Значит, статистическая взаимосвязь между исходной и сдвинутой последовательностью пренебрежимо мала, и можно сделать вывод о том, что исследуемые генераторы выдают независимые случайные величины.

Листинг

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <math.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <ctime>
#include <cmath>
#include <chrono>

#define N 1000

using namespace std;
const float RAND_MAX_F = RAND_MAX;

float get_rand() {
	return rand() / RAND_MAX_F;
}

float get_rand_range(const float min, const float max) {
	return get_rand() * (max - min) + min;
}
int main(){
	int k = 5;
	int count[k];
	int index =0;
	float numeric[N];

	mt19937 gen(time(0));
	uniform_real_distribution<> urd(0, 1);

	unsigned seed =
	std::chrono::system_clock::now().time_since_epoch().count();
	std::uniform_real_distribution<float> dist(0, 1);
	std::mt19937_64 rng(seed);

	for(int i = 0; i < k; i++)
	{
		count[i] = 0;
	}
	srand(time(NULL));
	float Q = 0.0;
	float sum = 0, sum_kvdr = 0;

	for(int i = 0; i < N; i++)
	{
		//float gch = get_rand_range(0, 1);
		//float gch = urd(gen);
		float gch = dist(rng);
		numeric[i] = gch;
		sum += gch;
		sum_kvdr += gch * gch;
		Q = 0.0;
		for(index = 0; index < k; index++){
			if(gch > Q && gch <= Q + (float)1/k){
				count[index]++;
				 break;
```

```
                }
                Q += (float)1/k
            }
        }
        cout << "N = " << N << "\nk = " << k << "\n\n";
        int interv = 1;
        for(int i = 0; i < k; i++){
            interv++;
        }
        float HI = 0.0;
        for(int i = 0; i < k; i++){
            HI += (float)count[i] * count[i] / (1 / (float)k);
        }
        HI /= (float)N;
        HI -= (float)N;

        cout << "\n" << "Хи-квадрат: " << HI << "\n";
        float Ex = 0.0;
        Ex = sum / (float) N;
        float S = 0.0;
        S = (sum_kvdr / (float) N) - (Ex * Ex);
        float autoc = 0.0;
        for(int taaay = 1; taaay < 500; taaay++){
            for(int i = 0; i < N - taaay; i++)
        {
        autoc +=(float) ((numeric[i] - Ex) * (float) (numeric[i + taaay] - Ex));
        }
            autoc /= (float)(N - taaay) * S ;
        }
        cout << "Автокорреляция: " << autoc << "\n";
        return 0;
}
```