ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГОБУ ВПО “СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ”

Кафедра вычислительных систем

Лабораторная работа №0
по дисциплине «Моделирование»

Выполнил:
студент гр. ИВ-622
Гайнулин Р.М.

Проверил:
Ассистент кафедры ВС
Петухова Я.В.

Новосибирск, 2020

Оглавление

Постановка задачи

Анализировать равномерности распределения трех генераторов случайных чисел, используя при этом параметры критерия «хи-квадрат» и автокорреляции.

Теоретические сведения

Генератор псевдослучайных чисел

ГПСЧ — алгоритм, который порождает последовательность чисел, элементы которой почти независимы друг от друга и подчиняются заданному распределению (обычно равномерному). ГПСЧ использует единственное начальное значение, откуда и следует его псевдослучайность. ГПСЧ имеет предсказуемую зависимость между числами и малую длину генерируемой последовательности случайных чисел. После этого генератор обязательно зациклится.

Rand (rand())

Rand() — стандартный генератор языка C/C++. Источником энтропии является счетчик тактов процессора, однако собирается только во время прерываний.

RdRand от Intel (__builtin_ia32_rdrand64_step())

RdRand — реализация цифрового генератора случайных чисел (DRNG). Источником энтропии являются шумы токов, ПСЧ строятся на основе случайного битового считывания значений от токов. Является очень быстрым и не застревает.

Mersenne twister (mt19937_64())

Вихрь Мерсенна — ГПСЧ, к которого источник энтропии основан на свойствах простых чисел Мерсенна. В конструкторе может инициализироваться величиной, от которой начинается генерация последовательности, либо специальным объектом seed_seq, являющимся зерном последовательности. Имеется 32-битный генератор и 64-битный.

χ^2-критерий

Критерий «хи-квадрат» (χ^2-критерий) — это один из самых известных статистических критериев; он является основным методом, используемым в сочетании с другими критериями.

Частотная диаграмма эталонного ГСЧ равномерная, 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). Реальный ГСЧ будет выдавать числа, распределенные не равномерно по k интервалам и в каждый интервал попадет по n_i чисел (в сумме $n_1 + n_2 + \ldots + n_k = N$).

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

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

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;

χ^2– критерий, который позволяет определить, удовлетворяет ли ГСЧ требования равномерного распределения или нет.

χ^2 проверяет значимость расхождения эмпирических (наблюдаемых) и теоретических (ожидаемых) значений. Чем больше количество значений N, тем большее значение χ^2-критерия, так как каждое слагаемое имеет вклад в общей сумме. То есть для разного количества значений N будет свое распределение.

Если случайные числа в действительности соответствуют ожидаемым значениям, то значение критерия будет относительно небольшим. То есть большинство отклонений находится около значения N/k. Так же если критерий имеет большое число, то это свидетельствует о том, что имеются существенные отклонения между теоретическим значением попадания в отрезок и действительными .

Автокорреляция a(τ)

Автокорреляция — это корреляционная связь между значениями одного и того же случайного процесса.

Математическое ожидание $Ex = \frac{1}{n}\sum_{i=1}^{n} x_i$

Выборочная дисперсия $S^2(x) = \frac{1}{N}\sum_{i=1}^{n} x_i^2 - (Ex)^2$

Автокорреляция $a(\tau) = \frac{\sum_{i=1}^{n-\tau}(x_i - Ex)(x_{i+\tau} - Ex)}{(n-\tau)*S^2(x)}$

Ex – математическое ожидание;

$S^2(x)$ – выборочная дисперсия;

$a(\tau)$ – автокорреляция;

x_i – множество псевдослучайных чисел;

$x_{i+\tau}$ – множество псевдослучайных чисел со смещением.

Значение автокорреляции *a(τ)* зависит от значения количества псевдослучайных чисел, а именно их количество на интервалах

Результаты экспериментов

Исследуем три случая с разными значениями случайных чисел N интервалов k. В качестве анализируемых данных возьмем:

- N = 100 000 случайных чисел и k = 100 интервалов;
- N = 100000 случайных чисел и k = 50 интервалов;
- N = 100000 случайных чисел и k = 100 интервалов.

И рассмотрим распределение чисел на интервалах в разных случаях использования трех разных генераторов псевдослучайных чисел.

χ^2-критерий

Таблица 1. Значения хи-квадрат для трех генераторов псевдослучайных чисел

Количество чисел, N	100 000	100 000	1 000 000
Количество интервалов, k	100	50	100
Стандартный ГПСЧ Rand	**110.682000**	**42.153000**	**105.3306**
ГПСЧ RdRand от Intel	**99.184000**	**39.407000**	**91.9238**
ГПСЧ Mersenne twister	**86.513000**	**36.394500**	**92.4076**

Для первого и третьего столбца табличное значение хи-квадрат $\chi^2_{\text{таб.}} = 113.1$, так же для второго столбца $\chi^2_{\text{таб.}} = 50.9$. По отношению $\chi^2_{\text{эксп.}} < \chi^2_{\text{таб.}}$ можно сказать, что гипотезы о равновероятном распределении в трех генераторах случайных чисел принимаются.

Таким образом, нам стало известно, что чем меньше значение критерия χ^2, тем более равномерное распределение, то есть равновероятное на всех интервалах. Так же, если больше значение количества случайных чисел, это значит, что будет больше и отклонение от равномерного распределения.

Автокорреляция *a(τ)*

Автокорреляция для каждого ГПСЧ при N = 1000000 и k = 100.

```
Autocorrelation Rand
offset = 1       a(τ) = 0.152891
offset = 2       a(τ) = -0.014829
offset = 3       a(τ) = -0.031406
offset = 4       a(τ) = -0.037964
offset = 5       a(τ) = -0.063582
offset = 6       a(τ) = 0.118890
offset = 7       a(τ) = -0.022728
offset = 8       a(τ) = -0.170422
offset = 9       a(τ) = -0.080772
offset = 10      a(τ) = 0.071430
offset = 11      a(τ) = -0.062511
offset = 12      a(τ) = 0.026266
offset = 13      a(τ) = -0.153286
offset = 14      a(τ) = 0.105913
offset = 15      a(τ) = 0.221582
offset = 16      a(τ) = 0.370421
offset = 17      a(τ) = -0.033138
offset = 18      a(τ) = -0.192713
offset = 19      a(τ) = -0.141493
offset = 20      a(τ) = -0.114539
```

Рис. 1 — Автокорреляция Rand

```
Autocorrelation RdRand
offset = 1       a(τ) = -0.198497
offset = 2       a(τ) = 0.182221
offset = 3       a(τ) = -0.079402
offset = 4       a(τ) = -0.031313
offset = 5       a(τ) = -0.056640
offset = 6       a(τ) = 0.056284
offset = 7       a(τ) = -0.235018
offset = 8       a(τ) = 0.160541
offset = 9       a(τ) = -0.334103
offset = 10      a(τ) = 0.082748
offset = 11      a(τ) = 0.077873
offset = 12      a(τ) = -0.097811
offset = 13      a(τ) = 0.078648
offset = 14      a(τ) = -0.106122
offset = 15      a(τ) = -0.182608
offset = 16      a(τ) = 0.226612
offset = 17      a(τ) = -0.100111
offset = 18      a(τ) = 0.085208
offset = 19      a(τ) = 0.004617
offset = 20      a(τ) = 0.001070
```

Рис. 2 — Автокорреляция RdRand

```
Autocorrelation Mersenne
offset = 1      a(τ) = 0.004726
offset = 2      a(τ) = 0.035118
offset = 3      a(τ) = -0.066145
offset = 4      a(τ) = -0.074673
offset = 5      a(τ) = 0.008863
offset = 6      a(τ) = 0.042499
offset = 7      a(τ) = -0.007142
offset = 8      a(τ) = -0.003078
offset = 9      a(τ) = 0.036837
offset = 10     a(τ) = -0.189711
offset = 11     a(τ) = -0.032548
offset = 12     a(τ) = -0.033522
offset = 13     a(τ) = 0.203877
offset = 14     a(τ) = -0.004602
offset = 15     a(τ) = 0.272798
offset = 16     a(τ) = -0.099812
offset = 17     a(τ) = -0.049434
offset = 18     a(τ) = 0.161678
offset = 19     a(τ) = -0.016669
offset = 20     a(τ) = -0.105518
```

Рис. 3 — Автокорреляция Mersenne twister

Во всех случаях автокорреляционная функция имеет очень слабую корреляцию. При изменении количества интервалов k получаем, что последовательность случайных сгенерированных чисел и последовательность сгенерированных чисел со смещением не имеет взаимосвязи между собой.

Таким образом, более равномерное распределение случайных чисел получается путем увеличения числа случайных чисел и уменьшения числа интервалов и иначе получаем менее равномерное распределение.

Заключение

В данной лабораторной работе был реализован программный код, в ходе написания которого использовались 3 генератора псевдослучайных чисел на основе функции rand, цифрового генератора случайных чисел и алгоритма Mersenne twister, значения генератора чисел образовывало последовательность случайных чисел, которые были распределены на некое заданное количество интервалов в зависимости от данных значений. Далее были написаны функции расчета значения критерия хи-квадрат и значения автокорреляции, так же построены графики распределения случайных величин по интервалам при варьировании количества случайных чисел и числа интервалов и было проведено исследование равномерного распределения по вышеуказанным критериям.

По результатам значений критерия хи-квадрат $\chi^2_{\text{эксп.}rand} = 110.682$, $\chi^2_{\text{эксп.}RdRand} = 99.184$, $\chi^2_{\text{эксп.}Mersenne} = 86.513$, и табличному значению критерия $\chi^2_{\text{таб.}} = 113.1$ и на данном отношении $\chi^2_{\text{эксп.}} < \chi^2_{\text{таб.}}$ можно сказать, что гипотезы о равновероятном распределении в генераторах случайных чисел принимаются.

На основе исследования можно сказать, что значение критерия хи-квадрат меньше тем, чем более равномерное распределение. Если случайные числа в действительности соответствуют тем самым ожидаемым значениям, то значение критерия будет относительно небольшим и распределение будет равномерным.

Так же, исходя из исследований, значение коэффициента автокорреляции, стремящееся к нулю, соответствует более равномерному распределению. При изменении значения смещения (τ) автокорреляционная функция приближена к нулю, следовательно, имеется очень слабая корреляция.

Приложение. Листинг

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <random>
#include <limits.h>

void rand_rdrand_64(int *num, int begin, int end) { // -mrdrnd
  unsigned long long rand64;
  if ( __builtin_ia32_rdrand64_step(&rand64) ) {
    *num = (int)((float)rand64/ULONG_MAX*(end - begin) + begin);
  }
  return;
}

int rand_mt19937_64(int begin, int end) {
    std::random_device rd;
    std::mt19937_64 gen(rd());
    std::uniform_int_distribution<int> uid(begin, end);
    return uid(gen);
}

int main() {
  srand(time(NULL));
  int max_n = 1000000, pseudo_random[max_n], pseudo_random_mt[max_n], temp;
  int max_interval = 100, pseudo_random_to_interval[max_interval];
  double num1;

  for (int i = 0; i < max_n; i++)
    pseudo_random[i] = 0;
  for (int i = 0; i < max_interval; i++)
    pseudo_random_to_interval[i] = 0;

  for (int i = 0; i < max_n; i++) {
    // pseudo_random[i] = rand() % max_n; // RAND
    // rand_rdrand_64(&pseudo_random[i], 0, max_n); //RDRAND
    pseudo_random[i] = rand_mt19937_64(0, max_n); //MERSENE TWISTER
    num1 = (double)pseudo_random[i] / (double)max_n;
    temp = (int)((double)num1 / (1.0/(double)max_interval));
    pseudo_random_to_interval[temp]++;
  }
  for (int i = 0; i < max_interval; i++) {
    double inter = (double)(i+1)/(double)max_interval;
    printf("\n%.2f %d", inter, pseudo_random_to_interval[i]);
  }

  //hi_exp2
  printf("\n\nHI_SQUARE");
  double hi_exp2 = 0.0;
  double p = 1.0 / (double)max_interval;
  for (int i = 0; i < max_interval; i++)
    hi_exp2 += (pow((double)pseudo_random_to_interval[i], 2) / p);
  hi_exp2 = (hi_exp2 / (double)max_n) - (double)max_n;
  printf("\nHI_exp2 = %f\n", hi_exp2);

  //autocorrelation
  printf("\nAutocorrelation Mersenne\n");
  double autocorrelation = 0.0, ex = 0.0;
  for (int i = 0; i < max_interval; i++)
      ex += pseudo_random_to_interval[i];
  ex /= max_interval;

  double sx2 = 0.0;
  for (int i = 0; i < max_interval; i++)
      sx2 += (pow(pseudo_random_to_interval[i], 2) - pow(ex, 2));
  sx2 /= max_interval;


  for (int offset = 1; offset <= max_interval / 2; offset++) {
      double dispersion = 0.0;
      /*double ex2_x = 0.0, x = 0.0;
      double ex2_y = 0.0, y = 0.0;*/
      autocorrelation = 0.0;
      for (int i = 0; i < max_interval - offset; i++) {
```

```
        autocorrelation += (pseudo_random_to_interval[i] - ex) * (pseudo_random_to_interval[i+offset] - ex);
      }
      autocorrelation /= ((max_interval-offset) * sx2);
    printf("offset = %d\ta( τ ) = %lf\n", offset, autocorrelation);
  }
  return 0;
}
```