Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение  
высшего образования

**«Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»**

**Институт Информационных технологий, математики и механики**

**Кафедра: программной инженерии**

Отчет по лабораторным работам по курсу «Компьютерное моделирование вероятностных процессов»:

**Выполнил:** студент группы 381603-3

Кумин

Алексей Александрович

Подпись

**Преподаватель:**

Ассистент

Кочеганова Мария Анатольевна

Подпись

Нижний Новгород

2019

**Оглавление**

[1 Лабораторная работа 1 - Поэтапное моделирование эксперимента. 5](#_Toc535847375)

[1.1 Постановка задачи 5](#_Toc535847376)

[1.2 Теоретическая часть 5](#_Toc535847377)

[1.2.1 Генератор случайных величин, используемый для решения задачи 5](#_Toc535847378)

[1.2.2 Моделирование эксперимента 6](#_Toc535847379)

[1.3 Функции, используемые в программе 6](#_Toc535847380)

[1.3.1 Функция моделирования эксперимента 6](#_Toc535847381)

[1.3.2 Функция вычисления относительной частоты при проведении N экспериментов 7](#_Toc535847382)

[1.4 Аналитическое вычисление вероятности (вычислительная часть) 7](#_Toc535847383)

[1.5 Проведение экспериментов 8](#_Toc535847384)

[1.6 Выводы 8](#_Toc535847385)

[2 Лабораторная работа 2 – Вычисление значения определенного интеграла методами Монте – Карло. 9](#_Toc535847386)

[2.1 Постановка задачи 9](#_Toc535847387)

[2.1.1 Задача А 9](#_Toc535847388)

[2.1.2 Задача Б 9](#_Toc535847389)

[2.2 Теоретическая часть 10](#_Toc535847390)

[2.2.1 Датчик случайных чисел 10](#_Toc535847391)

[2.2.2 Метод Монте-Карло 1 10](#_Toc535847392)

[2.2.3 Метод Монте-Карло 2 (вычисление интеграла методом Монте-Карло с помощью усреднения подынтегральной функции) 10](#_Toc535847393)

[2.2.4 Моделирование случайных точек 11](#_Toc535847394)

[2.3 Функции, используемые в программе 11](#_Toc535847395)

[2.3.1 Функция моделирования с.в. на отрезке [a,b] 11](#_Toc535847396)

[2.3.2 Метод Монте-Карло 1 для интеграла первого рода 11](#_Toc535847397)

[2.3.3 Метод Монте-Карло 2 для интеграла первого рода 12](#_Toc535847398)

[2.4 Вычисление интегралов аналитически 12](#_Toc535847399)

[2.4.1 Интеграл 1 рода (для задачи А) 12](#_Toc535847400)

[2.4.2 Интеграл 3 рода (для задачи Б) 12](#_Toc535847401)

[2.5 Проведение экспериментов 12](#_Toc535847402)

[2.5.1 Для задачи А 12](#_Toc535847403)

[2.5.2 Для задачи Б 13](#_Toc535847404)

[2.6 Выводы 13](#_Toc535847405)

[3 Лабораторная работа 3 14](#_Toc535847406)

[3.1 Постановка задачи 14](#_Toc535847407)

[3.2 Теоретическая часть 14](#_Toc535847408)

[3.2.1 Датчик случайных чисел 14](#_Toc535847409)

[3.2.2 Метод обратной функции 14](#_Toc535847410)

[3.2.3 Моделирование случайной величины на [0, 1] 15](#_Toc535847411)

[3.2.4 Понятия основных статистических характеристик 15](#_Toc535847412)

[3.3 Вычислительная часть 15](#_Toc535847413)

[3.3.1 Метод обратной функции 15](#_Toc535847414)

[3.3.2 Математическое ожидание 16](#_Toc535847415)

[3.3.3 Мода 16](#_Toc535847416)

[3.3.4 Медиана 16](#_Toc535847417)

[3.3.5 Дисперсия 16](#_Toc535847418)

[3.4 Функции, используемые в программе 16](#_Toc535847419)

[3.4.1 Функции реализации случайных величин 16](#_Toc535847420)

[3.4.2 Нахождение вариационного ряда 17](#_Toc535847421)

[3.4.3 Математическое ожидание 17](#_Toc535847422)

[3.4.4 Мода 17](#_Toc535847423)

[3.4.5 Дисперсия 17](#_Toc535847424)

[3.4.6 Медиана 18](#_Toc535847425)

[3.5 Проведение экспериментов 18](#_Toc535847426)

[3.6 Выводы 19](#_Toc535847427)

[4 Используемая литература 21](#_Toc535847428)

# Лабораторная работа 1 - Поэтапное моделирование эксперимента.

## Постановка задачи

15 экзаменационных билетов содержат по 2 вопроса, которые не повторяются. Билет берет студент, который знает только 25 вопросов. Для успешной сдачи экзамена достаточно ответить на два вопроса из своего билета или на один вопрос из своего билета и на указанные дополнительный вопрос из другого билета.

Повторить эксперимент (путем моделирования) N раз в одних и тех же условиях и найти относительную частоту события A = {студент сдаст экзамен}.

Найти вероятность события А теоретически.

Величина N вводится с клавиатуры.

*Входные данные:* N – кол-во повторений эксперимента.

*Выходные данные:* Относительная частота события A.

## Теоретическая часть

### Генератор случайных величин, используемый для решения задачи

Для создания случайных чисел, необходимых для моделирования эксперимента, в языке программирования C++ будем использовать генератор случайных чисел rand(). Данная функция выводит с равной вероятностью целые числа из диапазона [0..RAND\_MAX], где RAND\_MAX = 32767(Для Microsoft Visual Studio). Сама функция rand() один раз генерирует случайное число, а при последующих запусках программы отображает эти же самое число. Чтобы получать при каждом запуске программы разные числа необходимо воспользоваться функцией srand(unsigned int seed), которая принимает на вход целое положительное число типа unsigned int (семя генератора – seed) и на этой основе rand() генерирует случайные числа. Чтобы каждый раз выводились разные числа, необходимо чтобы аргумент srand() менялся. Для этого будем использовать функцию time(0), которая возвращает текущее календарное время в секундах (srand(time(0))). Для использования этой функции необходимо подключить библиотеку <ctime>, а для двух предыдущих функций - <random>. Таким образом при каждом запуске программы rand() будет возвращать разные целые случайные числа с равной вероятностью в промежутке [0..RAND\_MAX].

*Пример:*

#include <random>

#include <ctime>

int main()

{

srand(time(0));

int a = rand();

}

### Моделирование эксперимента

Моделирование эксперимента происходит в несколько этапов:

1. Чтобы представить нужное нам событие будем использовать вектор w размерности 3, где первый элемент вектора w[0] – первый вопрос из доставшегося студенту билета, w[1] – второй вопрос, w[2] –дополнительный вопрос.
2. Пронумеруем вопросы: 1 – 0, 2 – 1, … 29 – 28, 30 – 29, причем вопросы 0..24 те которые студент знает, а 25..29 не знает.
3. Чтобы случайно с равной вероятностью получить число из промежутка [0..29] применим генератор сл. ч. rand() и найдем остаток от деления от него:

w[0] = rand() % 30;

1. Заполним вектор w этими случайными числами так, так чтобы элементы вектора не были равны, если w[1] = w[0], то генерируем сл.ч. для w[1] заново пока w[1] и w[0] не станут разными, так же поступим и с w[2].
2. Таким образом если w[1] и w[2] = числу из промежутка [0..24], то студент знает вопросы из билета и сдаст экзамен, если один из этих элементов = числу из промежутка [25..29], то студент не знает одного вопроса из билета, и ему дается дополнительный вопрос w[2], если w[2] = числу из промежутка [0..24], то студент знает этот вопрос и сдаст экзамен, а во всех остальных случаях не сдаст.

## Функции, используемые в программе

### Функция моделирования эксперимента

int Model()

{

int w[3];

w[0] = rand() % 30;

w[1] = rand() % 30;

while (w[1] == w[0])

w[1] = rand() % 30;

w[2] = rand() % 30;

while ((w[2] == w[0]) || (w[2] == w[1]))

w[2] = rand() % 30;

if (((w[0] < 25) && (w[1] < 25)) || ((w[0] < 25) && (w[2] < 25)) || ((w[1] < 25) && (w[2] < 25)))

return 1;

else

return 0;

}

### Функция вычисления относительной частоты при проведении N экспериментов

double vN(int N)

{

srand(time(0));

int k = 0;

for (int i = 0; i < N; i++)

k += Model();

return (double)k / double(N);

}

## Аналитическое вычисление вероятности (вычислительная часть)

Для нахождения вероятности события A = {студент сдаст экзамен} воспользуемся классическим определением вероятности:

где n – число *несовместных равновероятных* *элементарных исходов*, благоприятствующих событию А, N – число всевозможных *элементарных исходов* N.

Обозначим множество всех элементарных исходов - Ω, а элементарные исходы - ꞷ, тогда:

где x – номер вопроса, тогда число всевозможных элементарных исходов, это мощность множества Ω:

Пусть 1, 2 … 24, 25 – вопросы, которые студент выучил, тогда событие A описывается:

Тогда вероятность события A:

## Проведение экспериментов

Относительная частота – это отношение числа экспериментов, удовлетворяющих событию, к общему числу проведенных экспериментов:

Ниже представлена таблица кол-ва проведенных экспериментов и относительной частоты события A:

|  |  |
| --- | --- |
| N – число проведенных экспериментов | Относительная частота эксп. A (υ(A)) |
| 10 | 0.8 |
| 100 | 0.91 |
| 1000 | 0.932 |
| 10000 | 0.9372 |
| 100000 | 0.93619 |
| 1000000 | 0.935942 |
| 10000000 | 0.935889 |

## Выводы

Как видно из таблицы, чем больше проведено экспериментов, тем меньше разница между относительной частотой события A и его вероятностью, найденной аналитически. Говорят, что относительная частота события сходится к его вероятности по вероятности:

# Лабораторная работа 2 – Вычисление значения определенного интеграла методами Монте – Карло.

## Постановка задачи

### Задача А

1. Реализовать 2 метода Монте-Карло для подсчета определенного интеграла вида
2. Посчитать точное значение интеграла аналитически
3. Провести серию из увеличивающегося количества испытаний и вывести результаты в таблицу следующего вида

|  |  |  |
| --- | --- | --- |
|  |  |  |

1. Вывести на экран графические представление 1го метода Монте-Карло (считающего частоту попадания под график функции) для заданного кол-ва испытаний.

### Задача Б

1. Реализовать 2 метода Монте-Карло для подсчета определенного интеграла вида
2. Посчитать аналитическое значение интеграла
3. Провести серию из увеличивающегося кол-ва испытаний и вывести результаты в таблицу следующего вида

|  |  |  |
| --- | --- | --- |
|  |  |  |

В качестве функций *f(x), f(x, y, z)* и соответствующих областей выбрать любые функции, для которых интегралы существуют и вычислимы алгебраически.

## Теоретическая часть

### Датчик случайных чисел

В качестве датчика случайных чисел возьмем функцию rand(), которая описывалась ранее (пункт 1.2.1.).

### Метод Монте-Карло 1

1. Пусть – независимые реализации случайных величин (точек) , где – равномерно распределена на , а на , где

Тогда или

В случае интеграла третьего рода используются вектора размерности 4:

где

1. Подсчитаем число точек и они попали в подграфик функции f(x)
2. Подсчитаем число точек и они попали в подграфик функции f(x)
3. Тогда интеграл будет вычисляться по формуле:

### Метод Монте-Карло 2 (вычисление интеграла методом Монте-Карло с помощью усреднения подынтегральной функции)

Пусть - величина равномерно распределенная на [a,b], g(x) ее плотность, тогда

Тогда математическое ожидание от с.в.:

И интеграл I будет выражаться через статистическое математическое ожидание:

### Моделирование случайных точек

Пусть x реализация случ. вел. равномерно распределенной на отрезке [a,b], а y – на [0,1], тогда:

по методу обратной функции.

Таким образом моделирование х осуществляется с помощью у, а у в свою очередь моделируется с помощью функции rand():

## Функции, используемые в программе

### Функция моделирования с.в. на отрезке [a,b]

double model(double a, double b)

{

return (a + ((b - a) / (double)RAND\_MAX) \* (double)rand());

}

### Метод Монте-Карло 1 для интеграла первого рода

double MK1(int N, double a, double b, double Fmax, double Fmin)

{

double x, f;

int m1 = 0, m2 = 0;

for (int i = 0; i < N; i++)

{

x = model(a, b);

f = model(Fmin, Fmax);

if ((f > 0) && (f <= F(x)))

m1++;

if ((f < 0) && (f >= F(x)))

m2++;

}

return ((double)(m1 - m2) / (double)N)\*(b - a)\*(Fmax - Fmin);

}

### Метод Монте-Карло 2 для интеграла первого рода

double MK2(int N, double a, double b)

{

double x, f;

double Mx = 0;

for (int i = 0; i < N; i++)

{

x = model(a, b);

Mx += F(x);

}

return Mx \* (b - a) / (double)N;

}

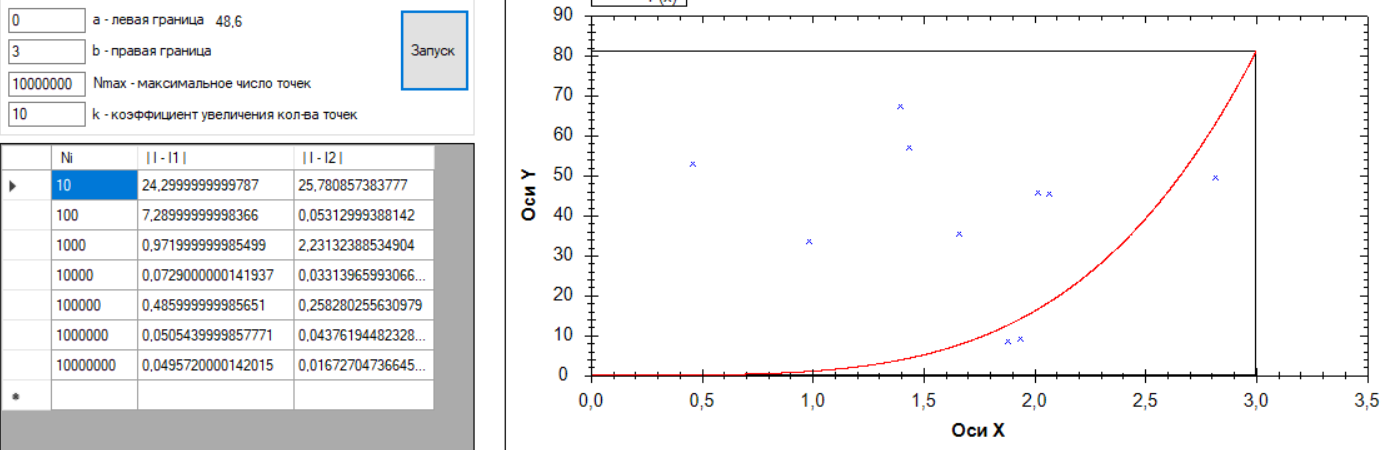
## Вычисление интегралов аналитически

### Интеграл 1 рода (для задачи А)

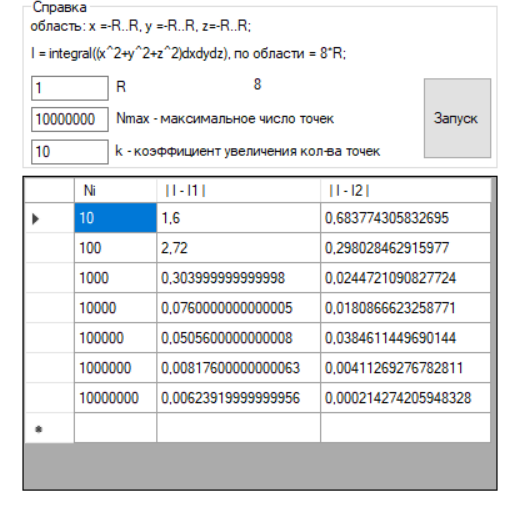
### Интеграл 3 рода (для задачи Б)

## Проведение экспериментов

### Для задачи А



### Для задачи Б



## Выводы

Как видно из проведенных экспериментов, чем большее кол-во точек мы берем, тем точнее вычисленный интеграл, правда это не всегда выполняется, что видно на приведенных выше примерах, но в общем случае это так. Нельзя сказать какой метод лучше, так как в данных экспериментах присутствует элемент случайности, и в одном эксперименте разброс точек может быть более удачным в одном методе, а в другом эксперименте во втором.

# Лабораторная работа 3

## Постановка задачи

Пусть  - непрерывная случайная величина с плотностью распределения вероятностей

.

Посчитать математическое ожидание и дисперсию случайной величины.

Используя метод обратной функции, получить выражение, позволяющее моделировать реализации случайной величины  по реализациям случайной величины , равномерно распределенной в интервале (0, 1).

Написать программу, решающую следующие задачи:

1) проведение N экспериментов по моделированию случайной величины (N задается с клавиатуры)

2) вывод результатов моделирования в виде вариационного ряда;

3) подсчет и вывод на экран основных статистических характеристик случайной величины (размах выборки, статистические математическое ожидание, мода, медиана и дисперсия)

## Теоретическая часть

### Датчик случайных чисел

В качестве датчика случайных чисел возьмем функцию rand(), которая описывалась ранее (пункт 1.2.1.).

### Метод обратной функции

Пусть - непрерывная случайная величина, тогда у нее есть плотность распределения:

Пусть x – реализация равномерной сл.вел. на [0,1], тогда , где у – реализация .

### Моделирование случайной величины на [0, 1]

Моделирование случайной величины равномерно распределенной на [0, 1] происходит с помощью генератора rand():

### Понятия основных статистических характеристик

*Вариационный ряд* – ряд, в котором представлены все случайные числа, полученные в ходе испытания, и расположены в порядке возрастания.

*Размах выборки* – разница между самым большим и самым меньшим значениями вариационного ряда (область, в которой лежат все полученные в ходе испытания числа).

*Математическое ожидание* – среднее значение случайной величины, при стремлении кол-ва выборок к бесконечности:

(для непрерывной сл.в.)

*Статистические математическое ожидание* – среднее значение случайной величины, среди всего имеющегося количества выборок, полученных в ходе испытания:

, где х – реализация .

*Мода* – это точка x, в которой функция плотности случайной величины принимает максимальное значение (точка в которой случайная величина чаще всего принимает значения)

*Медиана* – точка x, слева от которой случайная величина принимает свои значения с той же вероятностью что и справа

*Дисперсия* – мера разброса случайной величины относительно ее математического ожидания

## Вычислительная часть

### Метод обратной функции

– имеет показательное распределение,

x – реализация равномерно распределенной на [0, 1]

В нашем случае , тогда

### Математическое ожидание

### Мода

Для функции максимальное значение будет достигаться в точке x = 0, значит мода

### Медиана

По определению

### Дисперсия

## Функции, используемые в программе

### Функции реализации случайных величин

double relKsi()

{

double rel = rand();

while (rel == RAND\_MAX)

rel = rand();

return (((double)rel + 0.5) / (double)RAND\_MAX);

}

double relEta()

{

return (-log(1 - relKsi()) / 2);

}

### Нахождение вариационного ряда

void vectEta::vectE(int N)

{

srand(time(0));

for (int i = 0; i < N; i++)

{

Eta.push\_back(relEta());

if (Eta[i] > Etamax)

Etamax = Eta[i];

if (Eta[i] < Etamin)

Etamin = Eta[i];

}

quickSort(Eta, 0, Eta.size() - 1);

}

### Математическое ожидание

double vectEta::M()

{

double M = 0;

for (int i = 0; i < Eta.size(); i++)

M += Eta[i];

return M / (double)Eta.size();

}

### Мода

double vectEta::Mo()

{

double Max = 0, Mo = 0;

for (int i = 0; i < f.size(); i++)

if (f[i] > Max)

{

Max = f[i];

Mo = EtaMid[i];

}

return Mo;

}

### Дисперсия

double vectEta::D()

{

vectEta E2(Eta.size(), h);

for (int i = 0; i < E2.Eta.size(); i++)

{

double Eta2 = relEta();

E2.Eta[i] = Eta2 \* Eta2;

if (E2.Eta[i] > E2.Etamax)

E2.Etamax = E2.Eta[i];

if (Eta[i] < Etamin)

E2.Etamin = E2.Eta[i];

}

E2.h = h;

E2.vectX();

E2.vectV\_EtaMid();

double m = M();

return (E2.M() - m \* m);

}

### Медиана

double vectEta::Me()

{

if (Eta.size() % 2)

return Eta[(Eta.size() / 2) + 1];

else

return (Eta[(Eta.size() / 2)] + Eta[(Eta.size() / 2) + 1]) / (double)2;

}

## Проведение экспериментов

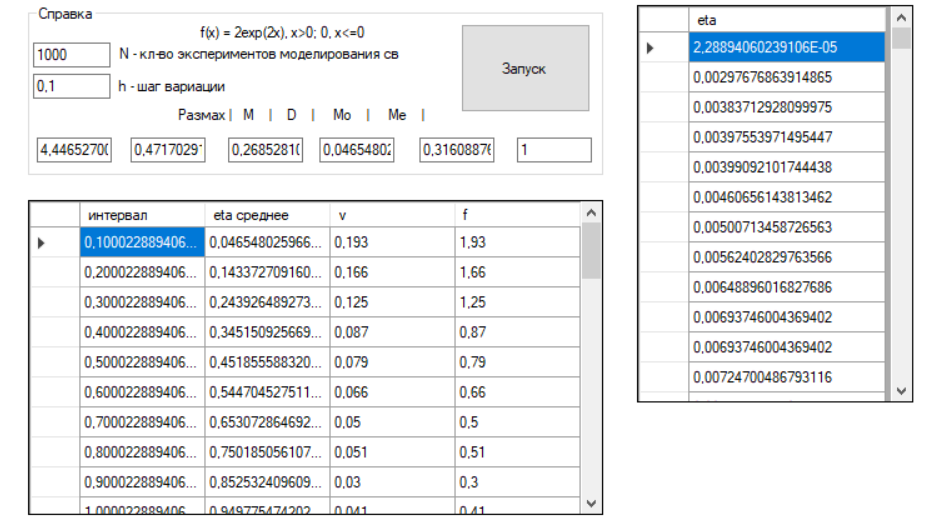


Рис.3 1000 экспериментов по моделированию случайной величины и статистические данные.

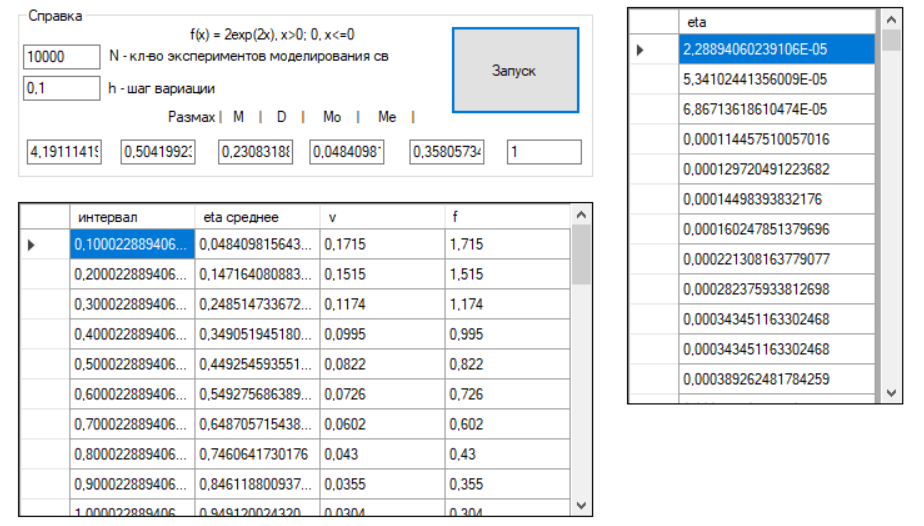


Рис.4 10000 экспериментов.

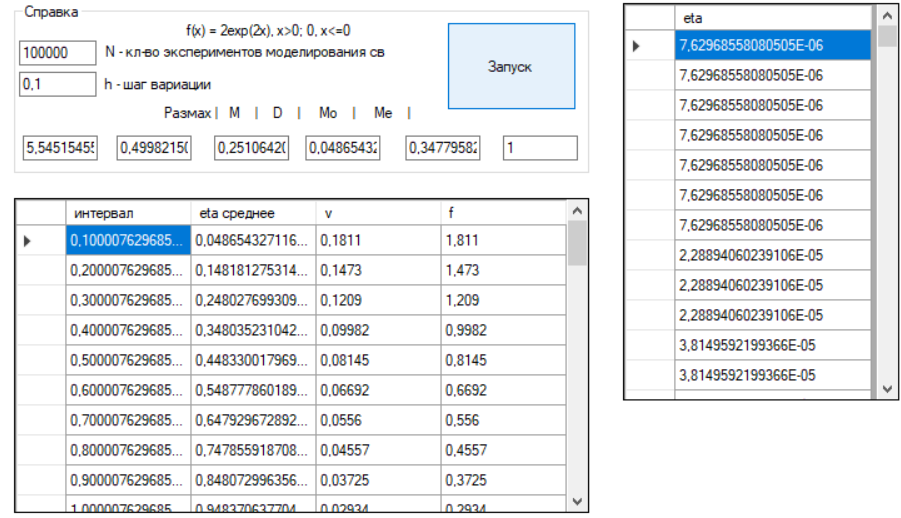


Рис.5 100000 экспериментов.

## Выводы

На основе проведенных опытов можно сделать вывод, что точность статистических данных зависит от величины вариационного ряда и соответственно от кол-ва проведенных экспериментов по моделированию случайной величины. Чем больше проведено экспериментов ближе статистические величины исследуемой случайной величины к этим же значениям, вычисленным аналитически. Отсюда вытекает основной вывод: с помощью метода обратной функции действительно можно моделировать непрерывные случайные величины с различной плотностью распределения, и он корректен для исследуемый случайной величины. Корректность метода подтверждена практически, на основе полученных опытов.

# Используемая литература

1. Лекционный материал курса Компьютерное моделирование вероятностных процессов 2018г.