Министерство науки и высшего образования РФ

Рязанский государственный радиотехнический университет им В.Ф.Уткина

Кафедра вычислительной и прикладной математики

Лабораторная работа №

«Тактическое планирование эксперимента»

Выполнил: ст. гр. 744

Овсянников С.А.

Проверил: профессор

Овечкин Г.В.

Рязань 2020

**Задание**

Вариант: 11

Одномерное случайное блуждание с переменным шагом.

На основе результатов лабораторной №7 осуществить планирование эксперимента – найти оптимальный объем выборки, при котором не будет потеряна точность, а также, не будет потрачено больше вычислительных ресурсов, чем необходимо

Кол-во интервалов разбиения: 16  
 Выборка: 570

Код:

static int experimentsCount = 100;

static double rightStepProbability = 0.5;

static int stepsCount = 10;

static Random random = new Random();

private static int circlesCount = 570;

public static double[] Calculate()

{

random = new Random();

double x1;

double x2;

double[] randomValues = new double[circlesCount];

double L;

double D;

for(int k = 0; k < circlesCount; k++)

{

x1 = 0;

x2 = 0;

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

{

double x = 0.0;

for (int j = 0; j < stepsCount; j++)

{

double r = random.NextDouble();

if (r < rightStepProbability)

{

x = DoStep(random, x, true);

}

else

{

x = DoStep(random, x, false);

}

}

x1 += Math.Abs(x);

x2 += x \* x;

}

L = x1 / experimentsCount;

D = x2 / experimentsCount - L \* L;

randomValues[k] = L;

}

return randomValues;

}

private static double DoStep(Random random, double x, bool direction)

{

double a = 0;

for(int j = 0; j < 100; j++)

{

a += Math.Exp(-j);

}

int i = 0;

double p = 0;

double r = random.NextDouble();

if (direction)

{

while (r > p)

{

i++;

double tmp = Math.Exp(-i);

double tmp2 = 2 \* a \* tmp;

p += tmp2;

}

x += i;

}

else

{

while (r > p)

{

i++;

double tmp = Math.Exp(-i);

double tmp2 = 2 \* a \* tmp;

p += tmp2;

}

x += -i;

}

return x;

}

double[] randomValues = WalkModeling.Calculate();

leftXBorder = randomValues.Min();

rightXBorder = randomValues.Max();

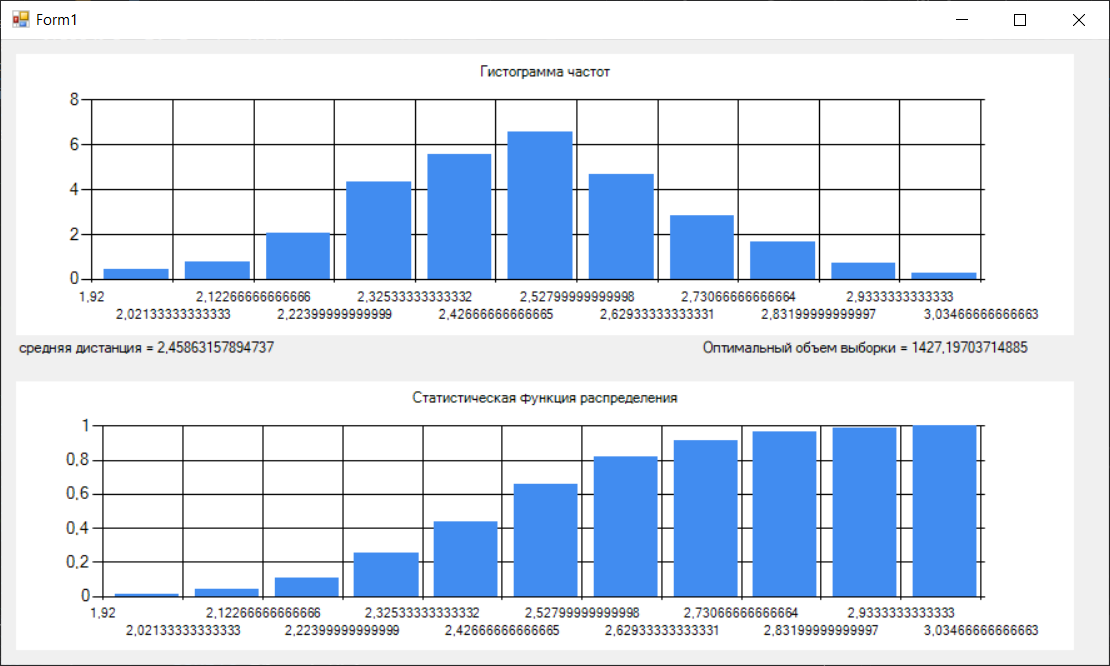
double averageDistance = randomValues.Sum() / randomValues.Length;

double mx = StatisticData.CalculateMathExpectation(randomValues);

double dx = StatisticData.CalculateDispersoin(randomValues,mx);

double N = tb \* tb \* dx / (d \* d);

Результат выполнения:



По оси X: значения дистанций, на которые «цель ушла от столба» в ту или иную сторону.