**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Кафедра Прикладной математики

Лабораторная работа № 3

по теории вероятностей и математической статистике

«Вычисление определенного интеграла методом Монте-Карло»

Вариант 26

Студент \_\_\_\_\_\_\_\_\_\_\_ Шутов А.С.

Группа ПМ-21-2

Руководитель

Ассистент каф. ПМ \_\_\_\_\_\_\_\_\_\_\_ Домашнева Е.Л.

Липецк 2023г.

# **Задание к лабораторной работе:**

**1 часть.**

Вспомнить встроенные функции пакета Excel: СЛЧИС(), СЛУЧМЕЖДУ(), СЧЁТ() и СЧЁТЕСЛИ(). Прочитать информацию о генераторе случайных чисел в Excel на странице 19 учебного пособия.

**2 часть.**

1) Изучить информацию по выполнению лабораторной работы на страницах 34 – 40 учебного пособия.

2) Выбрать функцию из таблицы ниже в соответствии с номером варианта.

3) Изобразить график функции, выделить область D для генерации пар случайных чисел.

4) Разработать программу, вычисляющую определенный интеграл методом Монте-Карло, чтобы с вероятностью p = 0,9 обеспечить точность вычисления интеграла ε = 0,01.

5) Провести 10 серий вычислений, после чего найти среднее арифметическое.

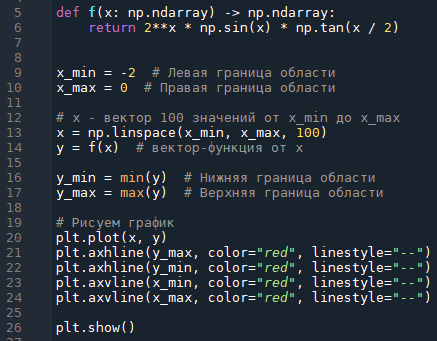
6) Сделать выводы по работе, оформить отчет.

|  |  |
| --- | --- |
| **Вариант** | **Задание** |
| 26 |  |

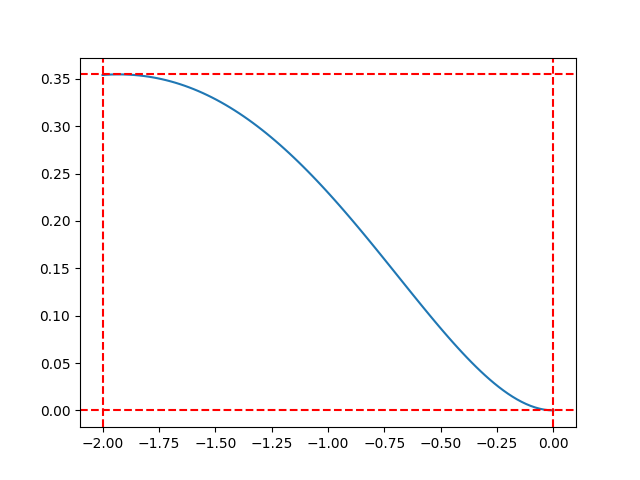
**Ход работы:**



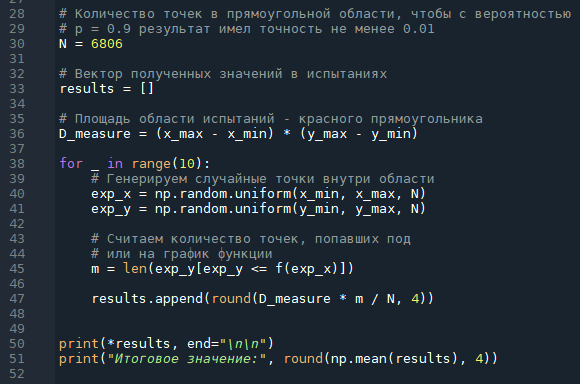
1. Для выполнения данной работы был использован язык программирования Python. Библиотека *matplotlib* нужна для построения графиков, *numpy* для поддержки больших многомерных массивов и очень быстрых математических функций для операций с этими массивами.



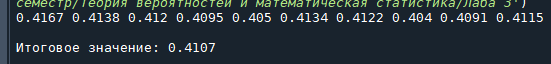
2. Определим функцию f (x) по варианту. Произведем отрисовку данной функции, далее изобразим пунктирной линией прямоугольник вокруг функции – область, в пределах которой будут проводиться статистические испытания.



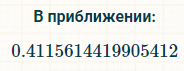
3. Результат выполнения приведённого блока кода.



4. Генерируем случайным образом точки внутри заданной области испытаний (в пределах красного прямоугольника). Для обеспечения точности вычислений ε = 0.01 с вероятностью р = 0.9 количество случайно сгенерированных точек равно N = 6806. Считаем количество точек, оказавшихся на графике функции или под ним, и по формуле находим приближенное значение интеграла заданной функции.



5. Получим результат: вектор значений, полученных в 10 испытаниях, и их среднее арифметическое как «итоговое» значение интеграла.



6. С помощью стороннего калькулятора интегралов вычислим заданный интеграл.

7. Результаты совпали с точностью до 0,01 и оказались достаточно близки.

**Код программы:**

**import** matplotlib**.**pyplot **as** plt

**import** numpy **as** np

**def** f**(**x**:** np**.**ndarray**)** **->** np**.**ndarray**:**

**return** 2**\*\***x **\*** np**.**sin**(**x**)** **\*** np**.**tan**(**x **/** 2**)**

x\_min **=** **-**2 # Левая граница области

x\_max **=** 0 # Правая граница области

# х - вектор 100 значений от x\_min до x\_max

x **=** np**.**linspace**(**x\_min**,** x\_max**,** 100**)**

y **=** f**(**x**)** # вектор-функция от х

y\_min **=** **min(**y**)** # Нижняя граница области

y\_max **=** **max(**y**)** # Верхняя граница области

# Рисуем график

plt**.**plot**(**x**,** y**)**

plt**.**axhline**(**y\_max**,** color**=**"red"**,** linestyle**=**"--"**)**

plt**.**axhline**(**y\_min**,** color**=**"red"**,** linestyle**=**"--"**)**

plt**.**axvline**(**x\_min**,** color**=**"red"**,** linestyle**=**"--"**)**

plt**.**axvline**(**x\_max**,** color**=**"red"**,** linestyle**=**"--"**)**

plt**.**show**()**

# Количество точек в прямоугольной области, чтобы с вероятностью

# р = 0.9 результат имел точность не менее 0.01

N **=** 6806

# Вектор полученных значений в испытаниях

results **=** **[]**

# Площадь области испытаний - красного прямоугольника

D\_measure **=** **(**x\_max **-** x\_min**)** **\*** **(**y\_max **-** y\_min**)**

**for** \_ **in** **range(**10**):**

# Генерируем случайные точки внутри области

exp\_x **=** np**.**random**.**uniform**(**x\_min**,** x\_max**,** N**)**

exp\_y **=** np**.**random**.**uniform**(**y\_min**,** y\_max**,** N**)**

# Считаем количество точек, попавших под

# или на график функции

m **=** **len(**exp\_y**[**exp\_y **<=** f**(**exp\_x**)])**

results**.**append**(round(**D\_measure **\*** m **/** N**,** 4**))**

**print(\***results**,** end**=**"\n\n"**)**

**print(**"Итоговое значение:"**,** **round(**np**.**mean**(**results**),** 4**))**

**Вывод:**

Метод Монте-Карло позволяет получать приближенные значения интегралов сложных функций, близкие к верным. Тем не менее, если требуется высокая точность, данный метод не подойдёт или потребует огромного количества вычислений, что не всегда возможно, и всё равно приведёт лишь к приближенным значениям с точностью до некоторого знака после запятой. Однако для исследования некоторых свойств изучаемых объектов данный метод статистических испытаний может применяться с большим успехом.