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

Кафедра ВС

Лабораторная работа 2
по дисциплине «Моделирование»

Выполнил: ст. гр.
ЗМП-41 Лёвкин И. А.

Проверил: Родионов А.С.

Новосибирск 2025

Задание

Построить датчик точек, равномерно распределённых в заданной области и по её периметру.

Ход работы:

1. Даны 2 функции:
   $y = 3(x-2)^2$
   $y = x^2 + 2x$
2. Точка пересечения фнукций
   $3(x-2)^2 = x^2 + 2x$
   $x = 1$

$x_{max} = 2$

$y_{max} = 3$

3. Нахождение функции распределения:

$$F(x) = \frac{S_0}{S}$$

$$S = \int_0^1 3(x-2)^2\,dx + \int_1^2 x^2 + 2x\,dx = 7 + \frac{16}{3} = \frac{37}{3}$$

$$S_0 = \int_0^{\frac{x}{2}} 3(t-2)^2\,dt + \int_{\frac{x}{2}}^{x} t^2 + 2t\,dt = \left(\frac{x^3}{8} - \frac{3x^2}{2} + 6x\right) + \left(\frac{7x^3}{24} + \frac{3x^2}{4}\right)$$

$$= \frac{5x^3}{12} - \frac{3x^2}{4} + 6x$$

$$F(x) = \frac{\left(\frac{5x^3}{12} - \frac{3x^2}{4} + 6x\right)}{\frac{37}{3}} = \frac{3\left(\frac{x^3}{12} - \frac{3x^2}{4} + 6x\right)}{37}$$

4. Рассчитаем длину дуг

$$\int_0^1 \sqrt{1 + ((3(t-2)^2)')^2}\ dx = \int_0^1 \sqrt{1 + 36(x-2)^2} = 10.677$$

$$\int_1^2 \sqrt{1 + ((x^2+2x)')^2}\ dx = \int_1^2 \sqrt{1 + (2x+2)^2} = 5.001$$

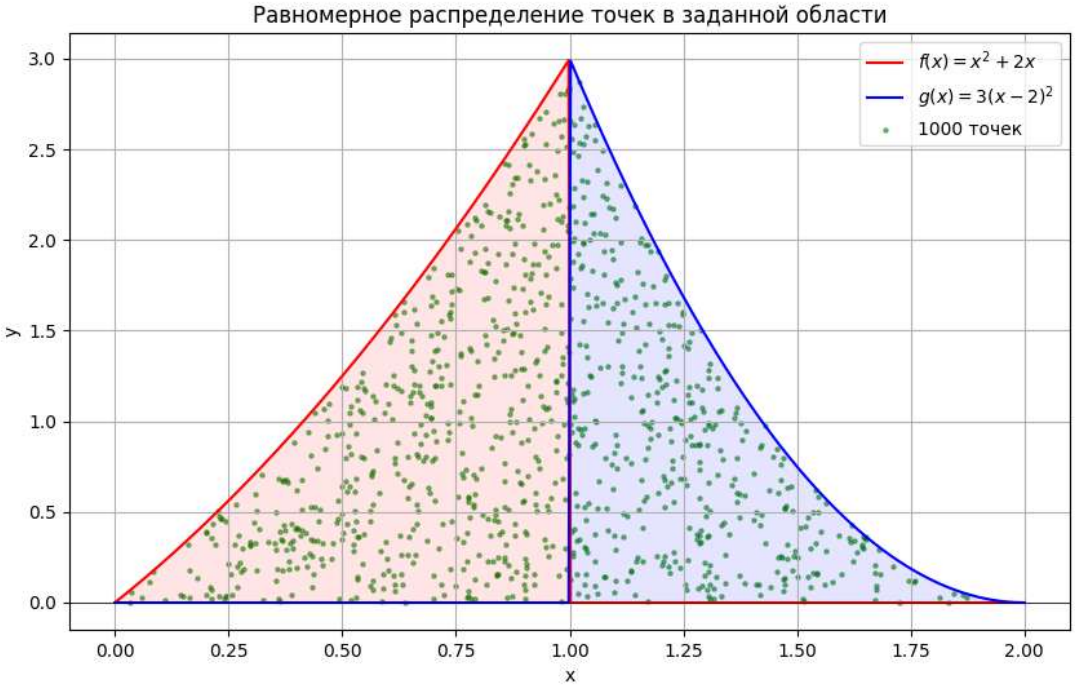
Введём дополнительную величину a:
$f_A(a)$ – равномерное распределение от 0 до a_{max},

$$a_{max} = 10.677 + 5.001 + 1 = 16.678$$

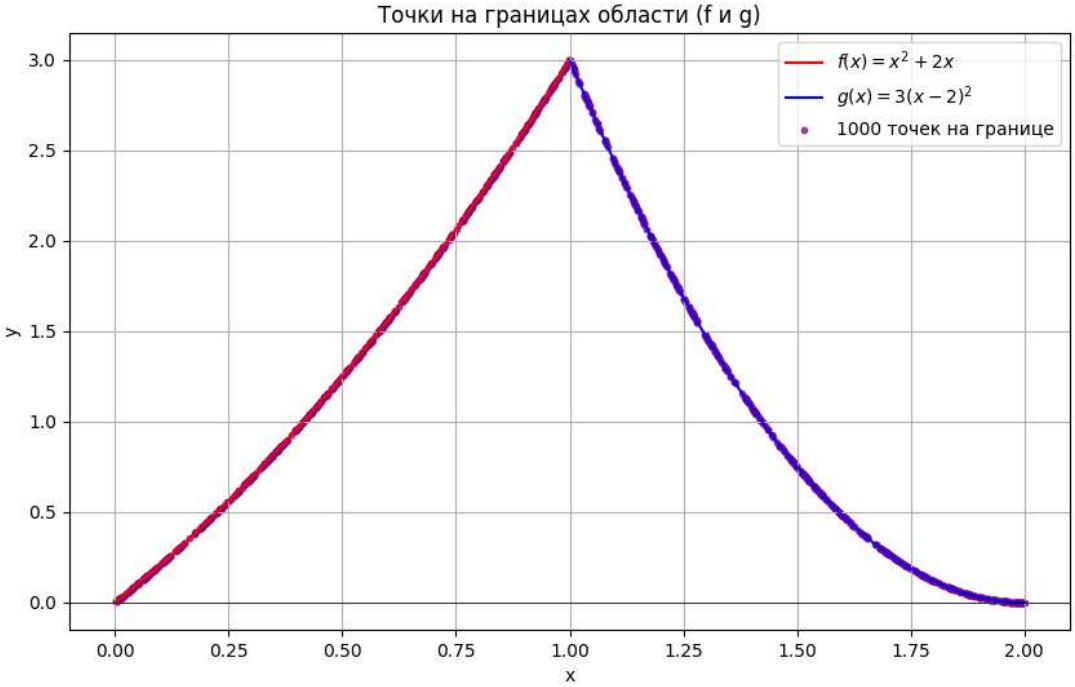
$$f(a,x) = \begin{cases} 3(x-2)^2, a\ \in [0; 10.677], x \in (0;1) \\ x^2 + 2x, a\ \in [10.677;\ 5.001], x\ \in [1;2) \\ 0, a\ \in [10.677;\ 16.678), x\ \in (0;\ 2) \end{cases}$$

Результаты:

Распределение по области:



Распределение по периметру:



Листинг

Часть 1

```
import numpy as np
import matplotlib.pyplot as plt


def f(x):
    return x**2 + 2 * x


def g(x):
    return 3 * (x - 2) ** 2


def generate_points(n):
    points = []
    while len(points) < n:
        x = np.random.uniform(0, 2)
        y = np.random.uniform(0, 3)
        if (x <= 1 and y <= f(x)) or (x > 1 and y <= g(x)):
            points.append((x, y))
    return np.array(points)


def plot_points(n):
    points = generate_points(n)
    x = np.linspace(0, 2, 500)
    y_f = [f(xi) if xi <= 1 else 0 for xi in x]
    y_g = [g(xi) if xi >= 1 else 0 for xi in x]
```

```
    plt.figure(figsize=(10, 6))
    plt.plot(x, y_f, "r-", label="$f(x) = x^2 + 2x$")
    plt.plot(x, y_g, "b-", label="$g(x) = 3(x-2)^2$")
    plt.scatter(
        points[:, 0], points[:, 1], s=5, c="green", alpha=0.5,
label=f"{n} точек"
    )
    plt.fill_between(x, y_f, color="red", alpha=0.1)
    plt.fill_between(x, y_g, color="blue", alpha=0.1)
    plt.axhline(0, color="black", linewidth=0.5)
    plt.xlabel("x")
    plt.ylabel("y")
    plt.legend()
    plt.title("Равномерное распределение точек в заданной области")
    plt.grid(True)
    plt.show()



plot_points(n=1000)
```

Часть 2

```
import numpy as np
import matplotlib.pyplot as plt



def f(x):
    return x**2 + 2 * x
```

```
def g(x):
    return 3 * (x - 2) ** 2


def generate_points_on_boundary(n):
    points = []
    for _ in range(n):
        if np.random.rand() < 0.5:
            # Точка на f(x): x ∈ [0, 1]
            x = np.random.uniform(0, 1)
            y = f(x)
        else:
            # Точка на g(x): x ∈ [1, 2]
            x = np.random.uniform(1, 2)
            y = g(x)
        points.append((x, y))
    return np.array(points)


def plot_boundary_points(n):
    points = generate_points_on_boundary(n)
    x = np.linspace(0, 2, 500)
    y_f = [f(xi) if xi <= 1 else np.nan for xi in x]
    y_g = [g(xi) if xi >= 1 else np.nan for xi in x]

    plt.figure(figsize=(10, 6))
    plt.plot(x, y_f, "r-", label="$f(x) = x^2 + 2x$")
    plt.plot(x, y_g, "b-", label="$g(x) = 3(x - 2)^2$")
    plt.scatter(
```

```
            points[:, 0],
            points[:, 1],
            s=10,
            c="purple",
            alpha=0.7,
            label=f"{n} точек на границе",
        )
        plt.axhline(0, color="black", linewidth=0.5)
        plt.xlabel("x")
        plt.ylabel("y")
        plt.legend()
        plt.title("Точки на границах области (f и g)")
        plt.grid(True)
        plt.show()


plot_boundary_points(n=1000)
```