МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ (ШКОЛА)

Департамент математического и компьютерного моделирования

ОТЧЁТ

к лабораторной работе №4 по дисциплине
«Дифференциальные уравнения»

Направление подготовки
01.03.02 «Прикладная математика и информатика»

Выполнил студент гр.
Б9121-01.03.02сп(1)
Держапольский Ю.В. ________
(Ф.И.О.) *(подпись)*

« 24 » мая 2023 г.

г. Владивосток

2023

Содержание

1. Введение

В этой лабораторной работе мы будем

2. Задание 1: Решить линейные уравнения

2.1. Постановка задачи

Для следующих линейных дифференциальных уравнений дать характеристику и найти общее решение:

1. $\left(x^2+1\right)u'' = 2u$;
2. $u'' - \dfrac{3}{x}u' + \dfrac{6}{x^2}u = 0$;
3. $y^{IV} + 6\,\dddot{y} + 18\ddot{y} + 30\dot{y} + 25y = \left(-8t^2 - 5t + 3\right)e^{-t}\sin t$;
4. $t^2 \cdot (2\ln t - 1) \cdot \ddot{y} + 4y = t \cdot (2\ln t + 1) \cdot \dot{y}$;
5. $(x-1)^3 y''' + 9(x-1)^2 y'' + 23(x-1)y' - 64y = \big(2 - 3\ln^2(x-1) + 8\ln(x-1)\big) \cdot (x-1)\sin^2 2\ln(x-1)$.

2.2. Решение

1. 1;

   Характеристика уравнения:

   Общее решение:

3. Задание 2: Решить задачи Коши

3.1. Постановка задачи

Для заданных уравнений указать тип в простой форме. Найти общее решение. Найти частное решение, удовлетворяющее заданным условиям. Построить график решения:

1. $x^2y'' - 3xy' = \dfrac{6y^2}{x^2} - 4y; y(1) = 1, y'(1) = 4;$
2. $y'y'' - \sqrt{1 + y'^2} = 0; y(0) = y'(0) = 0.$

3.2. Решение

1. $\begin{cases} 1, \\ 2; \end{cases}$

   Тип уравнения:

   Общее решение:

   Частное решение:

Рис. 1: График решения уравнения (1)

4. Задание 3: Решить линейные уравнения

4.1. Постановка задачи

На основе представленной ниже системы дифференциальных уравнений построить генератор псевдо-случайных чисел в диапазоне $[0, 1]$ с помощью метода Эйлера:

$$\dot{x} = \sigma \cdot (y - x); \quad \dot{y} = x \cdot (r - z) - y; \quad \dot{z} = x \cdot y - b \cdot z.$$

В качестве параметров зерна выбрать следующие: $x_0,\ y_0,\ z_0,\ dt$ и n. Здесь:

$$x_0 \in (2.8915, 3.2027),\ y_0 \in (1.4296, 1.7365),\ z_0 \in (15.2113, 16.1852),$$
$$dt \in (0, 0.1],\ n > 100,\ \sigma = 10,\ r = 28,\ b = 2.66;$$

В качестве источника энтропии использовать *Unix*-время, в качестве итератора использовать параметр n. Генератор чисел основывается на методе Эйлера, в качестве результата выдавать десятичную часть числа x_k из метода. На каждой итерации метода, x_k, y_k и z_k округлять до **тысяч**, чтобы избежать переполнения. Протестировать и представить графики соотношений значений с точностью до десятых и сотых относительно их количества. Приложить код, и примеры работы.

4.2. Решение

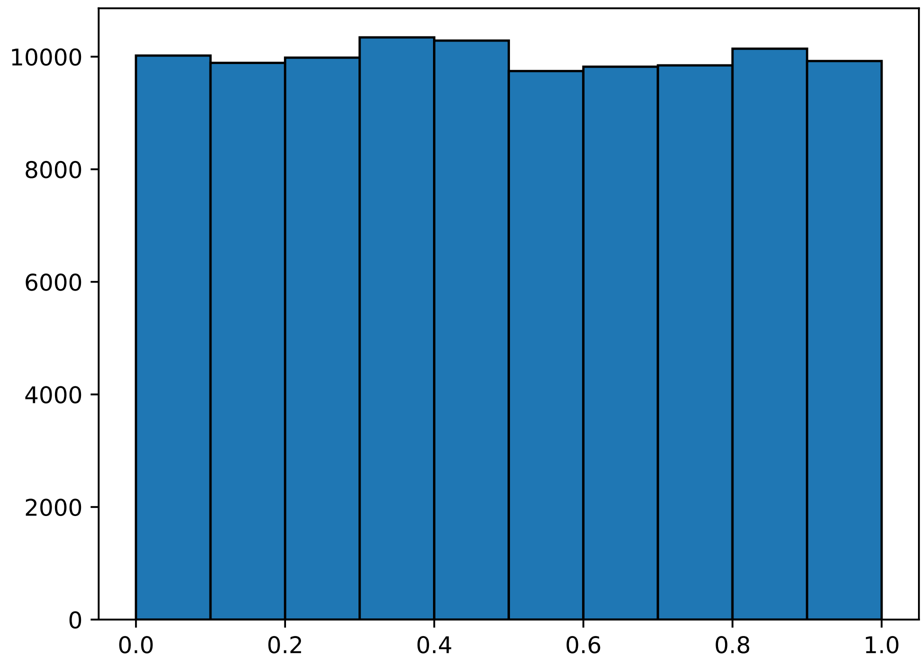

Рис. 2: Соотношение значений с точностью до десятых при выборке в 100000.

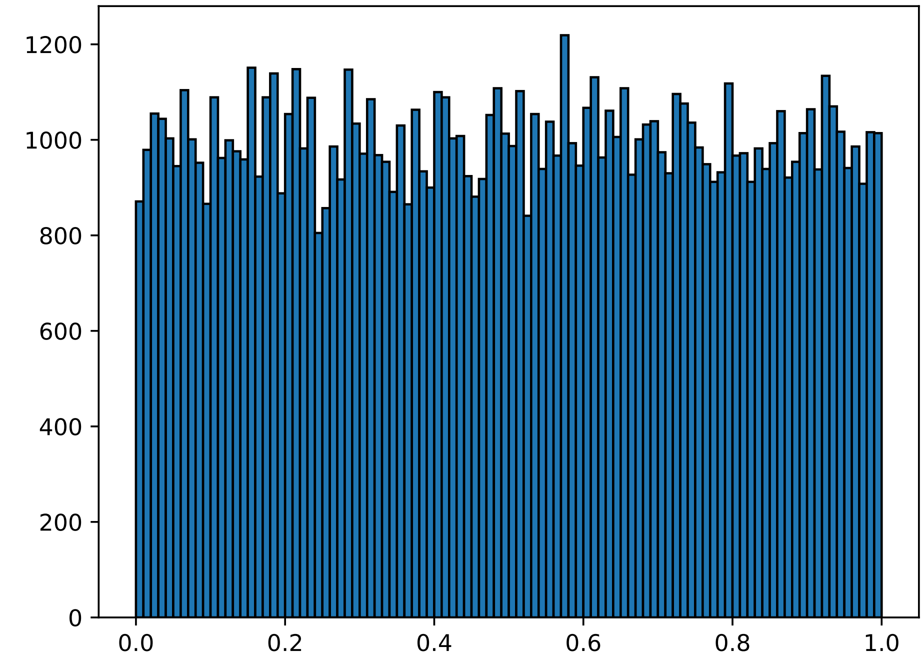

Рис. 3: Соотношение значений с точностью до сотых при выборке в 100000.

```python
import time
from math import fmod


class PseudoRandom:
    def __init__(self):
        self.sigma = 10
        self.r = 28
        self.b = 2.66
        self.n_min, self.n_max = 100, 1000
        self.x_range = (2.8915, 3.2027)
        self.y_range = (1.4296, 1.7365)
        self.z_range = (15.2113, 16.1852)
        self.dt_range = (1e-20, 0.1)
        self.entropy = 1

    def fx(self, x, y, z):
        return self.sigma * (y - x)

    def fy(self, x, y, z):
        return x * (self.r - z) - y

    def fz(self, x, y, z):
        return x * y - self.b * z

    @staticmethod
    def get_rand(seed, values: tuple, div):
        return values[0] + fmod(seed, div) * (values[1] - values[0]) / div

    def generate(self):
        self.entropy = time.time() * 1000

        x_i = self.get_rand(self.entropy, self.x_range, 463)
        y_i = self.get_rand(self.entropy, self.y_range, 539)
        z_i = self.get_rand(self.entropy, self.z_range, 822)
        dt = self.get_rand(self.entropy, self.dt_range, 1000)
        n = int(self.n_min + self.get_rand(self.entropy, (1, self.n_max - self.
            n_min), self.n_max - self.n_min))

        cut = 10000
```

```python
        for _ in range(n):
            x_ = x_i + dt * self.fx(x_i, y_i, z_i)
            y_ = y_i + dt * self.fy(x_i, y_i, z_i)
            z_ = z_i + dt * self.fz(x_i, y_i, z_i)
            x_i = fmod(x_, cut)
            y_i = fmod(y_, cut)
            z_i = fmod(z_, cut)

        return abs(fmod(x_i, 1))
```

Листинг 1: Код генератора псевдослучайных чисел

5. Заключение

В этой лабораторной работе мы решили