Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

Кафедра ПМиК

Лабораторная работа 7

по дисциплине «Современные проблемы информатики»

Выполнил: ст. гр. ЗМП-41 Лёвкин И. А.

Проверил: Лихачёв А.В.

**Новосибирск 2025**

**Постановка задачи**

Разработать компьютерную программу, реализующую алгоритм *моделирования истинных траекторий*. Предположим, что нейтрон испытал *k*-e рассеяние внутри пластинки в точке с абсциссой *xk* и после этого начал двигаться под углом к оси *X*, косинус которого равен . По формуле (7.1) Разыграем длину свободного пробега *λk*, и вычислим абсциссу следующего столкновения

. (7.4)

Проверим, пройдёт ли при этом нейтрон сквозь пластинку. Это означает, что имеет место . Если это условие выполнено, то расчёт траектории нейтрона заканчивается, и добавляется единица к счетчику прошедших частиц. В противном случае проверяем условие отражения: . Если оно выполнено, то расчёт траектории также заканчивается, а единица добавляется к счетчику отраженных частиц. Если же нейтрон остался внутри пластинки, т.е. оказалось, что , то это означает что, он испытал (*k*+1)-е столкновение, и надо продолжить моделирование траектории.

Сгенерируем очередное значение случайной величины *γ* и проверим условие поглощения: . Если это неравенство выполнено, то счёт траектории заканчивается и добавляется единица к счётчику поглощённых частиц. В противном случае мы считаем, что нейтрон испытал рассеяние в точке с абсциссой . Тогда разыгрывается новое направление скорости нейтрона, и затем повторяется весь цикл снова. После того как будут сосчитаны *N* траекторий, окажется, что *N*+ нейтронов прошли сквозь пластинку, *N*- нейтронов отразились от нее, а *N*0 нейтронов были поглощены. Тогда оценки искомых вероятностей будут отношения чисел *N*+, *N*-, *N*0 к *N.*

**Результаты:**

|  |
| --- |
| Оценки вероятностей:  Пройти сквозь пластину (P+): 0.0857 (857 частиц)  Отразиться (P-): 0.0800 (800 частиц)  Быть поглощённым (P0): 0.8343 (8343 частиц) |

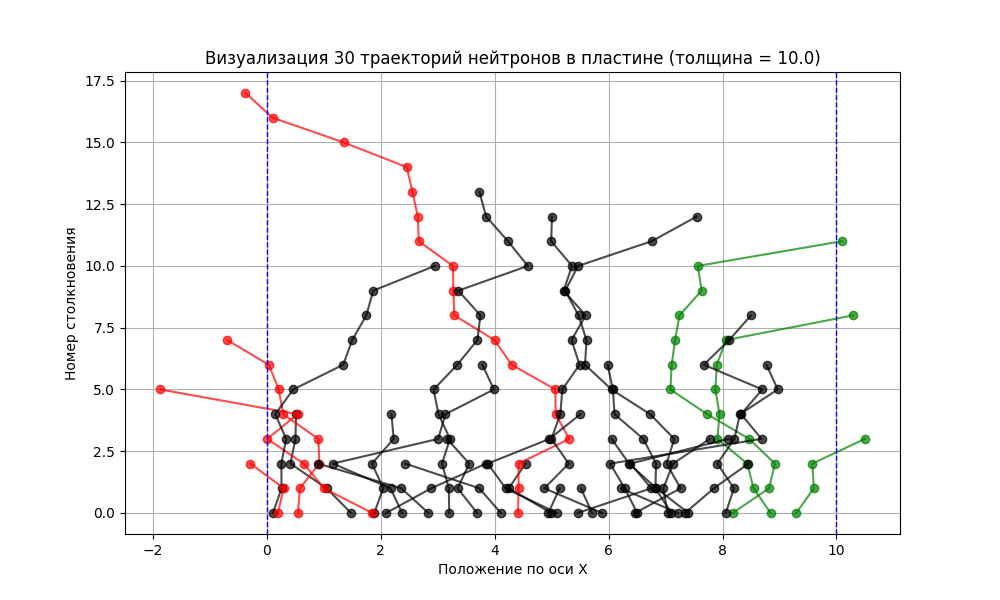
****

Рис. 1. Визуализация траекторий

* Прошли насквозь
* Отразились
* Поглощены

**Листинг**

import random

import math

from matplotlib import pyplot as plt

def simulate\_trajectory(d, mu, p\_absorb, trace=False):

"""

Симулирует одну траекторию нейтрона в пластине толщиной d.

:param d: толщина пластинки

:param mu: средняя длина свободного пробега

:param p\_absorb: вероятность поглощения при столкновении

:return: "pass", "reflect", or "absorb"

"""

x = random.uniform(0, d) # начальная абсцисса внутри пластины

cos\_theta = random.uniform(-1, 1) # направление движения

if trace:

xs = [x]

while True:

# 1. Случайная длина свободного пробега по экспоненциальному закону

lmbd = -mu \* math.log(random.random())

# 2. Вычисляем следующую абсциссу

x\_new = x + lmbd \* cos\_theta

if trace:

xs.append(x\_new)

# 3. Проверка выхода из пластинки

if x\_new >= d:

return "pass", xs if trace else "pass"

elif x\_new <= 0:

return "reflect", xs if trace else "reflect"

# 4. Проверка поглощения

gamma = random.random()

if gamma < p\_absorb:

return "absorb", xs if trace else "absorb"

# 5. Нейтрон остался внутри и рассеивается

x = x\_new

cos\_theta = random.uniform(-1, 1) # новое направление

def simulate\_many(N, d, mu, p\_absorb):

"""

Запускает моделирование N нейтронов и считает количество прошедших, отражённых и поглощённых частиц.

:return: словарь с оценками вероятностей и абсолютными числами

"""

passed = 0

reflected = 0

absorbed = 0

for \_ in range(N):

result = simulate\_trajectory(d, mu, p\_absorb)[0]

if result == "pass":

passed += 1

elif result == "reflect":

reflected += 1

elif result == "absorb":

absorbed += 1

return {

"P\_pass": passed / N,

"P\_reflect": reflected / N,

"P\_absorb": absorbed / N,

"N\_pass": passed,

"N\_reflect": reflected,

"N\_absorb": absorbed,

}

def visualize\_trajectories(M, d, mu, p\_absorb):

plt.figure(figsize=(10, 6))

for \_ in range(M):

result, xs = simulate\_trajectory(d, mu, p\_absorb, trace=True)

ys = list(range(len(xs))) # номер столкновения по оси Y

color = {"pass": "green", "reflect": "red", "absorb": "black"}[result]

plt.plot(xs, ys, marker="o", color=color, alpha=0.7)

plt.axvline(0, color="blue", linestyle="--", linewidth=1)

plt.axvline(d, color="blue", linestyle="--", linewidth=1)

plt.xlabel("Положение по оси X")

plt.ylabel("Номер столкновения")

plt.title(f"Визуализация {M} траекторий нейтронов в пластине (толщина = {d})")

plt.grid(True)

plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

d = 10.0

mu = 1.0

p\_absorb = 0.2

N = 10000

result = simulate\_many(N, d, mu, p\_absorb)

print("Оценки вероятностей:")

print(

f" Пройти сквозь пластину (P+): {result['P\_pass']:.4f} ({result['N\_pass']} частиц)"

)

print(

f" Отразиться (P-): {result['P\_reflect']:.4f} ({result['N\_reflect']} частиц)"

)

print(

f" Быть поглощённым (P0): {result['P\_absorb']:.4f} ({result['N\_absorb']} частиц)"

)

# Визуализация 30 траекторий

visualize\_trajectories(M=30, d=d, mu=mu, p\_absorb=p\_absorb)