Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

*Факультет программной инженерии и компьютерной техники*

Теория систем

Лабораторная работа №3

Группа: P3324

Выполнил:

Круглов Егор Ильич

Преподаватель:

Русак Алёна Викторовна

Санкт-Петербург

2025г.

# Задание

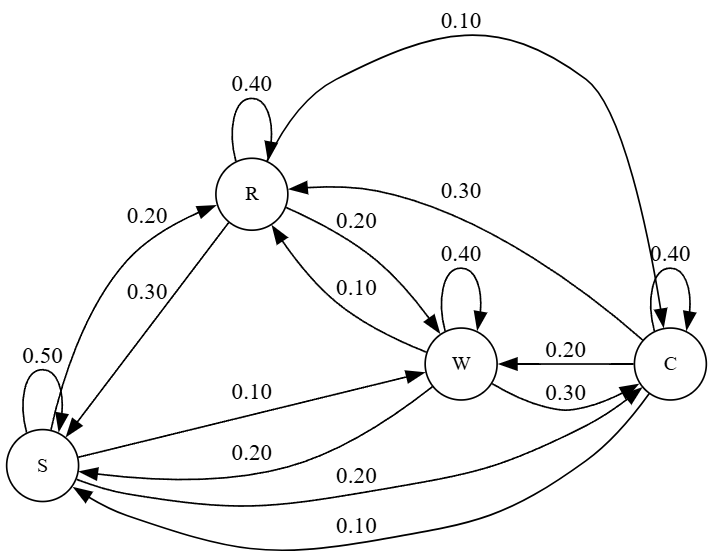
* Придумать эргодическую марковскую цепь, содержащую не менее 4-х состояний.
* Нарисовать диаграмму переходов и записать матрицу переходных вероятностей данной цепи.
* Промоделировать марковскую цепь пошагово с несколькими различными начальными  
  векторами вероятностей состояний и получить конечные вектора, к которым привело моделирование.
* Построить графики изменения компонентов финальных векторов, а также графики изменения среднеквадратического отклонения на каждом шаге моделирования для всех начальных векторов.
* Найти стационарное распределение.
* Сравнить вектора из пункта 3 и вектор, рассчитанный аналитически, между собой.

# Решение

Была выбрана погодная модель. Она состоит из 4 состояний:

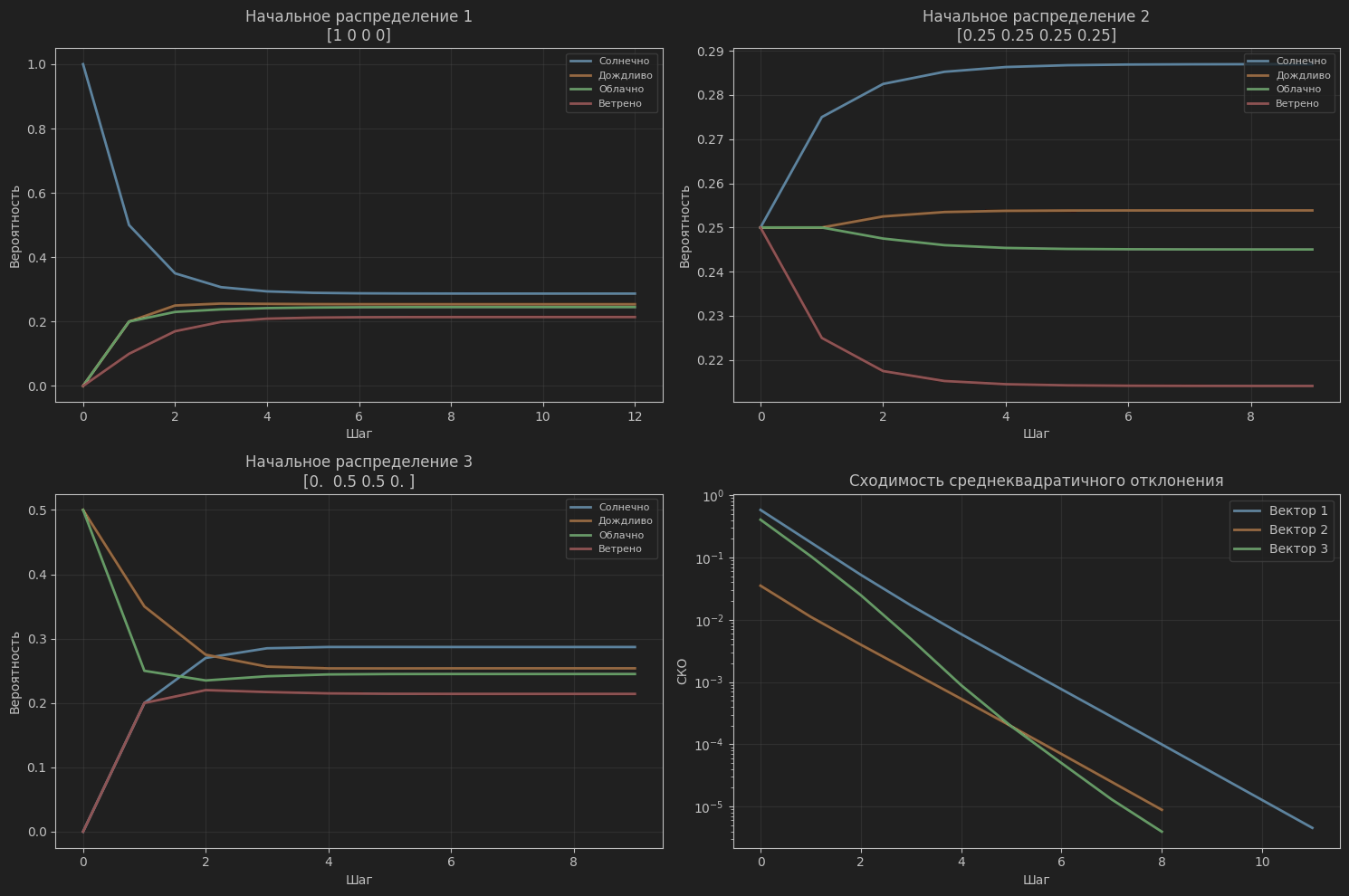
* Sunny
* Rainy
* Cloudy
* Windy

Диаграмма переходов:



Код был написан на python, в формате ipynb.

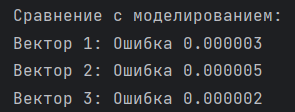
Графики:



Также был расчитано стационарное распределение:



Отклонение было небольшим:



Код представлен на GitHub - <https://github.com/KruglovEgor/System-Theory/tree/main/lab3>

import numpy as np  
import matplotlib.pyplot as plt

P = np.array([  
 [0.5, 0.2, 0.2, 0.1], # Sunny  
 [0.3, 0.4, 0.1, 0.2], # Rainy  
 [0.1, 0.3, 0.4, 0.2], # Cloudy  
 [0.2, 0.1, 0.3, 0.4] # Windy  
])

def simulate\_markov(P, initial\_vec, eps=1e-5, max\_steps=1000):  
 old = initial\_vec.copy()  
 history = [old]  
 for \_ in range(max\_steps):  
 new = old @ P  
 history.append(new)  
 if np.sqrt(np.sum((new - old)\*\*2)) < eps:  
 break  
 old = new  
 return np.array(history)  
  
# Начальные векторы  
initial\_vectors = [  
 np.array([1, 0, 0, 0]),  
 np.array([0.25, 0.25, 0.25, 0.25]),  
 np.array([0, 0.5, 0.5, 0])  
]  
  
# Моделирование  
results = []  
for vec in initial\_vectors:  
 results.append(simulate\_markov(P, vec))

states = ['Солнечно', 'Дождливо', 'Облачно', 'Ветрено']  
plt.figure(figsize=(15, 10))  
  
# Графики вероятностей для каждого начального вектора  
for i, res in enumerate(results):  
 plt.subplot(2, 2, i+1)   
 for j in range(4):  
 plt.plot(res[:, j], label=states[j], linewidth=2)  
 plt.title(f'Начальное распределение {i+1}\n{initial\_vectors[i]}')  
 plt.xlabel('Шаг')  
 plt.ylabel('Вероятность')  
 plt.grid(alpha=0.3)  
 plt.legend(loc='upper right', fontsize=8)  
  
# График среднеквадратичного отклонения  
plt.subplot(2, 2, 4)   
for i, res in enumerate(results):  
 diffs = [np.sqrt(np.sum((res[k] - res[k-1])\*\*2)) for k in range(1, len(res))]  
 plt.plot(diffs, label=f'Вектор {i+1}', linewidth=2)  
plt.title('Сходимость среднеквадратичного отклонения')  
plt.xlabel('Шаг')  
plt.ylabel('СКО')  
plt.yscale('log')  
plt.grid(alpha=0.3)  
plt.legend()  
  
plt.tight\_layout()  
plt.show()

A = P.T - np.eye(4)  
A[-1] = np.ones(4) # Замена последнего уравнения на нормировку  
b = np.zeros(4)  
b[-1] = 1  
  
stationary = np.linalg.solve(A, b)  
print("Стационарное распределение:", stationary)

print("\nСравнение с моделированием:")  
for i, res in enumerate(results):  
 final = res[-1]  
 diff = np.sqrt(np.sum((final - stationary)\*\*2))  
 print(f"Вектор {i+1}: Ошибка {diff:.6f}")

# Вывод

В ходе выполнения лабораторной работы была написана симуляция для марковской цепи. Также с помощью языка программировния python была сделана симуляция 1000 переходов и построены графики. Дополнительно было расчитано стационарное (аналитическое) распределение с помощью линейной алгебры (через библиотеку), которое окозалось очень близким с распределением, полученным путем симуляции.