



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____ «Информатика и системы управления»

КАФЕДРА _____ «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ по курсу «Моделирование»

«Марковские процессы»

Студент:	<u>ИУ7-73Б</u> (группа)	_____ (подпись, дата)	<u>М. Д. Маслова</u> (И. О. Фамилия)
Руководитель:		_____ (подпись, дата)	<u>И. В. Рудаков</u> (И. О. Фамилия)

2022 г.

СОДЕРЖАНИЕ

1	Задание	4
2	Теоретическая часть	5
2.1	Марковские процессы	5
2.2	Предельные вероятности состояний	5
2.3	Точки стабилизации	6
3	Практическая часть	7
3.1	Текст программы	7
3.2	Полученный результат	9

1 Задание

Разработать программное обеспечение, предоставляющее возможность определения вероятности и времени пребывания системы массового обслуживания в каждом состоянии в установившемся режиме работы.

Реализовать графический интерфейс, позволяющий задать количество состояний системы (их не более десяти) и матрицу интенсивностей переходов.

2 Теоретическая часть

2.1 Марковские процессы

Случайный процесс, протекающий в некоторой системе S , называется **марковским**, если для каждого момента времени вероятность любого состояния системы в будущем зависит только от ее состояния в настоящем времени и не зависит от того, когда и каким образом система пришла в это состояние, то есть не зависит от того, как процесс развивался в прошлом.

2.2 Предельные вероятности состояний

Для марковских процессов используются уравнения Колмогорова, составляющиеся по следующему правилу:

1. В левой части каждого уравнения стоит производная вероятности состояния.
2. Правая часть содержит столько членов, сколько стрелок связано с этим состоянием; если стрелка направлена из состояния соответствующий член имеет знак «-», если в состояние — знак «+».
3. Каждый член равен плотности вероятности перехода (интенсивности), соответствующей данной стрелке, умноженной на вероятность того состояния, из которого исходит стрелка.

То есть строится система уравнений, которые имеют вид:

$$P'_i(t) = \sum_{j=1}^n \lambda_{ji} P_j(t) - P_i(t) \sum_{j=1}^n \lambda_{ij}, \quad (2.1)$$

где $P_i(t)$ — вероятность того, что система находится в i -ом состоянии;

n — число состояний в системе;

λ_{ij} — интенсивность перехода системы из i -ого состояния в j -ое.

Одно из уравнений данной системы заменяется условием нормировки:

$$\sum_{i=1}^n P_i(t) = 1. \quad (2.2)$$

В силу того, что **предельные вероятности состояний постоянны**, для их определения в уравнениях Колмогорова необходимо **заменить их про-**

изводные нулями и решить полученную систему линейных алгебраических уравнений.

Отметим, что предельная вероятность состояния показывает *среднее относительное время пребывания* системы в этом состоянии.

2.3 Точки стабилизации

Для определения точек стабилизации системы определяются вероятности состояний с некоторым малым шагом Δt . Точка стабилизации считается найденной, если приращение вероятности, а также разница между ранее найденной предельной вероятностью состояния и вычисленной вероятностью, достаточно малы, то есть выполняются соотношения:

$$|P_i(t + \Delta t) - P_i(t)| < \varepsilon, \quad (2.3)$$

$$|P_i(t) - \lim_{t \rightarrow \infty} P_i(t)| < \varepsilon, \quad (2.4)$$

где ε — заданная точность.

3 Практическая часть

3.1 Текст программы

На листинге 3.1 представлены функции расчета предельных вероятностей по заданной матрице интенсивностей переходов.

Листинг 3.1 – Реализация функций расчета предельных вероятностей

```
1 import numpy as np
2 import scipy as sp
3
4 def GetKolmogorovEquations(matrix):
5     statesNumber = matrix.shape[0]
6
7     rightSide = np.zeros([statesNumber] * 2)
8
9     for state in range(statesNumber):
10         rightSide[state][state] = -sum(matrix[state, :])
11         rightSide[state] += matrix[:, state]
12
13     return rightSide
14
15
16 def GetNormalizationEquation(statesNumber):
17     return 1, np.zeros(statesNumber) + 1
18
19
20 def CalculateMarginalProbabilities(matrix):
21     statesNumber = matrix.shape[0]
22
23     leftSide = np.zeros(statesNumber)
24     rightSide = GetKolmogorovEquations(matrix)
25
26     leftSide[-1], rightSide[-1] = GetNormalizationEquation(
27         statesNumber)
28
29     return sp.linalg.solve(rightSide, leftSide)
```

На листинге 3.2 представлены функции расчета точек стабилизации.

Листинг 3.2 – Реализация функций расчета точек стабилизации

```
1 def GetProbabilitiesDerivatives(probabilities, _, matrix):
2     derivatives = np.zeros(len(probabilities))
3
4     for i, probability in enumerate(probabilities):
5         derivatives[i] = np.dot(probabilities, matrix[i, :])
6
7     return derivatives
8
9
10 def CalculateStabilizationTime(matrix, marginalProbabilities):
11     timeList = np.arange(0, TIME_MAX, TIME_STEP)
12
13     statesNumber = matrix.shape[0]
14     probabilities0 = np.array([0] * statesNumber)
15     probabilities0[0] = 1
16
17     kolmogorovMatrix = GetKolmogorovEquations(matrix)
18     probabilities = sp.integrate.odeint(
19         GetProbabilitiesDerivatives,
20         probabilities0,
21         timeList,
22         (kolmogorovMatrix,))
23
24     stabilizationTimes = np.zeros(statesNumber)
25
26     for state in range(statesNumber):
27         curStateProbabilities = probabilities[:, state]
28
29         found = False
30
31         for i in range(1, len(curStateProbabilities)):
32             previousProbability = curStateProbabilities[i - 1]
33             curProbability = curStateProbabilities[i]
34
35             if (abs(curProbability - previousProbability) < EPS
36                 and abs(curProbability
37                     - marginalProbabilities[state]) < EPS):
38                 found = True
39                 break
40
41         stabilizationTimes[state] = (timeList[i] if found
42                                     else None)
```

3.2 Полученный результат

На рисунках 3.1 представлен пример работы программы для системы с тремя состояниями. На рисунке 3.2 представлен соответствующий график зависимости вероятности от времени.

Маслова Марина ИУ7-73Б Лабораторная работа №2

Матрица интенсивностей переходов Задать

Количество состояний системы - + ☒ Генерация

	1		2		3
1	1.01	-	2.95	-	3.39
2	3.07	-	3.71	-	3.80
3	3.55	-	0.02	-	0.83

Результат Рассчитать Показать график

	1		2		3
P	0.35	-	0.15	-	0.50
t	0.78	-	1.19	-	1.18

Автор: Маслова Марина ИУ7-73Б

Рисунок 3.1 – Расчет для системы с тремя состояниями

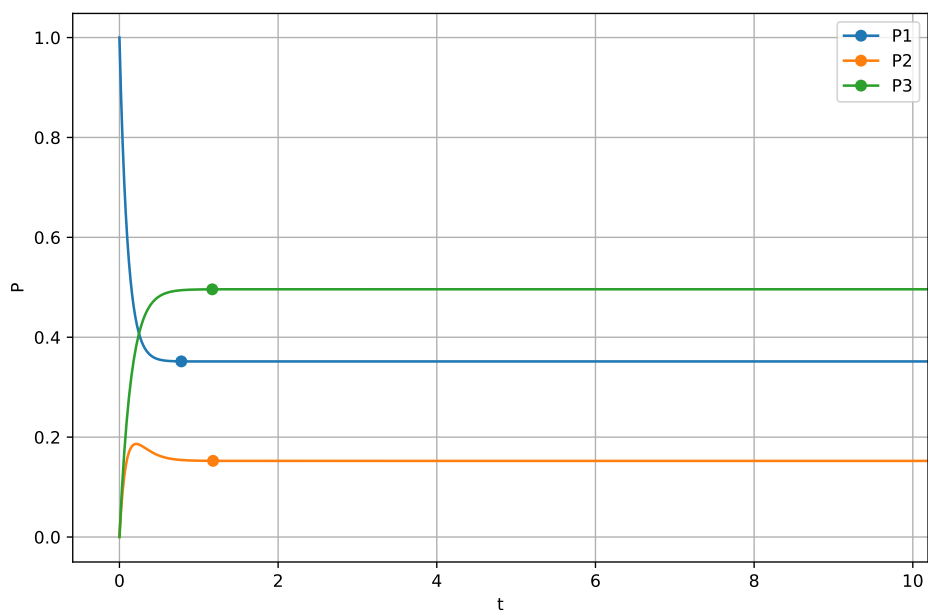


Рисунок 3.2 – График зависимости вероятности состояния от времени для системы с тремя состояниями

На рисунках 3.3 представлен пример работы программы для системы с пятью состояниями. На рисунке 3.4 представлен соответствующий график зависимости вероятности от времени.

Маслова Марина ИУ7-73Б Лабораторная работа №2

Матрица интенсивностей переходов Задать

Количество состояний системы - + ☒ Генерация

	1	2	3	4	5
1	2.45 - +	3.05 - +	0.31 - +	3.32 - +	1.73 - +
2	1.82 - +	2.32 - +	4.19 - +	1.25 - +	2.11 - +
3	2.91 - +	1.09 - +	2.05 - +	4.05 - +	2.99 - +
4	3.64 - +	0.42 - +	4.54 - +	3.94 - +	3.83 - +
5	2.07 - +	4.44 - +	1.27 - +	0.38 - +	4.69 - +

Результат Рассчитать Показать график

	1	2	3	4	5
P	0.23 - +	0.21 - +	0.17 - +	0.15 - +	0.24 - +
t	0.86 - +	0.75 - +	0.77 - +	0.77 - +	0.82 - +

Автор: Маслова Марина ИУ7-73Б

Рисунок 3.3 – Расчет для системы с пятью состояниями

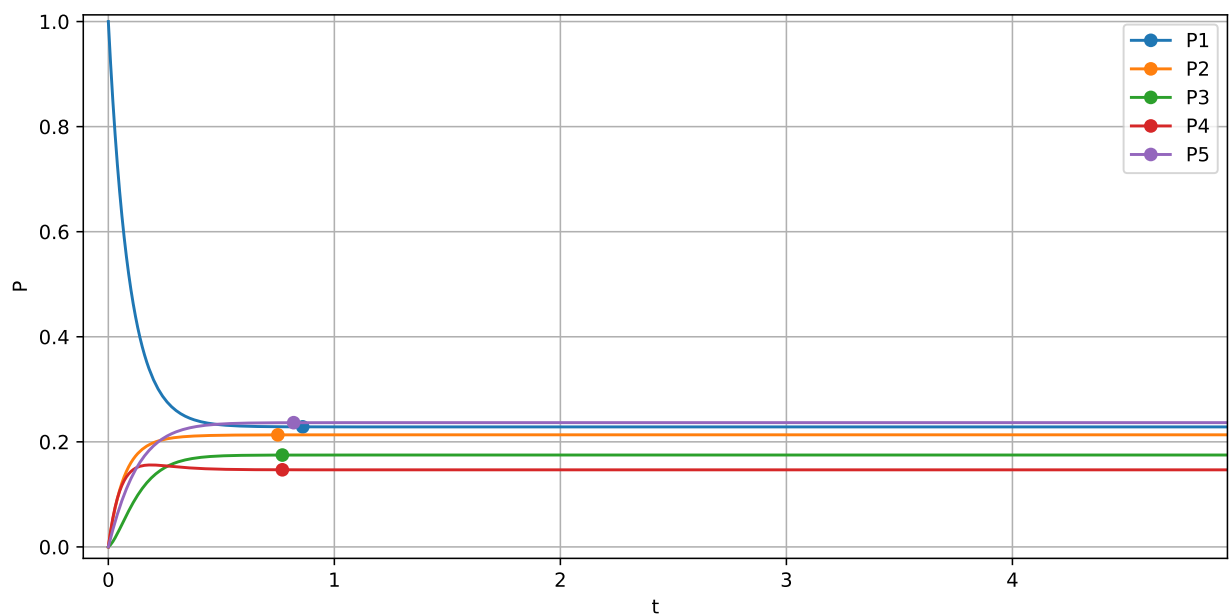


Рисунок 3.4 – График зависимости вероятности состояния от времени для системы с пятью состояниями