МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

ИНСТИТУТ КИБЕРНЕТИКИ

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 2

по курсу «**Случайные процессы**»

Тема: **Марковский процесс с непрерывным временем и пятью состояниями**

Выполнил:
Студент 4-го курса
Гогинян Б.А.
Группа: КМБО-03-16

МОСКВА 2019

Лабораторная работа по случайным процессам № 2

«Марковский процесс с непрерывным временем и пятью состояниями»

Задание

Дана матрица интенсивностей однородного марковского процесса с непрерывным временем

$$\Lambda = \begin{pmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} & \lambda_{14} & \lambda_{15} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} & \lambda_{24} & \lambda_{25} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} & \lambda_{34} & \lambda_{35} \\ \lambda_{41} & \lambda_{42} & \lambda_{43} & \lambda_{44} & \lambda_{45} \\ \lambda_{51} & \lambda_{52} & \lambda_{53} & \lambda_{53} & \lambda_{55} \end{pmatrix}$$

Следуя **Указаниям** нужно:

1. Построить граф состояний марковского процесса.

2. Написать систему дифференциальных уравнений Колмогорова для вероятностей состояний (с заданными интенсивностями).

3. Написать систему уравнений для нахождения стационарных вероятностей.

4. Найти стационарное распределение вероятностей состояний $(r_1, r_2, r_3, r_4, r_5)$.

5. Считая, что в начальный момент времени $t = 0$ система находится в состоянии 1, провести моделирование развития системы до события с номером K, при котором впервые будет выполнено неравенство

$$\delta_K = \max(|\nu_i(K) - \nu_i(K-1)|; i = 1,...,5) \le 0{,}0001,$$

где $\nu_i(K) = \dfrac{R_i(K)}{K}$ – относительная частота попадания системы в состояние i с 1-го по K событие (считаем, что $\nu_i(0) = 0$ при всех i), $R_i(K)$ – число попаданий системы в состояние i в событиях с 1-го по K. Событием считается переход системы из одного состояния в другое.

6. Составить таблицу 1 с данными о событиях:

– номер события l;

– момент $t_{соб}(l)$ наступления события l;

– состояние системы после события $C(l)$;

– время $\tau(l)$ пребывания системы в состоянии $C(l)$ с момента $t_{соб}(l)$ до перехода системы в другое состояние;

– значение отклонения δ_l.

В таблицу поместить данные о событиях с 1-го по 100, а также о событиях $K-5, K-4, K-3, K-2, K-1, K$, если $K>100$.

7. Составить таблицу 2 с данными о состояниях следующего вида:

– номер состояния i;

– число $R_i(100)$ попаданий системы в состояние i в событиях с 1-го по 100;

– относительная частота $\nu_i(100)$ попадания системы в состояние i в событиях с 1-го по 100;

– общее время $T_i(100)$ пребывания системы в состоянии i с момента $t_{соб}(1)$ до $t_{соб}(101)$, т.е. до времени наступления события 101.

– доля $\Delta_i(100)$ времени пребывания системы в состоянии i в период с $t_{соб}(1)$ до $t_{соб}(101)$.

8. Составить таблицу 3 с данными о состояниях следующего вида:

– номер состояния i;

– число $R_i(K)$ попаданий системы в состояние i в событиях с 1-го по K;

– относительная частота $\nu_i(K)$ попадания системы в состояние i в событиях с 1-го по K;

– общее время $T_i(K)$ пребывания системы в состоянии i с момента $t_{соб}(1)$ до $t_{соб}(K+1)$, т.е. до времени наступления события $K+1$.

– доля $\Delta_i(K)$ времени пребывания системы в состоянии i в период с $t_{соб}(1)$ до $t_{соб}(K+1)$.

Вычисления и вывод результатов проводить с точностью до 0,00001 .

Указания

В разделе отчета **Краткие теоретические сведения** следует дать определения марковского процесса с непрерывным временем и дискретным множеством состояний, однородного марковского процесса с непрерывным временем и дискретным множеством состояний, матрицы интенсивностей перехода, графа состояний, предельного распределения, стационарного распределения; привести дифференциальные уравнения Колмогорова для вероятностей состояний, формулы для нахождения стационарного распределения. В этом разделе должны быть описаны средства языка программирования, которые использованы в программе расчета.

В разделе отчета **Результаты расчетов** приводятся исходные данные и даются ответы на пункты задания, в том числе полностью заполненные таблицы 1, 2 и 3.

Граф состояний марковского процесса строится по данной матрице интенсивностей перехода. Вершины графа – прямоугольники – располагаются по окружности, если $\lambda_{ij} > 0 \ (i \neq j)$, то из вершины i в вершину j проводится дуга и над ней ставится значение λ_{ij}. При этом $\lambda_i = \sum_{i \neq j} \lambda_{ij} = -\lambda_{ii}$ – интенсивность выхода из состояния i.

Таблицы 1, 2 и 3 заполняются по результатам моделирования N событий, где $N = \max(K, 100)$. В начальный момент времени $t = 0$ система находится в состоянии 1 и определяется случайным образом в соответствии с показательным законом распределения с параметром λ_1 время, через которое система перейдет в следующее состояние $j = C(1)$. Это время равно моменту $t_{соб}(1)$ наступления первого события.

Если до наступления события l система находилась в состоянии i , то следующее состояние $C(l)$ находится в соответствии с правилом:

1) если $\lambda_i = \sum_{i \neq j} \lambda_{ij} = 0$, то $j = i$;

2) если $\lambda_i = \sum_{i \neq j} \lambda_{ij} > 0$, то определяется число k таких состояний j, что $\lambda_{ij} > 0$, и отрезок $[0,1]$ делится на k отрезков, длины которых равны $\frac{\lambda_{ij}}{\lambda_i}$ $(\lambda_{ij} > 0)$; каждый отрезок соответствует некоторому состоянию j, у которого $\lambda_{ij} > 0$; если случайным образом выбранное из отрезка $[0,1]$ число попало в отрезок, соответствующий состоянию j, то $C(l) = j$.

По этому правилу определяется и следующее состояние $j = C(1)$ в первом событии, при этом $i = 1$.

Если до наступления события l система находилась в состоянии i , то время $\tau(l)$ пребывания системы в состоянии $C(l) = j$ с момента $t_{соб}(l)$ до перехода системы в другое состояние определяется случайным образом в соответствии с показательным законом распределения с параметром λ_j. При этом $t_{соб}(l+1) = t_{соб}(l) + \tau(l)$.

Для каждого события l число попаданий $R_i(l)$ системы в состояние i в событиях с 1-го по l равно числу вхождений числа i в множество $\{C(1),...,C(l)\}$, $\nu_i(l) = \frac{R_i(l)}{l}$, $\delta_l = \max(|\nu_i(l) - -\nu_i(K-1)|; i = 1,...,5)$.

Общее время $T_i(K)$ пребывания системы в состоянии i с момента $t_{соб}(1)$ до $t_{соб}(K+1)$ находится следующим образом: $T_i(K) = \sum_{l=1}^{K} \tau(l) \cdot \delta_{iC(l)} = \sum_{\substack{i=l \\ 1 \le l \le K}} \tau(l)$.

Доля $\Delta_i(K)$ времени пребывания системы в состоянии i в период с $t_{соб}(1)$ до $t_{соб}(K+1)$ равна $\Delta_i(K) = \frac{T_i(K)}{t_{соб}(K+1) - t_{соб}(1)}$.

В разделе отчета **Анализ результатов и выводы** следует привести:

– граф состояний марковского процесса;

– значение K , удовлетворяющее условию из пункта 6 задания;

– таблицу следующего вида

i	**1**	**2**	**3**	**4**	**5**
r_i	r_1	r_2	r_3	r_4	r_5
$\nu_i(100)$					
$\nu_i(K)$					
$\Delta_i(100)$					
$\Delta_i(K)$					

Краткие теоретические сведения

Опр. Случайный процесс $X_t, t \geq 0$ называется марковским, если для любого целого неотрицательного m, любых моментов времени $0 \leq s_1 < s_2 < ... < s_m < s$, $t > 0$, любого набора состояний $E_{i_1}, E_{i_2}, ..., E_{i_m}, E_i, E_j$ выполнено равенство
$P(X_{s+t} = E_j | X_{s_1} = E_{i_1}; ...; X_{s_m} = E_{i_m}; X_s = E_i) = P(X_{s+t} = E_j | X_s = E_i)$

Опр. Процесс X_t называется однородным (по времени), если условная вероятность $P(X_{s+t} = E_j | X_s = E_i)$ перехода из состояния E_i в состояние E_j не зависит от s.

Опр. матрицы интенсивностей перехода

$p_{ij} = P(X_{s+t} = E_j | X_s = E_i)$;
$P(t) = ||p_{ij}(t)||$ - матрица вероятностей перехода за время t.
Предполагаем, что переходные вероятности $p_{ij}(t)$ дифференцируемы в нуле:
$p'_{ij}(0) = \lambda_{ij}$, при $i \neq j$ $p'_{ij}(t) = \lambda_{ij} t + o(t)$; $p'_{ii}(t) = 1 + \lambda_{ii} t + o(t)$
$P'(0) = \Lambda = ||\lambda_{ij}||$ - матрица интенсивностей (плотность вероятностей) перехода.

Опр.
Ориентированный граф состояний, вершины которого (обозначаемых прямоугольниками) служат состояния E_i системы, а стрелками обозначены возможные непосредственные переходы из состояния E_i в состояние E_j. При этом каждой стрелке прописана соответствующая плотность вероятности перехода λ_{ij}, $(\lambda_{ij} > 0)$.

Опр.
Пределы $p_i = lim\, p_i(t)$, $t \to \infty$, если они существуют, называются *предельными* (или финитными) *вероятностями состояний*.

Опр. Распределение вероятностей состояний, которое не зависит от времени $p_i(t + \tau) = p_i(t) = p_i$ для любых $t, \tau \geq 0$ и любых i=1,2,… называется стационарным.

Опр. Дифференциальные уравнения Колмогорова для вероятностей состояний:

$$\frac{dp_i}{dt} = \lambda_{ii} p_i(t) + \sum_{i \neq j} \lambda_{ji} p_j(t) = \sum_{i \neq j} \lambda_{ji} p_j(t) - p_i(t) \sum_{i \neq j} \lambda_{ij}, i = 1,2,...$$

Формулы для нахождения стационарного распределения:

$$\begin{cases} \sum_{i \neq j} \lambda_{ji} p_j - p_i \sum_{i \neq j} \lambda_{ij} = 0, i = 1,2,\dots,n \\ p_1 + p_2 + \cdots + p_n = 1 \end{cases}$$

Средства языка программирования Python, которые использованы в программе расчета:

np.dot(A, B) - умножение матриц A и B

np.max(A) - находит максимальный элемент в матрице

np.abs(A) - модуль

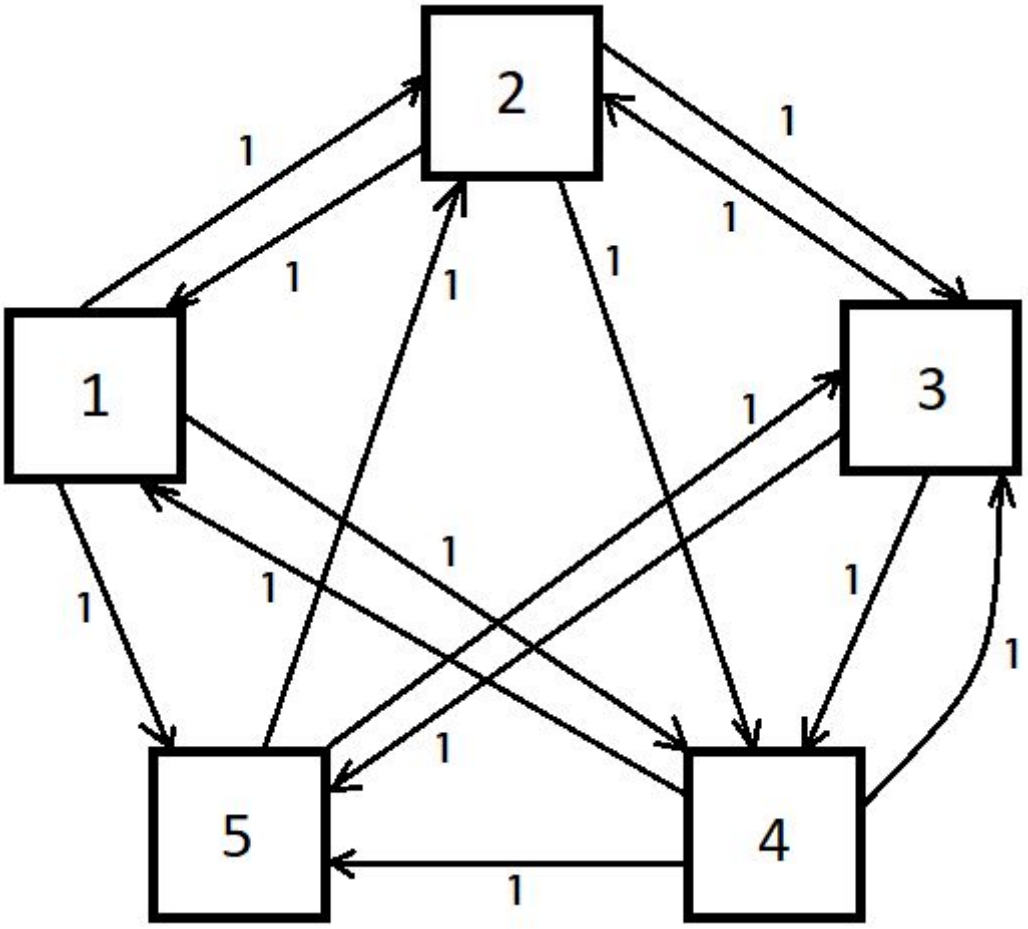
np.random.random_sample() - возвращает случайное число из полуинтервала [0,1)

Результаты расчетов

Матрица интенсивностей однородного марковского процесса с непрерывным временем Λ =

```
[-3  1  0  1  1]
[1  -3  1  1  0]
[0  1  -3  1  1]
[1  0  1  -3  1]
[0  1  1  0  -2]
```

1 Построить граф состояний марковского процесса.



2 Выписать систему дифференциальных уравнений Колмогорова для вероятностей состояний (с заданными интенсивностями).

$$
\begin{aligned}
p_1'(t) &= p_2 + p_4 - 3p_1 \\
p_2'(t) &= p_1 + p_3 + p_5 - 3p_2 \\
p_3'(t) &= p_2 + p_4 + p_5 - 3p_3 \\
p_4'(t) &= p_1 + p_2 + p_3 - 3p_4 \\
p_5'(t) &= p_1 + p_3 + p_4 - 2p_5 \\
p_1 + p_2 + p_3 + p_4 + p_5 &= 1
\end{aligned}
$$

3. Написать систему уравнений для нахождения стационарных вероятностей.

$$
\begin{array}{lcl}
0 = p_2 + p_4 - 3p_1 & & 3p_1 = p_2 + p_4 \\
0 = p_1 + p_3 + p_5 - 3p_2 & & 3p_2 = p_1 + p_3 + p_5 \\
0 = p_2 + p_4 + p_5 - 3p_3 & \Rightarrow & 3p_3 = p_2 + p_4 + p_5 \\
0 = p_1 + p_2 + p_3 - 3p_4 & & 3p_4 = p_1 + p_2 + p_3 \\
0 = p_1 + p_3 + p_4 - 2p_5 & & 2p_5 = p_1 + p_3 + p_4 \quad \text{- можно отбросить} \\
p_1 + p_2 + p_3 + p_4 + p_5 = 1 & & p_1 + p_2 + p_3 + p_4 + p_5 = 1
\end{array}
$$

4. Найти стационарное распределение вероятностей состояний $(r_1, r_2, r_3, r_4, r_5)$

Решая систему выше методом Гаусса получим: $r = (\frac{19}{147}, \frac{10}{49}, \frac{32}{147}, \frac{9}{49}, \frac{13}{49})$

5. Считая, что в начальный момент времени t_0 система находится в состоянии 1, провести моделирование развития системы до события с номером K.

Приложение

6. Составить таблицу 1 с данными о событиях:

Таблица 1

l	$t_{соб}(l)$	$C(l)$	$\tau(l)$	$\delta(l)$
1	0.2071724587682856	4	0.36482469258730654	1.0
2	0.5719971513555921	5	0.870854586967099	0.5
3	1.442851738322691	3	0.0584456532950035	0.3333333
4	1.5012973036521915	5	0.35187397018415123	0.1666667
5	1.8531712738363426	3	0.35333675912402795	0.15
6	2.206508032960371	4	0.6231203548853144	0.1333333
7	2.829628387845685	5	0.09370954352233879	0.0952381
8	2.9233379313680237	3	0.7751411833028935	0.0892857
9	3.698479114670917	5	0.08495641951894357	0.0694444
10	3.7834355341898607	2	0.06754179723264647	0.1
11	3.850977331422507	1	0.18072282499669867	0.0909091
12	4.031700156419205	4	0.07822303761358021	0.0681818
13	4.109923194032786	5	0.20493969617152935	0.0512821
14	4.314862890204315	3	0.07223520624525961	0.0549451

15	4.387098096449575	5	0.6280640413699321	0.0428571
16	5.015162137819507	2	0.03371353610954905	0.0583333
17	5.048875673929056	1	0.16464961180415388	0.0551471
18	5.21352528573321	5	0.4755850915343966	0.0359477
19	5.689110377267607	3	0.1702207945293779	0.0409357
20	5.859331171796985	5	0.2667605741895696	0.0315789
21	6.126091745986554	3	0.1561411058657603	0.0357143
22	6.2822328518523145	5	0.5645307094374262	0.0281385
23	6.846763561289741	2	0.27210503247037154	0.0395257
24	7.118868593760112	1	0.45555250625068744	0.0380435
25	7.5744211000108	4	0.25058271065064636	0.035
26	7.825003810661446	3	0.08449667183511707	0.0292308
27	7.909500482496563	4	0.3116057114963377	0.031339
28	8.2211061939929	1	0.12618294726773682	0.031746
29	8.347289141260637	2	0.34254400531183427	0.0307882
30	8.689833146572472	1	0.6109822963429734	0.0287356
31	9.300815442915445	2	0.4071678642915746	0.027957
32	9.70798330720702	3	0.11425074060534199	0.0241935
33	9.822234047812362	2	0.05952967787923397	0.0255682
34	9.881763725691595	3	0.26484697889062037	0.0222816
35	10.146610704582216	5	0.7207564590091576	0.0210084
36	10.867367163591373	2	0.2887008608526554	0.0230159
37	11.156068024444028	1	0.18717979254052153	0.0232733
38	11.34324781698455	5	0.12158379075444292	0.0192034
39	11.464831607738992	3	0.03959313455916019	0.0195682
40	11.504424742298152	2	0.21615368897072867	0.0205128
41	11.72057843126888	3	0.6267765724405541	0.0182927
42	12.347355003709435	4	0.11593244393450883	0.0209059
43	12.463287447643944	1	0.006569210060604855	0.0199336
44	12.469856657704549	4	0.040141249394493116	0.019556
45	12.509997907099041	5	0.4655708715089359	0.0166667
46	12.975568778607977	3	0.4088444253869517	0.0164251
47	13.38441320399493	4	0.003771035719139266	0.0180389
48	13.38818423971407	3	0.21533224745289894	0.0155142

49	13.603516487166969	5	0.42032612890793425	0.0153061
50	14.023842616074903	3	0.14798790439984066	0.0146939
51	14.171830520474744	2	0.5126903891655238	0.0164706
52	14.684520909640268	1	0.19420561207673281	0.0165913
53	14.878726521717	2	0.2526159595894296	0.0156023
54	15.13134248130643	1	0.012223541262214886	0.0157233
55	15.143566022568644	5	0.7540833244522498	0.0138047
56	15.897649347020893	2	0.13032747590446464	0.0146104
57	16.027976822925357	1	0.28365646541096795	0.0147243
58	16.311633288336324	4	0.1469569821804439	0.0148215
59	16.45859027051677	1	0.22459621640231014	0.0140269
60	16.68318648691908	5	0.9581490488172707	0.0127119
61	17.64133553573635	2	0.08880507253668936	0.013388
62	17.73014060827304	4	0.1126605754757716	0.0137493
63	17.84280118374881	3	0.10724510248558035	0.0122888
64	17.950046286234393	2	0.05863231769718089	0.0126488
65	18.008678603931575	4	0.19359022113569468	0.0129808
66	18.202268825067268	1	0.379038034854981	0.0125874
67	18.58130685992225	5	1.3984099054707932	0.0115332
68	19.979716765393043	2	0.32336111364431824	0.0118525
69	20.30307787903736	1	0.194604783172245	0.0119352
70	20.497682662209606	5	0.5302788816951487	0.0109731
71	21.027961543904755	2	0.4174187339985589	0.0112676
72	21.445380277903315	4	0.3215273787525897	0.0117371
73	21.766907656655903	3	0.8657608535218065	0.0108447
74	22.63266851017771	5	0.07091158092265502	0.0103665
75	22.703580091100363	3	0.11593326775575802	0.0104505
76	22.81951335885612	5	0.14700309914445267	0.01
77	22.966516458000573	3	0.2766843430739584	0.010082
78	23.243200801074533	2	0.041809590268242304	0.010323
79	23.285010391342777	1	0.2844967866244482	0.0105485
80	23.569507177967225	5	0.577293885924168	0.0094937
81	24.14680106389139	2	0.49319373550967965	0.0098765
82	24.639994799401073	1	0.7914431335277927	0.0100873

83	25.431437932928866	2	0.4748755997170352	0.0095504
84	25.9063135326459	4	0.07222125766513644	0.0101836
85	25.978534790311038	1	0.03446677661601817	0.0096639
86	26.013001566927056	2	0.08714713872657673	0.0091655
87	26.10014870565363	1	0.4490408885207332	0.0093558
88	26.549189594174365	2	0.01698669691003637	0.0088819
89	26.5661762910844	3	0.24115104603989804	0.0089377
90	26.8073273371243	4	0.8384295170464957	0.0094881
91	27.645756854170795	3	0.031012781706990594	0.0086691
92	27.676769635877786	2	0.7106542402525218	0.0084806
93	28.387423876130306	3	0.7604152684611672	0.0084151
94	29.147839144591472	2	0.5536885437372241	0.0082361
95	29.701527688328696	4	0.023178178352750725	0.0089586
96	29.724705866681447	3	0.2977988251257589	0.008114
97	30.022504691807207	5	0.23223052199862057	0.0081615
98	30.254735213805827	2	0.2688141137145485	0.0078898
99	30.523549327520374	3	0.7302449282624908	0.0078334
100	31.253794255782864	4	0.5817556921892121	0.0084848
...	...	...	...	...
7661	2882.0156602418415	2	0.07502718597208265	0.0001009
7662	2882.0906874278135	4	0.5576377327060493	0.0001046
7663	2882.6483251605196	3	0.16092787840988257	0.0001
7664	2882.8092530389295	5	0.7684430132759215	0.0001043
7665	2883.5776960522053	2	0.18910845870471726	0.0001009
7666	2883.76680451091	3	0.5317396291254142	0.0001

7. Составить таблицу 2 с данными о состояниях следующего вида:

Таблица 2

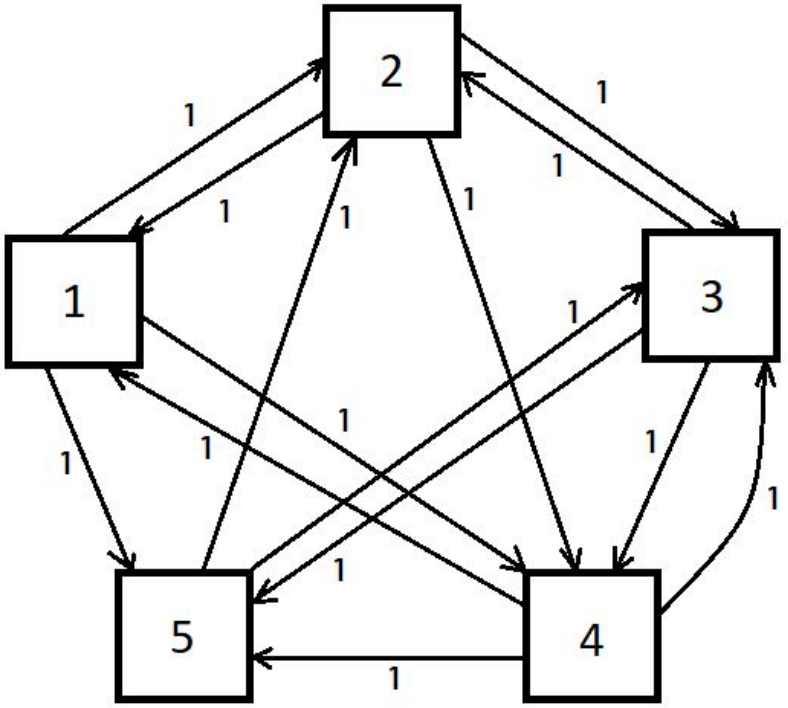
i	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
1	17	0.17	4.78678	0.15418
2	23	0.23	6.11848	0.19707
3	23	0.23	6.9139	0.22269
4	16	0.16	3.49677	0.11263
5	21	0.21	9.93787	0.320095

8. Составить таблицу 3 с данными о состояниях следующего вида:

Таблица 3

i	$R_i(K)$	$v_i(K)$	$T_i(K)$	$\Delta_i(K)$
1	1073	0.13997	362.9253	0.12586
2	1740	0.22698	589.84544	0.20455
3	1792	0.23376	600.74503	0.20833
4	1521	0.19841	536.19230	0.18595
5	1540	0.20089	794.05873	0.27537

Анализ результатов и выводы



K = 7666

i	1	2	3	4	5
r_i	0.12925	0.20408	0.21769	0.18367	0.26531
$v_i(100)$	0.17	0.23	0.23	0.16	0.21
$v_i(K)$	0.13997	0.22698	0.23376	0.19841	0.20089
$\Delta_i(100)$	0.15418	0.19707	0.22269	0.11263	0.3201
$\Delta_i(K)$	0.12586	0.20455	0.20833	0.18595	0.27537

Литература по теории случайных процессов

1. Булинский А. В., А. Н. Ширяев А. Н. Теория случайных процессов: Учебник для вузов. — М.: ФИЗМАТЛИТ, 2005.
2. Вентцель Е. С., Овчаров Л. А. Теория случайных процессов и ее инженерные приложения: Учеб. пособие для вузов. — М.: Высшая школа, 2007.
3. Лобузов А.А., Гумляева С.Д., Норин Н.В. Задачи по теории случайных процессов. — М.: МИРЭА, 1993.
4. Письменный Д. Т. Конспект лекций по теории вероятностей, математической статистике и случайным процессам — М.: Айрис-пресс, 2007.
5. Прохоров А. В., Ушаков В. Г., Ушаков Н. Г. Задачи по теории вероятностей. Основные понятия, предельные теоремы, случайные процессы. — М.: КДУ, 2009.
6. Сборник задач по теории вероятностей, математической статистике и теории случайных функций: Учеб. пособие для вузов / Б.Г. Володин, М.П.Ганин, И.Я. Динер и др.; Под ред. А. А. Свешникова. — СПб.: Лань, 2008.
7. Кемени Д., Снелл Д. Конечные цепи Маркова. — М.: Наука, 1970.
8. Кемени Д., Снелл Д., Кнепп А. Счетные цепи Маркова. — М.: Наука, 1987.
9. Чжун Кай-Лай. Однородные цепи Маркова. — М.: Мир, 1964.
10. Карлин С. Основы теории случайных процессов. — М.: Мир, 1971.
11. Гихман И.И., Скороход А.В. Введение в теорию случайных процессов: Учеб. пособие для вузов. — М.: Наука, 1975.
12. Ивченко Г. И., Каштанов В. А., Коваленко И. Н. Теория массового обслуживания: Учеб. пособие для вузов. — М.: Либроком, 2012.

Приложение

```
import numpy as np

A = np.array([[-3, 1, 0, 1, 1], [1, -3, 1, 1, 0], [0, 1, -3, 1, 1], [1, 0, 1, -3, 1], [0, 1, 1, 0, -2]])


# интенсивность выхода из состояния i
def OutputIntensity(l, i): # l- lambda_ij - строка матрицы A, i - string number
  sum = 0
  for k in range(len(l)):
    if k != i:
      sum += l[k]
  return  sum

def GetItFrequency(sequenceOfStates, numberOfReturns, returnFrequency, pastFrequecy):
  #print("сработала функция GetItFreq")
  for i in range(5):
    numberOfReturns[i] = sequenceOfStates.count(i)
    #print("количество входов в состояние "+str(i)+":", numberOfReturns[i])
    pastFrequency[i] = returnFrequency[i]
    returnFrequency[i] = numberOfReturns[i] / len(sequenceOfStates)
    #print("частота вхождений в состояние(returnFreq)"+str(i)+":", returnFrequency[i])


def PrCheck(freq, pastfreq): # проверка условия
  #print("PrCheck work...")
  a=[]
  for i in range(5):
     a.append(np.abs(freq[i] - pastfreq[i])) # разница последней и предпоследней частоты
  # print("массив разниц частот:",a)
  # print("Значение отклонения delta_l =", round(np.max(a), 8))
  return np.max(a) # максимум среди них

def Delta(T, tK, t1):
  return T/(tK - t1)



time = [0] # время
state = [0, 1, 2, 3, 4] # - состояния системы
i = state[0]      # начальное состояние
j = 0
numberOfReturns = [0, 0, 0, 0, 0] # R(l) -число попаданий системы в состояние i=1...5
returnFrequency = [0, 0, 0, 0, 0] # v(l) -частота попаданий
pastFrequency = [0, 0, 0, 0, 0]
T = [0, 0, 0, 0, 0] # - общее время нахождения в i-том состоянии
sequenceOfStates = []
# time.append(np.random.exponential(scale = 1/OutputIntensity(A[0], 0))) # scale =
1/lambda[1]

# переменные для вывода
l = 0 # номер события
tl = 0 # момент наступления события l
# tau_l = time[i] - время пребывания в l-ом состоянии
```

```
# delta = round(np.max(a), 6) - отклонение


#print("Определение первого состояния...")

lam = OutputIntensity(A[i], i) # лямбда - интенсивность
time.append(np.random.exponential(scale = 1/OutputIntensity(A[i], i))) # время нахождения в новом i-том состоянии

#print("lambda_i =", lam)
if lam == 0:
  j = i  # j - следующее состояние
elif lam > 0:
  num=0
  cutState = []  # список соответствующих состояний
  cutLength = [] # список длин отрезков

  for k in range(len(A[i])):
    if A[i][k] > 0:
      num += 1   # подсчет количества отрезков
      cutLength.append(A[i][k]/lam)
      cutState.append(k)      # длина i-го отрезка
  nextState = np.random.random_sample() # выбираем рандомное число из отрезка [0, 1], по которому определяем следующее состояние
  #print("рандомное число ", nextState)
  #print("список длин отрезков", cutLength)
  #print("список соответствующих состояний", cutState)
  cutStart = 0
  cutEnd = cutLength[0]
  buf = 0
  cut = False
  while cut == False:
    #print("проверка отрезка")
    #print("[", round(cutStart, 5), ",", round(cutEnd, 5), "]")
    if cutStart <= nextState and nextState < cutEnd:
      #print("для начала подходит")
      cut = True
      st = cutEnd - cutStart
      j = cutState[buf]
      sequenceOfStates.append(j)
      l+=1
      tl += time[len(time)-1]
      T[i] += time[len(time)-1]
      #print("j=", j, end="\n")
      GetItFrequency(sequenceOfStates, numberOfReturns, returnFrequency, pastFrequency)
    else:
      #print("для начала не подходит, следующий отрезок")
      buf+=1
      cutEnd = cutEnd + cutLength[buf]
      cutStart += cutLength[buf-1]
  tsob1 = tl
  time.append(np.random.exponential(scale = 1/OutputIntensity(A[j], j)))
  print(l, "\t", tl, "\t", j+1, "\t", time[len(time)-1], "\t", round(PrCheck(returnFrequency, pastFrequency) ,7))


while PrCheck(returnFrequency, pastFrequency) >= 0.0001:
  if l == 100:
```

```
    print(100, "\tRi(100) \tvi(100) \tTi(100)   \tDelta_i(100) --------")
    for i in range(5):
      print(i+1, "\t", numberOfReturns[i], "\t", returnFrequency[i], "\t", T[i], "\t",
Delta(T[i], tl, tsob1), end="\n")
    print("-----------------------------------------------")
  i = j                 # предыдущее состояние
  lam = OutputIntensity(A[i], i)
  #time.append(np.random.exponential(scale = 1/OutputIntensity(A[i], i))) # время нахождения
в новом i-том состоянии
  #print("Определение следующего состояния...")
  #print("lambda_i =", lam)
  if lam == 0:
    j = i           # j - следующее состояние
  elif lam > 0:
    num=0
    cutState = []  # список соответствующих состояний
    cutLength = [] # список длин отрезков
    for k in range(len(A[i])):
      if A[i][k] > 0:
        num += 1    # подсчет количества отрезков
        cutLength.append(A[i][k]/lam)
        cutState.append(k)      # длина i-го отрезка
    nextState = np.random.random_sample() # выбираем рандомное число из отрезка [0, 1], по
которому определяем следующее состояние
    # print("рандомное число ", nextState)
    #print("список длин отрезков", cutLength)
    #print("список соответствующих состояний", cutState)
    cutStart = 0
    cutEnd = cutLength[0]
    cut = False
    buf = 0
    while cut == False:
      # print("проверка отрезка [", round(cutStart, 5), ",", round(cutEnd, 5), "]")
      if cutStart <= nextState and nextState < cutEnd:
        #print("подходит")
        cut = True
        st = cutEnd - cutStart
        j = cutState[buf]  # определили новое состояние
        sequenceOfStates.append(j)
        l+=1
        tl += time[len(time)-1]
        T[i] += time[len(time)-1] # общее время пребывания в событии j
        #print("j=", j, end="\n")
        GetItFrequency(sequenceOfStates, numberOfReturns, returnFrequency, pastFrequency)
      else:
        #print("не подходит, следующий отрезок")
        buf+=1
        cutEnd = cutEnd + cutLength[buf]
        cutStart += cutLength[buf-1]
    time.append(np.random.exponential(scale = 1/OutputIntensity(A[j], j)))
    print(l, "\t", tl, "\t", j+1, "\t", time[len(time)-1], "\t",
round(PrCheck(returnFrequency, pastFrequency) ,7))


print()
for i in range(5):
  print(i+1, "\t", numberOfReturns[i], "\t", returnFrequency[i], "\t", T[i], "\t",
Delta(T[i], tl, tsob1), end="\n")
```