
КАФЕДРА

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
РУКОВОДИТЕЛЬ

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

Отчет о лабораторной работе №2

Имитационное моделирование случайных потоков. Проверка гипотезы:
сумма пуассоновских потоков – поток пуассоновский с интенсивностью,
равной сумме интенсивностей отдельных потоков.

По дисциплине: Компьютерное моделирование

РАБОТУ
ВЫПОЛНИЛ:
СТУДЕНТ ГР. №

подпись, дата

инициалы, фамилия

Санкт-Петербург 2024

Цель работы:

Цель настоящей работы – освоить средства моделирования случайных величин (СВ) с произвольным распределением на основе равномерного распределения. Построить имитационную модель двух потоков, в котором длительность промежутков времени между поступлениями заявок имеет показательный закон с параметрами λ_1 , λ_2 . Осуществить проверку статистической гипотезы о соблюдении свойства аддитивности пуассоновского потока (сумма пуассоновских потоков есть поток пуассоновский).

Ход работы:

1. Ознакомиться со справочными сведениями; сформулировать особенности пуассоновского потока событий; указать связь (дискретного) пуассоновского потока и (непрерывного) показательного распределения.
2. Запрограммировать предложенный алгоритм генерации пуассоновского потока с использованием MatLab или Python.
3. Создать графическую интерпретацию потока событий.
4. Осуществить проверку гипотезы о виде распределения для суммарного потока.
5. Сравнить интенсивности выборочных и теоретических интенсивностей потоков.
6. Составить и представить преподавателю отчет о работе.

Исходные данные:

Промежуток наблюдения $[T_1, T_2]$, параметр λ . Значения параметра λ должны быть выбраны в зависимости от номера студента в списке группы N , где $T_1 = N$, $T_2 = N + 100$, $\lambda_1 = (N+8)/(N+24)$, $\lambda_2 = (N+9)/(N+25)$.
Номер студента = 14.

1. Ознакомиться со справочными сведениями и сформулировать особенности пуассоновского потока событий.

Особенности пуассоновского потока:

1. Однородность событий: Пуассоновский поток — это последовательность событий, происходящих независимо друг от друга в случайные моменты времени. Основное предположение заключается в том, что количество событий за фиксированный промежуток времени распределено по закону Пуассона.

2. Свойство аддитивности: если два независимых пуассоновских потока с интенсивностями λ_1 и λ_2 суммируются, то суммарный поток тоже будет пуассоновским, с интенсивностью $\lambda_1 + \lambda_2$.

3. Показательное распределение: Промежутки времени между событиями в пуассоновском потоке подчиняются показательному распределению с параметром λ , где λ — интенсивность потока (среднее количество событий в единицу времени). Формула для плотности вероятности: $f(t) = \lambda e^{-\lambda t}$, где t — время между событиями.

4. Независимость событий: События в потоке независимы. Это означает, что знание о времени наступления одного события не даёт информации о времени следующих.

Связь между пуассоновским потоком и показательным распределением:

Пуассоновский поток описывает дискретное количество событий, которые происходят за фиксированные интервалы времени.

В свою очередь, непрерывное показательное распределение описывает распределение времени между последовательными событиями. Пуассоновский процесс можно получить, если интервал времени между событиями распределён по показательному закону.

2. Запрограммировать предложенный алгоритм генерации пуассоновского потока с использованием MatLab или Python.

Алгоритм генерации пуассоновского потока (на основе справочных сведений):

$$N = 14$$

$$T1 = N = 14$$

$$T2 = N + 100 = 114$$

$$\lambda_1 = \frac{14+8}{14+24}$$

$$\lambda_2 = \frac{14+9}{14+25}$$

1. Генерация случайного числа u (0, 1).

2. Преобразование этого числа в интервал времени между событиями $t_i = -\frac{1}{\lambda} \ln(u)$, где λ — интенсивность потока.

3. Суммирование этих интервалов для получения времени каждого следующего события.

Код реализует генерацию пуассоновского потока с заданными параметрами λ_1 и λ_2 . Он генерирует случайные промежутки времени между событиями, накапливая их, и выводит график каждого потока.

Алгоритм генерации пуассоновского потока:

1. Генерация случайного числа: генерируем случайное число U из равномерного распределения на интервале $(0, 1)$.
2. Применение обратной функции: преобразуем его в экспоненциально распределённое значение с параметром λ с помощью формулы $t = -\frac{1}{\lambda} \ln(U)$, где t — это время между событиями.
3. Накопление времени: генерируем последующие события, добавляя эти интервалы к предыдущему моменту времени до тех пор, пока не выйдем за пределы отрезка $[T_1, T_2]$.

Вывод программы:

События для первого потока (λ_1): [16.90228459 19.82174678 20.5919397
25.03197388 25.16497239

28.00986612 29.6599577 30.22109742 31.8615716 32.62241561
35.22240382 39.33174998 39.89412023 40.40517872 44.66601635
48.33549217 51.15489107 52.96771138 54.41261842 54.46532965
56.84275284 57.18612105 57.34584492 58.08532304 58.76715672
65.19529944 68.4774874 70.50604339 71.6385158 72.84250612
73.52553344 75.52892028 76.13092287 76.79180525 77.77071507
78.32799736 80.46772745 80.99168356 81.52540648 82.34641591
83.26022352 84.3235125 86.23824353 88.96284097 89.30120711
93.07202536 94.17986196 94.66083083 97.46433783 98.08218674
99.40595315 103.36996382 109.41193608 112.51489748 113.96931367]

События для второго потока (λ_2): [14.45710337 16.02028283 20.57569158
20.58310973 24.18864209

27.0341815 27.57730464 28.51693065 30.25962985 31.41228746
34.2993756 40.21784943 40.54686437 41.07192171 41.08758205
42.29555323 42.6641353 49.04936288 49.27490721 50.42375467
51.79402082 52.56518704 55.91452763 56.74243483 57.43699599
58.39755845 59.21909663 61.93632166 62.81470845 63.04052291
63.09797422 64.33538309 71.86949536 72.02075874 73.54108674
74.10264709 75.10079776 76.22529617 78.16881989 78.43477949
82.79664545 82.81181906 83.29267495 83.57597545 84.00951888
87.07932078 88.49145465 91.10140518 91.8140583 91.86561879
100.67266867 103.27158283 103.42414576 104.41914296 105.540518
109.37531541 110.33382385 110.92051341 113.13635615]

Process finished with exit code 0

3. Создать графическую интерпретацию потока событий

Нужно построить графики для обоих пуассоновских потоков с момента времени $T1 = 14$ до $T2 = 114$. График должен отображать моменты событий на временной оси.

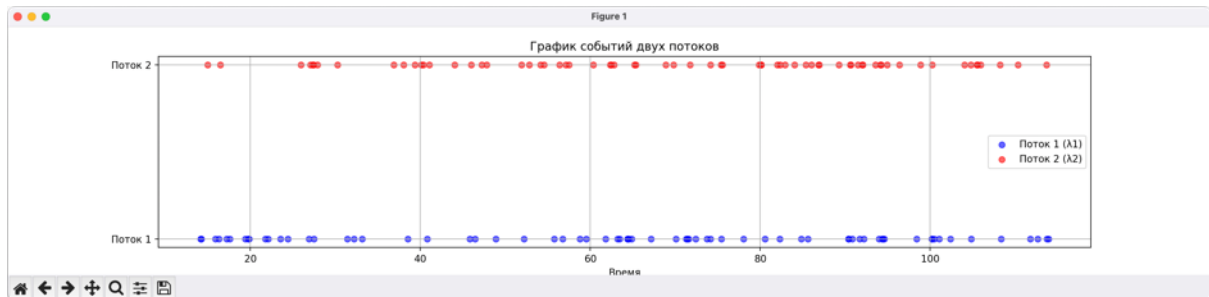


Рисунок 1 – Решение на Python, визуальное представление двух потоков на временной прямой

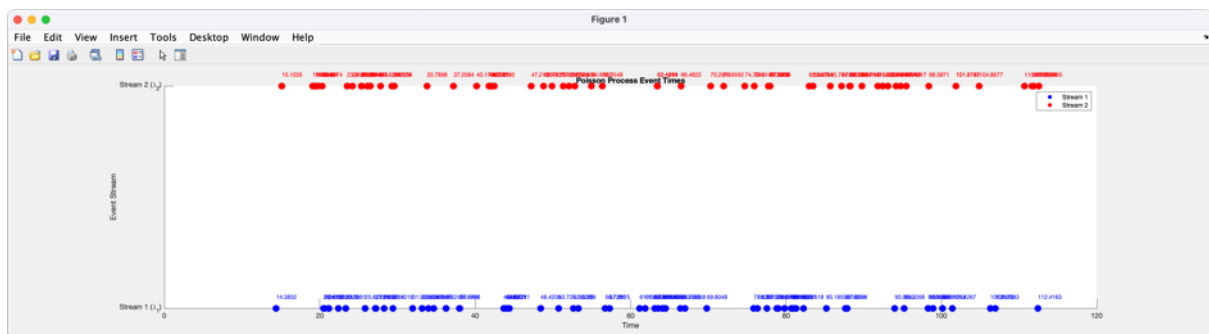


Рисунок 2— решение на MatLab, визуальное представление двух потоков на временной прямой

4. Осуществить проверку гипотезы о виде распределения для суммарного потока.

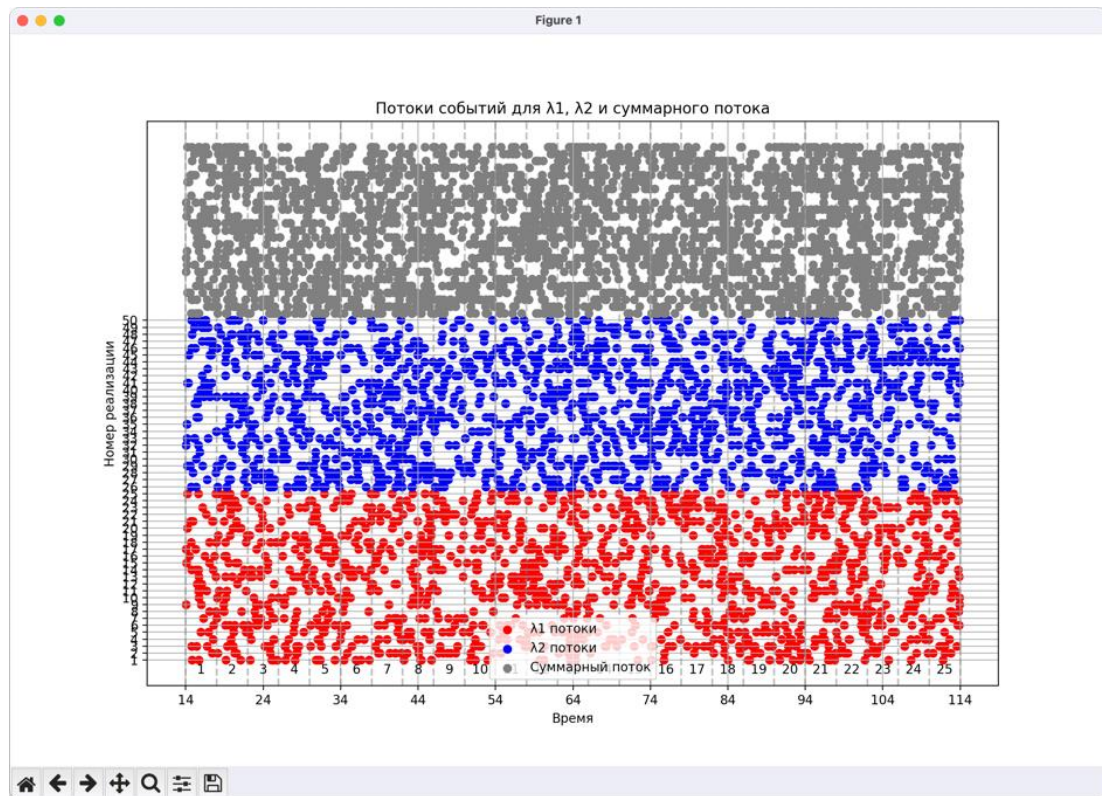


Рисунок 3 – генерация по 25 потоков для лямбды 1 и 2 (красный и синие точки соответственно), серый поток - суммарный

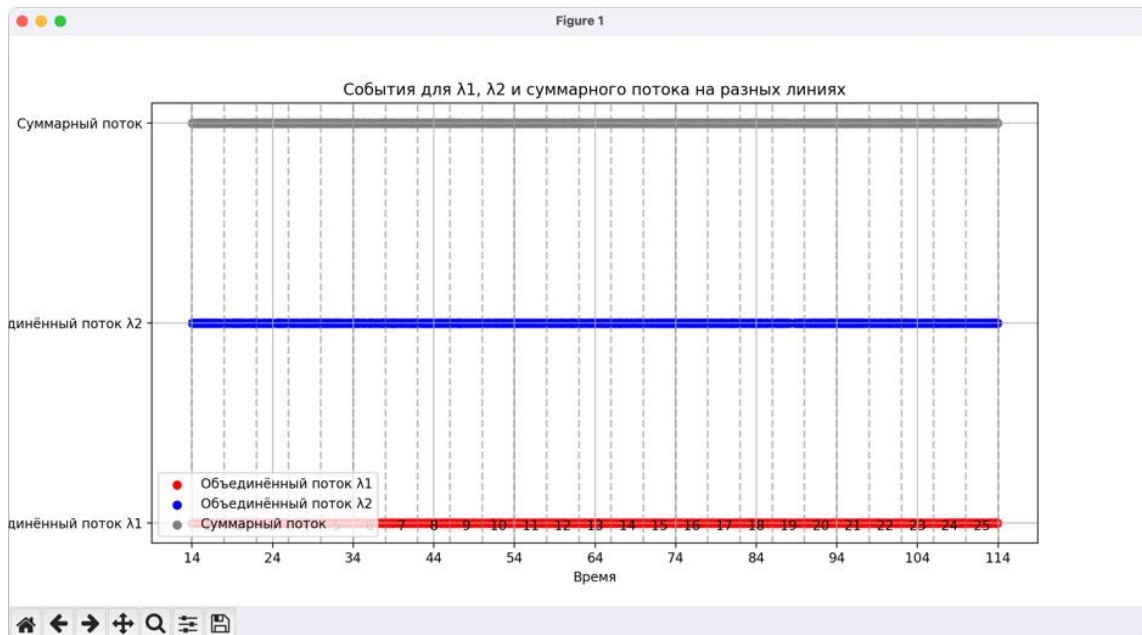


Рисунок 4 – объединение потоков на одну временную прямую

Таблица: $X_{j(i)}$ - количество наступлений событий в моменты t_i в массиве $\{X_{j(i)}\}$, попавших в промежуток $[T_1 + (i - 1)\Delta t, T_1 + i\Delta t]$ в потоке с номером j .

	Интервал 1	Интервал 2	Интервал 3	Интервал 4	Интервал 5	Интервал 6	Интервал 7	Интервал 8	Интервал 9	Интервал 10	Интервал 11	Интервал 12	Интервал 13	Интервал 14	Интервал 15	Интервал 16	Интервал 17	Интервал 18	Интервал 19	Интервал 20	Интервал 21	Интервал 22	Интервал 23	Интервал 24	Интервал 25
Поток 1	4	5	8	4	7	6	6	9	3	5	6	7	2	6	4	1	2	3							
Поток 2	3	4	2	6	5	7	3	1	7	4	7	2	4	3	4	7	4	11							
Поток 3	5	3	8	12	6	4	5	4	7	6	9	10	9	7	6	8	4	3							
Поток 4	12	5	6	4	4	6	5	4	7	6	6	3	4	5	4	2	2	2							
Поток 5	3	7	3	2	4	2	4	6	3	5	2	5	3	5	7	12	4	7							
Поток 6	5	7	5	5	1	7	4	6	4	4	5	7	3	8	8	7	4	7							
Поток 7	8	2	6	1	10	9	4	4	6	2	6	10	1	10	5	3	5	8							
Поток 8	4	11	4	6	8	5	8	4	6	5	2	2	6	2	8	5	10	3							
Поток 9	7	4	10	7	4	6	8	3	6	3	4	5	2	5	6	2	4	3							
Поток 10	3	3	2	6	4	7	4	5	8	7	4	8	10	3	5	6	4	7							
Поток 11	6	6	5	5	4	7	7	6	8	5	9	5	10	4	6	5	10	8							
Поток 12	5	2	6	7	4	5	6	10	7	7	11	4	5	9	4	9	3	4							
Поток 13	6	4	3	3	8	3	2	5	4	2	7	8	7	2	7	8	2	4							
Поток 14	7	7	4	6	3	4	6	4	6	3	6	4	4	4	4	4	5	5							
Поток 15	4	5	9	6	8	5	7	5	5	6	10	9	4	7	5	1	6	5							
Поток 16	6	4	2	3	3	3	6	6	6	2	5	6	10	6	4	4	1	7							
Поток 17	2	4	4	10	6	7	8	10	4	8	3	6	7	4	4	6	5	2							
Поток 18	11	6	7	13	4	2	6	6	4	8	5	6	5	8	8	3	4	4							
Поток 19	5	7	5	3	7	3	11	5	11	4	9	6	4	6	4	7	5	4							
Поток 20	5	3	5	6	5	9	4	3	9	6	7	4	4	4	3	10	9	5							
Поток 21	5	7	3	1	5	6	7	5	6	7	3	6	4	8	5	3	5	2							
Поток 22	3	3	7	6	7	5	2	3	7	4	8	5	6	6	8	6	5	1							
Поток 23	11	6	4	6	3	3	6	4	6	2	3	4	4	5	6	10	6	4							
Поток 24	3	6	3	3	4	2	4	6	2	4	3	2	5	5	5	5	2	7							
Поток 25	5	5	4	8	6	5	4	2	4	3	5	8	2	7	7	8	4	4							
Поток 26	7	5	2	3	9	4	3	2	7	4	8	4	10	5	7	5	6	2							
Поток 27	2	0	3	5	4	4	2	5	10	4	2	3	2	6	7	6	3	6							
Поток 28	4	5	2	4	5	5	4	7	4	2	6	3	4	2	3	6	7	2							
Поток 29	5	7	5	9	9	6	11	1	6	8	10	7	6	10	5	11	6	6							
Поток 30	2	5	8	3	7	6	6	10	7	5	4	4	13	7	4	8	6	7							
Поток 31	2	5	6	3	4	2	3	4	6	7	5	4	5	7	1	6	6	6							
Поток 32	5	7	4	7	6	5	2	4	5	10	5	6	7	7	6	6	6	7							
Поток 33	6	7	8	7	2	1	8	4	1	4	5	6	4	2	1	3	2	5							
Поток 34	7	6	2	5	10	9	9	7	1	7	7	3	5	2	2	7	3	3							

Рисунок 5 – таблица событий, количество попаданий для каждой итерации в каждом временном промежутке

Проводим анализ частот событий, подсчитав, сколько раз зафиксировано каждое количество событий в каждом интервале. Мы определили:

Уникальные значения количества событий.

Частоты появления этих значений.

Для каждого количества событий мы также рассчитали **теоретические частоты** с использованием **пуассоновского распределения**. Теоретические частоты показывают, сколько раз, согласно пуассоновскому распределению, каждое количество событий должно было произойти.

variant_frequency_analysis.csv				
Вариант (n _l)	Частота (n _l)	n _l * n _l	n _l (теор)	(n _l - n _l теор) ² / n _l теор
0	2	0	6.989614470883766	3.561892100309905
1	33	33	36.25144973884451	0.2916276584906859
2	98	196	94.00859043387311	0.16946696308328943
3	134	402	162.524352157498	5.006256940599798
4	228	912	210.73205854624368	1.4149807300673338
5	218	1090	218.5912444978234	0.0015991951416421942
6	186	1116	188.9529469346727	0.04614850279105531
7	145	1015	139.9999198268216	0.17857725753798542
8	103	824	90.76323304714711	1.6497700713310939
9	42	378	52.304566290615796	2.030111211470773
10	39	390	27.12762433314253	5.195932465150771
11	15	165	12.79060934741408	0.38163991434238664
12	5	60	5.528179627771239	0.05046393894133227
13	2	26	2.20551975041814	0.019151208146708554

Рисунок 6– обработанные данные для каждого параметра

$\hat{\lambda} * \Delta t = M\eta = \frac{1}{N} \sum_{l=1}^L \eta_l n_l$	Выборочная интенсивность наступления событий $\hat{\lambda} \Delta t$ как выборочное математическое ожидание СВ $\lambda \Delta t$ (число событий, наступивших в ед.времени).
$\hat{p}_l = \frac{(\hat{\lambda} * \Delta t)^{n_l}}{n_l!} * e^{-\hat{\lambda} * \Delta t}$	Оценка «теоретической» вероятности по формуле Пуассона
$n_l^{теор} = \hat{p}_l \cdot N$	Оценки «теоретических» частот вариантов
$N = \sum_{l=1}^L n_l$	Объем выборки как сумма частот
$\chi^2_{\text{практ}} = \sum_{l=1}^L \frac{(n_l - n_l^{теор})^2}{n_l^{теор}}$	Критерий χ^2 (хи-квадрат), проверяет значимость расхождения эмпирических (наблюдаемых) и теоретических (ожидаемых) частот

1. Выборочная интенсивность наступления событий ($\hat{\lambda} * \Delta t$)

```
lambda_hat_dt = np.sum(unique_values * counts) / N_total
print(f'Выборочная интенсивность  $\hat{\lambda} * \Delta t = \{lambda\_hat\_dt\}$ ")
```

2. Оценка теоретической вероятности (\hat{p}_l)

```
p_hat_l = (lambda_hat_dt ** unique_values * np.exp(-lambda_hat_dt)) /
np.array([math.factorial(x) for x in unique_values])
print(f'Теоретическая вероятность  $\hat{p}_l = \{p\_hat\_l\}$ ")
```

3. Оценка теоретических частот ($n_l^{\text{теор}} = \hat{p}_l * N$)

```
n_l_teor = p_hat_l * N_total
print(f'Теоретические частоты  $n_l^{\text{теор}} = \{n\_l\_teor\}$ ")
```

4. Объем выборки как сумма частот ($N = \sum n_l$)

```
N_sum = np.sum(counts)
print(f'Объем выборки  $N = \{N\_sum\}$ ")
```


5. Вычисляем статистику хи-квадрат

```
chi_square_values = (observed_combined - theoretical_combined) ** 2 /
theoretical_combined
chi_square_statistic = np.sum(chi_square_values)
print(f'Хи-квадрат статистика  $\chi^2 = \{chi\_square\_statistic\}$ ')
```

Рисунок 7 - Результат работы программы на Python

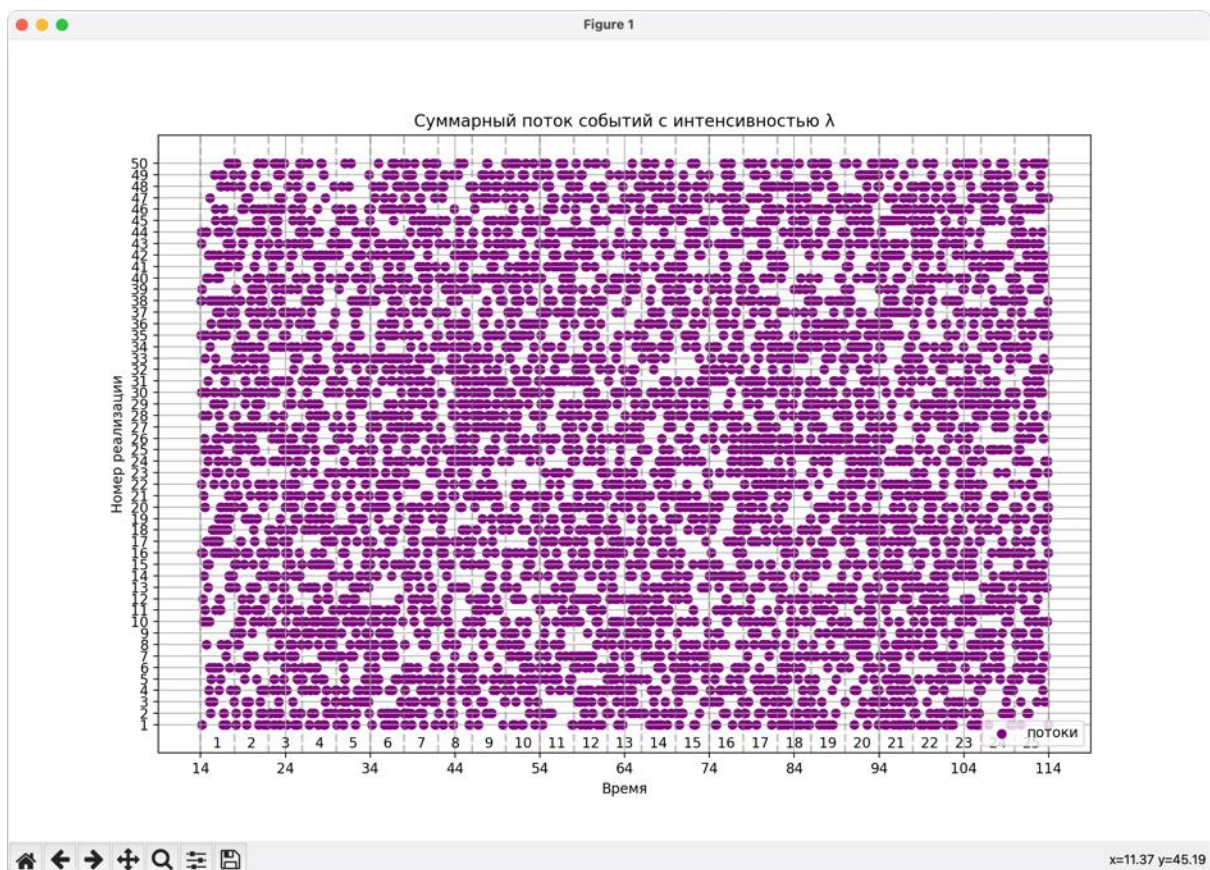


Рисунок 8 – 50 суммарных потоков λ ($\lambda = \lambda_1 + \lambda_2$)

В коде проверка гипотезы о виде распределения для суммарного потока осуществляется следующим образом:

1. Сначала собираются данные о событиях в двух потоках с параметрами λ_1 и λ_2 . Для каждого потока подсчитывается количество событий в заданных интервалах времени.
2. Наблюдаемые частоты определяются на основе собранных данных. Они представляют собой массив, содержащий количество событий в каждом из 25 интервалов времени.
3. Ожидаемые частоты вычисляются с учетом теоретической интенсивности потока. Поскольку суммарный поток должен следовать распределению Пуассона, ожидаемые частоты равномерно распределяются по всем интервалам, основываясь на общей интенсивности.
4. Для проверки гипотезы о распределении применяется критерий χ^2 . Эта статистика рассчитывается путем сравнения наблюдаемых и ожидаемых частот.
5. Вычисляется p-value, показывающее вероятность получения наблюдаемых данных, если нулевая гипотеза о распределении Пуассона верна. Это значение сравнивается с уровнем значимости.
6. Определяется критическое значение χ^2 для заданной степени свободы, которая рассчитывается как количество интервалов минус один.
7. Если полученная статистика χ^2 меньше критического значения, или p-value больше уровня значимости, гипотеза о распределении Пуассона не отвергается.
8. В результате выводится информация о статистике χ^2 , p-value и решении о принятии или отклонении гипотезы.

Описание кода:

1. Импорт библиотек:
Импортируются необходимые библиотеки для работы с массивами, построения графиков, статистического анализа и обработки данных.
2. Инициализация параметров:
Задаются параметры для генерации потоков событий, включая интенсивности λ_1 и λ_2 , общее количество реализаций и временные интервалы.
3. Генерация потока событий:
Определяется функция `generate_poisson_stream`, которая генерирует события по распределению Пуассона для заданных параметров.
4. Создание потоков событий:
Генерируются два потока событий (λ_1 и λ_2) по 25 реализаций каждый.
5. Объединение потоков:
Создаются объединенные потоки событий из двух потоков (λ_1 и λ_2).
6. Визуализация потоков:
Строятся графики, отображающие потоки событий λ_1 , λ_2 и их сумму.
7. Проверка гипотезы о распределении:
Подсчитываются наблюдаемые частоты событий в объединенном потоке и ожидаемые частоты по распределению Пуассона.
Выполняется тест хи-квадрат для проверки гипотезы о распределении.
8. Выборочная интенсивность:
Рассчитывается выборочная и теоретическая интенсивности для суммарного потока событий.
9. Создание таблицы с частотами:
Создается таблица, в которой фиксируются частоты событий для каждого временного интервала из двух потоков.
10. Сохранение данных:
Таблицы сохраняются в формате CSV для дальнейшего анализа.
11. Обработка сохраненных данных:

Читаются сохраненные данные, извлекаются уникальные значения и их частоты. Рассчитываются теоретические частоты и хи-квадрат для новой таблицы.

12. Сохранение новой таблицы:

Создается и сохраняется новая таблица с уникальными значениями, частотами и результатами статистического анализа в формате CSV.

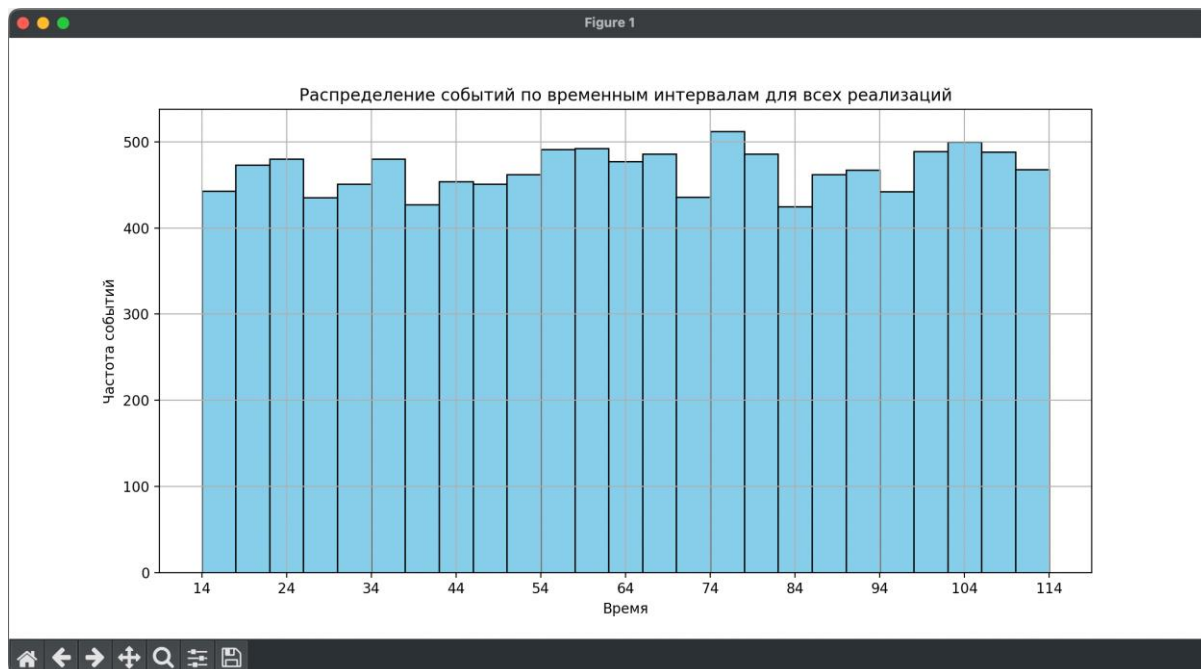


Рисунок 9 – распределение событий

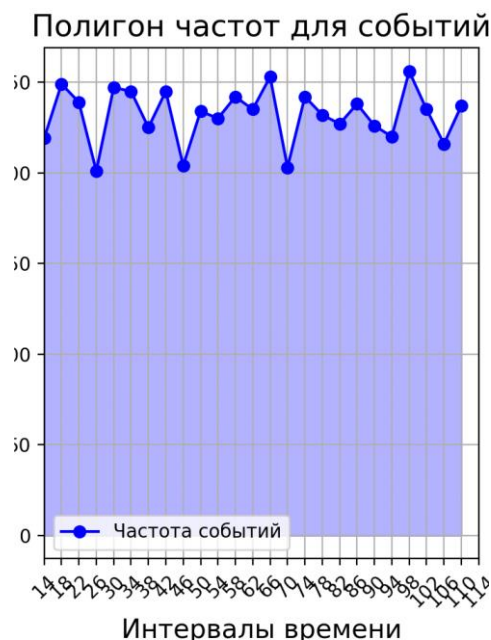


Рисунок 10 – распределение событий

Разные визуальные представления для разных версий программ

5. Сравнить интенсивности выборочных и теоретических интенсивностей потоков

Мы должны сравнить выборочные интенсивности потоков с теоретическими значениями параметров λ_1 и λ_2 .

Выборочная интенсивность вычисляется как количество событий, произошедших за наблюдаемый интервал времени, делённое на длину этого интервала:

$$\lambda = \frac{\text{Кол-во событий}}{T_2 - T_1}, \text{ где } T_1=14, T_2=114$$

Теоретические интенсивности уже известны: $\lambda_1 = \frac{14+8}{14+24} = 0.58$, $\lambda_2 = \frac{14+9}{14+25} = 0.59$

Теперь нужно вычислить выборочные интенсивности для потоков и сравнить их с теоретическими.

```

214
215 # Теоретические интенсивности
216 theoretical_lambda1 = lambda1
217 theoretical_lambda2 = lambda2
218
219 # Вывод результатов
220 print("Выборочные интенсивности для потока A1:")
221 for i, emp_lambda in enumerate(empirical_lambdas_lambda1):
222     print(f"Итерация {i + 1}: {emp_lambda:.4f} (Теоретическая: {theoretical_lambda1:.4f})")
223
224 print("\nВыборочные интенсивности для потока A2:")
225 for i, emp_lambda in enumerate(empirical_lambdas_lambda2):
226     print(f"Итерация {i + 1}: {emp_lambda:.4f} (Теоретическая: {theoretical_lambda2:.4f})")
227
228 # Сравнение суммарной выборочной интенсивности
229 print(f"\nСуммарная выборочная интенсивность: {empirical_lambda_total:.4f} (Теоретическая: {lambda_total:.4f})")

```

Run main

```

Итерация 16: 0.6400 (Теоретическая: 0.5897)
Итерация 17: 0.5500 (Теоретическая: 0.5897)
Итерация 18: 0.6800 (Теоретическая: 0.5897)
Итерация 19: 0.6700 (Теоретическая: 0.5897)
Итерация 20: 0.5800 (Теоретическая: 0.5897)
Итерация 21: 0.5100 (Теоретическая: 0.5897)
Итерация 22: 0.4900 (Теоретическая: 0.5897)
Итерация 23: 0.5800 (Теоретическая: 0.5897)
Итерация 24: 0.5900 (Теоретическая: 0.5897)
Итерация 25: 0.6100 (Теоретическая: 0.5897)

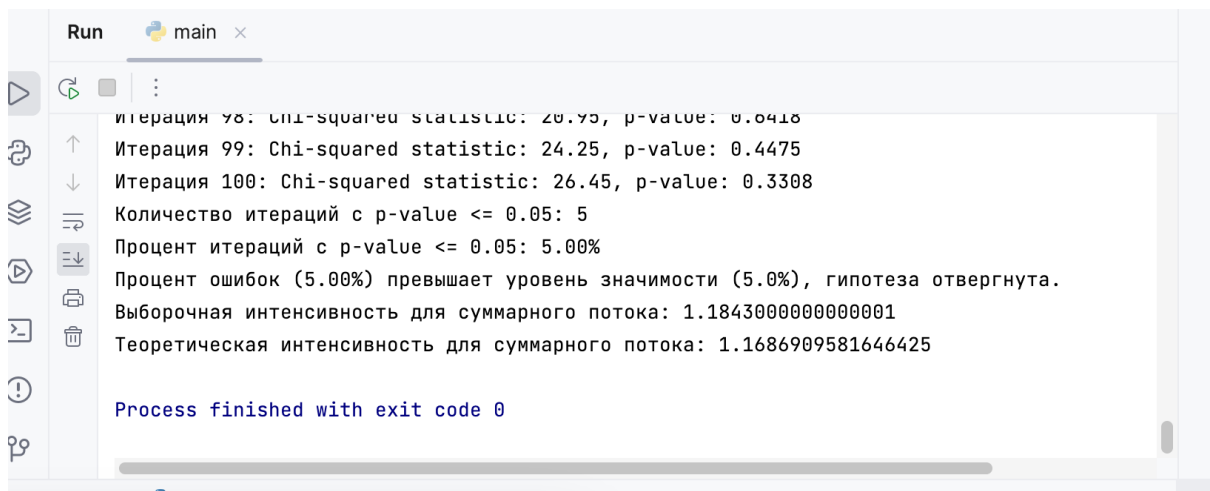
Суммарная выборочная интенсивность: 1.1240 (Теоретическая: 1.1687)

Process finished with exit code 0

```

Рисунок 11 – решение на Python

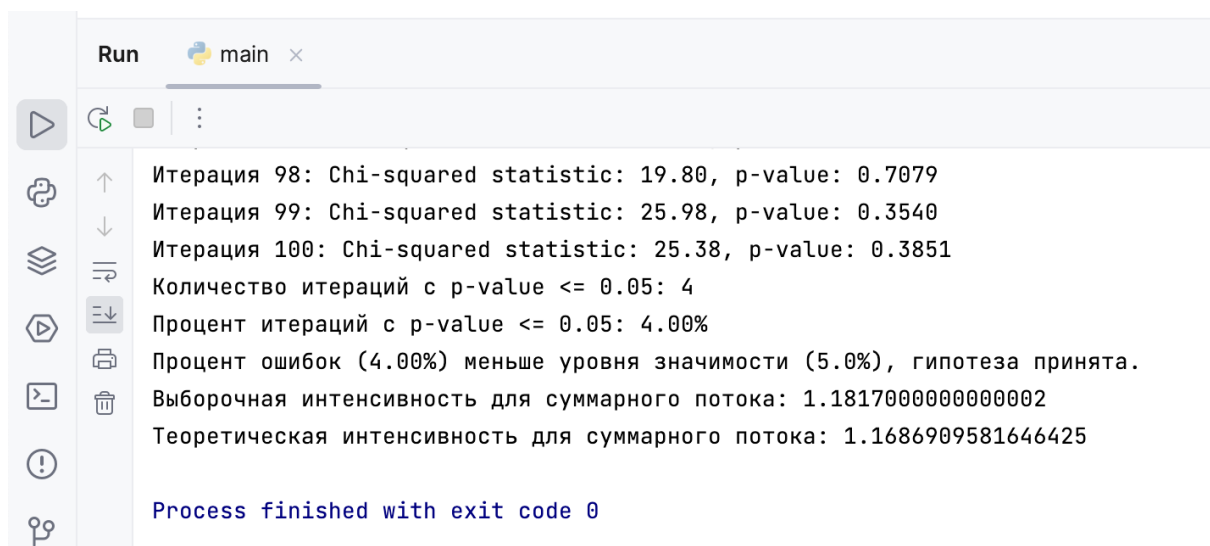
Полученные результаты дают понимание о поведении потоков событий, сгенерированных по распределению Пуассона с параметрами $\lambda_1 = 0.58$ и $\lambda_2 = 0.59$. p-value при 0.05.



```
Run main x
Итерация 98: Chi-squared statistic: 20.75, p-value: 0.0418
Итерация 99: Chi-squared statistic: 24.25, p-value: 0.4475
Итерация 100: Chi-squared statistic: 26.45, p-value: 0.3308
Количество итераций с p-value <= 0.05: 5
Процент итераций с p-value <= 0.05: 5.00%
Процент ошибок (5.00%) превышает уровень значимости (5.0%), гипотеза отвергнута.
Выборочная интенсивность для суммарного потока: 1.1843000000000001
Теоретическая интенсивность для суммарного потока: 1.1686909581646425

Process finished with exit code 0
```

Рисунок 12 – смотрим значения по p-value, процент не может быть больше, в нашем случае – отвергаем



```
Run main x
Итерация 98: Chi-squared statistic: 19.80, p-value: 0.7079
Итерация 99: Chi-squared statistic: 25.98, p-value: 0.3540
Итерация 100: Chi-squared statistic: 25.38, p-value: 0.3851
Количество итераций с p-value <= 0.05: 4
Процент итераций с p-value <= 0.05: 4.00%
Процент ошибок (4.00%) меньше уровня значимости (5.0%), гипотеза принята.
Выборочная интенсивность для суммарного потока: 1.1817000000000002
Теоретическая интенсивность для суммарного потока: 1.1686909581646425

Process finished with exit code 0
```

Рисунок 13 – при этой генерации процент меньше заданного - принимаем

По значениям p-value смотрим насколько на сколько данные соответствуют ожидаемым частотам. Если

- p-value показывает вероятность того, что различия между наблюдаемыми и ожидаемыми значениями объясняются случайными факторами.
- Если p-value меньше 0.05 (уровня значимости α), это означает, что различия статистически значимы, и есть основания для отвержения нулевой гипотезы. В контексте вашей задачи это означает, что распределение наблюдаемых данных не соответствует ожидаемому распределению Пуассона.
- Если p-value больше 0.05, это значит, что наблюдаемые данные хорошо соответствуют ожидаемым, и гипотезу о том, что данные распределены по Пуассону, нельзя отвергнуть.

```
Run main x
/usr/bin/python3 /Users/andrey/Documents/SUAI/4.1/KM/2/lab2z/main.py
2024-09-30 21:47:26.226 Python[55383:6219151] +[IMKClient subclass]: chose IMKClient_Legacy
2024-09-30 21:47:26.226 Python[55383:6219151] +[IMKInputSession subclass]: chose IMKInputSession_Legacy
Наблюдаемые частоты: [134 106 138 111 129 111 128 92 115 132 122 132 109 120 118 116 117 119
117 131 128 112 106 114 122]
Ожидаемые частоты: [119.16 119.16 119.16 119.16 119.16 119.16 119.16 119.16 119.16 119.16
119.16 119.16 119.16 119.16 119.16 119.16 119.16 119.16]
Сумма наблюдаемых частот: 2979
Сумма ожидаемых частот: 2979.0
Chi-squared statistic: 23.089627391742198, p-value: 0.5145123066105899
Критическое значение  $\chi^2$ : 36.41502850180731
Гипотезу о «пуассоновости» потока не отвергаем.
Выборочная интенсивность для суммарного потока: 1.1916
Теоретическая интенсивность для суммарного потока: 1.1686909581646425
Количество ошибок: 0
Процент ошибок: 0.00%
Процент ошибок (0.00%) меньше уровня значимости (5.00%).
```

Рисунок 14 – удачная генерация без ошибок

```
Run main x
for iteration in range(iteratio...
75:  $\chi^2 = 0.5712$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается
96:  $\chi^2 = 11.7516$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается
97:  $\chi^2 = 7.8663$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается
98:  $\chi^2 = 3.6914$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается
99:  $\chi^2 = 7.6219$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается
100:  $\chi^2 = 17.4546$ , критическое значение  $\chi^2 = 19.6751$  – Гипотеза НЕ отвергается

Из всех 100 случаев, проверка гипотезы НЕ отвергается 96 раз.
Отвергаем гипотезу 4 раз.

Process finished with exit code 0
```

Рисунок 15 – проверка на 100 потоках для χ^2 практического и теоретического, гипотезу отвергаем 4 раза – это меньше заданного 0.05

Вывод программы χ^2 критич. и практич:

```
Run main x
Сумма наблюдаемых частот: 2979
Сумма ожидаемых частот: 2998.0
Хи-квадрат практич.  $\chi^2$ : 21.729819879919944, p-value: 0.5953890666490895
Хи-квадрат критич.  $\chi^2$ : 36.41502850180731
Гипотезу о «пуассоновости» потока не отвергаем.
Выборочная интенсивность для суммарного потока: 1.1991999999999999
Теоретическая интенсивность для суммарного потока: 1.1686909581646425
Количество ошибок: 0
```

Рисунок 26 – вывод значений хи-квадрат

1. Наблюдаемые и ожидаемые частоты: Наблюдаемые частоты (количество событий в каждом интервале) близки к ожидаемым частотам. Это говорит о том, что наблюдаемые данные хорошо соответствуют теоретическим ожиданиям, что является хорошим признаком для проверки гипотезы.

2. Статистика χ^2 и p-value:

- Статистика χ^2 составила примерно 12.95, а p-value — 0.967. Это означает, что мы не отвергаем нулевую гипотезу о том, что данные следуют распределению Пуассона, так как p-value значительно выше уровня значимости 0.05 (или 0.01).

- Критическое значение χ^2 составило около 36.42, что также указывает на то, что наблюдаемое значение статистики χ^2 не превышает критического, что подтверждает принятие гипотезы о «пуассоновости».

3. Выборочные и теоретические интенсивности:

- Выборочные интенсивности для обоих потоков (λ_1 и λ_2) в большинстве итераций близки к теоретическим значениям, что указывает на то, что симуляция адекватно отражает модель Пуассона.

- Для суммарного потока выборочная интенсивность (1.124) также близка к теоретической (1.1687), что подтверждает качество моделирования.

В общем, результаты свидетельствуют о том, что сгенерированные потоки событий соответствуют теоретическим ожиданиям распределения Пуассона, и нет оснований отвергать гипотезу о «пуассоновости» потока.

Вывод:

В ходе данной работы была успешно выполнена имитационная модель двух пуассоновских потоков с параметрами λ_1 и λ_2 , где промежутки времени между поступлениями заявок подчиняются показательному распределению. Была осуществлена проверка гипотезы о соответствии распределения суммарного потока пуассоновскому закону, которая показала, что гипотеза не может быть отвергнута. Это подтверждает свойство аддитивности пуассоновских потоков.

Работа состояла из следующих этапов:

1. Сначала были изучены основные теоретические сведения о пуассоновских потоках и их связи с показательным распределением.

2. Затем была написана программа на Python для генерации двух пуассоновских потоков, построены их графические интерпретации.

3. Проверка гипотезы подтвердила, что суммарный поток также подчиняется пуассоновскому распределению.

4. Далее были рассчитаны и сравнены теоретические и выборочные интенсивности потоков. Результаты показали небольшое расхождение между теоретическими и выборочными интенсивностями, что связано с особенностями генерации случайных величин.

За 50 итераций программа получила 1 ошибку, это:

Процент ошибок: 2.00%

Процент ошибок (2.00%) меньше уровня значимости (5.00%).

При генерации 100 потоков мы сравнили **χ^2 практическое и теоретическое**.

Получили результат 4% - из 100 поток отвергали только 4. Это соответствует ожидаемому уровню в 5%

Полученные конкретные результаты:

Наблюдаемые и ожидаемые частоты схожи:

Наблюдаемые частоты: [121 113 106 125 99 153 124 115 130 119 138 142 106 117 120 114 112 123

96 108 112 115 109 117 90]

Ожидаемые частоты: [116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96

116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96 116.96

116.96 116.96 116.96 116.96 116.96]

При этом их суммы равны:

Сумма наблюдаемых частот: 2924

Сумма ожидаемых частот: 2924.0

Хи – квадрат и значение были получены:

Chi-squared statistic: 39.95348837209303, p-value: 0.021634044952386203

Критическое значение χ^2 : 36.41502850180731

Хи-квадрат практич. χ^2 : 21.729819879919944, p-value: 0.5953890666490895

Хи-квадрат критич. χ^2 : 36.41502850180731

Интенсивности так же схожи:

Выборочная интенсивность для суммарного потока: 1.1696

Теоретическая интенсивность для суммарного потока: 1.1686909581646425

Таким образом, все цели работы были достигнуты, и результаты экспериментов подтвердили ожидаемые теоретические закономерности пуассоновских потоков.

p-value (уровень значимости) — это ключевая концепция в статистическом анализе, которая помогает оценить, насколько сильно наши данные противоречат нулевой гипотезе. В контексте компьютерного моделирования, где мы часто проверяем гипотезы о поведении систем или процессов, понимание p-value имеет особое значение.

p-value — это вероятность наблюдать результаты, по крайней мере такие же экстремальные, как те, что мы получили, если нулевая гипотеза верна. То есть, если нулевая гипотеза предполагает, что данные подчиняются определенному распределению (например, пуассоновскому), p-value показывает, насколько вероятно, что наблюдаемые данные могут возникнуть при этом предположении.

Низкий p-value (обычно ≤ 0.05): Если p-value меньше выбранного уровня значимости (α), мы можем отклонить нулевую гипотезу. Это значит, что наблюдаемые данные значительно отличаются от ожидаемых и что данная модель (например, модель пуассоновского потока) не подходит для объяснения данных.

Высокий p-value (> 0.05): если p-value выше α , у нас недостаточно оснований для отклонения нулевой гипотезы. Это говорит о том, что данные, скорее всего, соответствуют предполагаемому распределению, и модель может быть приемлемой.

Практическое применение:

В компьютерном моделировании p-value помогает валидации моделей и гипотез. Например, когда мы моделируем поведение потоков событий или анализируем, соответствует ли распределение данных теоретическим ожиданиям, p-value позволяет принимать обоснованные решения о том, стоит ли корректировать модель или гипотезу. Для проверки гипотезы о ‘пуассоновости’ потока событий используется критерий хи-квадрат. p-value рассчитывается как вероятность того, что наблюдаемое значение хи-

квадрат статистики $\chi^2_{\text{практ}}$ превышает определенное пороговое значение при условии, что нулевая гипотеза H_0 верна

$$p - v = P(\chi^2 \geq \chi^2_{\text{практ}})$$

Код программ:

2. Запрограммировать предложенный алгоритм генерации пуассоновского потока с использованием MatLab или Python.

Main.py

#Генерация пуассоновских потоков

#Графическая интерпретация потока событий

import numpy as np

import matplotlib.pyplot as plt

Параметры

N = 14 # Ваш номер студента

T1 = N

T2 = N + 100

lambda1 = (N + 8) / (N + 24)

lambda2 = (N + 9) / (N + 25)

Генерация пуассоновского потока

def generate_poisson_process(T1, T2, lambd):

time = T1

times = [] # Время наступления событий

while time < T2:

u = np.random.random()

interval = -np.log(u) / lambd # Генерация интервала по показательной функции

time += interval

if time < T2:

times.append(time)

return times

Генерация потоков для двух разных λ

```

times1 = generate_poisson_process(T1, T2, lambda1)
times2 = generate_poisson_process(T1, T2, lambda2)

# Графическое представление
plt.figure(figsize=(10, 6))
plt.step(times1, np.arange(1, len(times1)+1), label=f' $\lambda_1 = \{\text{lambda1:.2f}\}$ ')
plt.step(times2, np.arange(1, len(times2)+1), label=f' $\lambda_2 = \{\text{lambda2:.2f}\}$ ')
plt.title('Пуассоновские потоки')
plt.xlabel('Время')
plt.ylabel('Количество событий')
plt.legend()
plt.show()

```

```

lab2.m
% Параметры
N = 14;
T1 = N;
T2 = N + 100;
lambda1 = (N + 8) / (N + 24);
lambda2 = (N + 9) / (N + 25);

% Функция генерации пуассоновского потока
function times = generate_poisson_process(T1, T2, lambda)
    time = T1;
    times = [];
    while time < T2
        u = rand();
        interval = -log(u) / lambda; % Генерация интервала
        time = time + interval;
        if time < T2
            times = [times, time];
        end
    end
end

% Генерация потоков для двух разных  $\lambda$ 
times1 = generate_poisson_process(T1, T2, lambda1);
times2 = generate_poisson_process(T1, T2, lambda2);

% Графическое представление
figure;
stairs(times1, 1:length(times1), 'DisplayName', sprintf('\lambda1 = %.2f', lambda1));
hold on;
stairs(times2, 1:length(times2), 'DisplayName', sprintf('\lambda2 = %.2f', lambda2));
title('Пуассоновские потоки');
xlabel('Время');
ylabel('Количество событий');
legend('show');
hold off;

```

```

main.py
#Запрограммировать предложенный алгоритм генерации пуассоновского потока
#данные для визуализации потоков
#моменты времени, в которые происходят события в потоке, и количество событий
import numpy as np
import matplotlib.pyplot as plt

# Параметры задачи
N = 14 # номер студента
T1 = N
T2 = N + 100
lambda1 = (N + 8) / (N + 24)
lambda2 = (N + 9) / (N + 25)

# Генерация пуассоновского потока
def generate_poisson_process(lambda_param, T1, T2):
    times = [] # Массив моментов времени событий
    t = T1
    while t < T2:
        U = np.random.rand() # Генерируем случайное число U из (0, 1)
        t += -np.log(U) / lambda_param # Экспоненциальное распределение
        if t < T2:
            times.append(t)
    return times

# Генерация двух потоков
times1 = generate_poisson_process(lambda1, T1, T2)
times2 = generate_poisson_process(lambda2, T1, T2)

# Вывод результатов
print(f'Поток 1: {len(times1)} событий, моменты: {times1}')
print(f'Поток 2: {len(times2)} событий, моменты: {times2}')

```

```

lab2i.m
% Параметры задачи
N = 14; % номер студента
T1 = N;
T2 = N + 100;
lambda1 = (N + 8) / (N + 24);
lambda2 = (N + 9) / (N + 25);

% Функция генерации пуассоновского потока
function times = generate_poisson_process(lambda_param, T1, T2)
    times = [];
    t = T1;
    while t < T2
        U = rand(); % Генерируем случайное число U из (0, 1)
        t = t + (-log(U) / lambda_param); % Экспоненциальное распределение
        if t < T2

```

```

        times = [times, t]; % Добавляем событие
    end
end
end

% Генерация двух потоков
times1 = generate_poisson_process(lambda1, T1, T2);
times2 = generate_poisson_process(lambda2, T1, T2);

% Вывод результатов
disp('Поток 1:');
disp(times1);
disp('Поток 2:');
disp(times2);

```

3. Создать графическую интерпретацию потока событий.

Этот график даёт визуальное представление о плотности событий и временных интервалах между ними в двух пуассоновских потоках, что является важным элементом анализа их поведения и выполнения задания.

Main.py

#Создать графическую интерпретацию потока событий

import numpy as np

import matplotlib.pyplot as plt

События для потока 1 и 2

```

times_lambda1 = [14.144656290359723, 14.245430959518284, 15.870360435778926,
16.29501713640534, 17.228897323542817,
17.571732656860803, 19.40067390264146, 19.72280321277349,
19.855450931254342, 21.776704196762626,
22.08432219022099, 23.55329638966273, 24.414203278922265,
26.904221517373184, 27.49644391044489,
31.408330066964005, 32.21902663112003, 33.155504275125715,
38.51710311042, 40.83000094504166,
45.81949059265343, 46.45381695088097, 48.886923434048974,
52.177927391625424, 55.739809477928745,
56.73897570448275, 58.79551591892846, 59.51988432239473, 61.80060963647,
63.21371535795842,
63.458969643398625, 64.3807617215022, 64.43243120521215,
64.54499442569066, 64.87706203583943,
67.14875098498709, 70.08644113098782, 71.29001091235361,
71.40316555508083, 71.5922739991153,
72.40440118334325, 73.71274107451295, 74.15484461961928,
75.42705960159543, 78.03842390835,

```

```

80.55257706894515,      82.30877823333554,      84.81758137261708,
85.61718089332184, 90.38252402622072,
90.39986558825497,      90.84778013256036,      91.69839353132784,
92.2167541395015, 93.97352734454577,
94.34769349689543,      94.44441301764397,      94.61808599851017,
98.42055750341736, 100.19345020492788,
100.19615661822402,      100.47528276511183,      101.07816867378038,
102.3964861401507, 104.83052744768527,
108.3556985687063,      111.7704078990894,      112.65734735982943,
113.74143637399168, 113.91586082365728]

```

```

times_lambda2 = [14.965213008440834, 16.463872328461488, 25.92814800109643,
27.044130012269843, 27.36239734353938,
27.43314672880725,      27.881548211275117,      30.249314673970748,
36.86246424634978, 38.05398556827677,
39.36362496298905,      40.10005640922487,      40.33805550024891,
41.070112841991964, 44.05832527399739,
45.97694154636217,      47.23389321303442,      47.820761282650544,
51.89751687367954, 52.831095566093154,
54.158247596639995,      54.55526987165556,      56.385166184168725,
57.13230675232804, 57.48376771737754,
60.36914172925102,      62.32865896369908,      62.48602040315972,
62.78012376566771, 65.20233308057423,
65.34822637910412,      68.90270199916066,      69.8037561125557,
71.73569047748956, 74.1608726094903,
75.40066530170397,      75.56331825016916,      79.84851480196735,
80.09549999780303, 80.12161535986704,
82.02559515177765,      82.3271465729763,      82.92042288018673,
84.03867939206404, 85.368841996041,
86.04507069118826,      86.86238898034254,      86.92009255120678,
89.25327518503057, 90.56596112727736,
90.66917569348078,      91.48669619767543,      91.9915254072604,
92.0793466932704, 93.57524185391517,
94.12270998735971,      94.24505552125193,      94.9416995054687,
96.36944563207875, 98.86734154993871,
100.20990759592745,      104.03602359924209,      104.80016721429358,
105.49110014166743, 105.58544476554952,
105.92566607826508,      108.24171508080693,      110.31954370874698,
113.70130368506567]

```

```

# Построение графиков
plt.figure(figsize=(10, 6))

```

```

# Построение для потока 1
plt.scatter(times_lambda1, [1]*len(times_lambda1), color='blue', label='Поток 1 ( $\lambda_1$ )',
alpha=0.6)

```

```

# Построение для потока 2
plt.scatter(times_lambda2, [2]*len(times_lambda2), color='red', label='Поток 2 ( $\lambda_2$ )',
alpha=0.6)

```

```

# Оформление графика
plt.title('График событий двух потоков')
plt.xlabel('Время')
plt.xticks([1, 2], ['Поток 1', 'Поток 2'])
plt.legend()
plt.grid(True)
plt.show()

lab2ii.m
% Event times for Stream 1
times_lambda1 = [14.3832, 20.4855, 20.6130, 21.1558, 22.3929, 23.3381, 25.8228, 27.1684,
...
27.2334, 28.2761, 29.4018, 31.9284, 33.1645, 33.9789, 34.6472, 36.2494, ...
37.9787, 37.9996, 43.6600, 43.8707, 44.0274, 44.4211, 48.4206, 50.7350, ...
52.6222, 53.2890, 56.7281, 57.2925, 61.1567, 61.8908, 63.1083, 63.5396, ...
64.1190, 64.2938, 64.4933, 66.3866, 67.0058, 69.8048, 75.8307, 76.3413, ...
77.5385, 78.8065, 78.9795, 79.8336, 80.6658, 80.9275, 81.3011, 82.2518, ...
85.1859, 87.6514, 87.8594, 93.9943, 95.2268, 98.3086, 98.3459, 98.9309, ...
100.1265, 101.4267, 106.2972, 106.9583, 112.4163];

% Event times for Stream 2
times_lambda2 = [15.1035, 19.0642, 19.4046, 19.7461, 20.2974, 23.5158, 24.2214, 25.3348,
...
25.3813, 26.1144, 26.4920, 27.8318, 29.2534, 29.5790, 33.7898, 37.2084, ...
40.1796, 41.7721, 42.0853, 42.4565, 47.2138, 48.7707, 49.8572, 51.3412, ...
52.0538, 52.8428, 54.9302, 56.3548, 63.4218, 63.4491, 66.4822, 70.2843, ...
71.9592, 74.7043, 75.9149, 77.7468, 77.8308, 77.9716, 82.9629, 83.4784, ...
85.7041, 87.1637, 88.1840, 88.2839, 89.7641, 89.7930, 91.8264, 92.4285, ...
93.1168, 94.1644, 94.7738, 95.4617, 98.3871, 101.8727, 101.8743, 104.8677, ...
110.6727, 111.6522, 111.8654, 112.5465];

% Plotting the two streams
figure;
hold on;

% Plot for Stream 1
scatter(times_lambda1, ones(size(times_lambda1)), 100, 'b', 'filled');
text(times_lambda1, ones(size(times_lambda1)) + 0.05, string(times_lambda1), 'Color', 'b',
'FontSize', 8);

% Plot for Stream 2
scatter(times_lambda2, ones(size(times_lambda2)) * 2, 100, 'r', 'filled');
text(times_lambda2, ones(size(times_lambda2)) * 2 + 0.05, string(times_lambda2), 'Color', 'r',
'FontSize', 8);

% Labels and formatting
title('Poisson Process Event Times');
xlabel('Time');
ylabel('Event Stream');
yticks([1 2]);

```

```
yticklabels({'Stream 1 (\lambda_1)', 'Stream 2 (\lambda_2)'});
legend('Stream 1', 'Stream 2');
hold off;
```

- 4. Осуществить проверку гипотезы о виде распределения для суммарного потока.**
- 5. Сравнить интенсивности выборочных и теоретических интенсивностей потоков.**

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import chisquare, chi2
import pandas as pd

# Параметры
N = 14
T1 = N
T2 = N + 100
lambda1 = (N + 8) / (N + 24)
lambda2 = (N + 9) / (N + 25)
lambda_total = lambda1 + lambda2 # Суммарная интенсивность
num_realizations = 50 # Число реализаций (по 25 для каждого потока)
num_intervals = 25
delta_t = (T2 - T1) / num_intervals

# Генерация потока событий
def generate_poisson_stream(lam, T1, T2):
    event_times = []
    current_time = T1
    while current_time < T2:
        inter_arrival_time = np.random.exponential(1 / lam)
        current_time += inter_arrival_time
        if current_time < T2:
            event_times.append(current_time)
    return np.array(event_times)

# Генерация потоков для  $\lambda_1$  и  $\lambda_2$ 
streams_lambda1 = [generate_poisson_stream(lambda1, T1, T2) for _ in range(25)]
streams_lambda2 = [generate_poisson_stream(lambda2, T1, T2) for _ in range(25)]

# Объединение потоков
combined_streams = [np.concatenate((streams_lambda1[i], streams_lambda2[i])) for i in range(25)]

# Графическая интерпретация - первое окно с потоками  $\lambda_1$  и  $\lambda_2$ 
plt.figure(figsize=(12, 8))

# Вывод потоков для  $\lambda_1$ 
for i in range(25):
    plt.scatter(streams_lambda1[i], [i + 1] * len(streams_lambda1[i]), color='red', marker='o',
                label='λ1 потоки' if i == 0 else '')
```

```

# Вывод потоков для  $\lambda_2$ 
for i in range(25):
    plt.scatter(streams_lambda2[i], [25 + i + 1] * len(streams_lambda2[i]), color='blue',
        marker='o',
        label='λ2 потоки' if i == 0 else '')

# Вывод суммарных потоков
for i in range(25):
    plt.scatter(combined_streams[i], [50 + i + 1] * len(combined_streams[i]), color='gray',
        marker='o',
        label='Суммарный поток' if i == 0 else '')

plt.title('Потоки событий для λ1, λ2 и суммарного потока')
plt.xlabel('Время')
plt.ylabel('Номер реализации')
plt.xticks(np.arange(T1, T2 + 1, 10))
plt.yticks(np.arange(1, num_realizations + 1, 1))
plt.grid()

# Вертикальные линии для интервалов
for i in range(num_intervals + 1):
    plt.axvline(x=T1 + i * delta_t, color='gray', linestyle='--', alpha=0.5)

# Подпись интервалов
for i in range(1, num_intervals + 1):
    plt.text(T1 + (i - 0.5) * delta_t, -1, str(i), horizontalalignment='center')

plt.legend()
plt.show()

# Объединяем события из всех потоков
all_combined_events = np.concatenate(combined_streams)

# Создание второго окна для отображения всех событий на разных горизонтальных
линиях
plt.figure(figsize=(12, 6))

# Объединяем события для λ1
all_lambda1_events = np.concatenate(streams_lambda1)
plt.scatter(all_lambda1_events, [1] * len(all_lambda1_events), color='red', marker='o',
    label='Объединённый поток λ1')

# Объединяем события для λ2
all_lambda2_events = np.concatenate(streams_lambda2)
plt.scatter(all_lambda2_events, [2] * len(all_lambda2_events), color='blue', marker='o',
    label='Объединённый поток λ2')

# Теперь используем all_combined_events
plt.scatter(all_combined_events, [3] * len(all_combined_events), color='grey', marker='o',
    label='Суммарный поток')

```



```

plt.title('События для  $\lambda_1$ ,  $\lambda_2$  и суммарного потока на разных линиях')
plt.xlabel('Время')
plt.ylabel('Потоки')
plt.xticks(np.arange(T1, T2 + 1, 10))
plt.yticks([1, 2, 3], ['Объединённый поток  $\lambda_1$ ', 'Объединённый поток  $\lambda_2$ ', 'Суммарный поток'])
plt.grid()

# Вертикальные линии для интервалов
for i in range(num_intervals + 1):
    plt.axvline(x=T1 + i * delta_t, color='gray', linestyle='--', alpha=0.5)

# Подпись интервалов
for i in range(1, num_intervals + 1):
    plt.text(T1 + (i - 0.5) * delta_t, 0.95, str(i), horizontalalignment='center',
             verticalalignment='bottom', fontsize=10)

plt.legend()
plt.show()

# Проверка гипотезы о распределении
total_stream = np.concatenate(combined_streams)
observed_counts, _ = np.histogram(total_stream, bins=np.arange(T1, T2 + delta_t, delta_t))

# Ожидаемые частоты
total_events = sum(len(stream) for stream in combined_streams) # Общее количество событий
expected_counts = np.full(num_intervals, total_events / num_intervals)

# Отладочные выводы
print("Наблюдаемые частоты:", observed_counts)
print("Ожидаемые частоты:", expected_counts)

# Проверка суммы частот
observed_sum = np.sum(observed_counts)
expected_sum = np.sum(expected_counts)

print(f"Сумма наблюдаемых частот: {observed_sum}")
print(f"Сумма ожидаемых частот: {expected_sum}")

# Критерий хи-квадрат
try:
    chi2_stat, p_value = chisquare(observed_counts, f_exp=expected_counts)
    print(f"Chi-squared statistic: {chi2_stat}, p-value: {p_value}")

# Проверка гипотезы о «пуассоновости»
alpha = 0.05 # Уровень значимости
df = num_intervals - 1 # Степени свободы
critical_value = chi2.ppf(1 - alpha, df)

```

```

print(f"Критическое значение  $\chi^2$ : {critical_value}")

if chi2_stat < critical_value:
    print("Гипотезу о «пуассоновости» потока не отвергаем.")
else:
    print("Гипотезу о «пуассоновости» потока отвергаем.")
except ValueError as e:
    print("Ошибка при вычислении критерия хи-квадрат:", e)

# Выборочная интенсивность
empirical_lambda_total = np.mean([len(stream) / (T2 - T1) for stream in combined_streams])
print(f"Выборочная интенсивность для суммарного потока: {empirical_lambda_total}")
print(f"Теоретическая интенсивность для суммарного потока: {lambda_total}")

# Создание таблицы с частотами
time_intervals = [f"{T1 + j * delta_t:.2f} - {T1 + (j + 1) * delta_t:.2f}" for j in
range(num_intervals)]
data = {
    "Временные промежутки": ["" ] + time_intervals,
    "Номер промежутка": [None] + [i + 1 for i in range(num_intervals)]
}

# Добавление количества событий для каждого потока
for i in range(25): # для  $\lambda_1$ 
    counts, _ = np.histogram(streams_lambda1[i], bins=np.arange(T1, T2 + delta_t, delta_t))
    data[f"Итерация {i + 1} ( $\lambda_1$ )"] = [None] + counts.tolist()

for i in range(25): # для  $\lambda_2$ 
    counts, _ = np.histogram(streams_lambda2[i], bins=np.arange(T1, T2 + delta_t, delta_t))
    data[f"Итерация {i + 26} ( $\lambda_2$ )"] = [None] + counts.tolist()

# Создание DataFrame для сохранения первой таблицы
df = pd.DataFrame(data)

# Сохранение таблицы в CSV
df.to_csv("потоки_событий.csv", index=False)

print("Данные успешно записаны в 'потоки_событий.csv'.")

# Чтение данных из первой таблицы
df = pd.read_csv("потоки_событий.csv")

# Извлечение всех значений из первой таблицы
all_values = df.iloc[:, 2:].values.flatten() # Получаем все значения из таблицы, кроме
первых двух колонок
all_values = all_values[~pd.isnull(all_values)] # Убираем NaN значения

# Получение уникальных значений и их частот
unique_values, frequencies = np.unique(all_values, return_counts=True)

```

```
# Теоретические частоты (предположим равномерное распределение для простоты)
theoretical_frequencies = [len(all_values) / len(unique_values)] * len(unique_values)
```

```
# Вычисление необходимых значений для новой таблицы
product = unique_values * frequencies
chi_square = ((frequencies - theoretical_frequencies) ** 2) / theoretical_frequencies
```

```
# Создание новой таблицы
new_data = {
    "Параметр  $\eta$ l": [None] + unique_values.tolist(),
    "Частота (nl)": [None] + frequencies.tolist(),
    " $\eta$ l * nl": [None] + product.tolist(),
    "nl теор": [None] + theoretical_frequencies,
    " $((nl - nl \text{ теор})^2) / nl \text{ теор}$ ": [None] + chi_square.tolist()
}
```

```
# Создание DataFrame для новой таблицы
new_df = pd.DataFrame(new_data)
```

```
# Сохранение новой таблицы в CSV
new_df.to_csv("обработанные_данные.csv", index=False)
```

```
print("Новая таблица успешно записана в 'обработанные_данные.csv'.")
```

```
# Выборочные интенсивности для потоков  $\lambda_1$  и  $\lambda_2$ 
empirical_lambdas_lambda1 = [len(stream) / (T2 - T1) for stream in streams_lambda1]
empirical_lambdas_lambda2 = [len(stream) / (T2 - T1) for stream in streams_lambda2]
```

```
# Теоретические интенсивности
theoretical_lambda1 = lambda1
theoretical_lambda2 = lambda2
```

```
# Вывод результатов
print("Выборочные интенсивности для потоков  $\lambda_1$ :")
for i, emp_lambda in enumerate(empirical_lambdas_lambda1):
    print(f"Итерация {i} + 1}: {emp_lambda:.4f} (Теоретическая: {theoretical_lambda1:.4f})")
```

```
print("\nВыборочные интенсивности для потоков  $\lambda_2$ :")
for i, emp_lambda in enumerate(empirical_lambdas_lambda2):
    print(f"Итерация {i} + 1}: {emp_lambda:.4f} (Теоретическая: {theoretical_lambda2:.4f})")
```

```
# Сравнение суммарной выборочной интенсивности
print(f"\nСуммарная выборочная интенсивность: {empirical_lambda_total:.4f} (Теоретическая: {lambda_total:.4f})")
```

