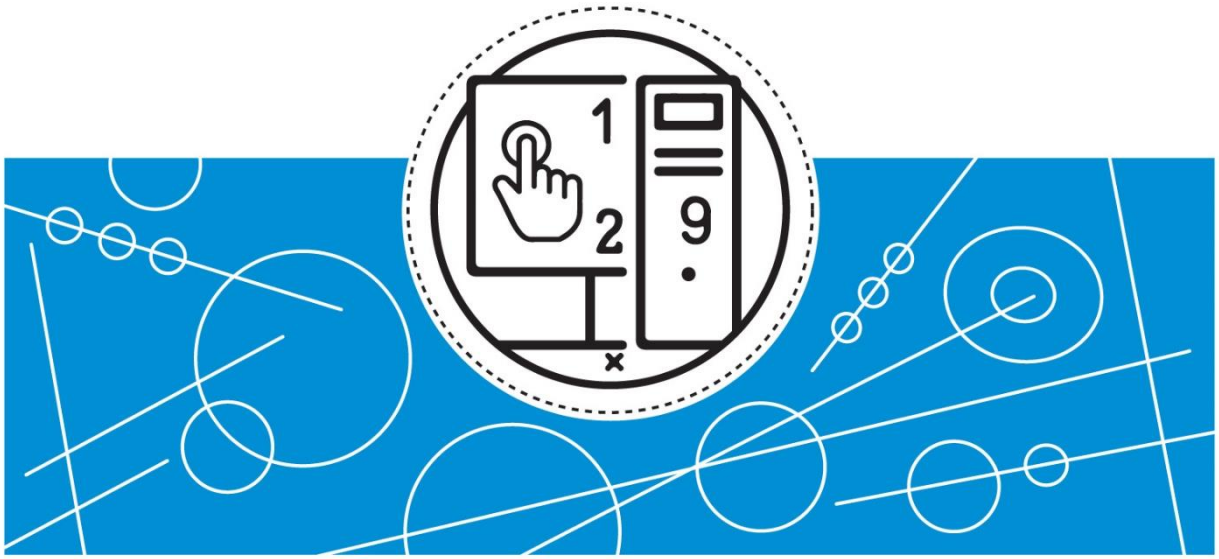
ИТМО

И. С. СМИРНОВ, М. В. ЛУКИНА,
Т. Е. МИТИНА

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В ЗАДАЧАХ ТЕОРИИ ВЕРОЯТНОСТЕЙ



Санкт-Петербург
2026

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

И.С. Смирнов, М.В. Лукина, Т.Е. Митина

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ В ЗАДАЧАХ ТЕОРИИ ВЕРОЯТНОСТЕЙ

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

РЕКОМЕНДОВАНО К ИСПОЛЬЗОВАНИЮ В УНИВЕРСИТЕТЕ ИТМО
по направлению подготовки 09.03.03, 09.03.04, 11.03.02, 12.03.01, 12.03.03, 12.03.04, 15.03.06, 18.03.02, 27.03.04, 27.03.05 в качестве учебно-методического пособия для реализации основных профессиональных образовательных программ высшего образования бакалавриата

Санкт-Петербург

2026

Смирнов И.С., Лукина М.В., Митина Т.Е., Компьютерное моделирование в задачах теории вероятностей – СПб: Университет ИТМО, 2026. – 65 с.

Рецензент:

Бойцев Антон Александрович, кандидат физико-математических наук, доцент (квалификационная категория "ординарный доцент") института математики, Университета ИТМО.

Учебно-методическое пособие предназначено для организации работы обучающихся по дисциплине «Теория вероятностей». Пособие включает две расчётно-графические работы, объединяющие восемь лабораторных работ, и направлено на формирование навыков применения методов теории вероятностей с использованием вычислительного эксперимента.

ИТМО (Санкт-Петербург) – национальный исследовательский университет, научно-образовательная корпорация. Альма-матер победителей международных соревнований по программированию. Приоритетные направления: IT и искусственный интеллект, фотоника, робототехника, квантовые коммуникации, трансляционная медицина, Life Sciences, Art&Science, Science Communication.

Лидер федеральной программы «Приоритет-2030», в рамках которой реализовывается программа «Университет открытого кода». С 2022 ИТМО работает в рамках новой модели развития – научно-образовательной корпорации. В ее основе академическая свобода, поддержка начинаний студентов и сотрудников, распределенная система управления, приверженность открытому коду, бизнес-подходы к организации работы. Образование в университете основано на выборе индивидуальной траектории для каждого студента.

ИТМО пять лет подряд – в сотне лучших в области Automation & Control (кибернетика) Шанхайского рейтинга. По версии SuperJob занимает первое место в Петербурге и второе в России по уровню зарплат выпускников в сфере IT. Университет в топе международных рейтингов среди российских вузов. Входит в топ-5 российских университетов по качеству приема на бюджетные места. Рекордсмен по поступлению олимпиадников в Петербурге. С 2019 года ИТМО самостоятельно присуждает ученые степени кандидата и доктора наук.

Оглавление

Введение

Классический курс теории вероятностей традиционно сопровождается типовым расчётом: набором задач на вычисление вероятностей в различных схемах – от бросания костей и монет, до анализа выборок изделий из партий. Такая работа хорошо тренирует владение формулами, но в основном опирается на «бумажный» подход к вероятностям: студент подставляет числа в готовые формулы и получает ответ.

Современная прикладная математика, анализ данных и машинное обучение делают акцент на **вычислительном эксперименте**. Практикующему специалисту важно уметь не только вывести формулу, но и реализовать модель, провести серию симуляций, визуализировать результаты и интерпретировать их, учитывая случайные отклонения и погрешности.

В методических указаниях предлагается несколько **лабораторных работ**, разбитых на две расчётно-графические работы (РГР):

- Расчётно-графическая работа №1 – лабораторные работы, посвящённые базовым моделям случайных испытаний, закону больших чисел, условным вероятностям и простейшим задачам геометрической вероятности.
- Расчётно-графическая работа №2 – лабораторные работы, направленные на моделирование распределений, иллюстрацию центральной предельной теоремы, работу с совместными распределениями и зависимостями.

Важный акцент пособия – сопоставление теоретических результатов с данными, полученными в вычислительном эксперименте. В каждой работе требуется планировать серию испытаний, получать эмпирические оценки и представлять их в наглядной форме. Отдельное внимание уделяется расхождениям с теорией, связанным со случайными колебаниями и конечным объёмом выборки.

Каждая лабораторная работа строится вокруг моделирования реальной вероятностной схемы и содержит:

- цель работы – что именно нужно понять и уметь по завершении выполнения;
- задание (условие) – описание моделируемой вероятностной схемы и требуемых результатов;
- формирование варианта – правила соответствия числовых параметров задачи номеру варианта;
- теоретическую справку – определения, формулы и теоремы, необходимые для выполнения задания;
- описание вычислительного эксперимента – рекомендации по выбору числа испытаний, построению таблиц и графиков, формулировке выводов;
- программную реализацию и визуализацию в Python – краткие примеры кода и рекомендации по моделированию и обработке результатов;
- примеры оформления результатов – требования к рисункам, таблицам и пояснениям в отчёте.

Организация выполнения работ

В ходе выполнения каждой лабораторной работы предлагается действовать по следующему алгоритму:

1. Ознакомиться с целью работы и краткой теоретической справкой. Убедиться, что понятны используемые обозначения и формулы.
2. Определить номер варианта согласно правилам распределения вариантов (см. следующий раздел «Распределение вариантов») и подставить его во все формулы параметров задачи.
3. Спланировать вычислительный эксперимент: выбрать несколько значений числа испытаний (малое, среднее и большое), продумать, какие величины будут измеряться и в какой форме они будут представлены в отчёте.
4. Реализовать моделирование, следуя указаниям в описании лабораторной работы.
5. Построить необходимые графики и таблицы, сравнить эмпирические результаты с теоретическими значениями.
6. Сделать выводы, кратко ответив на вопросы: как ведут себя случайные величины при увеличении числа испытаний, насколько хорошо совпадает теория с результатами моделирования, как на них влияют параметры варианта.

Рекомендуемая структура отчёта по каждой лабораторной работе:

- титульный лист, оформленный по требованиям преподавателя;
- цель работы;
- исходные данные и номер варианта;
- правило формирования параметров по варианту;
- краткая теоретическая справка – основные формулы и определения;
- описание алгоритма моделирования и использованных параметров;
- фрагменты кода – по необходимости;
- результаты моделирования: таблицы, графики;
- выводы.

Распределение вариантов

Для индивидуализации заданий в работах используется параметр «вариант» ($variant$). Конкретное правило распределения вариантов задаёт преподаватель.

В дальнейшем номер варианта обозначается через v. Все параметры задач (число испытаний, вероятности, размеры геометрических объектов и т. п.) задаются **именно через эту величину** v. Зная свой номер варианта v, необходимо подставить его в указанные формулы, не изменяя структуру задания.

Используемые инструменты

В качестве основного инструмента моделирования в работах рекомендуется использовать язык **Python**. Удобно применять его в интерактивных вычислительных средах, например в Jupyter Notebook, JupyterLab

или Google Colab. Эти среды позволяют в одном документе объединять текстовые пояснения и формулы, писать и запускать код, строить графики и гистограммы, а также сразу видеть результат и при необходимости корректировать ход эксперимента.

В расчётно-графических работах используются стандартные модули Python и распространённые библиотеки для обработки данных и визуализации. Для моделирования случайных испытаний применяется модуль *random*, обеспечивающий генерацию случайных чисел и выбор случайных элементов из заданных множеств. Для вычисления математических выражений (корни, тригонометрические функции, число π и т. д.) используется модуль *math*. Для подсчёта частот появления значений удобно применять модуль *collections*, в первую очередь класс *Counter*, позволяющий автоматически формировать таблицы частот. Построение графиков, траекторий и гистограмм выполняется с помощью библиотеки *matplotlib*. При необходимости в задачах, связанных с обработкой больших выборок и векторными вычислениями, дополнительно используется библиотека *numpy*.

РГР 1. Лабораторные работы 1-4

Первый блок включает четыре лабораторные работы, посвящённые базовым вероятностным моделям и закону больших чисел. В этих заданиях студент:

- знакомится с простейшими схемами случайных испытаний (монета, кубик, схема Бернулли);
- учится моделировать условные вероятности и применять формулу Байеса;
- осваивает работу с дискретными распределениями (биномиальное, пуассоновское) и гистограммами;
- знакомится с геометрической вероятностью и методом Монте-Карло для оценки числа π.

Во всех работах акцент делается на сходимости частот к теоретическим вероятностям и на влиянии числа испытаний на качество аппроксимации. Отдельное внимание в заданиях уделяется выбору нескольких уровней числа экспериментов (малое, среднее, большое N), чтобы можно было увидеть, как меняются результаты и насколько быстро «успокаивается» случайный разброс.

Лабораторная работа №1. Монетка или кубик

Цель работы. На примере подбрасывания монеты или игрального кубика проиллюстрировать закон больших чисел и влияние числа испытаний на сходимость относительной частоты к теоретической вероятности.

Условие. Смоделировать N независимых экспериментов.

- Если вариант **чётный** – бросается монета. Результат одного испытания:

$$X = \begin{cases} 1, & \text{«орёл»;} \\ 0, & \text{«решка»,} \end{cases} \qquad P(X=1) = P(X=0) = \frac{1}{2}$$

  Событие «успех» – «выпал орёл».

- Если вариант **нечётный** – бросается игральный кубик. Результат одного испытания – число очков $k \in \{1; 2; 3; 4; 5; 6\}$, причём

$$P(X=k) = \frac{1}{6}, \qquad k = 1; \dots; 6.$$

  Событие «успех» – «выпала шестёрка».

Формирование варианта. Номер варианта обозначим через v.

1. Выбор модели:
   - если v чётное, моделируется монета;
   - если v нечётное, моделируется кубик.
2. Количество испытаний. Один из возможных вариантов задания параметров:

$$N_1(v) = 100 + 10 \cdot (v \bmod 10),$$
$$N_2(v) = 250 + 25 \cdot (v \bmod 10),$$
$$N_3(v) = 500 + 50 \cdot (v \bmod 10)$$

3. Длина начального участка:

$$n_0(v) = 10 + 5 \cdot (v \bmod 10)$$

Теоретическая справка. Схема Бернулли. Закон больших чисел.

Схема Бернулли. Рассматривается последовательность независимых испытаний, в каждом из которых фиксировано событие A с вероятностью $P(A) = p,\ 0 < p < 1$. В каждом испытании возможны ровно два исхода: «успех» (событие A произошло) и «неудача» (событие A не произошло).

Обозначим через $\nu_n(A)$ число наступлений события A в n испытаниях, а через

$$\hat{p}(A) = \frac{\nu_n(A)}{n} \qquad (1)$$

– относительную частоту события A в этих n испытаниях.

Закон больших чисел. Пусть в независимых испытаниях схемы Бернулли событие A имеет вероятность $p = P(A)$. Тогда для любого $\varepsilon > 0$

$$P(|\hat{p}(A) - p| \geq \varepsilon) \underset{n\to\infty}{\longrightarrow} 0.$$

Иначе говоря, при большом числе испытаний относительная частота наступления события A с большой вероятностью становится сколь угодно близкой к его вероятности p.

Описание вычислительного эксперимента.

1. Провести моделирование для нескольких значений числа испытаний. Рассмотреть не менее трёх значений:
   - малое N_1;
   - среднее N_2;
   - большое N_3.

   Конкретные значения задаются через номер варианта (см. раздел «Формирование варианта»).
2. Для каждого N построить серию из N независимых испытаний.
   - Сгенерировать список результатов:
     - 0 или 1 – для монеты,
     - 1-6 – для кубика.
   - Для каждого $n = 1, 2, ..., N$ посчитать число успехов к моменту n и относительную частоту по формуле (1).
   - Начальный участок траектории – первые n_0 испытаний, где n_0 задаётся как функция от номера варианта.
3. Для каждого N построить два графика:
   - Полная траектория относительной частоты успеха, показывающая характер полной выборки;
   - Начальный участок траектории.

4. На всех графиках провести горизонтальную линию $y = p$, соответствующую теоретической вероятности успеха.
5. Сравнить графики и сделать выводы:
   - как ведёт себя траектория P_n на начальном участке (насколько велик разброс);
   - как меняется характер колебаний при переходе от N_1 к N_2 и N_3;
   - насколько близко P_n подходит к теоретическому уровню p при больших n;
   - как проявляется на практике закон больших чисел.

Программная реализация и визуализация в Python.

Для моделирования применяется модуль *random*, для построения графиков – *matplotlib.pyplot*. После генерации серии испытаний вычисляется траектория относительной частоты $\hat{p}_n$ и строятся два графика: полная траектория при $N = N_3$ и начальный участок (первые n_0 испытаний). На графиках нужно провести горизонтальную линию $y = p$.

Ниже приводится ячейка с импортом библиотек и настройками оформления графиков (в том числе форматирование десятичных дробей с запятой и сохранение рисунков в хорошем качестве).

```
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

def comma_tick(x, pos=None, digits=2):
    if abs(x - round(x)) < 1e-9:
        return f"{int(round(x))}"
    return f"{x:.{digits}f}".replace(".", ",")

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
})
```

Далее строится график полной траектории $\hat{p}_n$ при $N = N_3$. Предполагается, что к этому моменту уже вычислены значения p и последовательность $\hat{p}_n$ длины N_3.

```
ns = range(1, N3 + 1)
```

```
fig, ax = plt.subplots(figsize=(8.2, 4.6))
ax.plot(ns, p_hat, linewidth=1.6, label=r"относительная частота $\hat{p}_n$")
ax.axhline(p, linestyle="--", linewidth=1.6, label=r"теоретическая вероятность $p$")

ax.set_xlabel("Номер испытания $n$")
ax.set_ylabel(r"Относительная частота $\hat{p}_n$")
ax.set_title(rf"Траектория относительной частоты успеха при $N={N3}$")
ax.legend()

ax.yaxis.set_major_formatter(FuncFormatter(lambda x, pos: comma_tick(x, pos, digits=2)))

fig.tight_layout()
fig.savefig("lr1_fig1.png", bbox_inches="tight")
plt.show()
```

После этого строится график начального участка траектории по первым n_0 испытаниям.

```
n0_eff = min(n0, len(p_hat))
ns0 = range(1, n0_eff + 1)

fig, ax = plt.subplots(figsize=(8.2, 4.6))
ax.plot(ns0, p_hat[:n0_eff], linewidth=1.8,
        label=rf"относительная частота $\hat{{p}}_n$ (первые ${n0_eff}$ испытаний)")
ax.axhline(p, linestyle="--", linewidth=1.6, label=r"теоретическая вероятность $p$")

ax.set_xlabel("Номер испытания $n$")
ax.set_ylabel(r"Относительная частота $\hat{p}_n$")
ax.set_title(rf"Начальный участок траектории (первые ${n0_eff}$ испытаний)")
ax.legend()

ax.yaxis.set_major_formatter(FuncFormatter(lambda x, pos: comma_tick(x, pos, digits=2)))

fig.tight_layout()
fig.savefig("lr1_fig2.png", bbox_inches="tight")
plt.show()
```

Примеры оформления графиков.

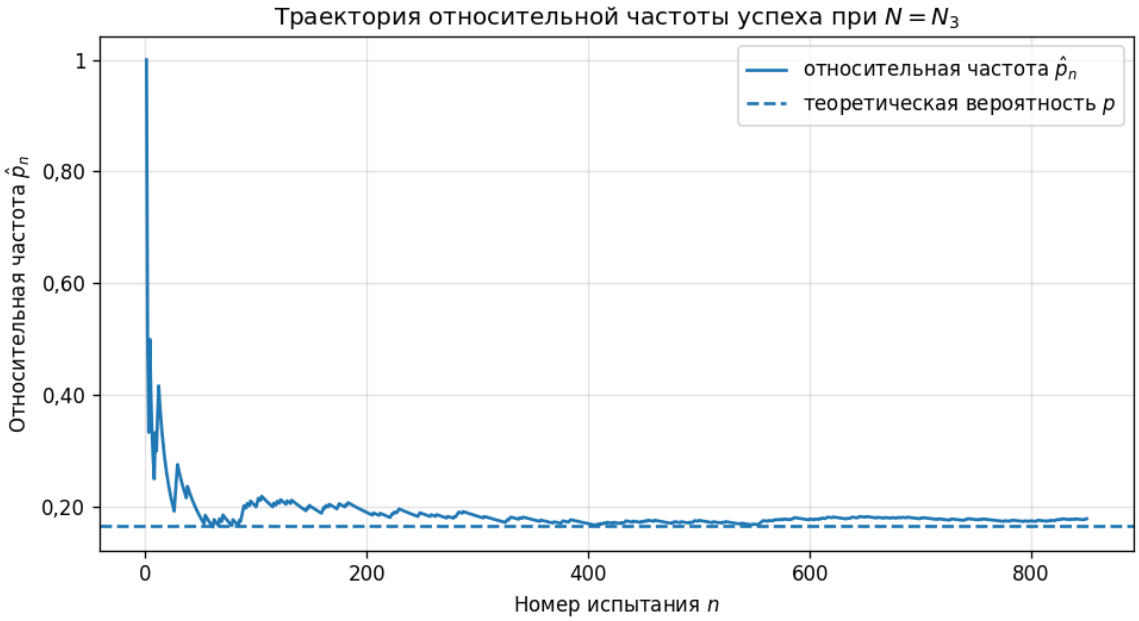


Рис. 1. Траектория относительной частоты успеха при $N = N_3$

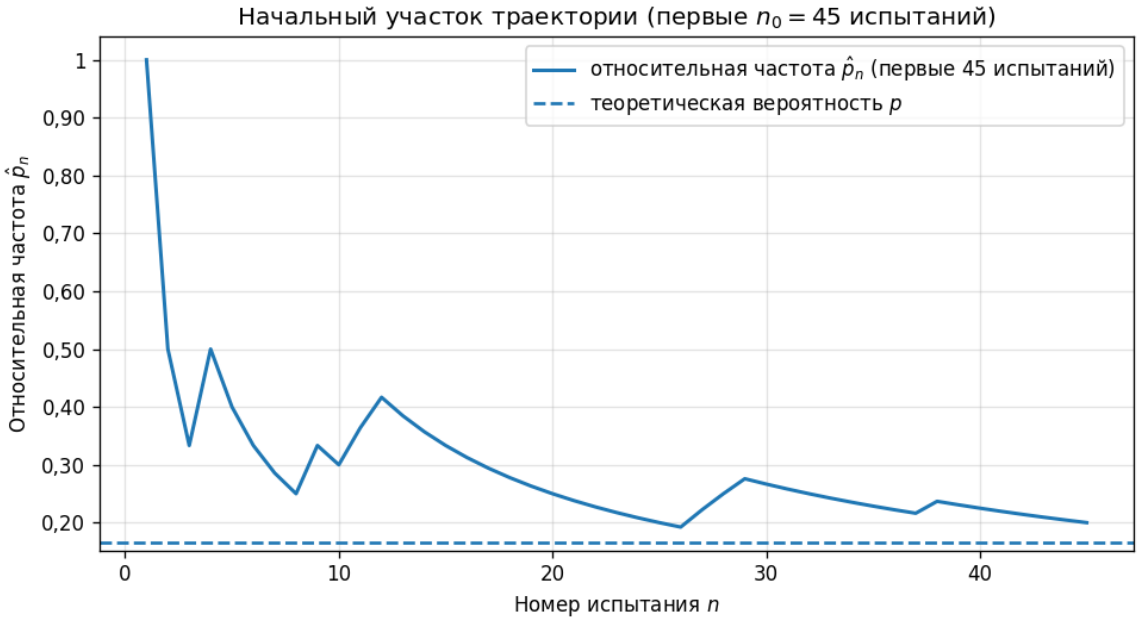


Рис. 2. Начальный участок траектории (первые $n_0(v)$ испытаний)

Лабораторная работа №2. Условная вероятность и формула Байеса

Цель работы. Смоделировать работу диагностического теста, проиллюстрировать понятие условной вероятности и формулы Байеса, исследовать влияние объёма выборки на точность эмпирической оценки апостериорной вероятности.

Условие. Рассматривается медицинский тест для диагностики некоторого заболевания. Обозначим:

- событие H_1 – «пациент болен»;
- событие H_2 – «пациент здоров»;
- событие A – «результат теста положительный»;
- событие $\overline{A}$ – «результат теста отрицательный».

Пусть:

- $p = P(H_1)$ – вероятность того, что случайно выбранный пациент болен;
- чувствительность теста (a) – вероятность получить положительный результат у больного:

$$a = P(A|H_1)$$

- специфичность теста (b) – вероятность получить отрицательный результат у здорового:

$$b = P\left(\overline{A}\middle|H_2\right)$$

Числовые значения параметров p, a, b задаются через номер варианта (см. раздел «Формирование варианта»).

Формирование варианта. Номер варианта обозначим через v.

1. Параметры теста:

$$p(v) = 0{,}01 + 0{,}001 \cdot (v \bmod 20),$$
$$a(v) = 0{,}85 + 0{,}01 \cdot (v \bmod 5),$$
$$b(v) = 0{,}9 + 0{,}01 \cdot (v \bmod 5).$$

2. Объём выборки (число пациентов):

$$N_1(v) = 500 + 100 \cdot (v \bmod 10),$$
$$N_2(v) = 2500 + 250 \cdot (v \bmod 10),$$
$$N_3(v) = 5000 + 500 \cdot (v \bmod 10).$$

Теоретическая справка. Условная вероятность и формула Байеса.

Условная вероятность события A при условии, что произошло событие C, определяется как

$$P(A|C) = \frac{P(A \cap C)}{P(C)}, \qquad P(C) > 0.$$

Пусть H_1, H_2, ..., H_n – попарно несовместные гипотезы, образующие полную группу событий, то есть выполняется

$$H_1 \cup H_2 \cup \ldots \cup H_n = \Omega, \qquad H_i \cap H_j = \emptyset \text{ при } i \neq j.$$

Пусть A – наблюдаемое событие с $P(A) > 0$. Тогда формула Байеса (2) даёт условную вероятность гипотезы H_k при условии, что произошло событие A:

$$P(H_k|A) = \frac{P(H_k)P(A|H_k)}{\sum_{i=1}^{n} P(H_i)P(A|H_i)}. \quad (2)$$

В нашей задаче апостериорная вероятность того, что пациент болен при положительном тесте равна:

$$P(H_1|A) = \frac{P(H_1)P(A|H_1)}{P(H_1)P(A|H_1) + P(H_2)P(A|H_2)}. \quad (3)$$

Подставляя в формулу (3) условные обозначения, получим следующее выражение:

$$P(H_1|A) = \frac{p \cdot a}{p \cdot a + (1-p) \cdot (1-b)} \quad (4)$$

Важно понимать, что даже при больших значениях чувствительности и специфичности (большие a и b) при малой распространённости болезни (малое p) доля действительно больных среди пациентов с положительным тестом может оказаться невелика.

Описание вычислительного эксперимента.

1. Выбрать несколько значений числа пациентов N. Например, можно взять три уровня:
   - малое N_1,
   - среднее N_2,
   - большое N_3.
2. Для каждого выбранного значения N смоделировать N пациентов. Для каждого пациента:
   - случайно определить, болен он или здоров, согласно вероятности $P(H_1)$;
   - в зависимости от состояния сгенерировать результат теста, а именно:
     - если пациент болен (верна гипотеза H_1):
       - событие A с вероятностью a;
       - событие $\overline{A}$ с вероятностью $1 - a$;
     - если пациент здоров (верна гипотеза H_2):
       - событие A с вероятностью $1 - b$;
       - событие $\overline{A}$ с вероятностью b;
3. По результатам моделирования для каждого N подсчитать:
   - общее число пациентов с положительным тестом $n(A)$;

- число больных с положительным тестом $n(H_1 \cap A)$ – число пациентов, для которых одновременно выполняются H_1 и A.

4. Вычислить эмпирическую апостериорную вероятность:

$$\hat{P}(H_1|A) = \frac{n(H_1 \cap A)}{n(A)}$$

5. Вычислить теоретическое значение $P(H_1|A)$ по формуле (4) и сравнить его с эмпирическим.
6. Для каждого N найти:
   - абсолютную погрешность:

$$\Delta(N) = \left|\hat{P}(H_1|A) - P(H_1|A)\right|$$

   - относительную погрешность:

$$\delta(N) = \frac{\Delta}{P(H_1|A)}$$

7. Сравнить результаты для разных N и сделать вывод о том, как объём выборки влияет на точность эмпирической оценки. Полученные данные рекомендуется оформить в виде следующей таблицы:

Таблица 1. Результаты моделирования апостериорной вероятности.

N	$\hat{P}(H_1\|A)$	$P(H_1\|A)$	$\Delta(N)$	$\delta(N)$
N_1				
N_2				
N_3				

Программная реализация и визуализация в Python.

Для моделирования удобно использовать *numpy.random*: он позволяет быстро генерировать состояния пациентов и результаты теста в векторном виде. Для каждого размера выборки $N \in \{N_1, N_2, N_3\}$ вычисляются величины $n(A)$ и $n(H_1 \cap A)$, затем эмпирическая оценка $\hat{P}(H_1|A)$ и ошибки $\Delta(N)$, $\delta(N)$. Теоретическое значение $P(H_1|A)$ вычисляется по формуле (4).

Сначала подключим библиотеки и зададим общие настройки оформления графиков (шрифт, сетка, формат подписей осей).

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
```

```
# Формат подписей с десятичной запятой
def comma_formatter(digits=3):
    def _fmt(x, pos=None):
        if abs(x - round(x)) < 1e-12:
            return f"{int(round(x))}"
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
})
```

Далее зададим параметры варианта и вычислим теоретическое значение $P(H_1|A)$.

```
v = 7  # номер варианта

# Параметры (подставить формулы из методички)
p = ...  # P(H1)
a = ...  # P(A|H1)
b = ...  # P(Ā|H2)

# Объёмы выборки (подставить формулы из методички)
N1 = ...
N2 = ...
N3 = ...
Ns = np.array([N1, N2, N3], dtype=int)

# Теоретическая апостериорная вероятность
P_theory = (p * a) / (p * a + (1 - p) * (1 - b))
P_theory
```

После этого выполним моделирование для $N \in \{N_1, N_2, N_3\}$ и сформируем таблицу результатов, используемую далее при построении рисунков.

```
rng = np.random.default_rng(42)
```

```
rows = []
for N in Ns:
    H1 = rng.random(N) < p

    A = np.empty(N, dtype=bool)
    A[H1] = rng.random(H1.sum()) < a
    H2 = ~H1
    A[H2] = rng.random(H2.sum()) < (1 - b)

    nA = int(A.sum())
    nH1A = int(np.logical_and(H1, A).sum())

    P_hat = nH1A / nA if nA > 0 else np.nan
    Delta = abs(P_hat - P_theory) if np.isfinite(P_hat) else np.nan
    delta = Delta / P_theory if (P_theory > 0 and np.isfinite(Delta)) else
np.nan
    rows.append({
        "N": N,
        "P_hat": P_hat,
        "P_theory": P_theory,
        "Delta": Delta,
        "delta": delta
    })

df = pd.DataFrame(rows)
df
```

Примеры оформления графиков.

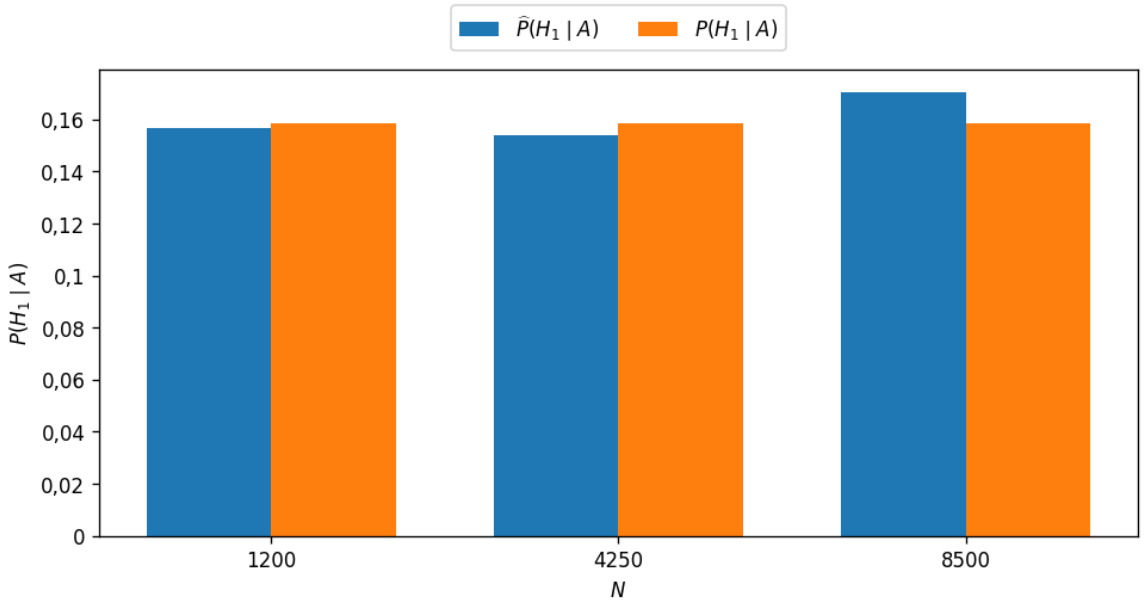


Рис. 3. Эмпирическая и теоретическая апостериорная вероятность

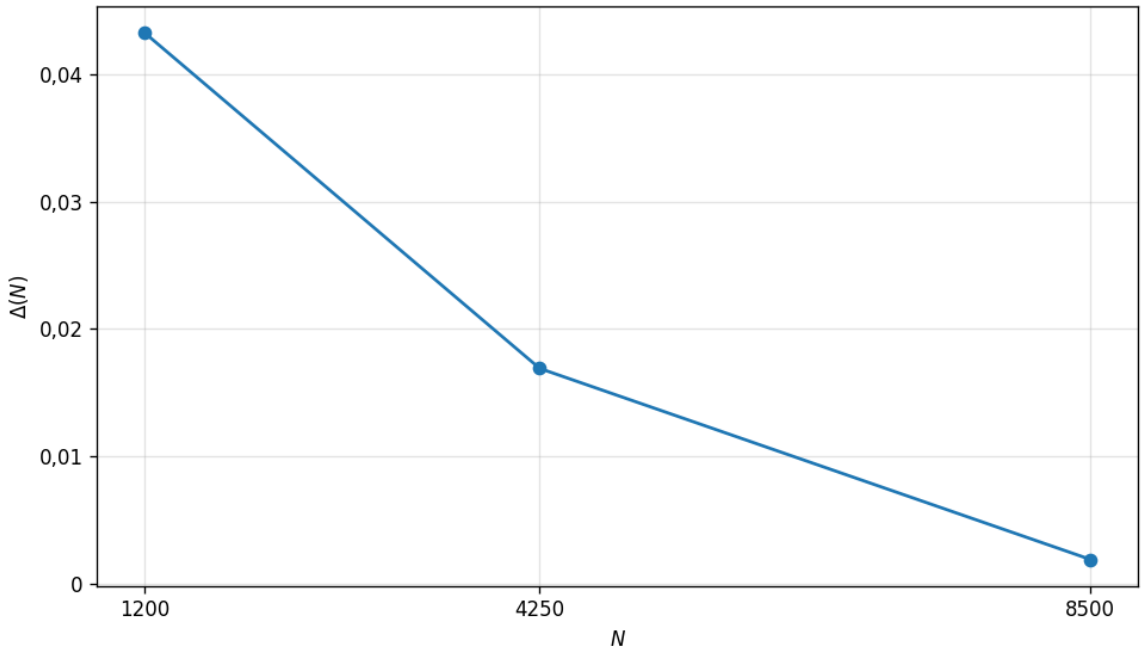


Рис. 4. Абсолютная ошибка в зависимости от числа пациентов

Лабораторная работа №3. Схема Бернулли и аппроксимация распределением Пуассона

Цель работы. Исследовать биномиальное распределение числа успехов в серии независимых испытаний, а также качество его аппроксимации распределением Пуассона при малой вероятности успеха и большом числе испытаний.

Условие. Рассматривается схема Бернулли. В каждом испытании фиксируется событие A («успех»), происходящее с вероятностью p, $0 < p < 1$. Проводится серия из n независимых испытаний.

Обозначим через ξ число наступления события A в серии из n испытаний. Тогда ξ имеет биномиальное распределение с параметрами n и p.

Для аппроксимации биномиального распределения при малой вероятности успеха и большом числе испытаний используется распределение Пуассона с параметром λ.

Формирование варианта. Номер варианта обозначим через v.

Параметры схемы Бернулли:

- Число испытаний в серии:

$$n(v) = 50 + 5 \cdot (v \bmod 6).$$

- Вероятность успеха:

$$p(v) = 0{,}02 + 0{,}001 \cdot (v \bmod 5).$$

- Параметр распределения Пуассона:

$$\lambda(v) = n(v)p(v)$$

Число серий для моделирования:

- Для иллюстрации влияния объёма эксперимента зададим три уровня числа серий:

$$N_1(v) = 200 + 20 \cdot (v \bmod 10),$$
$$N_2(v) = 600 + 30 \cdot (v \bmod 10),$$
$$N_3(v) = 1500 + 50 \cdot (v \bmod 10).$$

Теоретическая справка. Биномиальное распределение. Аппроксимация распределением Пуассона.

Если проводится n независимых испытаний, в каждом из которых событие A происходит с вероятностью p, то число наступлений события A обозначим через ξ. Тогда ξ имеет биномиальное распределение $B_{n,p}$ с параметрами n и p и принимает значения $k = 0, 1, \dots, n$ с вероятностями:

$$P(\xi = k) = C_n^k p^k (1-p)^{n-k}, \qquad k = 0{,}1, \dots, n$$

Если p мало, а n велико, так что $\lambda = np$ остаётся умеренным ($\lambda < 2$), то биномиальное распределение можно приближённо заменить распределением Пуассона Π_λ:

$$P(\xi = k) = \frac{\lambda^k}{k!} e^{-\lambda}, \qquad k = 0,1,2, \ldots$$

Описание вычислительного эксперимента.

Для иллюстрации влияния числа серий моделирования на вид эмпирического распределения предлагается рассмотреть несколько значений числа серий N:

- малое N_1;
- среднее N_2;
- большое N_3.

Требуется:

1. Для каждого выбранного значения N:
   - Смоделировать N независимых серий, каждая серия состоит из n независимых испытаний. В каждом испытании событие «успех» происходит с вероятностью p;
   - В каждой серии подсчитать число успехов ξ (сколько раз произошло событие A).
2. Получить эмпирическое распределение числа успехов.
   - Для каждого возможного значения $k = 0,\ 1,\ \ldots,\ n$ определить относительную частоту

$$\hat{P}_N(\xi = k) = \frac{\text{число серий, в которых зафиксировано } \xi = k}{N}.$$

   - Рассмотреть, как меняется «гладкость» гистограммы при переходе от N_1 к N_2 и N_3.
3. Построить график эмпирического распределения:
   - по горизонтальной оси откладывается число успехов k;
   - по вертикальной оси – эмпирические вероятности $\hat{P}_N(\xi = k)$;
   - эмпирическое распределение удобно изобразить столбчатой диаграммой.
4. На том же графике изобразить теоретические распределения:
   - точки биномиального распределения $P(\xi = k)$ для $\xi \sim B_{n,p}$;
   - точки распределения Пуассона $P(\eta = k)$ для $\eta \sim \Pi_\lambda$.
5. Сравнить для каждого N три кривые (эмпирическую, биномиальную, пуассоновскую) и сделать выводы:
   - как влияет увеличение числа серий N на «гладкость» эмпирической гистограммы;
   - насколько хорошо распределение Пуассона приближает биномиальное при выбранных n, p и $\lambda = np$.

- при каких значениях k аппроксимация Пуассоном выглядит лучше, а при каких заметны отличия.

При желании можно дополнительно указать в отчёте численное значение $\lambda = np$ и сравнить эмпирическое математическое ожидание числа успехов (среднее по сериям) с теоретическими значениями $E\xi = np$ и $E\eta = \lambda$.

Программная реализация и визуализация в Python.

Для моделирования удобно использовать *numpy.random*: это позволяет быстро генерировать числа успехов в сериях испытаний и строить эмпирическое распределение. Для каждого значения числа серий $N \in \{N_1, N_2, N_3\}$ вычисляются относительные частоты $\hat{P}_N(\xi = k)$, которые затем сравниваются с теоретическими вероятностями биномиального распределения $B_{n,p}$ и пуассоновского распределения Π_λ, где $\lambda = np$.

Сначала подключим библиотеки и зададим общие настройки оформления графиков.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter
from math import comb, exp, factorial

# Формат подписей с десятичной запятой
def comma_formatter(digits=3):
    def _fmt(x, pos=None):
        if abs(x - round(x)) < 1e-12:
            return f"{int(round(x))}"
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
})
```

Далее зададим параметры варианта и значения N_1, N_2, N_3.

```
v = 7  # номер варианта
```

```
# Параметры схемы Бернулли (формулы из ЛР №3)
n = 50 + 5 * (v % 6)
p = 0.02 + 0.001 * (v % 5)
lam = n * p

N1 = 200 + 20 * (v % 10)
N2 = 600 + 30 * (v % 10)
N3 = 1500 + 50 * (v % 10)
Ns = np.array([N1, N2, N3], dtype=int)

ks = np.arange(n + 1)

# Теоретические вероятности
binom_probs = np.array([comb(n, k) * (p ** k) * ((1 - p) ** (n - k)) for k
in ks], dtype=float)
poisson_probs = np.array([exp(-lam) * (lam ** k) / factorial(k) for k in
ks], dtype=float)
```

После этого выполним моделирование для одного выбранного значения N и сформируем таблицу эмпирических вероятностей $\hat{P}_N(\xi = k)$, используемую далее при построении графика.

```
rng = np.random.default_rng(42)

N = N1

xi = rng.binomial(n=n, p=p, size=N)

counts = np.bincount(xi, minlength=n + 1)
emp_probs = counts / N

df_plot = pd.DataFrame({
    "k": ks,
    "P_hat": emp_probs,
    "P_binom": binom_probs,
    "P_pois": poisson_probs
})

df_plot.head(10)
```

Примеры оформления графиков.

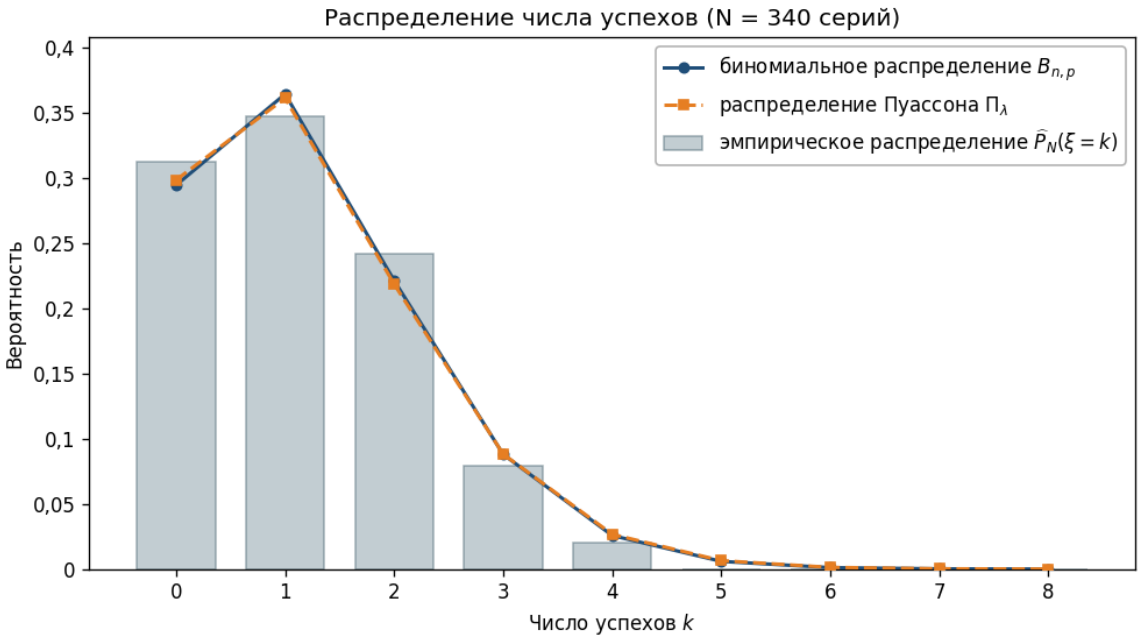


Рис. 5. Сравнение эмпирического распределения числа успехов с теоретическими распределениями при малом числе серий

Лабораторная работа №4. Геометрическая вероятность – игла Бюффона

Цель работы. Смоделировать классический опыт Бюффона и по результатам моделирования получить приближённое значение числа π. Познакомиться с геометрической вероятностью и оценить, как меняется точность оценки при увеличении числа испытаний.

Условие. Пусть на плоскости нанесены параллельные прямые на расстоянии d друг от друга (например, как линии на тетрадном листе). На эту плоскость случайным образом бросается игла длиной l. Пусть длина иглы не превосходит расстояния между прямыми: $l \leq d$. Предполагается, что:

- положение центра иглы выбирается равновероятно по вертикали между соседними линиями;
- угол наклона иглы относительно направления прямых также выбирается случайно и равновероятно.

Эксперимент повторяется многократно. Для трёх различных значений числа бросков N_1, N_2, N_3 требуется выполнить моделирование и по результатам получить приближённые значения числа π.

Формирование варианта. Номер варианта обозначим через v.

1. Геометрические параметры:
   - расстояние между параллельными прямыми: $d = 4$;
   - длина иглы:

$$l(v) = 2 + (v \bmod 3)$$

2. Число бросков:
   Зададим три уровня числа бросков:

$$N_1(v) = 100 + 10 \cdot (v \bmod 10),$$
$$N_2(v) = 500 + 50 \cdot (v \bmod 10),$$
$$N_3(v) = 2000 + 100 \cdot (v \bmod 10)$$

Теоретическая справка. Опыт Бюффона и вероятность пересечения.

Рассмотрим классический опыт Бюффона: на плоскость с параллельными прямыми, расположенными на расстоянии d друг от друга, бросается игла длиной l, причём $l \leq d$. Пусть:

- Y – расстояние от центра иглы до ближайшей прямой;
- φ – угол между иглой и направлением прямых

Выбирая положение центра иглы и её угол наклона случайно и равновероятно, можно считать, что $Y \sim U[0; 0{,}5d]$, $\varphi \sim U[0; 0{,}5\pi]$ и Y и φ независимы. Условие пересечения иглы с прямой имеет вид:

$$Y \leq \frac{l}{2} \sin \varphi \,. \qquad (5)$$

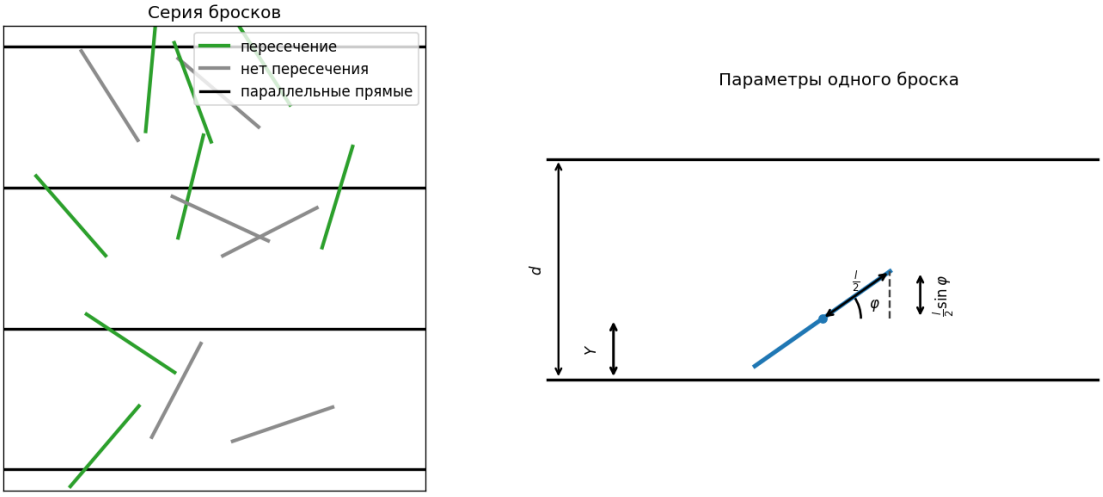


Рис. 6. Опыт Бюффона: серия бросков иглы и геометрия одного броска

Вероятность пересечения вычисляется как отношение площадей области под графиком к площади прямоугольника

$$P_{\text{пересечения}} = P\left(Y \le \frac{l}{2}\sin\varphi\right) = \frac{2}{\pi d}\int_0^{0{,}5\pi} \frac{l}{2}\sin\varphi\, d\varphi = \frac{2l}{\pi d}. \qquad (6)$$

Именно эта формула (5) лежит в основе вероятностной оценки числа π: выразив π через $P_{\text{пересечения}}$ и заменив неизвестную истинную вероятность её эмпирической оценкой $\hat{p}_N$, можно получить приближённое значение числа π.

Эта оценка является случайной величиной, и, в силу закона больших чисел, при увеличении числа бросков N в среднем будет всё ближе к истинному значению π. Однако сходимость довольно медленная: даже для больших N наблюдаются заметные флуктуации.

Описание вычислительного эксперимента.

Для каждого из трёх значений числа бросков N_1, N_2, N_3:

1. Смоделировать N бросков иглы. В рамках моделирования достаточно считать, что:
   - расстояние от центра иглы до ближайшей линии равномерно распределено на отрезке $[0; 0{,}5d]$: $Y \sim U[0; 0{,}5d]$;
   - угол наклона иглы φ относительно направления линий равномерно распределён на некотором интервале $[0; \pi)$. *Подумайте, почему при программной реализации достаточно моделировать φ только на отрезке $[0; 0{,}5\pi]$?* То есть $\varphi \sim U[0; 0{,}5\pi]$
2. Для каждого броска проверить выполнения условия (5). Если оно выполняется, считаем, что при данном броске произошло пересечение.
3. Для каждого значения N определить число бросков, в которых произошло пересечение. Обозначим это число через K_N.
4. Вычислить эмпирическую вероятность пересечения

$$\hat{p}_N = \frac{K_N}{N}$$

5. Используя теоретическую формулу для вероятности пересечения (6), получить приближённое значение числа π:

$$\hat{\pi}_N = \frac{2l}{d\hat{p}_N}$$

В отчёте сделать выводы:

- как сильно колеблется оценка $\hat{\pi}_N$ при малых и больших N;
- заметно ли улучшение точности при переходе от N_1 к N_2 и N_3;
- остаются ли заметные случайные отклонения даже при большом числе бросков.

Для всех рассматриваемых значений N свести результаты в таблицу (пример оформления – таблица 2) и проанализировать, как изменяется точность оценки $\hat{\pi}_N$ при увеличении числа испытаний.

Таблица 2. Результаты моделирования опыта Бюффона при различных значениях N.

N	$N_{hit}(N)$	$\hat{p}_N$	$\hat{\pi}_N$	$\Delta = \lvert\hat{\pi}_N - \pi\rvert$	$\delta = \frac{\Delta}{\pi}$
N_1					
N_2					
N_3					

Программная реализация и визуализация в Python.

Для моделирования опыта Бюффона удобно использовать *numpy.random*: это позволяет быстро генерировать параметры броска и выполнять вычисления для больших серий испытаний. В каждом броске моделируются величины Y и φ согласно распределениям, заданным в теоретической части, после чего фиксируется факт пересечения иглы с ближайшей прямой. По серии из N бросков вычисляются число пересечений K_N, эмпирическая вероятность пересечения $\hat{p}_N$ и оценка $\hat{\pi}_N$. Для $N \in \{N_1, N_2, N_3\}$ формируется таблица результатов. Для визуализации строится график зависимости $\hat{\pi}_k$ от числа бросков k и проводится горизонтальная линия истинного значения π.

Сначала подключим библиотеки и зададим общие настройки оформления графиков.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
import math
from matplotlib.ticker import FuncFormatter

# Формат подписей с десятичной запятой (фиксированное число знаков)
def comma_fixed(digits=3):
    def _fmt(x, pos=None):
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

def comma_int(x, pos=None):
    return f"{int(round(x))}"

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
})
```

Далее зададим параметры варианта и три уровня числа бросков N_1, N_2, N_3.

```
v = 7  # номер варианта

d = 4
l = 2 + (v % 3)

# Три уровня числа бросков (ваши новые формулы)
N1 = 100 + 10 * (v % 10)
N2 = 500 + 50 * (v % 10)
N3 = 2000 + 100 * (v % 10)
Ns = np.array([N1, N2, N3], dtype=int)

d, l, Ns
```

После этого выполним моделирование до N_3 (одной серией) и сформируем таблицу результатов для N_1, N_2, N_3.

```
rng = np.random.default_rng(42)

Y = rng.uniform(0.0, d/2.0, size=N3)
```

```
phi = rng.uniform(0.0, math.pi/2.0, size=N3)

hits = (Y <= (l/2.0) * np.sin(phi)).astype(int)

K = np.cumsum(hits)                    # K_k
k = np.arange(1, N3 + 1)               # 1..N3
p_hat_k = K / k                        # \hat p_k

with np.errstate(divide="ignore", invalid="ignore"):
    pi_hat_k = (2.0 * l) / (d * p_hat_k)

# первые шаги могут давать бесконечности, если K_k=0
pi_hat_k[~np.isfinite(pi_hat_k)] = np.nan

rows = []
for N in Ns:
    K_N = int(K[N-1])
    pN = float(p_hat_k[N-1])
    piN = float(pi_hat_k[N-1])
    rows.append([int(N), K_N, pN, piN])

df = pd.DataFrame(rows, columns=["N", "K_N", "p_hat_N", "pi_hat_N"])
df
```

Примеры оформления графиков.

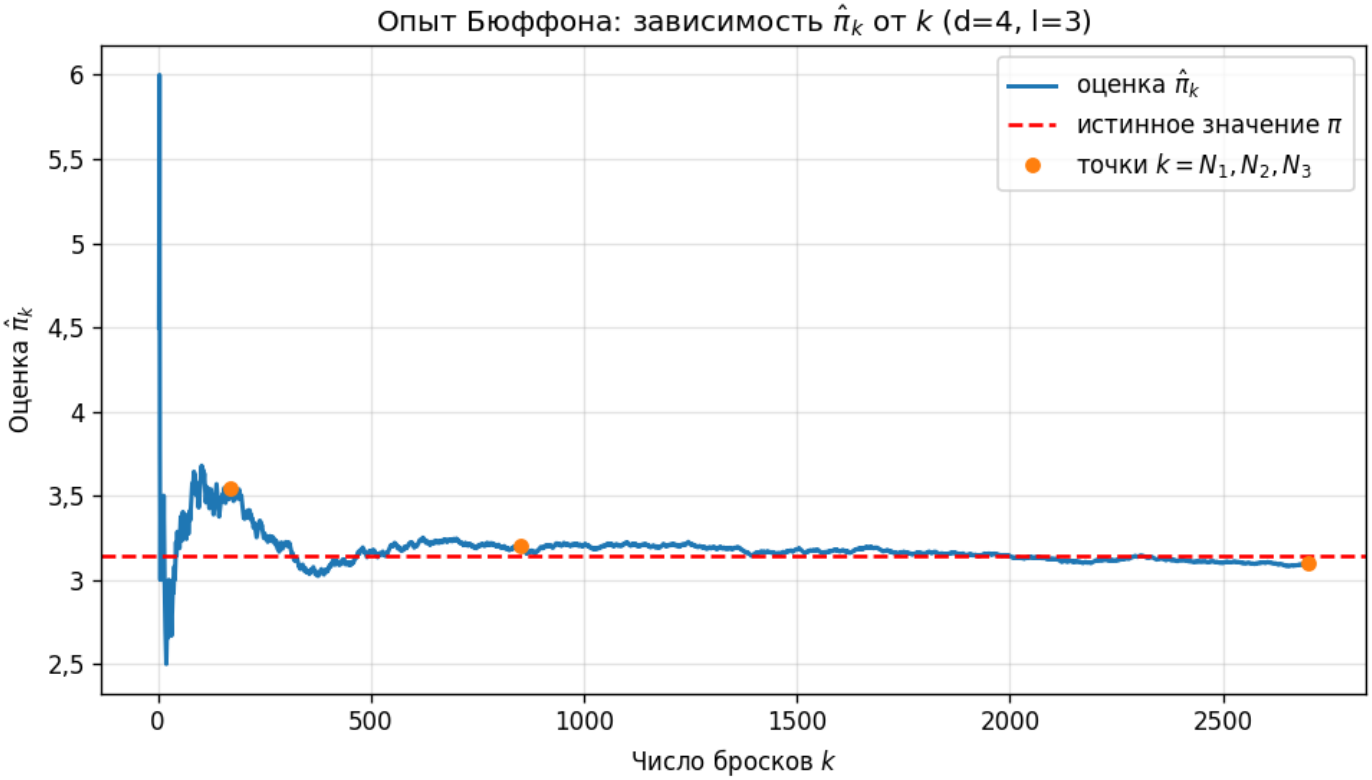


Рис. 7. Зависимость оценки $\hat{\pi}_k$ от числа бросков k при моделировании опыта Бюффона

РГР 2. Лабораторные работы 5-8

Во втором блоке рассматриваются распределения случайных величин и их приближения. Задания этого блока направлены на то, чтобы студент:

- познакомился на практике с типовыми распределениями (биномиальным, геометрическим, показательным, нормальным) и их свойствами;
- увидел, как выборочные характеристики (среднее, дисперсия, мода, медиана) приближают теоретические значения при увеличении объёма выборки;
- проиллюстрировал центральную предельную теорему и нормальную аппроксимацию биномиального распределения;
- освоил метод получения случайных величин с заданным распределением с помощью преобразований равномерной случайной величины;
- научился работать с совместным распределением, ковариацией и коэффициентом корреляции, а также интерпретировать диаграммы рассеяния.

Во всех работах второго блока особое внимание уделяется сравнению эмпирических характеристик, полученных из моделирования, с теоретическими формулами, а также влиянию объёма выборки на качество аппроксимации. Студенту предстоит научиться:

- строить гистограммы и столбчатые диаграммы для дискретных и непрерывных распределений;
- вычислять выборочные характеристики и сравнивать их с теоретическими;
- делать содержательные выводы о «качестве попадания» модели в данные при увеличении числа экспериментов.

Лабораторная работа №5. Дискретные и непрерывные распределения

Цель работы. Познакомиться на практике с типовыми распределениями случайных величин (биномиальным, геометрическим, показательным и нормальным), научиться строить для них столбчатые диаграммы и гистограммы, эмпирические функции распределения, вычислять выборочные характеристики и сравнивать их с теоретическими значениями.

Условие. Рассматриваются четыре случайные величины:

- ξ_1 – биномиальное распределение с параметрами n и p_1;
- ξ_2 – геометрическое распределение с параметром p_2;
- ξ_3 – показательное распределение с параметром λ;
- ξ_4 – нормальное распределение с параметрами a и σ.

Смоделировать выборки объёма N для каждой случайной величины ξ_1, ξ_2, ξ_3, ξ_4. Для каждой случайной величины требуется:

- для дискретных распределений (ξ_1, ξ_2) построить столбчатую диаграмму эмпирических вероятностей (относительных частот) и на том же графике изобразить теоретические вероятности;

- для непрерывных распределений (ξ_3, ξ_4) построить гистограмму плотности и на том же графике изобразить теоретическую плотность;
- вычислить выборочные характеристики: среднее, дисперсию, моду, медиану;
- записать теоретические значения математического ожидания и дисперсии и сравнить их с выборочными.

Построить для каждой случайной величины эмпирическую функцию распределения и сравнить её с теоретической функцией распределения. Сделать выводы:

- насколько хорошо выборочные характеристики приближают теоретические значения;
- как зависит качество аппроксимации от типа распределения и объёма выборки.

Формирование варианта. Номер варианта обозначим через v.

Один из возможных вариантов задания параметров:

- Объём выборки:

$$N(v) = 2000 + 500 \cdot (v \bmod 5).$$

- Параметры биномиального распределения:

$$n(v) = 10 + (v \bmod 6),$$
$$p_1(v) = 0{,}3 + 0{,}1 \cdot (v \bmod 5).$$

- Параметр геометрического распределения:

$$p_2(v) = 0{,}25 + 0{,}05 \cdot (v \bmod 6).$$

- Параметр показательного распределения:

$$\lambda(v) = 0{,}5 + 0{,}1 \cdot (v \bmod 6).$$

- Параметры нормального распределения:

$$a(v) = -1{,}5 + 0{,}5 \cdot (v \bmod 7),$$
$$\sigma(v) = 0{,}7 + 0{,}1 \cdot (v \bmod 6).$$

Теоретическая справка. Биномиальное, геометрическое, показательное и нормальное распределения.

1. Биномиальное распределение.

Рассматривается последовательность из n независимых испытаний, в каждом из которых событие «успех» происходит с вероятностью p_1, а «неудача» – с вероятностью $q_1 = 1 - p_1$. Обозначим через ξ_1 число успехов в этих n испытаниях. Говорят, что ξ_1 имеет биномиальное распределение с параметрами n и p_1. $\xi \in \mathrm{B}_{n,p}$.

Случайная величина ξ_1 принимает значения $k = 0, 1, ..., n$ с вероятностями

$$\mathrm{P}(\xi_1 = k) = C_n^k p_1^k q_1^{n-k}, \qquad k = 0,1, \dots, n, \tag{7}$$

где C_n^k – биномиальный коэффициент.

Математическое ожидание и дисперсия:

$$\mathrm{E}\xi_1 = np_1, \qquad \mathrm{D}\xi = np_1q_1. \tag{8}$$

2. Геометрическое распределение.

Будем считать, что случайная величина ξ_2 – число испытаний до первого успеха в независимых испытаниях, в каждом из которых событие «успех» имеет вероятность p_2, а «неудача» – вероятность $q_2 = 1 - p_2$. $\xi \in \mathrm{G}_p$.

Тогда ξ_2 принимает значения $k = 1, 2, \ldots$ с вероятностями

$$\mathrm{P}(\xi_2 = k) = q_2^{k-1}p_2, \qquad k = 1,2, \ldots \tag{9}$$

Математическое ожидание и дисперсия:

$$\mathrm{E}\xi_2 = \frac{1}{p_2}, \qquad \mathrm{D}\xi_2 = \frac{q_2}{p_2^2}. \tag{10}$$

3. Показательное распределение.

Случайная величина ξ_3 имеет показательное распределение с параметром $\lambda > 0,\ \xi \in \mathrm{E}_\alpha$, если её плотность распределения имеет вид

$$f_{\xi_3}(x) = \begin{cases} \lambda e^{-\lambda x}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{11}$$

Интегрируя эту зависимость (11) по соответствующим промежуткам, можно получить функцию распределения:

$$F_{\xi_3}(x) = \begin{cases} 0, & x < 0 \\ 1 - e^{-\lambda x}, & x \geq 0 \end{cases}$$

Математическое ожидание и дисперсия:

$$\mathrm{E}\xi_3 = \frac{1}{\lambda}, \qquad \mathrm{D}\xi_3 = \frac{1}{\lambda^2}. \tag{12}$$

4. Нормальное распределение.

Случайная величина ξ_4 имеет нормальное распределение с параметрами $a \in R$ и $\sigma > 0, \xi \in \mathrm{N}_{a,\sigma}$,если её плотность распределения имеет вид:

$$f_{\xi_4}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2\sigma^2}}, \qquad x \in R.$$

Математическое ожидание и дисперсия:

$$\mathrm{E}\xi_4 = a, \qquad \mathrm{D}\xi_4 = \sigma^2. \tag{13}$$

При достаточно большом объёме выборки N выборочные характеристики (среднее, дисперсия, мода, медиана) становятся хорошим приближением к теоретическим значениям соответствующих распределений (следствие из закона больших чисел).

Описание вычислительного эксперимента.

1. Для каждой случайной величины ξ_1, ξ_2, ξ_3, ξ_4 сгенерировать выборку объёма N независимых реализаций:
   - для биномиального распределения ξ_1 – моделировать N раз число успехов в n испытаниях схемы Бернулли с вероятностью успеха p_1;
   - для геометрического распределения ξ_2 – моделировать «число испытаний до первого успеха»;
   - для показательного распределения ξ_3 – использовать готовую функцию генерации показательной случайной величины или преобразование равномерной случайной величины (подробные примеры – в разделе «Примеры программной реализации моделирования на языке Python – блок 2»);
   - для нормального распределения ξ_4 – использовать генератор нормальной случайной величины с параметрами a и σ.
2. Для дискретных распределений ξ_1 и ξ_2:
   - составить таблицу значений k и соответствующих относительных частот $\hat{p}(k)$;
   - построить столбчатую диаграмму эмпирических вероятностей (по оси абсцисс – значения k, по оси ординат – $\hat{p}(k)$);
   - на том же графике изобразить теоретические вероятности $P(\xi_i = k)$ для соответствующего распределения по формулам (7) и (9).
3. Для непрерывных распределений ξ_3 и ξ_4:
   - выбрать число интервалов гистограммы (например, от 15 до 30, в зависимости от N);
   - построить гистограмму с нормировкой по плотности (так, чтобы площадь под гистограммой была равна 1);
   - на том же графике изобразить теоретическую плотность $f_{\xi_i}(x)$.
4. Для каждой случайной величины ξ_1, ξ_2, ξ_3, ξ_4 вычислить:
   - выборочное среднее;
   - выборочную дисперсию;
   - выборочную моду;
   - выборочную медиану.
5. Сравнить полученные выборочные значения с теоретическими $\mathrm{E}\xi_i$ и $\mathrm{D}\xi_i$ по формулам (8), (10), (12), (13) и рассчитать абсолютные и относительные погрешности.
6. Построить эмпирическую функцию распределения для каждой случайной величины:
   - для дискретных распределений – ступенчатую функцию, показывающую на каждом значении k накопленную относительную частоту $\hat{F}(k) = \hat{P}(X \leq k)$;

- для непрерывных распределений – ступенчатую функцию по отсортированной выборке: при каждом значении x эмпирическая функция равна доле наблюдений, не превосходящих x.
- на этих же графиках изобразить теоретические функции распределения соответствующих распределений.

7. Сделать выводы по результатам эксперимента:
   - насколько хорошо выборочные характеристики приближают теоретические значения для каждого распределения;
   - для каких распределений (при данном N) графики гистограмм и эмпирических функций распределения визуально ближе к теоретическим;
   - как изменились бы результаты при увеличении объёма выборки (по аналогии с законом больших чисел)

Программная реализация и визуализация в Python.

Для моделирования выборок удобно использовать *numpy.random*: это позволяет генерировать реализации распределений векторно и быстро строить таблицы частот, гистограммы и эмпирические функции распределения. В работе моделируются выборки объёма N для четырёх распределений: биномиального ξ_1, геометрического ξ_2, показательного ξ_3 и нормального ξ_4. Далее вычисляются выборочные характеристики и строятся графики для сравнения с теоретическими зависимостями.

Сначала подключим библиотеки и зададим общие настройки оформления графиков.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
from matplotlib.ticker import FuncFormatter

def comma_fixed(digits=2):
    def _fmt(x, pos=None):
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

def comma_int(x, pos=None):
    return f"{int(round(x))}"

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
```

```
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
})
```

Далее зададим параметры варианта.

```
v = 7  # номер варианта

# Объём выборки
N = 2000 + 500 * (v % 5)

# Биномиальное распределение
n = 10 + (v % 6)
p1 = 0.3 + 0.1 * (v % 5)

# Геометрическое распределение
p2 = 0.25 + 0.05 * (v % 6)

# Показательное распределение
lam = 0.5 + 0.1 * (v % 6)

# Нормальное распределение
a = -1.5 + 0.5 * (v % 7)
sigma = 0.7 + 0.1 * (v % 6)

N, n, p1, p2, lam, a, sigma
```

После этого выполним моделирование выборок заданного объёма для четырёх распределений, вычислим выборочные характеристики и подготовим таблицы частот и плотностей, которые далее используются при оформлении рисунков.

```
rng = np.random.default_rng(42)

binom_sample = rng.binomial(n=n, p=p1, size=N)          # биномиальное
geom_sample  = rng.geometric(p=p2, size=N)              # геометрическое
exp_sample   = rng.exponential(scale=1.0/lam, size=N)   # показательное
norm_sample  = rng.normal(loc=a, scale=sigma, size=N)   # нормальное

def sample_mode_discrete(x: np.ndarray) -> int:
    counts = np.bincount(x.astype(int))
```

```
    return int(np.argmax(counts))

def sample_mode_hist(x: np.ndarray, bins: int = 25) -> float:
    dens, edges = np.histogram(x, bins=bins, density=True)
    idx = int(np.argmax(dens))
    return float(0.5 * (edges[idx] + edges[idx + 1]))

def sample_stats(x: np.ndarray, mode_kind: str, bins_for_mode: int = 25):
    mean = float(np.mean(x))
    var  = float(np.var(x, ddof=1))
    med  = float(np.median(x))
    if mode_kind == "discrete":
        mode = float(sample_mode_discrete(x))
    else:
        mode = float(sample_mode_hist(x, bins=bins_for_mode))
    return mean, var, med, mode

m1, s21, med1, mod1 = sample_stats(binom_sample, "discrete")
m2, s22, med2, mod2 = sample_stats(geom_sample,  "discrete")
m3, s23, med3, mod3 = sample_stats(exp_sample,   "hist", bins_for_mode=25)
m4, s24, med4, mod4 = sample_stats(norm_sample,  "hist", bins_for_mode=25)

E1, D1 = n * p1, n * p1 * (1 - p1)
E2, D2 = 1 / p2, (1 - p2) / (p2 ** 2)
E3, D3 = 1 / lam, 1 / (lam ** 2)
E4, D4 = a, sigma ** 2

df_stats = pd.DataFrame(
    [
        ["Биномиальное",   m1, s21, med1, mod1, E1, D1],
        ["Геометрическое", m2, s22, med2, mod2, E2, D2],
        ["Показательное",  m3, s23, med3, mod3, E3, D3],
        ["Нормальное",     m4, s24, med4, mod4, E4, D4],
    ],
    columns=["Распределение", "x¯", "S²", "Me", "Mo", "E", "D"]
)

for col_s, col_t, name in [("x¯", "E", "ΔE"), ("S²", "D", "ΔD")]:
    df_stats[name] = (df_stats[col_s] - df_stats[col_t]).abs()
    df_stats["δ" + name[1:]] = df_stats[name] / df_stats[col_t].abs()
```

```
df_stats
```

Примеры оформления графиков.

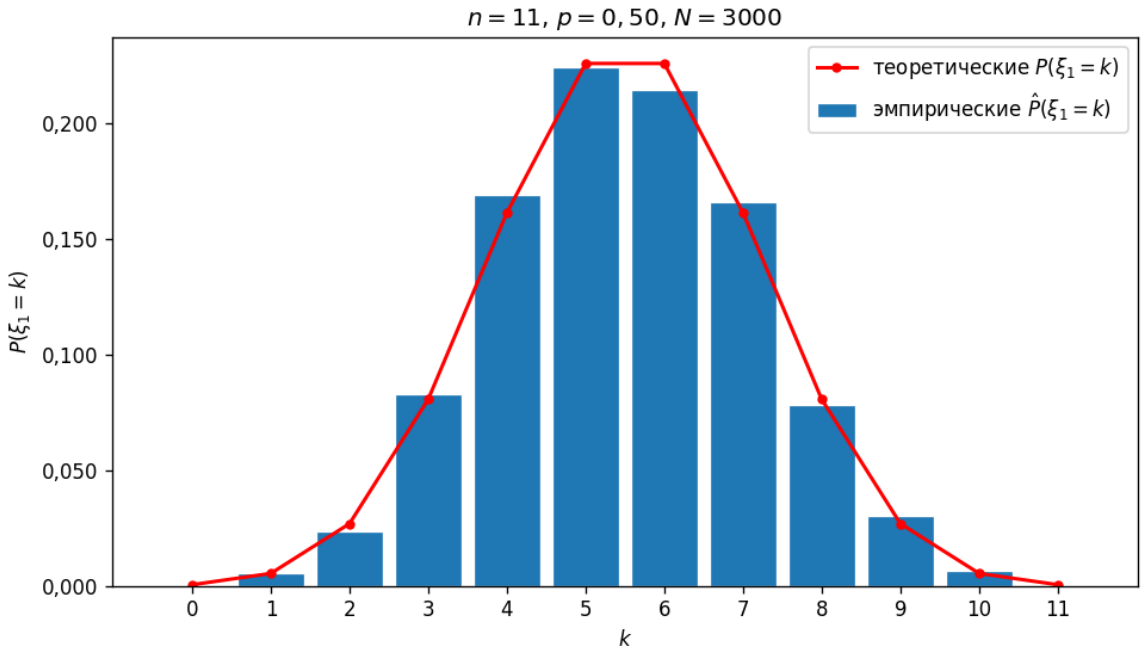


Рис. 8. Биномиальное распределение: эмпирические и теоретические вероятности

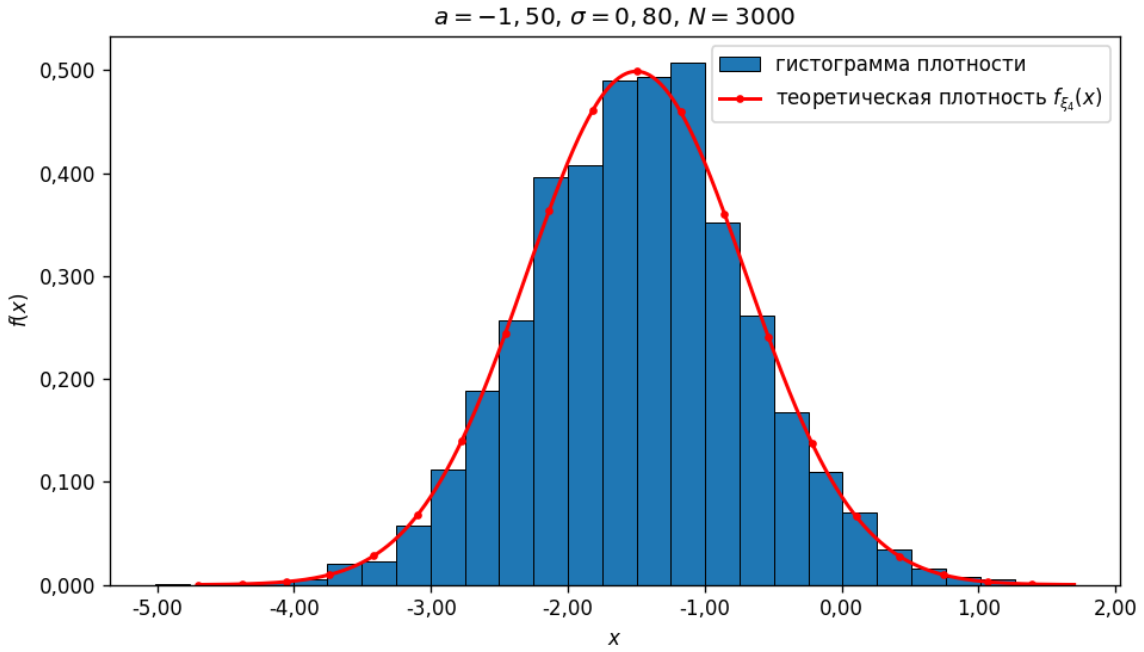


Рис. 9. Нормальное распределение: гистограмма и теоретическая плотность

Лабораторная работа №6. Центральная предельная теорема и нормальная аппроксимация биномиального распределения

Цель работы. На примере схемы Бернулли проиллюстрировать центральную предельную теорему и предельную теорему Муавра-Лапласа:

- сравнить эмпирическое биномиальное распределение с теоретическим;
- исследовать распределение центрированной и нормированной суммы и его приближение стандартным нормальным законом.

Условие. Рассматривается схема Бернулли: в каждом независимом испытании событие «успех» происходит с вероятностью p, событие «неудача» – с вероятностью $q = 1 - p$.

Пусть ξ – число успехов в серии из n независимых испытаний схемы Бернулли. Тогда ξ имеет биномиальное распределение с параметрами n и p. Для трёх различных значений числа испытаний $n \in \{n_1; n_2; n_3\}$ смоделировать N независимых серий и в каждой серии подсчитать число успехов ξ.

Для каждого значения n:

1. Построить столбчатую диаграмму эмпирической функции вероятностей числа успехов ξ и на том же графике изобразить теоретические биномиальные вероятности.
2. Вычислить выборочное среднее и выборочную дисперсию числа успехов и сравнить их с теоретическими значениями.
3. Построить центрированную и нормированную случайную величину

$$\eta = \frac{\xi - np}{\sqrt{npq}} \tag{14}$$

   и по тем же сериям получить выборку значений η.

4. Построить гистограмму эмпирической плотности распределения η и том же графике изобразить теоретическую плотность стандартного нормального распределения.
5. Построить эмпирическую функцию распределения для η и сравнить её с теоретической функцией распределения стандартного нормального закона.

Сделать выводы:

- как изменяется форма биномиального распределения при увеличении n;
- при каких значениях n нормальная аппроксимация биномиального распределения работает лучше;
- насколько хорошо распределение центрированной и нормированной суммы приближается стандартным нормальным распределением.

Формирование варианта. Номер варианта обозначим через v.

1. Вероятность успеха в одном испытании:

$$p(v) = 0{,}2 + 0{,}1 \cdot (v \bmod 7).$$

2. Число испытаний в серии. Один из возможных вариантов задания параметров:

$$n_1(v) = 25 + (v \bmod 5), \qquad n_2(v) = 2n_1(v), \qquad n_3 = 5n_1(v).$$

3. Число серий (объём выборки для каждого n):

$$N(v) = 1000 + 100 \cdot (v \bmod 6).$$

Теоретическая справка.

1. Схема Бернулли и биномиальное распределение.

Пусть в каждом испытании событие A происходит с вероятностью p, а не происходит – с вероятностью $q = 1 - p$.

Рассматривается серия из n независимых испытаний. Число успехов (осуществлений события A) в n испытаниях обозначим через ξ. Тогда $\xi \in \mathrm{B}_{n,p}$ имеет биномиальное распределение с параметрами n и p, и

$$\mathrm{P}(\xi = k) = C_n^k p^k q^{n-k}, \qquad k = 0,1, \dots, n. \tag{15}$$

Математическое ожидание и дисперсия биномиальной случайной величины:

$$\mathrm{E}\xi = np, \qquad \mathrm{D}\xi = npq. \tag{16}$$

2. Стандартное нормальное распределение.

Случайная величина η имеет стандартное нормальное распределение $\eta \in \mathrm{N}_{0,1}$, если её плотность равна

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad x \in R. \tag{17}$$

Функция распределения обозначается $\Phi_{0,1}(x)$.

3. Центральная предельная теорема (частный случай для схемы Бернулли).

Пусть ξ_1, ξ_2, ..., ξ_n – независимые одинаково распределённые случайные величины, $\mathrm{E}X\xi_1 = \mu$, $\mathrm{D}\xi_1 = \sigma^2 > 0$. Тогда при $n \to \infty$

$$\frac{\xi_1 + \dots + \xi_n - n\mu}{\sigma\sqrt{n}}$$

сходится по распределению к стандартному нормальному закону $\mathrm{N}_{0,1}$.

В схеме Бернулли ξ_1, ξ_2, ..., ξ_n можно считать индикаторами успеха в каждом испытании ($\xi_i = 1$ при успехе и $\xi_i = 0$ при неудаче), тогда $\xi = \xi_1 + \dots + \xi_n$ – число успехов в серии. В этом случае $\mu = p$, $\sigma^2 = pq$ и

$$\eta = \frac{\xi - np}{\sqrt{npq}} \approx \mathrm{N}_{0,1}, \qquad n \to \infty$$

Это частный случай предельной теоремы Муавра-Лапласа для схемы Бернулли.

Описание вычислительного эксперимента.

Для заданного варианта, фиксированного числа экспериментов N и каждого n:

1. Моделирование схемы Бернулли:
   - выполнить N независимых «серий»;
   - в каждой серии выполнить n независимых испытаний схемы Бернулли с вероятностью успеха p;
   - подсчитать число успехов X в каждой серии и сохранить полученные значения $X_1, X_2, ..., X_n$.
2. Эмпирическое биномиальное распределение:
   - по выборке $X_1, X_2, ..., X_n$ подсчитать относительные частоты $\hat{p}_k$:
   - построить столбчатую диаграмму эмпирических вероятностей: по оси абсцисс – значения k, по оси ординат – $\hat{p}_k$;
   - на том же графике изобразить теоретические биномиальные вероятности, рассчитанные по формуле (15).
3. Для каждого N построить два графика:
   - Полная траектория относительной частоты успеха;
   - Начальный участок траектории – первые n_0 испытаний.
4. Сравнение выборочных и теоретических характеристик:
   - вычислить выборочное среднее

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i ;$$

   - вычислить выборочную дисперсию

$$S^2 = \frac{1}{N-1}\sum_{i=1}^{N} (X_i - X)^2 ;$$

   - сравнить их с теоретическими значениями числовых характеристик биномиального распределения, вычисленными по формуле (16);
   - при желании рассчитать абсолютные и относительные погрешности.
5. Центрированная и нормированная сумма:
   - по тем же значениям X_i построить выборку Z_i по формуле (14);
   - построить гистограмму выборки $\{Z_i\}$ с нормировкой по плотности (так, чтобы площадь под гистограммой была равна 1);
   - на том же графике изобразить теоретическую плотность стандартного нормального распределения (17);
   - насколько близко f_n подходит к теоретическому уровню p при больших n.
   - как проявляется на практике закон больших чисел.
6. Эмпирическая функция распределения для Z.
   - отсортировать выборку $\{Z_i\}$;

- построить ступенчатый график эмпирической функции распределения $F_N(z)$, равной доле наблюдений, не превосходящих точку z;
- на том же графике изобразить теоретическую функцию распределения $\Phi_{0,1}(z)$.

7. Сделать выводы по результатам эксперимента:
   - вид биномиального распределения при переходе от n_1 к n_3;
   - качество совпадения эмпирических и теоретических биномиальных вероятностей;
   - форма гистограммы для Z по сравнению с плотностью $\varphi(x)$;
   - насколько хорошо эмпирическая функция распределения $F_N(z)$ приближает $\Phi_{0,1}(z)$ при разных n.

Программная реализация и визуализация в Python.

Для моделирования удобно использовать *numpy.random*: это позволяет векторно генерировать реализации случайных величин и быстро получать выборочные характеристики, частотные таблицы и данные для визуализации. В работе моделируются серии испытаний схемы Бернулли при нескольких значениях числа испытаний, выполняется сравнение эмпирических и теоретических зависимостей, а также анализируется центрированная и нормированная величина.

Сначала подключим библиотеки и зададим функции для оформления подписей осей с десятичной запятой.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
from matplotlib.ticker import FuncFormatter

def comma_fixed(digits=2):
    def _fmt(x, pos=None):
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

def comma_int(x, pos=None):
    return f"{int(round(x))}"
```

Далее зададим параметры варианта.

```
v = 7  # номер варианта

p = 0.2 + 0.1 * (v % 7)
```

```
q = 1 - p

n1 = 25 + (v % 5)
n2 = 2 * n1
n3 = 5 * n1
ns = [n1, n2, n3]

N = 1000 + 100 * (v % 6)

p, n1, n2, n3, N
```

После этого выполним моделирование, вычислим выборочные характеристики и подготовим таблицы эмпирических и теоретических значений, используемые далее при визуализации.

```
rng = np.random.default_rng(42)

def binom_pmf(k, n, p):
    return math.comb(n, k) * (p ** k) * ((1 - p) ** (n - k))

def phi(z):
    return (1.0 / math.sqrt(2 * math.pi)) * np.exp(-0.5 * z**2)

rows_stats = []
blocks = []

eta_sample = None
n_for_eta = n3  # для нормировки берём максимальное n

for n in ns:
    xi = rng.binomial(n=n, p=p, size=N)

    ks = np.arange(0, n + 1)
    counts = np.bincount(xi, minlength=n + 1)
    p_hat = counts / N
    p_theory = np.array([binom_pmf(k, n, p) for k in ks], dtype=float)

    blocks.append(pd.DataFrame({
        "n": n,
        "k": ks,
        "P_hat": p_hat,
        "P_theory": p_theory
```

```
    }))

    mean_emp = float(np.mean(xi))
    var_emp  = float(np.var(xi, ddof=1))
    mean_th  = n * p
    var_th   = n * p * (1 - p)

    rows_stats.append({
        "n": n,
        "x̄": mean_emp,
        "S²": var_emp,
        "E": mean_th,
        "D": var_th
    })

    if n == n_for_eta:
        eta_sample = (xi - n * p) / math.sqrt(n * p * (1 - p))

df_stats = pd.DataFrame(rows_stats)
df_tab = pd.concat(blocks, ignore_index=True)

bins = 25
dens, edges = np.histogram(eta_sample, bins=bins, density=True)
centers = 0.5 * (edges[:-1] + edges[1:])

df_eta = pd.DataFrame({
    "z_center": centers,
    "hist_density": dens,
    "phi_theory": phi(centers)
})

df_stats, df_tab.head(10), df_eta.head(10)
```

Примеры оформления графиков.

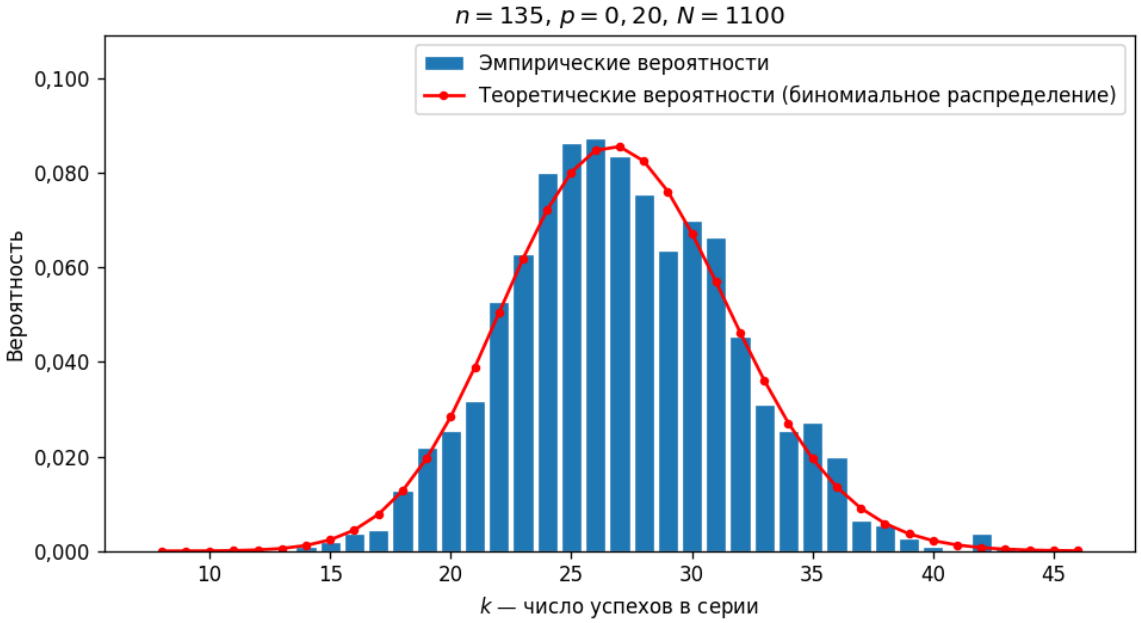


Рис. 10. Биномиальное распределение: эмпирические и теоретические вероятности

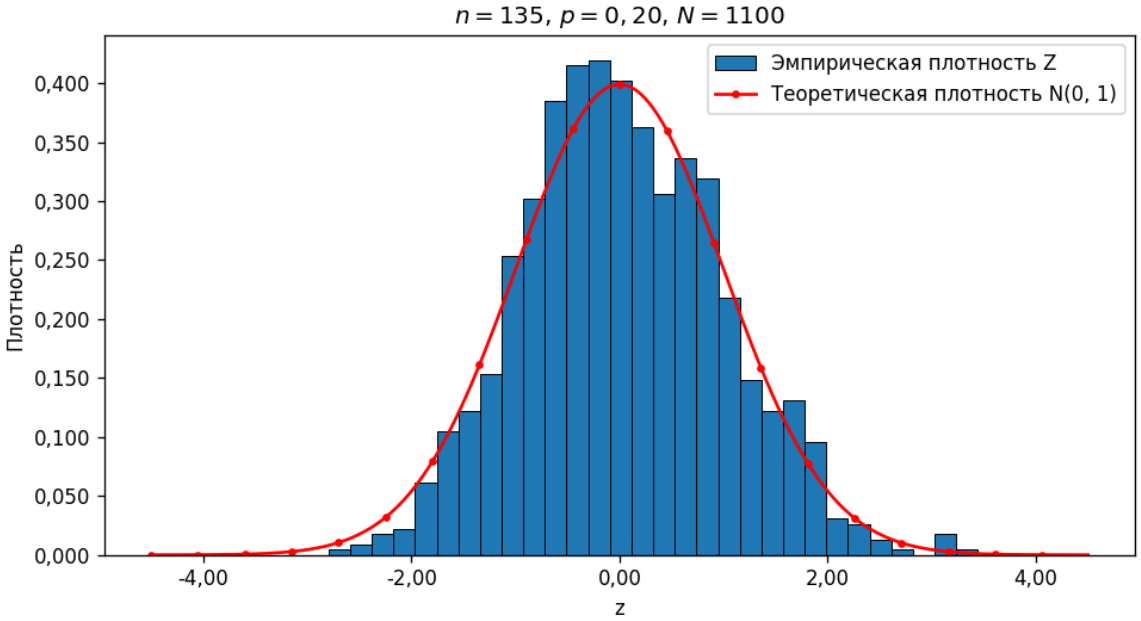


Рис. 11. Центрированная и нормированная сумма: гистограмма и теоретическая плотность стандартного нормального распределения

Лабораторная работа №7. Преобразование равномерной случайной величины в показательную

Цель работы. Ознакомиться с методом генерации показательно распределённой случайной величины с помощью преобразования равномерной случайной величины; на примере проверить моделированием, что полученная выборка действительно соответствует показательному распределению с заданным параметром и выборочные характеристики приближают теоретические.

Условие. Пусть ξ – случайная величина, равномерно распределённая на отрезке $[0;1]$. То есть $\xi \sim \mathrm{U}_{[0;1]}$. Количество экспериментов N и параметр показательного распределения $\lambda > 0$ задаются через номер варианта v (см. раздел «Формирование варианта»).

Рассматривают преобразование

$$\eta = g(\xi). \qquad (36-18)$$

В аналитической части исследования необходимо:

- найти функцию распределения $F_\eta(x)$ и плотность $f_\eta(x)$;
- показать, что η имеет показательное распределение с параметром λ;
- найти теоретические значения математического ожидания и дисперсии случайной величины η.

В вычислительной части требуется:

- сгенерировать N независимых реализаций ξ_1, ξ_2, ..., ξ_N равномерно распределённой случайной величины ξ;
- для каждого i по формуле (18) построить выборку $\eta_1, \eta_2, ..., \eta_N$;
- построить гистограмму выборки η_i, нормированную по плотности, и на том же графике изобразить теоретическую плотность показательного распределения с параметром λ;
- вычислить выборочные характеристики $\overline{\eta}$, S_η^2, медиану и моду (приближённо по гистограмме);
- сравнить выборочные характеристики с теоретическими значениями и сделать вывод о корректности метода генерации показательной случайной величины через равномерную.

Формирование варианта. Номер варианта обозначим через v.

1. Параметр показательного распределения:

$$\lambda(v) = 0{,}5 + 0{,}1 \cdot (v \bmod 6).$$

2. Объём выборки:

$$N(v) = 2000 + 200 \cdot (v \bmod 6).$$

Теоретическая справка. Равномерное и показательное распределения. Преобразование равномерной случайной величины.

1. Равномерное распределение на $[0;1]$.

Случайная величина ξ имеет равномерное распределение на отрезке $[0;1]$, если её плотность равна

$$f_\xi(x) = \begin{cases} 1, & x \in [0;1] \\ 0, & x \notin [0;1] \end{cases}$$

Функция распределения:

$$F_\xi(x) = \begin{cases} 0, & x < 0 \\ x, & 0 \le x \le 1 \\ 1, & x > 1 \end{cases}$$

Математическое ожидание и дисперсия:

$$\mathrm{E}\xi = \frac{1}{2}, \qquad \mathrm{D}\xi = \frac{1}{12}$$

2. Показательное распределение.

Случайная величина η имеет показательное распределение с параметром $\lambda > 0$, если её плотность распределения имеет вид

$$f_\eta(x) = \begin{cases} \lambda e^{-\lambda x}, & x \ge 0 \\ 0, & x < 0 \end{cases} \tag{19}$$

Функция распределения:

$$F_\eta(x) = \begin{cases} 0, & x < 0 \\ 1 - e^{-\lambda x}, & x \ge 0 \end{cases}$$

Математическое ожидание и дисперсия:

$$\mathrm{E}\eta = \frac{1}{\lambda}, \qquad \mathrm{D}\eta = \frac{1}{\lambda^2}. \tag{20}$$

3. Преобразование равномерной случайной величины в показательную.

Пусть $\xi \sim \mathrm{U}_{[0;1]}$. Рассмотрим преобразование

$$\eta = -\frac{1}{\lambda}\ln(1-\xi) \tag{21}$$

Тогда для $x > 0$:

$$P(\eta \le x) = P\left(-\frac{1}{\lambda}\ln(1-\xi) \le x\right) = P\left(1-\xi \ge e^{-\lambda x}\right) = P\left(\xi \le 1 - e^{-\lambda x}\right)$$

Поскольку ξ равномерно распределена на $[0;1]$, для $t \in [0;1]$ – $P(\xi \le t) = t$. Отсюда $F_\eta(x) = P(\eta \le x) = 1 - e^{-\lambda x}$ для $x > 0$, а для $x \le 0$ имеем $F_\eta(x) = 0$. Дифференцируя $F_\eta(x)$, получаем плотность (19)

$$f_\eta(x) = \lambda e^{-\lambda x}, \qquad x > 0.$$

Именно это соотношение используется как обоснование метода моделирования: если сгенерировать $\xi \sim \mathrm{U}_{[0;1]}$ и затем вычислить η по формуле (21), то полученная η должна иметь показательное распределение с параметром λ.

Описание вычислительного эксперимента.

1. Определение параметров варианта.
   - По номеру варианта v вычислить объём выборки N и параметр λ по формулам из раздела «Формирование варианта»;
   - Зафиксировать эти значения в начале отчёта.
2. Генерация равномерной выборки.
   - Сгенерировать N независимых реализаций $\xi_1, \xi_2, ..., \xi_N$ равномерной случайной величины на $[0; 1]$.
3. Преобразование в показательную случайную величину.
   - Для каждого $i = 1, 2, ..., N$ вычислить η_i по формуле (21);
   - Получается выборка $\eta_1, \eta_2, ..., \eta_N$, которая далее рассматривается в качестве выборки из показательного распределения.
4. Графический анализ: гистограмма и теоретическая плотность.
   - Построить гистограмму выборки $\eta_1, \eta_2, ..., \eta_N$ с нормировкой по плотности (площадь под гистограммой равна 1);
   - На том же графике построить теоретическую плотность показательного распределения (19);
   - В отчёте следует кратко прокомментировать гладкость полученной гистограммы.
5. Числовые характеристики по выборке:
   - вычислить выборочное среднее

$$\overline{\eta} = \frac{1}{N}\sum_{i=1}^{N} \eta_i ;$$

   - вычислить выборочную дисперсию

$$S_\eta^2 = \frac{1}{N-1}\sum_{i=1}^{N} (\eta_i - \eta)^2 ;$$

   - найти выборочную медиану (по упорядоченной выборке) и оценить моду по полученной гистограмме;
   - сравнить полученные числовые характеристики распределения с теоретическими значениями, вычисленными по формуле (20);
   - при желании рассчитать абсолютные и относительные погрешности.
6. Сделать выводы по результатам эксперимента:
   - Насколько хорошо гистограмма выборки η_i визуально соответствует теоретической плотности.
   - Насколько близки выборочные характеристики $\overline{\eta}$ и S_η^2 к теоретическим значениям $\mathrm{E}\eta$ и $\mathrm{D}\eta$.
   - Как, по вашему мнению, изменилось бы качество совпадения при увеличении или уменьшении объёма выборки N.

Программная реализация и визуализация в Python.

Для моделирования удобно использовать *numpy.random*: это позволяет векторно генерировать реализации случайных величин и быстро получать выборочные характеристики, частотные таблицы и данные для визуализации. В работе моделируется выборка для показательного распределения, полученная методом преобразования равномерно распределённой случайной величины. Далее вычисляются выборочные характеристики и строится график для сравнения эмпирической плотности с теоретической.

Сначала подключим библиотеки и зададим функции форматирования подписей осей с десятичной запятой.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
from matplotlib.ticker import FuncFormatter

def comma_fixed(digits=2):
    def _fmt(x, pos=None):
        if abs(x - round(x)) < 1e-12:
            return f"{int(round(x))}"
        return f"{x:.{digits}f}".replace(".", ",")
    return FuncFormatter(_fmt)

def fmt_comma(x, digits=2):
    return f"{x:.{digits}f}".replace(".", ",")
```

Далее зададим параметры варианта.

```
v = 7  # номер варианта

lam = 0.5 + 0.1 * (v % 6)

# сокращённый объём выборки (рекомендуемый)
N = 2000 + 200 * (v % 6)

lam, N
```

После этого выполним моделирование: сгенерируем равномерную выборку, применим преобразование и подготовим числовые характеристики и данные для гистограммы.

```
rng = np.random.default_rng(42)
```

```
# равномерная выборка на [0, 1)
u = rng.random(N)

# преобразование в показательную (эквивалентно -ln(u)/lam)
x = - (1.0 / lam) * np.log(1.0 - u)

# выборочные характеристики
x_mean = float(np.mean(x))
x_var  = float(np.var(x, ddof=1))
x_med  = float(np.median(x))

# мода по гистограмме (приближённо)
bins = 30
dens, edges = np.histogram(x, bins=bins, density=True)
centers = 0.5 * (edges[:-1] + edges[1:])
x_mode = float(centers[int(np.argmax(dens))])

# теоретические значения
E_th = 1.0 / lam
D_th = 1.0 / (lam ** 2)
Me_th = math.log(2) / lam  # медиана

df_stats = pd.DataFrame([{
    "N": N, "λ": lam,
    "x̄": x_mean, "S²": x_var, "Me": x_med, "Mo~": x_mode,
    "E": E_th, "D": D_th, "Me_theory": Me_th
}])

# таблица для гистограммы (плотность в центрах)
df_hist = pd.DataFrame({
    "x_left": edges[:-1],
    "x_right": edges[1:],
    "x_center": centers,
    "hist_density": dens
})

df_stats, df_hist.head(10)
```

Примеры оформления графиков.

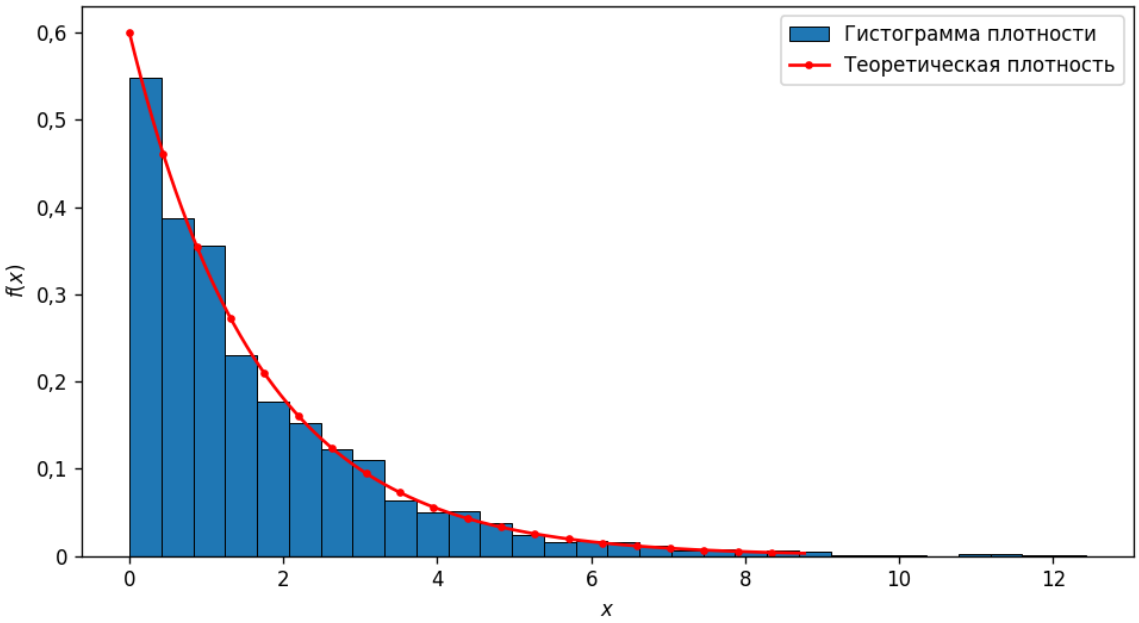


Рис. 12. Гистограмма выборки и теоретическая плотность показательного распределения

Лабораторная работа №8. Совместное распределение и корреляция

Цель работы. Изучить на моделировании совместное распределение пары зависимых случайных величин, иллюстрируя линейную модель, а также на численных примерах показать связь между дисперсией шума, формой «облака точек» на диаграмме рассеяния и величиной коэффициента корреляции.

Условие. Рассматривается линейная модель зависимых случайных величин:

$$Y = aX + \varepsilon.$$

Пусть X и ε – независимые случайные величины:

- X имеет стандартное нормальное распределение $\mathrm{N}_{0,1}$;
- ε имеет нормальное распределение с нулевым средним и дисперсией σ^2, то есть $\varepsilon \sim \mathrm{N}_{0,\sigma^2}$.

Количество экспериментов N, коэффициент a и несколько значений дисперсии шума σ_1^2, σ_2^2, σ_3^2 задаются через номер варианта v (см. раздел «Формирование варианта»).

Для каждого значения σ_k $(k = 1, 2, 3)$ независимо моделируются выборки:

- X_1, X_2, ..., X_N – реализация стандартно нормально распределённой случайной величины X;
- $\varepsilon_1^{(k)}$, $\varepsilon_2^{(k)}$, ..., $\varepsilon_N^{(k)}$ – реализации случайной величины $\varepsilon_k \sim \mathrm{N}_{0,\sigma_k^2}$, не зависимой от X.

Для каждого k рассматривается выборка пар

$$\left(X_i, Y_i^{(k)}\right), \qquad Y_i^{(k)} = aX_i + \varepsilon_i^{(k)}, \qquad i = 1, \dots, N.$$

Необходимо:

1. Для каждого $k = 1, 2, 3$ построить диаграмму рассеяния $\left(X, Y^{(k)}\right)$.
2. Для каждой выборки $\left(X_i, Y_i^{(k)}\right)$ вычислить:
   - выборочные средние $\overline{X}$, $\overline{Y}^{(k)}$;
   - выборочные дисперсии S_X^2, $S_{Y^{(k)}}^2$;
   - выборочную ковариацию $\widehat{cov}\left(X, Y^{(k)}\right)$;
   - выборочный коэффициент корреляции.
3. Сравнить выборочные значения с теоретическими.
4. Сравнить диаграммы рассеяния при разных значениях σ_k: насколько «плотно» точки располагаются вдоль прямой и как меняется «размазанность» облака точек.
5. Сделать выводы:
   - как увеличение дисперсии шума σ^2 влияет на форму облака точек $(X; Y)$;
   - как при росте σ^2 изменяется модуль коэффициента корреляции;

- насколько хорошо выборочные характеристики приближают теоретические.

Формирование варианта. Номер варианта обозначим через v.

1. Число экспериментов (объём выборки пар $(X;Y)$):

$$N(v) = 2000 + 200 \cdot (v \bmod 6).$$

2. Коэффициент a:

$$a(v) = 0{,}5 + 0{,}5 \cdot (v \bmod 4).$$

3. Уровни шума:

$$\sigma_1(v) = 0{,}4 + 0{,}1 \cdot (v \bmod 5), \qquad \sigma_2(v) = 2\sigma_1(v), \qquad \sigma_3(v) = 3\sigma_1(v).$$

Теоретическая справка. Нормальное распределение, ковариация, корреляция, линейная модель.

1. Нормальное распределение.

Случайная величина ξ имеет нормальное распределение с параметрами $a \in R$ и $\sigma > 0$, если её плотность распределения имеет вид

$$f_\xi(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-a)^2}{2\sigma^2}}, \qquad x \in R$$

Математическое ожидание и дисперсия:

$$E\xi = a, \qquad D\xi = \sigma^2$$

2. Линейное преобразование случайной величины.

Пусть ξ – случайная величина, $E\xi = \mu$, $D\xi = \sigma^2$ и пусть $\eta = \alpha\xi + \beta$, где α, β – произвольные числа. Тогда:

$$E\eta = \alpha E\xi + \beta, \tag{22}$$

$$D\eta = \alpha^2 D\xi. \tag{23}$$

Если ξ имеет нормальное распределение, то и η имеет нормальное распределение (линейное преобразование нормальной СВ).

3. Ковариация.

Пусть ξ и η – случайные величины. Ковариация определяется как

$$cov(\xi;\eta) = E[(\xi - E\xi)(\eta - E\eta)].$$

Эквивалентная запись:

$$cov(\xi;\eta) = E[\xi\eta] - E\xi \cdot E\eta.$$

Если ξ и η независимы, то $cov(\xi;\eta) = 0$.

4. Коэффициент корреляции.

Коэффициент корреляции случайных величин ξ и Y определяется формулой

$$\rho_{XY} = \frac{cov(X;Y)}{\sqrt{DX} \cdot \sqrt{DY}}, \qquad |\rho_{XY}| \le 1. \tag{24}$$

Если $|\rho_{XY}|$ близко к 1, линейная связь сильная, а если $|\rho_{XY}|$ близко к 0 – линейная связь слабая.

5. Линейная модель $Y = aX + \varepsilon$.

Пусть:

- $X \sim \mathrm{N}_{0,1}$;
- $\varepsilon \sim \mathrm{N}_{0,\sigma^2}$;
- X и ε независимы;

Тогда:

- Математическое ожидание по формуле (22):

$$EX = 0, E\varepsilon = 0 \Rightarrow EY = E(aX + \varepsilon) = aEX + E\varepsilon = 0.$$

- Дисперсия по формуле (23):

$$DX = 1, D\varepsilon = \sigma^2 \Rightarrow DY = D(aX + \varepsilon) = a^2 DX + D\varepsilon = a^2 + \sigma^2.$$

- Ковариация:

$$cov(X; Y) = cov(X; aX + \varepsilon) = acov(X; X) + cov(X; \varepsilon) = aDX + 0 = a.$$

- Коэффициент корреляции по формуле (24):

$$\rho_{XY} = \frac{cov(X; Y)}{\sqrt{DX} \cdot \sqrt{DY}} = \frac{a}{\sqrt{1} \cdot \sqrt{a^2 + \sigma^2}} = \frac{a}{\sqrt{a^2 + \sigma^2}}$$

При увеличении σ^2 знаменатель растёт, поэтому $|\rho_{XY}|$ уменьшается: облако точек $(X; Y)$ на диаграмме рассеяния становится «более размазанным» вокруг прямой.

Описание вычислительного эксперимента.

1. Определение параметров варианта.
   - По номеру варианта v найти число экспериментов N, коэффициент a и значения σ_1, σ_2, σ_3;
   - Отметить все эти значения в начале отчёта.
2. Генерация выборки X.
   - Сгенерировать N независимых реализаций X_1, X_2, ..., X_N стандартно нормально распределённой случайной величины.
3. Генерация шума и выборок $Y^{(k)}$.
   Для каждого $k = 1, 2, 3$:
   - Сгенерировать N независимых реализаций шума $\varepsilon_1^{(k)}$, $\varepsilon_2^{(k)}$, ..., $\varepsilon_N^{(k)}$, где $\varepsilon_N^{(k)} \sim \mathrm{N}_{0,\sigma_k^2}$;
   - Вычислить значения $Y_i^{(k)} = aX_i + \varepsilon_i^{(k)}$, $i = 1,$ ..., N. В результате получаем три выборки пар $\left(X_i, Y_i^{(k)}\right)$, соответствующие разным уровням шума σ_1, σ_2, σ_3.
4. Диаграммы рассеяния. Для каждого k:

- Построить диаграмму рассеяния: по оси абсцисс – X_i, по оси ординат – $Y_i^{(k)}$;
- Убедиться, что при малом шуме точки располагаются плотнее вдоль некоторой прямой, а при большем шуме облако точек становится более «толстым» и расплывчатым;
- нанести на график прямую зависимости $Y = aX$ (её можно рассматривать как «линию тренда» в рамках заданной модели) и границы интервалов трёх сигм: $Y = aX + 3\sigma_k$ и $Y = aX - 3\sigma_k$.

5. Проверка правила одной, двух и трёх сигм. Для каждого k:
   - для всех точек посчитать вертикальные отклонения от прямой $Y = aX$: $\Delta_i^{(k)} = Y_i^{(k)} - aX_i$;
   - вычислить долю точек, для которых выполняется $\left|\Delta_i^{(k)}\right| \le \sigma_k$, $\left|\Delta_i^{(k)}\right| \le 2\sigma_k$, $\left|\Delta_i^{(k)}\right| \le 3\sigma_k$;
   - оформить результаты в таблицу и сравнить с ориентиром для нормального шума: примерно 68%, 95%, 99,7%.
6. Выборочные характеристики. Для каждой выборки $\left(X_i, Y_i^{(k)}\right)$ вычислить:
   - выборочное среднее

$$\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i\,, \qquad \overline{Y}^{(k)} = \frac{1}{N}\sum_{i=1}^{N} Y_i^{(k)}\,;$$

   - выборочную дисперсию

$$S_X^2 = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - X)^2\,, \qquad S_{Y^{(k)}}^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left({Y^{(k)}}_i - Y^{(k)}\right)^2\,;$$

   - выборочную ковариацию

$$\widehat{cov}\left(X; Y^{(k)}\right) = \frac{1}{N-1}\sum_{i=1}^{N}\left(X_i - \overline{X}\right)\left(Y_i^{(k)} - \overline{Y}^{(k)}\right);$$

   - выборочный коэффициент корреляции

$$\hat{\rho}_{XY^{(k)}} = \frac{\widehat{cov}\left(X; Y^{(k)}\right)}{\sqrt{S_X^2}\cdot\sqrt{S_{Y^{(k)}}^2}}.$$

7. Сравнить полученные выборочные характеристики с теоретическими значениями.
8. Сделать выводы по результатам эксперимента:
   - как изменилось облако точек на диаграмме рассеяния при переходе от σ_1 к σ_3;

- как изменился модуль $|\hat{\rho}_{XY(k)}|$ и насколько он близок к теоретическому $|\rho_{XY(k)}|$;
- подтверждается ли на практике теоретическая формула для коэффициента корреляции ρ_{XY} в линейной модели.

Программная реализация и визуализация в Python.

Для моделирования удобно использовать *numpy.random*: это позволяет векторно генерировать реализации случайных величин и быстро получать выборочные характеристики и данные для визуализации. В работе моделируется линейная зависимость $Y = aX + \varepsilon$ при нескольких уровнях шума и анализируется, как меняются диаграмма рассеяния и коэффициент корреляции.

Сначала подключим библиотеки и зададим функции форматирования подписей осей с десятичной запятой.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import math
from matplotlib.ticker import FuncFormatter

def comma_trim(max_digits=3):
    def _fmt(x, pos=None):
        s = f"{x:.{max_digits}f}".replace(".", ",")
        s = s.rstrip("0").rstrip(",")
        return "0" if s == "-0" else s
    return FuncFormatter(_fmt)
```

Далее зададим параметры варианта.

```
v = 7  # номер варианта

N = 2000 + 200 * (v % 6)
a = 0.5 + 0.5 * (v % 4)

sigma1 = 0.4 + 0.1 * (v % 5)
sigma2 = 2 * sigma1
sigma3 = 3 * sigma1

N, a, sigma1, sigma2, sigma3
```

После этого выполним моделирование и вычислим выборочные характеристики для трёх уровней шума.

```
rng = np.random.default_rng(42)

X = rng.normal(loc=0.0, scale=1.0, size=N)

sigmas = [sigma1, sigma2, sigma3]
rows = []
samples = {}  # будем хранить (X,Y) для каждого sigma

for s in sigmas:
    eps = rng.normal(loc=0.0, scale=s, size=N)
    Y = a * X + eps
    samples[s] = (X, Y)

    x_mean = float(np.mean(X))
    y_mean = float(np.mean(Y))
    x_var  = float(np.var(X, ddof=1))
    y_var  = float(np.var(Y, ddof=1))
    cov_xy = float(np.mean((X - x_mean) * (Y - y_mean)))  # ковариация (с
1/N)
    rho_xy = cov_xy / (math.sqrt(x_var) * math.sqrt(y_var))

    # теоретические значения для модели (X~N(0,1), eps~N(0,s^2))
    # E[X]=0, D[X]=1, E[Y]=0, D[Y]=a^2 + s^2, cov(X,Y)=a,
rho=a/sqrt(a^2+s^2)
    rho_th = a / math.sqrt(a*a + s*s)

    rows.append({
        "sigma": s,
        "x̄": x_mean, "ȳ": y_mean,
        "Sx²": x_var, "Sy²": y_var,
        "cov(X,Y)": cov_xy,
        "rho_hat": rho_xy,
        "rho_theory": rho_th
    })

df_stats = pd.DataFrame(rows)
df_stats
```

Примеры оформления графиков.

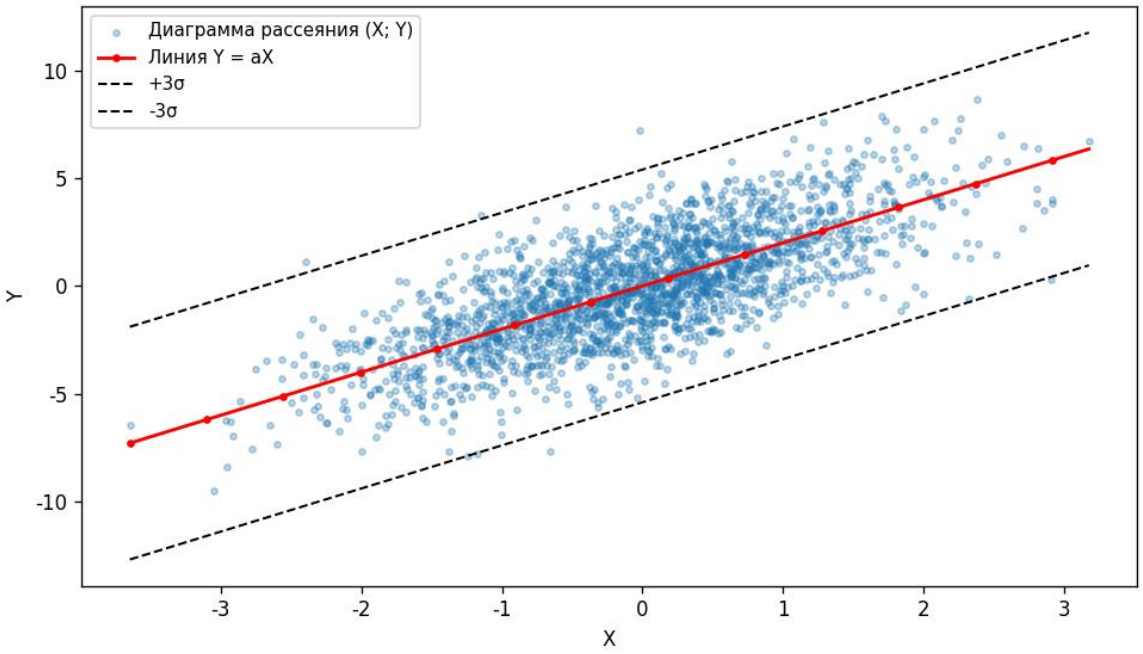


Рис. 13. Диаграмма рассеяния пары $(X; Y)$ при «большом» шуме σ_3

Рекомендации по моделированию, обработке и визуализации результатов средствами Python

В данном разделе приведены примеры программной реализации основных приёмов, используемых в расчётно-графических работах. Эти фрагменты кода:

- не являются готовыми решениями лабораторных работ;
- предназначены как шаблоны и подсказки по работе с языком Python при моделировании случайных величин, построении графиков и обработке результатов эксперимента.

Примеры можно копировать и адаптировать под свои задачи, однако необходимо разбираться в коде и понимать смысл каждой строки программы, уметь объяснять логику построения вычислительного эксперимента.

Также рекомендуется:

- фиксировать параметры варианта (в том числе объёмы выборок);
- задавать генератор случайных чисел с фиксированным значением *seed* для воспроизводимости результатов;
- оформлять рисунки и таблицы аккуратно (подписи, легенда, единый стиль).

В этой ячейке подключаются необходимые библиотеки и задаются базовые параметры оформления рисунков: поддержка кириллицы, сетка, качество сохранения изображений.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import FuncFormatter

plt.rcParams.update({
    "font.family": "DejaVu Sans",
    "axes.unicode_minus": False,
    "axes.grid": True,
    "grid.alpha": 0.35,
    "figure.dpi": 120,
    "savefig.dpi": 300,
    "legend.frameon": True,
})
```

При оформлении результатов экспериментов рекомендуется использовать запятую в качестве разделителя десятичной дроби. Ниже приведены функции, которые задают такой формат подписей осей.

```
def comma_tick(x, pos=None, digits=2):
```

```
    # Подписи осей с десятичной запятой
    if abs(x - round(x)) < 1e-12:
        return f"{int(round(x))}"
    return f"{x:.{digits}f}".replace(".", ",")

def comma_formatter(digits=2):
    return FuncFormatter(lambda x, pos: comma_tick(x, pos, digits=digits))

def apply_comma_axes(ax, digits=2):
    ax.xaxis.set_major_formatter(comma_formatter(digits=digits))
    ax.yaxis.set_major_formatter(comma_formatter(digits=digits))
```

Для воспроизводимости вычислительного эксперимента используют генератор *default_rng(seed)*. Значение *seed* желательно фиксировать в отчёте (например, связать с номером варианта).

```
v = 7  # номер варианта (пример)
seed = 1000 + v
rng = np.random.default_rng(seed)

seed
```

Чтобы элементы на рисунке размещались корректно, рекомендуется перед сохранением вызывать *tight_layout()*, а сохранять с параметром *bbox_inches="tight"*.

```
def save_figure(fig, filename: str):
    fig.tight_layout()
    fig.savefig(filename, bbox_inches="tight")
```

Далее приведены заготовки, которые часто используются при построении траекторий относительных частот и при моделировании схемы Бернулли. Сначала формируется серия испытаний (массив из нулей и единиц), затем по ней вычисляется траектория относительной частоты $\hat{p}_n$. Такой приём применяется, когда нужно проанализировать поведение оценки вероятности при увеличении числа испытаний.

```
def bernoulli_series(rng: np.random.Generator, p: float, N: int) ->
np.ndarray:
    return (rng.random(N) < p).astype(int)

def freq_trajectory(successes: np.ndarray) -> np.ndarray:
    return np.cumsum(successes) / np.arange(1, len(successes) + 1)
```

Для оформления графика траектории удобно использовать единый шаблон: строится линия $\hat{p}_n$, а также проводится горизонтальная линия на уровне теоретического значения вероятности. Подписи осей и легенда должны быть осмысленными, а формат чисел на осях задаётся через функцию *apply_comma_axes*, приведённую выше.

```
def plot_trajectory(p_hat: np.ndarray, p_theory: float, title: str,
filename: str | None = None):
    n = np.arange(1, len(p_hat) + 1)

    fig, ax = plt.subplots(figsize=(8.2, 4.6))
    ax.plot(n, p_hat, linewidth=1.6, label="Относительная частота")
    ax.axhline(p_theory, linestyle="--", linewidth=1.6,
label="Теоретическое значение")

    ax.set_xlabel("Номер испытания n")
    ax.set_ylabel("Относительная частота")
    ax.set_title(title)
    ax.legend()

    apply_comma_axes(ax, digits=2)

    if filename is not None:
        save_figure(fig, filename)
    plt.show()
```

В задачах с дискретными распределениями часто требуется получить эмпирические вероятности значений ξ по результатам моделирования. Для этого удобно использовать *np.bincount*, который подсчитывает частоты появления значений. Далее эмпирические вероятности можно сравнивать с теоретическими вероятностями (например, для биномиального или пуассоновского распределения).

```
from math import comb, exp, factorial

def pmf_binomial(k: np.ndarray, n: int, p: float) -> np.ndarray:
    return np.array([comb(n, int(kk)) * (p**kk) * ((1 - p)**(n - kk)) for
kk in k], dtype=float)

def pmf_poisson(k: np.ndarray, lam: float) -> np.ndarray:
    return np.array([exp(-lam) * (lam**int(kk)) / factorial(int(kk)) for
kk in k], dtype=float)
```

```
def empirical_pmf(samples: np.ndarray, k_max: int | None = None) ->
np.ndarray:
    if k_max is None:
        k_max = int(np.max(samples))
    counts = np.bincount(samples.astype(int), minlength=k_max + 1)
    return counts / samples.size
```

Ниже приведён шаблон построения рисунка для сравнения эмпирических вероятностей с одной или несколькими теоретическими зависимостями. Эмпирические вероятности изображаются столбцами, теоретические – линиями с маркерами.

```
def plot_discrete_compare(k: np.ndarray, p_emp_full: np.ndarray,
                          theory_list: list[tuple[str, np.ndarray]],
                          title: str, filename: str | None = None):
    fig, ax = plt.subplots(figsize=(8.2, 4.6))

    p_emp = p_emp_full[k]
    ax.bar(k, p_emp, alpha=0.6, label="Эмпирические вероятности")

    for name, p_th in theory_list:
        ax.plot(k, p_th, marker="o", linewidth=1.3, label=name)

    ax.set_xlabel("k")
    ax.set_ylabel("P(ξ = k)")
    ax.set_title(title)
    ax.legend()

    apply_comma_axes(ax, digits=3)

    if filename is not None:
        save_figure(fig, filename)
    plt.show()
```

Для непрерывных распределений обычно строят гистограмму с нормировкой по плотности *(density=True)* и накладывают график теоретической плотности $f(x)$. Диапазон по оси x имеет смысл задавать осмысленно, чтобы на рисунке не доминировали редкие «хвостовые» наблюдения.

```
def plot_continuous_hist(samples: np.ndarray, pdf,
                         bins: int = 25,
                         x_min: float | None = None,
```

```
                          x_max: float | None = None,
                          title: str = "",
                          filename: str | None = None):
    fig, ax = plt.subplots(figsize=(8.2, 4.6))

    ax.hist(samples, bins=bins, density=True, alpha=0.6,
label="Гистограмма (плотность)")

    if x_min is None:
        x_min = float(np.min(samples))
    if x_max is None:
        x_max = float(np.max(samples))

    xs = np.linspace(x_min, x_max, 400)
    ax.plot(xs, pdf(xs), linewidth=1.8, label="Теоретическая плотность")

    ax.set_xlabel("x")
    ax.set_ylabel("f(x)")
    ax.set_title(title)
    ax.legend()

    apply_comma_axes(ax, digits=3)

    if filename is not None:
        save_figure(fig, filename)
    plt.show()
```

Для сравнения распределений полезна эмпирическая функция распределения. Она строится по отсортированной выборке и представляет собой ступенчатую функцию. При необходимости на тот же рисунок можно добавить теоретическую функцию распределения $F(x)$.

```
def ecdf(samples: np.ndarray):
    x = np.sort(samples)
    y = np.arange(1, len(x) + 1) / len(x)
    return x, y

def plot_ecdf(samples: np.ndarray, F_theory=None,
              title: str = "", filename: str | None = None):
    x, y = ecdf(samples)

    fig, ax = plt.subplots(figsize=(8.2, 4.6))
```

```
    ax.step(x, y, where="post", linewidth=1.6, label="Эмпирическая функция
распределения")

    if F_theory is not None:
        xs = np.linspace(float(x[0]), float(x[-1]), 400)
        ax.plot(xs, F_theory(xs), linewidth=1.6, label="Теоретическая
функция распределения")

    ax.set_xlabel("x")
    ax.set_ylabel("F(x)")
    ax.set_title(title)
    ax.legend()

    apply_comma_axes(ax, digits=3)

    if filename is not None:
        save_figure(fig, filename)
    plt.show()
```

При обработке результатов часто требуются выборочные характеристики: среднее, дисперсия, медиана и мода. Для дискретных данных моду удобно определять как наиболее часто встречающееся значение. Для непрерывных данных можно использовать приближённую оценку моды по гистограмме.

```
def sample_mode_discrete(x: np.ndarray) -> int:
    counts = np.bincount(x.astype(int))
    return int(np.argmax(counts))

def sample_mode_hist(x: np.ndarray, bins: int = 25) -> float:
    dens, edges = np.histogram(x, bins=bins, density=True)
    idx = int(np.argmax(dens))
    return float(0.5 * (edges[idx] + edges[idx + 1]))

def sample_stats(x: np.ndarray, mode_type: str = "discrete",
bins_for_mode: int = 25):
    mean = float(np.mean(x))
    var  = float(np.var(x, ddof=1))
    med  = float(np.median(x))

    if mode_type == "discrete":
        mode = float(sample_mode_discrete(x))
    else:
        mode = float(sample_mode_hist(x, bins=bins_for_mode))
```

```
    return mean, var, med, mode
```

Для оформления результатов удобно собирать вычисленные показатели в таблицу *DataFrame*. Если требуется выгрузить таблицу в CSV так, чтобы числа записывались с запятой (например, для открытия в Excel), используют разделитель «;» и параметр *decimal=","*.

```
def to_report_table(rows: list[dict]) -> pd.DataFrame:
    return pd.DataFrame(rows)

def export_csv_ru(df: pd.DataFrame, filename: str):
    df.to_csv(filename, index=False, sep=";", decimal=",")
```

В некоторых экспериментах на первых шагах могут возникать недопустимые операции (например, деление на ноль или логарифм неположительного числа). В таких случаях полезно явно контролировать предупреждения и заменять некорректные значения на *np.nan*, чтобы не ломать построение графиков.

```
def safe_divide(a: np.ndarray, b: np.ndarray, fill_value=np.nan) ->
np.ndarray:
    with np.errstate(divide="ignore", invalid="ignore"):
        y = a / b
    y[~np.isfinite(y)] = fill_value
    return y
```

Список литературы

1. Чернова, Н. И. Введение в теорию вероятностей : учебное пособие / Н. И. Чернова. — 2-е изд. — Москва : ИНТУИТ, 2016. — 170 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/100736
2. Вентцель, Е. С. Теория вероятностей : учебник для студентов вузов. — 8-е изд., стер. — Москва : Высшая школа, 2002. — 575 с. — ISBN 5-06-003650-2.
3. Гмурман, В. Е. Теория вероятностей и математическая статистика : учебник для прикладного бакалавриата / В. Е. Гмурман. — 12-е изд. — Москва : Издательство Юрайт, 2016. — 479 с. — (Бакалавр. Прикладной курс). — ISBN 978-5-9916-6484-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/389102
4. Хилл, К. Научное программирование на Python / пер. с англ. А. Снастина. — Москва : ДМК Пресс, 2021. — 646 с. — ISBN 978-5-97060-914-9.
5. МакКинни, У. Python и анализ данных / пер. с англ. А. А. Слинкиной. — 3-е изд. — Москва : ДМК Пресс, 2023. — 536 с. — ISBN 978-5-93700-174-0.
6. Сомова, М. Н., Беличенко, О. М. Компьютерное моделирование в обучении теории вероятностей // Актуальные проблемы преподавания математики в техническом вузе. — 2019. — № 7. — С. 299–303. — DOI: 10.25206/2307-5430-2019-7-299-303.

Смирнов Игорь Сергеевич
Лукина Марина Владимировна
Митина Татьяна Евгеньевна

Компьютерное моделирование в задачах теории вероятностей

Учебно-методическое пособие

В авторской редакции
Редакционно-издательский отдел Университета ИТМО
Зав. РИО Н.Ф. Гусарова
Подписано к печати
Заказ №
Тираж
Отпечатано на ризографе

Редакционно-издательский отдел
Университета ИТМО
197101, Санкт-Петербург, Кронверкский пр., 49, литер А