Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования
Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Отчет по лабораторной работе

«Моделирование случайных величин и проверка гипотез о виде распределения»

Выполнила:
студентка группы 381903_3
Троегубова А. А.

Проверил:
Преподаватель кафедры ТВиАД,
Кудрявцев Е. В.

Нижний Новгород
2022

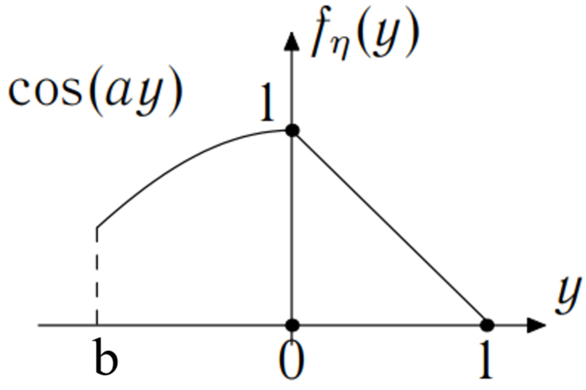
Оглавление

Введение

В данной лабораторной работе рассматривается моделирование случайной величины с заданным законом распределения непрерывного типа, для которой вычисляются её выборочное распределение и выборочные числовые характеристики. Кроме того, с помощью критерия согласия хи-квадрат, проверяются гипотезы о виде распределения смоделированной случайной величины.

Постановка задачи

Вариант 23: Плотность распределения с. в. η задана графически:



Моделирование случайной величины

Постановка задачи

1. Получить номер варианта от преподавателя и ознакомиться с текстом задачи в Приложении.
2. Изучить теоретический материал части 1 и выбрать подходящий способ решения задачи.
3. Написать первую часть программы – «розыгрыш» значений случайной величины. Эта часть должна включать в себя отображение содержания задачи, пользовательский интерфейс для ввода необходимых параметров, вывод результатов.

Для непрерывной случайной величины значения требуется расположить в порядке возрастания: $x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(N)}$. Здесь $x_{(j)}$ — j-е по возрастанию число среди наблюдений $x_{(1)}, x_{(2)}, \dots, x_{(N)}$.

y_i	y_1	y_2	$\cdots$	y_k
n_i	n_1	n_2	$\cdots$	n_k
$\frac{n_i}{n}$	$\frac{n_1}{n}$	$\frac{n_2}{n}$	$\cdots$	$\frac{n_k}{n}$

Здесь $n_j =$число$\{x_i: x_i = y_j, i = 1,2 \dots n\}$.

Общее описание программы

Программа реализована на языке Python с использованием среды разработки Jupyter Notebook.

Непрерывная случайная величина задана своей плотностью распределения $f_\eta(y)$. Функция распределения $F_\eta(y)$ непрерывна, она монотонно возрастает на некотором интервале (y_{min}, y_{max}), $-\infty < y_{min} < y_{max} < +\infty$ и постоянна вне его. Рассмотрим функцию G(t): (0, 1) $\rightarrow (y_{min}, y_{max})$, обратную к $F_\eta(y)$ для $y \in (y_{min}, y_{max})$. Тогда случайная величина G(t) имеет функцию распределения $F_\eta(y)$.

Идея моделирования заключается в биективном отображении интервала (0; 1) на числовую ось OY (в рамках задачи является горизонтальной осью, соответствующей значениям с. в.). В качестве правила отображения принимаем обратную функцию $G(t)$ к функции распределения н. с. в $F_\eta(y)$, которую находится по известной плотности распределения $f_\eta(y)$. Вид функций отображен на рисунке 1.

Плотность распределения н. с. в.

$$f_\eta(y) = \begin{cases} 0 & y \le -\frac{1}{a} arcsin\frac{a}{2} \\ cos(ay) & -\frac{1}{a} arcsin\frac{a}{2} \le y \le 0 \\ -y+1 & 0 \le y \le 1 \\ 0 & y \ge 1 \end{cases}$$

Функция распределения н. с. в.

$$F_\eta(y) = \begin{cases} 0 & y \le -\frac{1}{a} arcsin\frac{a}{2} \\ \frac{1}{a} sin(ay) - \frac{1}{2} & -\frac{1}{a} arcsin\frac{a}{2} \le y \le 0 \\ -\frac{y^2}{2} + y + \frac{1}{2} & 0 \le y \le 1 \\ 1 & y \ge 1 \end{cases}$$

Обратная функция к функции распределения н. с. в.

$$G(t) = \begin{cases} \frac{1}{a} arcsin(at + sin\frac{a}{2}) & 0 \le t \le 0.5 \\ -\sqrt{2(1-t)} + 1 & 0.5 \le t \le 1 \end{cases}$$

Рисунок 1.

Описание алгоритмов

Сперва программа запрашивает у пользователя значение count_it - размер выборки и значение параметра a такое, что $|a| \le 2$, данное условие было выведено из условия нормировки $\int_{-\infty}^{+\infty} f_\eta(y)\, dy = 1 \quad => \int_b^0 \cos(ay)\, dy = \frac{1}{2}$. Затем в цикле count_it раз находится значение с. в. Для нахождения одного значения необходимо при помощи генерации случайного целого числа U, получить случайное значение F, принадлежащее интервалу (0, 1). А затем отображаем F на ось OY, согласно обратной ф-ии G(t). В результате работы цикла, получаем набор значений с. в., затем сортируем его для вывода в таблицу 1. Алгоритм изображен на рисунке 2. Полный алгоритм данной части работы находится в приложении 1.

```
for i in range(0, count_it):
    U = random.randint(0, N)
    F = U/N
    u[i] = U
    f[i] = F
    if F < 0.5:
        y[i] = (1/a)* math.asin(a * F - a / 2)
    else:
        y[i] = 1 - math.sqrt(2 - 2 * F)

order = y.argsort()
ind = order.argsort()
```

Рисунок 2.

Результат работы алгоритма отображается в таблице 1, где первый столбец соответствует значению случайной величины, а третий – значению вероятности, то есть значению выборочной функции распределения.

	Значения с. в.	Рандомное число U	U/N на [0, 1]
0	-0.558744	4410181.0	0.004410
1	-0.557954	4939255.0	0.004939
2	-0.556872	5664233.0	0.005664
3	-0.556309	6042243.0	0.006042
4	-0.552911	8331066.0	0.008331
...	...	...	...
995	0.887146	993632008.0	0.993632
996	0.920027	996802134.0	0.996802
997	0.929392	997507262.0	0.997507
998	0.944703	998471109.0	0.998471
999	0.960632	999225090.0	0.999225

Таблица 1.

Статистические характеристики случайных величин

Постановка задачи

Определить теоретические и выборочные числовые характеристики: Eη, Dη, $\bar{x}, S^2$, $\widehat{Me}$, $\widehat{R}$. Составить таблицу:

Eη	$\bar{x}$	$\lvert E\eta - \bar{x} \rvert$	Dη	S^2	$\lvert D\eta - S^2 \rvert$	$\widehat{Me}$	$\widehat{R}$
…	…	…	…	…	…	…	…

Построить графики теоретической $F_\eta(y)$ и выборочной $\widehat{F}_\eta(y)$ функций распределения. Вычислить меру их расхождения D.

Определить из условий задачи закон распределения случайной величины η, если он не указан явно. Организовать ввод границ промежутков $\Delta'_1, \Delta'_2, \ldots, \Delta'_k$. Построить гистограмму. Вычислить теоретическую плотность распределения $f(z_j)$ в точке z_j — середине промежутка Δ'_j. Результаты оформить в виде таблицы:

z_j	z_1	z_2	$\ldots$	z_k
$f_\eta(z_j)$	$f_\eta(z_1)$	$f_\eta(z_2)$	$\ldots$	$f_\eta(z_k)$
$\dfrac{n_j}{n\lvert\Delta'_j\rvert}$	$\dfrac{n_1}{n\lvert\Delta'_1\rvert}$	$\dfrac{n_2}{n\lvert\Delta'_2\rvert}$	$\ldots$	$\dfrac{n_k}{n\lvert\Delta'_k\rvert}$

$$\max_{j=\overline{1,\,k}} \left| \frac{n_j}{n\lvert\Delta'_j\rvert} - f_\eta(z_j) \right| = \ldots$$

Общее описание программы

Программа реализована на языке Python с использованием среды разработки Jupyter Notebook.

Для составления таблицы числовых характеристик необходимо:

1. Вычислить аналитически значения числовых характеристик.

Математическое ожидание с. в. η:

$$E = \frac{1}{a^2} + \frac{\sqrt{1-\left(\frac{a}{2}\right)^2}}{a^2} - \frac{1}{2a}\arcsin\left(\frac{a}{2}\right) + \frac{1}{6}$$

Дисперсия с. в. η:

$$D = \frac{1}{2a^2}\left(\arcsin\frac{a}{2}\right)^2 + \frac{2}{a^3}\arcsin\frac{a}{2}\sqrt{1-\left(\frac{a}{2}\right)^2} - \frac{1}{a^2} + \frac{1}{12} - E^2$$

2. Определить выборочные аналоги числовых характеристик η:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ — выборочное среднее,}$$

$$S^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2 \text{ — выборочная дисперсия,}$$

$$\widehat{R} = x_{(n)} - x_{(1)} \text{ — размах выборки,}$$

$$\widehat{Me} = x_{(k+1)} \text{ при } n = 2k+1 \text{ или}$$

$$\widehat{Me} = (x_{(k)} + x_{(k+1)})/2 \text{ при } n = 2k \text{ — выборочная медиана.}$$

Пусть $x_{(1)}, x_{(2)} \dots x_{(N)}$ — выборочные значения случайной величины η. По аналогии с функцией распределения $F_\eta(y)$ случайной величины введём выборочную функцию распределения:

$$\widehat{F}_\eta(x) = \frac{1}{n} \times \text{число}\{x_i\colon x_i < x,\ i = 1,\ 2,\ \dots,\ n\}.$$

Полезно вычислить величину: мера расхождения $F_\eta(y)$ и $\hat{F}_\eta(y)$

$$D = \max_{-\infty<x<\infty} |\widehat{F}_\eta(x) - F_\eta(x)|.$$

Учитывая тот факт, что функция распределения не убывает, а выборочная функция распределения имеет конечное число скачков, величину D можно вычислять по формуле:

$$D = \max_{1\leqslant j\leqslant n}\left(\frac{j}{n} - F_\eta(x_{(j)}+0),\ F_\eta(x_{(j)}) - \frac{j-1}{n}\right),$$

Где $x_{(1)} \le x_{(2)} \le \cdots \le x_{(N)}$ - упорядоченные по возрастанию значения $x_{(1)}, x_{(2)} \dots x_{(N)}$.

Для построения гистограммы необходимо разбить числовой промежуток, в который попали наблюдения $x_{(1)}, x_{(2)} \dots x_{(N)}$, на примыкающие промежутки $\Delta'_1, \Delta'_2, \dots, \Delta'_k$. Пусть n_i = число $\{x_i\colon x_i \in \Delta'_j,\ i = 1, 2, \dots n\}$. В плоскости XOY на оси OX отметим промежутки $\Delta'_1, \Delta'_2, \dots$,

Δ'_k и над j-м из них построим прямоугольник высотой $\dfrac{n_i}{n_i|\Delta'_j|}$. Гистограммой называют фигуру, составленную из этих прямоугольников. Гистограмма является аналогом функции плотности распределения, поэтому площадь гистограммы всегда равна 1. Подробные алгорится

Результат работы алгоритмов

На рисунках 2 и 3:

График теоретической функции распределения $F_\eta(y)$ – зелёный.

График выборочной функции распределения $\hat{F}_\eta(y)$ – синий.

Красная точка – значение с. в., при котором $F_\eta(y)$ и $\hat{F}_\eta(y)$ имеют максимальное расхождение *D*.

Желтая линия – вертикаль, проведённая, через максимальное расхождение *D*, изображена для наглядности величины *D*.

Видно, что с увеличением выборки $\hat{F}_\eta(y)$ стремится к $F_\eta(y)$. Код алгоритма построения $F_\eta(y)$ и $\hat{F}_\eta(y)$ и их отрисовки указан в приложении 2.

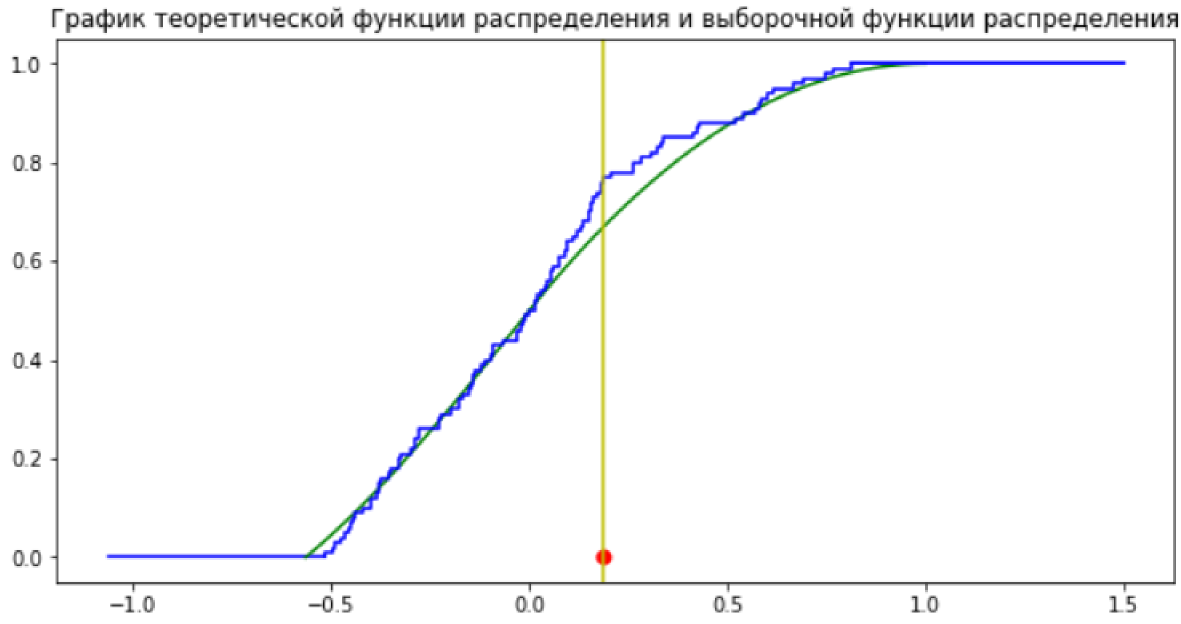


Рисунок 2. Выборка равна 100.

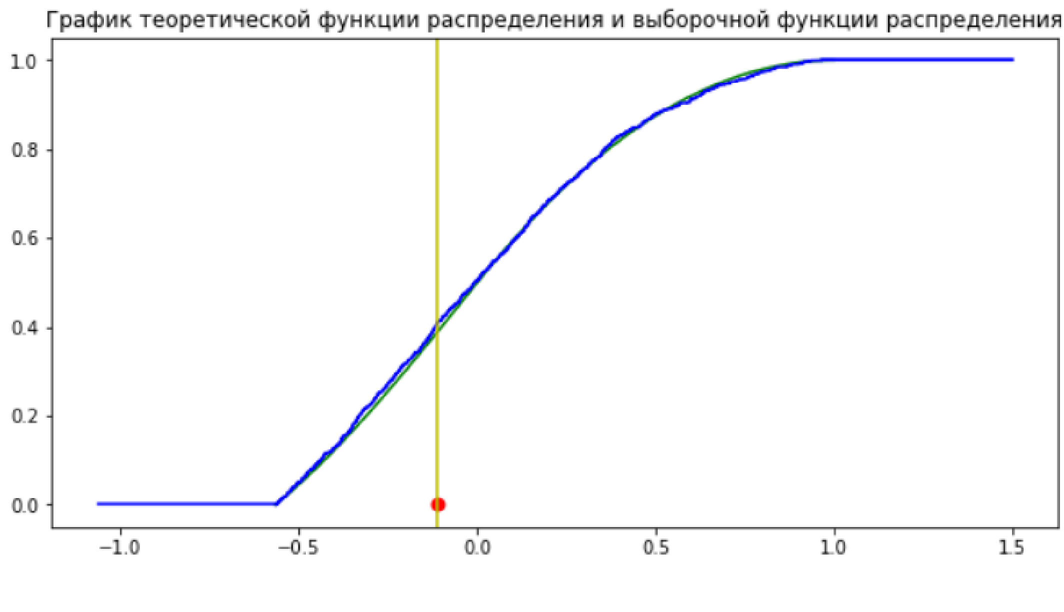


Рисунок 3. Выборка равна 1000.

На таблицах 2 и 3 представлены теоретические и выборочные числовые характеристики для выборка равной 100 и 1000 соответственно. Видно, что с увеличением выборки выборочные среднее и выборочная дисперсия стремятся к своим теоретическим аналогам. Код расчёта числовых характеристик и вывод таблицы указан в приложении 2.

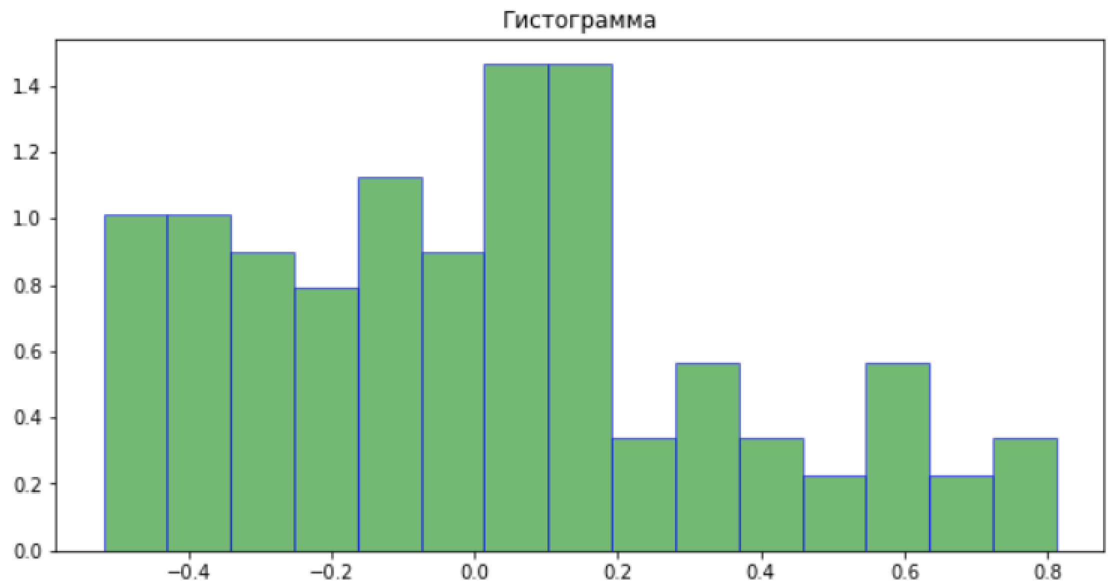
	Мат. ожидание	Выборочное среднее	Модуль разности мат. ожидания и выборочного среднего	Дисперсия	Выборочная дисперсия	Модуль разности дисперсии и выборочной дисперсии	Выборочная медиана	Размах выборки	Мера расхождения D
0	0.034451	0.010944	0.023507	0.129935	0.111243	0.018692	0.013408	1.3301	0.103172

Таблица 2. Выборка равна 100.

	Мат. ожидание	Выборочное среднее	Модуль разности мат. ожидания и выборочного среднего	Дисперсия	Выборочная дисперсия	Модуль разности дисперсии и выборочной дисперсии	Выборочная медиана	Размах выборки	Мера расхождения D
0	0.034451	0.029559	0.004892	0.129935	0.135783	0.005848	-0.006543	1.532163	0.02184

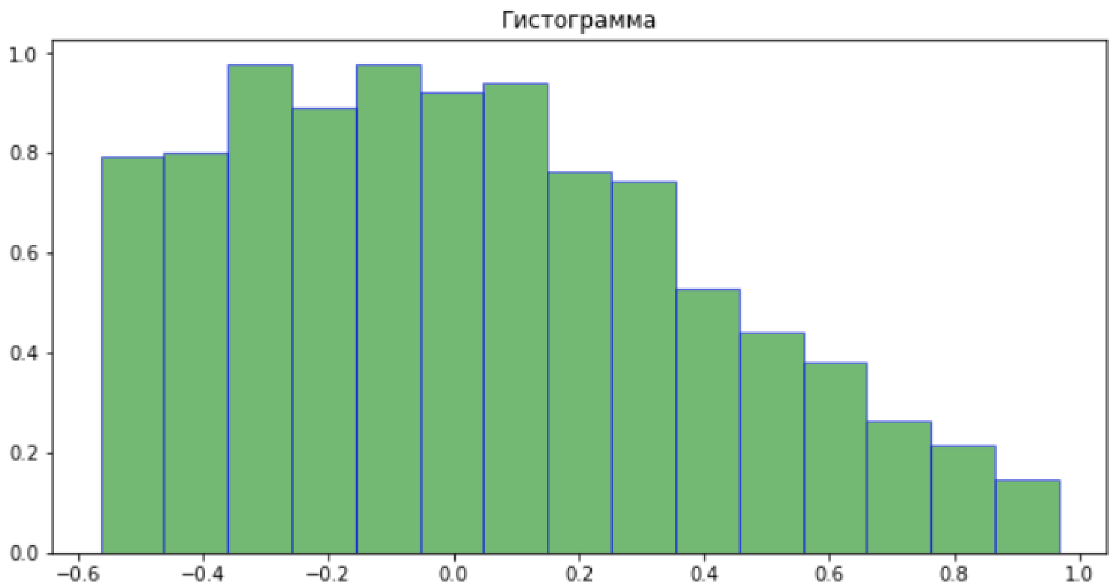
Таблица 3. Выборка равна 1000.

На рисунках 4 и 5 представлены гистограммы для выборок равных 100 и 1000 соответственно. Количество промежутков равно 15, причем промежутки равны. Однако пользователь может ввести своё количество промежутков и их границы. В таблицах 4 и 5 представлены сведенья о значениях теоретической плотности распределения и значениях гистограммы в точках – серединах промежутков. Видно, что при увеличении выборки (и при увеличении кол-ва промежутков) гистограмма стремится к $f_\eta(y)$, максимальная мера расхождения теоретической плотности распределения и гистограммы стремится к 0. Код построения гистограммы, её отрисовка и вывод таблицы 4 (5) указан в приложении 2.




```
Максимальная мера расхождения гистограммы и теор. плотности распределения: 0.6130323478495265
```

Рисунок 4. Выборка равна 100.




```
Максимальная мера расхождения гистограммы и теор. плотности распределения: 0.0850857193007919
```

Рисунок 5. Выборка равна 1000.

	Значение точки - середины j-ого промежутка Zj	Значение теор. плотности распределения в точке Zj	Значение гисторгамы в точке Zj
0	-0.473736	0.757968	1.014961
1	-0.385063	0.837779	1.014961
2	-0.296389	0.902790	0.902188
3	-0.207716	0.951852	0.789414
4	-0.119043	0.984100	1.127735
...	...	...	...
10	0.412998	0.587002	0.338320
11	0.501671	0.498329	0.225547
12	0.590344	0.409656	0.563867
13	0.679018	0.320982	0.225547
14	0.767691	0.232309	0.338320

Таблица 4. Выборка равна 100.

	Значение точки - середины j-ого промежутка Zj	Значение теор. плотности распределения в точке Zj	Значение гисторгамы в точке Zj
0	-0.514139	0.717069	0.792996
1	-0.411995	0.815044	0.802787
2	-0.309850	0.893922	0.979008
3	-0.207706	0.951857	0.890897
4	-0.105562	0.987490	0.979008
...	...	...	...
10	0.507303	0.492697	0.440554
11	0.609448	0.390552	0.381813
12	0.711592	0.288408	0.264332
13	0.813736	0.186264	0.215382
14	0.915880	0.084120	0.146851

Таблица 5. Выборка равна 1000.

Проверка гипотезы о виде распределения

Постановка задачи:

Заключительная часть программы должна включать в себя:

1. Ввод числа k интервалов $\Delta_1" = (-\infty, z_1), \Delta_2" = [z_1, z_2), . . . , \Delta_k" = [z_{k-1}, \infty)$. Выбор границ интервалов z_1 , $z_2, \dots z_{k-1}$.

2. Отображение гипотезы в виде теоретических вероятностей q_1 , $q_2, \dots q_k$.

3. Ввод уровня значимости α .

4. Отображение вычисленного значения $\bar{F}(R_0)$ и решения о принятии или отвержении гипотезы H_0.

Общее описание программы:

Программа реализована на языке Python с использованием среды разработки Jupyter Notebook.

Предположение о распределении, которому подчиняются выборочные значения, будем называть гипотезой. Проверяемая гипотеза называется нулевой гипотезой и обозначается H_0.

Под статистическим критерием для проверки нулевой гипотезы понимается правило (функция), которое каждому набору выборочных значений $x_1 x_2, \dots, x_n$, приписывает решение «принять нулевую гипотезу» или «отклонить нулевую гипотезу». Чтобы задать такое правило, достаточно задать разбиение множества наборов выборочных значений на два подмножества — область принятия гипотезы и область отклонение гипотезы. Область отклонения гипотезу называют критической областью.

Рассмотрим подробнее критерий χ^2для проверки согласия данных с распределением

$F\eta$(x). Разобьём числовую ось на интервалы $\Delta_1" = (-\infty, z_1), \Delta_2" = [z_1, z_2), . . . , \Delta_k" = [z_{k-1}, \infty)$. Интервалы следует выбирать так, чтобы каждый содержал хотя бы одну точку, а лучше несколько.

В качестве статистики критерия выберем величину

$$R_0 = \sum_{j=1}^{k} \frac{(n_j - nq_j)^2}{nq_j}$$

Величина R_0 характеризует меру расхождения между наблюдавшимися частотами и ожидаемым числом попаданий в интервал при нулевой гипотезы. Не согласующимися с нулевой гипотезой являются значения больше R_0. 0 . Выберем критическую область вида ($\chi^2_{\alpha;k-1}$,∞). Пусть $F_{\chi^2_{k-1}}(x)$ − функция распределения χ^2, тогда рассмотрим $\bar{F}(x) = 1 - F_{\chi^2_{k-1}}(x)$ − невозврастающая. Значит, $R_0 > \chi^2_{\alpha;k-1}$ тогда и только тогда, когда $\bar{F}(R_0) < \bar{F}(\chi^2_{\alpha;k-1}) = \alpha$.

Формулы для численных расчётов

Плотность распределения χ^2 с r степенями свободы имеет вид:

$$f_{\chi^2_r}(x) = \begin{cases} 0, & x \leqslant 0, \\ 2^{-r/2}[\Gamma(r/2)]^{-1} x^{r/2-1} e^{-x/2}, & x > 0. \end{cases}$$

Исходя из вида плотности, вычислить

$$\overline{F}(R_0) = \int_{R_0}^{\infty} f_{\chi^2_r}(x)\,dx = 1 - \int_0^{R_0} f_{\chi^2_r}(x)\,dx$$

можно, например, методом трапеций:

$$\int_a^b g(x)\,dx \approx \sum_{k=1}^{n} \Big(g\Big(a + (b-a)\frac{k-1}{n}\Big) + g\Big(a + (b-a)\frac{k}{n}\Big)\Big) \frac{b-a}{2n}.$$

Результат работы алгоритмов:

Так как случайная величина смоделирована по конкретной теоретической функции распределения, то решение о принятии гипотезы должно соответствовать критерию уровня значимости α.

Тогда для проверки случайной величины на правильность моделирования применим критерий χ^2 для ста различных выборок, величина каждой из них равна 1000. Каждую выборку будем проверять трижды, с различным уровнем критерием значимости $\alpha_1 = 0.1\,, \alpha_2 = 0.5, \alpha_3 = 0.9$. На рисунке 6 отображен результат проверок. Видно, что примерно α * (кол-во выборок) = 100α выборок отклоняется, и $100(1-\alpha)$ выборок принимаются. Это значит, что программа работает корректно, случайная величина замоделирована правильно. Код алгоритма принятия гипотез о распределении с. в. указан в приложении 3.

```
alfa = 0.1: Гипотиза принята  89  раз; Гипотеза отклонена  11  раз
alfa = 0.5: Гипотиза принята  52  раз; Гипотеза отклонена  48  раз
alfa = 0.9: Гипотиза принята  11  раз; Гипотеза отклонена  89  раз
```

Рисунок 6.

Кроме того, пользователь может одну конкретную выборку на соответствии какому-либо распределению. Для этого программа просит ввести количество промежутков k и значение уровня значимости α. Чтоб избежать случая, когда в промежуток не попадает ни одного значения с. в., границы промежутков вычисляются так, чтоб попадание в них с. в. было равновероятно. В результате проверки (рисунок 7) на принятие гипотезы, программа оглашает вердикт, выводит значение статистики R_0 и таблицу теоретических вероятностей промежутков, как было замечено выше, они равны между собой. Код алгоритма принятия гипотезы о распределении с. в. и вывод результатов (рисунок 7) указан в приложении 3.

```
Введите количество промежутков k: 10
Введите уровень значимости alfa: 0.1
F(R0) с чертой:  0.999999992647553
Гипотеза H0 принята :)
```

	Значения теоретических вероятности qj, характеризующих гипотезу H0
0	0.1
1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.1
7	0.1
8	0.1
9	0.1

Рисунок 7.

Литература

Кнут Д. Э. Искусство программирования. Том 3. Сортировка и поиск = The Art of Computer Programming. Volume 3. Sorting and Searching / под ред. В. Т. Тертышного (гл. 5) и И. В. Красикова (гл. 6). – 2-е изд. – Москва: Вильямс, 2007. – Т. 3. – 832 с.

Гренандер, У. Краткий курс вычислительной вероятности и статистики / У. Гренандер, В. Фрайбергер. — М.: Наука, 1978. — 192 с.

Чистяков, В. П. Курс теории вероятностей: Учебник. — 3-е издание, исправленное. — М.: Наука, 1987. — 240 с.

Приложение 1 – моделирование с. в.

```
import math
import random
import numpy as np
import pandas as pd
import scipy.stats as sps

a = float(input("Введите значение параметра a, удовлетворяющее условию |a|<=2: "))
print(a)
count_it = int(input("Введите значение параметра N: "))

u = np.zeros(count_it)
y = np.zeros(count_it)
f = np.zeros(count_it)
N = 1000000000
for i in range(0, count_it):
    U = random.randint(0, N)
    F = U/N
    u[i] = U
    f[i] = F
    if F < 0.5:
        y[i] = (1/a)* math.asin(a * F - a / 2)
    else:
        y[i] = 1 - math.sqrt(2 - 2 * F)

order = y.argsort()
ind = order.argsort()
data = {'Значения с. в.': pd.Series(y, index=ind),
        'Рандомное число U': pd.Series(u, index=ind),
        'U/N на [0, 1]': pd.Series(f, index=ind)}
# создаем датафрейм:
df = pd.DataFrame(data, range(0,count_it))
#pd.set_option('display.max_rows', None)
pd.set_option('display.max_rows', 10)
df
```

Приложение 2 - статистические характеристики с. в.

Построение графиков выборочной и теоретической функций распределения, и отыскания меры расхождения

```
import matplotlib.pyplot as plt

y1 = lambda x: 0 * x1
y2 = lambda x: (1/a) * np.sin(a*x2) + 1/2
y3 = lambda x: 1/2 + x3 - x3*x3/2
y4 = lambda x: 1 + 0 * x4

f = np.sort(f)
y = np.sort(y)

x1 = np.linspace(-0.5- (1/a) * math.asin(a / 2), y[0], 100)
x2 = np.linspace(- (1/a) * math.asin(a / 2), 0, 100)
x3 = np.linspace(0, 1, 100)
x4 = np.linspace(y[count_it-1], 1.5, 100)

plt.figure(figsize=(10, 5))
plt.plot(x1, y1(x1), color ='b')
plt.plot(x2, y2(x2), color ='g')
plt.plot(x3, y3(x3), color ='g')
plt.plot(x4, y4(x4), color ='b')
ax = plt.gca()
ax.set_title('График теоретической функции распределения и выборочной функции распределения')

num = count_it
F = []
x = y
for i in range(0, num):
    res = 0
    for j in range(0, count_it):
        if y[j] < x[i]:
            res = res + 1;
    F.append(res/count_it)

F.append(1.0)
x6 = np.append(x, 1.5)
plt.step(x6, F, color = 'b')
H = F
#мера расхождения
D = 0
res_x = res_f = 0
for i in range(0, count_it):
    if x[i] < 0:
        c = math.fabs(F[i] - (1/a) * np.sin(a*x[i]) - 1/2)
        if (i+1 < count_it):
            b = math.fabs(F[i+1] - (1/a) * np.sin(a*x[i]) - 1/2)
        if (c>b and D<c):
```

```
            D = c
            res_x = x[i]
        if (b>c and D<b):
            D = b
            res_x = x[i]
    else:
        c = math.fabs(F[i] - 1/2 - x[i] + x[i]*x[i]/2)
        if (i + 1 < count_it):
            b = math.fabs(F[i + 1] - 1/2 - x[i] + x[i]*x[i]/2)
        if (c>b and D<c):
            D = c
            res_x = x[i]
        if (b>c and D<b):
            D = b
            res_x = x[i]

plt.plot(res_x, 0, marker = '.', mew = 5, color = 'r')
plt.axvline(x=res_x, color = 'y')
plt.axvline(x=y[0], ymin = 0, ymax = H[0], color = 'b')
plt.show()
```

Расчёт таблицы теоретических и выборочных числовых характеристик с.в.

```
x = np.sort(y)
#мат. ожидание
E = 1/(a*a) - math.sqrt(1-(a/2)**2)/(a**2) - math.asin(a / 2)/(2*a) + 1/6
Dis = (math.asin(a / 2)**2)/(2*a**2) + math.asin(a / 2) * 2*math.sqrt(1-(a/2)**2)/(a**3) -
1/(a**2)+1/12 - E**2
#размах выборки
R = x[count_it-1]-x[0]

#выборочное средние
xx = 0
for i in range(0, count_it):
    xx = xx + x[i]
xx = (1.0/ count_it) * xx

#выборачная дисперссия
S = 0
for i in range(0, count_it):
    S = S +(x[i] - xx)**2
S = (1.0/count_it) * S

#выборочная медиана
Me = 0
if count_it % 2 == 1:
    if (count_it == 1):
        Me = x[0]
    else:
        Me = x[int((count_it - 1)/2) + 1]
```

```
else:
    Me = (x[int(count_it/2)] +  x[int(count_it/2) + 1])/2


data = {'Мат. ожидание': pd.Series(E),
        'Выборочное среднее': pd.Series(xx),
        'Модуль разности мат. ожидания и выборочного среднего': pd.Series(math.fabs(E - xx)),
       'Дисперсия': pd.Series(Dis),
       'Выборочная дисперсия': pd.Series(S),
       'Модуль разности дисперсии и выборочной дисперсии': pd.Series(math.fabs(Dis-S)),
       'Выборочная медиана': pd.Series(Me),
       'Размах выборки': pd.Series(R),
       'Мера расхождения D': pd.Series(D)}

# создаем датафрейм:
df = pd.DataFrame(data, range(0,1))
#pd.set_option('display.max_rows', None)
pd.set_option('display.max_rows', 10)
df
```

Построение гистограммы и таблицы 4

```
k = 15
bool = 1
k = float(input("Введите количество промежутков k: "))
bool = float(input("Введите 1, чтобы сгенирировать равные по величине промежутки\nВведите 2, чтобы ввести свои границы промежутков: "))

if (bool == 1):
    lamd = np.linspace(start = x[0], stop = x[count_it-1], num = int(k+1))
else:
    i = k - 1
    print("Левая граница: ", x[0], "  Правая граница: ", x[count_it - 1])
    lamd = []
    lamd.append(x[0])
    while(i != 0):
        print("Введите правую границу промежутка №", int(k - i), " :")
        val = float(input())
        lamd.append(val)
        i = i-1
    lamd.append(x[count_it - 1])

f_ht = []
Df = 0
n = []
Z = []
for i in range(1, int(k + 1)):

    m = (lamd[i] + lamd[i-1])/2.0
    Z.append(m)
    if (m < 0):
        f_ht.append(math.cos(a*m))
    else:
```

```
        f_ht.append(- m + 1)

    count_x = 0
    for j in range(0, count_it):
        if x[j] <= lamd[i] and x[j] >= lamd[i-1]:
            count_x = count_x + 1;
    n.append(count_x / (count_it * (lamd[i] - lamd[i-1])))

    bf = math.fabs(n[i - 1] - f_ht[i - 1])
    if (Df < bf):
        Df = bf

plt.figure(figsize=(10, 5))
ax = plt.gca()
ax.set_title('Гистограмма')
plt.hist( x, bins = lamd, edgecolor = 'b', facecolor='g', density = 1, alpha=0.55)
plt.show()
print("Максимальная мера расхождения гистограммы и теор. плотности распределения:", Df)

data = {'Значение точки - середины j-ого промежутка Zj ': pd.Series(Z),
        'Значение теор. плотности распределения в точке Zj': pd.Series(f_ht),
        'Значение гистограмы в точке Zj': pd.Series(n)}
# создаем датафрейм:
df = pd.DataFrame(data, range(0,k))
#pd.set_option('display.max_rows', None)
pd.set_option('display.max_rows', 10)
df
```

Приложение 3 – проверка гипотезы о виде распределения

Проверка одной выборки и вывод результатов (рисунок 7)

```
from scipy.special import gamma
from scipy import integrate


def val_f(p1, p2):
    if p1 < 0:
        f_p1 = (1./a) * math.sin(a*p1) + 1/2
    else:
        f_p1 = 1/2 + p1 - p1 * p1 / 2
    if p2 < 0:
        f_p2 = (1./a) * math.sin(a*p2) + 1/2
    else:
        f_p2 = 1/2 + p2 - p2 * p2 / 2
    return f_p2 - f_p1


#k = 15
bool = 1
k = int(input("Введите количество промежутков k: "))
#bool = float(input("Введите 1, чтобы сгенирировать равные по величине промежутки\nВведите 2, чтобы ввести свои границы промежутков: "))
alfa = float(input("Введите уровень значимости alfa: "))
k = 10
u = 2
lamd0 = []
lamd0.append((1/a)* math.asin(- a / 2.0))
t = np.linspace(start = 0, stop = 1, num = u + 1)
for i in range(u-1):
    if t[i+1] < 0.5:
        lamd0.append((1/a)* math.asin(a * t[i+1] - a / 2.0))
    else:
        lamd0.append(1 - math.sqrt(2 - 2 * t[i+1]))

lamd0.append(1)

if (bool == 1):
    lamd = []
    lamd.append((1/a)* math.asin(- a / 2.0))
    t = np.linspace(start = 0, stop = 1, num = k + 1)
    for i in range(k-1):
        if t[i+1] < 0.5:
            lamd.append((1/a)* math.asin(a * t[i+1] - a / 2.0))
        else:
            lamd.append(1 - math.sqrt(2 - 2 * t[i+1]))

    lamd.append(1)
else:
    i = k - 1
```

```
    print("Левая граница: ", x[0], "  Правая граница: ", x[count_it - 1])
    lamd = []
    lamd.append(x[0])
    while(i != 0):
        print("Введите правую границу промежутка №", int(k - i), " :")
        val = float(input())
        lamd.append(val)
        i = i-1
    lamd.append(x[count_it - 1])

R0 = 0
q = []
n = []
for i in range(1, int(k + 1)):

    #число наблюдений попавших в промежуток
    count_x = 0
    for j in range(count_it):
        if x[j] <= lamd[i] and x[j] >= lamd[i-1]:
            count_x = count_x + 1;
    n.append(count_x)

    #теоритическая вероятность попадания с. в. в промежуток

    q.append(val_f(lamd[i-1], lamd[i]))

    R0 = R0 + (n[i-1]- count_it * q[i-1])**2 / count_it * q[i-1]

g = gamma((k-1)/2)
l = lambda x: (x**((k-1)/2-1)) / (2**((k-1)/ 2)*g*(math.e**(x/2)))
b = integrate.quad(l, 0, R0)
FR0 = 1.0 - b[0]
print('F(R0) с чертой: ', FR0)
if FR0 < alfa:
    print("Гипотеза H0 отклонена :()")
else:
    print("Гипотеза H0 принята :)")

data = {'Значения теоретических вероятности qj, характеризующих гипотезу H0 ': pd.Series(q)}
# создаем датафрейм:
df = pd.DataFrame(data, range(0,k))
#pd.set_option('display.max_rows', None)
pd.set_option('display.max_rows', 10)
df
```

Проверка 100 разных выборок на разных уровнях значимости

```
def f(x):
    if (x == 0):
        return 0
    g = gamma((k-1)/2)
    return (x**((k-1)/2-1)) / (2**((k-1)/ 2)*g*(math.e**(x/2)))
```

```
def met(a, b, n):
    res = 0
    for i in range(n):
        res = res + (f(a + (b - a) * (i - 1) / n) + f(a + (b - a) * (i) /  n)) * (b - a) / (2 * n)
    return res

def exper(alff, count):

    res = [0, 0]
    for i in range(count):
        u = np.zeros(count_it)
        y = np.zeros(count_it)
        f = np.zeros(count_it)
        N = 1000000000
        for i in range(0, count_it):
            U = random.randint(0, N)
            F = U/N
            u[i] = U
            f[i] = F
            if F < 0.5:
                y[i] = (1/a)* math.asin(a * F - a / 2)
            else:
                y[i] = 1 - math.sqrt(2 - 2 * F)

        x = np.sort(y)

        R0 = 0
        q = []
        n = []
        for i in range(1, int(k + 1)):

            #число наблюдений попавших в промежуток
            count_x = 0
            for j in range(count_it):
                if x[j] <= lamd[i] and x[j] >= lamd[i-1]:
                    count_x = count_x + 1;
            n.append(count_x)

            #теоритическая вероятность попадания с. в. в промежуток

            q.append(val_f(lamd[i-1], lamd[i]))

            R0 = R0 + (n[i-1]- count_it * q[i-1])**2 / (count_it * q[i-1])

        #print(R0)
        #g = gamma((k-1)/2)
        #l = lambda x: (x**((k-1)/2-1)) / (2**((k-1)/ 2)*g*(math.e**(x/2)))
        #b = integrate.quad(l, 0, R0)
        #print(b)
        FR0 = 1.0 - b[0]
```

```
        #print('F(R0) с чертой: ', FR0)
        c = met(0, R0, 100)
        FR0 = 1.0 - c.real
       # print('F(R0) с чертой: ', FR0)

        if FR0 < alff:
            res[0] = res[0] + 1;
        else:
            res[1] = res[1] + 1;
    return res;

ex1 = exper(0.1, 100)
ex2 = exper(0.5, 100)
ex3 = exper(0.9, 100)
#res[0] - раз отклонена, res[1] - раз принята
print("alfa = 0.1: Гипотиза принята ", ex1[1], " раз; Гипотеза отклонена ", ex1[0], " раз" )
print("alfa = 0.5: Гипотиза принята ", ex2[1], " раз; Гипотеза отклонена ", ex2[0], " раз" )
print("alfa = 0.9: Гипотиза принята ", ex3[1], " раз; Гипотеза отклонена ", ex3[0], " раз" )
```