

ОТЧЕТ ПО КУРСОВОЙ РАБОТЕ

по дисциплине «Программные технологии построения управляющих оболочек СДО»

Выполнил cтудент группы М80-101М-21

Фейзуллин К.М.

Руководитель:

Мхитарян Г. А.

# Содержание

[Содержание 2](#_Toc104471592)

[Задание 3](#_Toc104471593)

[Выполнение 4](#_Toc104471594)

[Задание 1 4](#_Toc104471595)

[Задание 2 5](#_Toc104471596)

[Задание 3 6](#_Toc104471597)

[Задание 4 6](#_Toc104471598)

[Задание 5 9](#_Toc104471599)

[Вывод 13](#_Toc104471600)

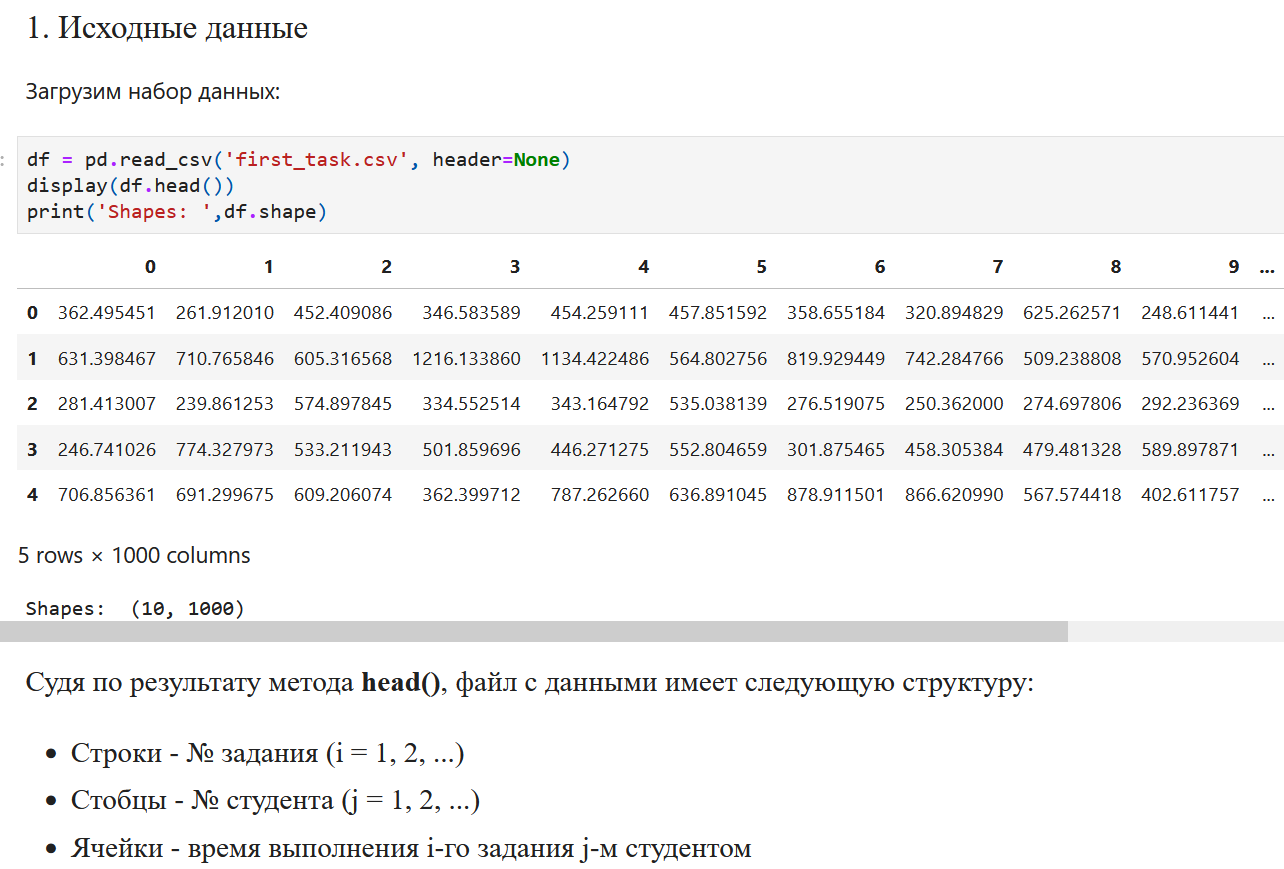
[Приложение 14](#_Toc104471601)

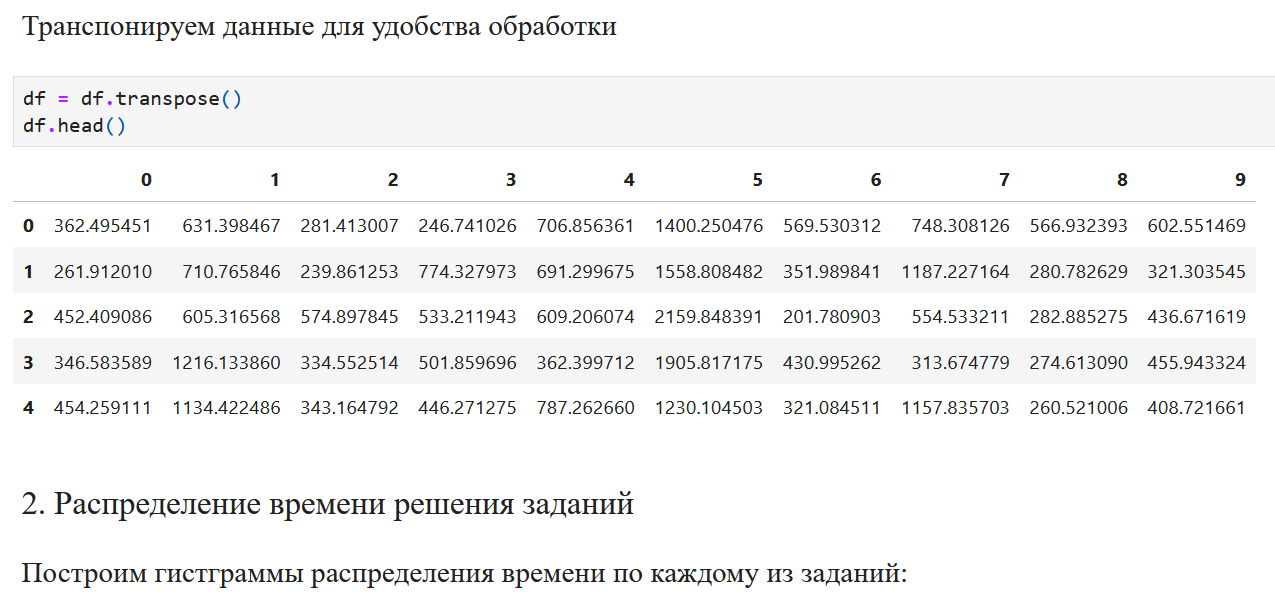
# Задание

1. Получить датасет: студент-задание-время.
2. Построить гистограмму по выборке.
3. Сформулировать гипотезу о принадлежности выборки распределению.
4. Проверить гипотезу о соответствии выбранному распределению.
5. Получить параметры подходящего гамма-распределения.

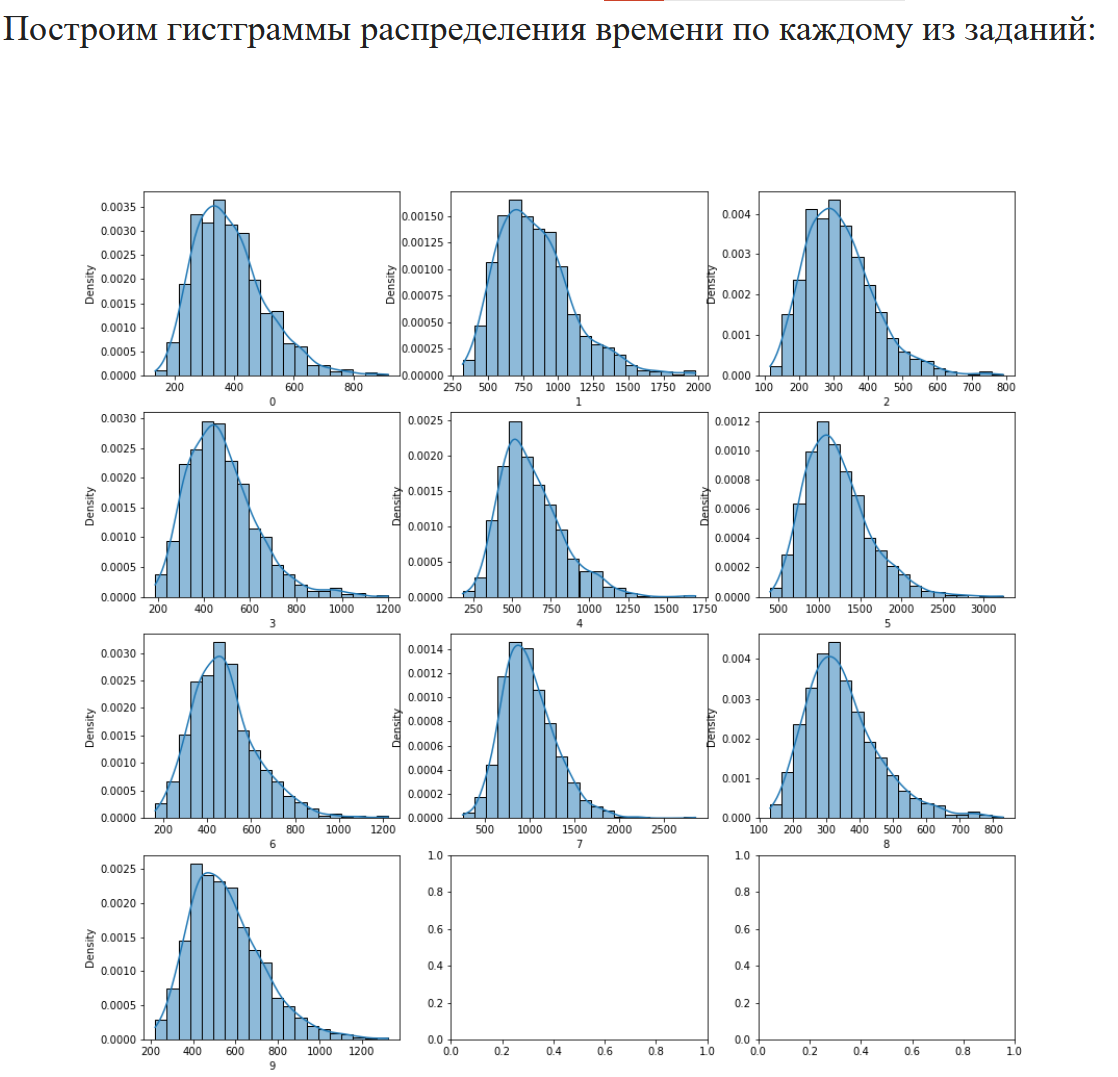
# Выполнение

## Задание 1



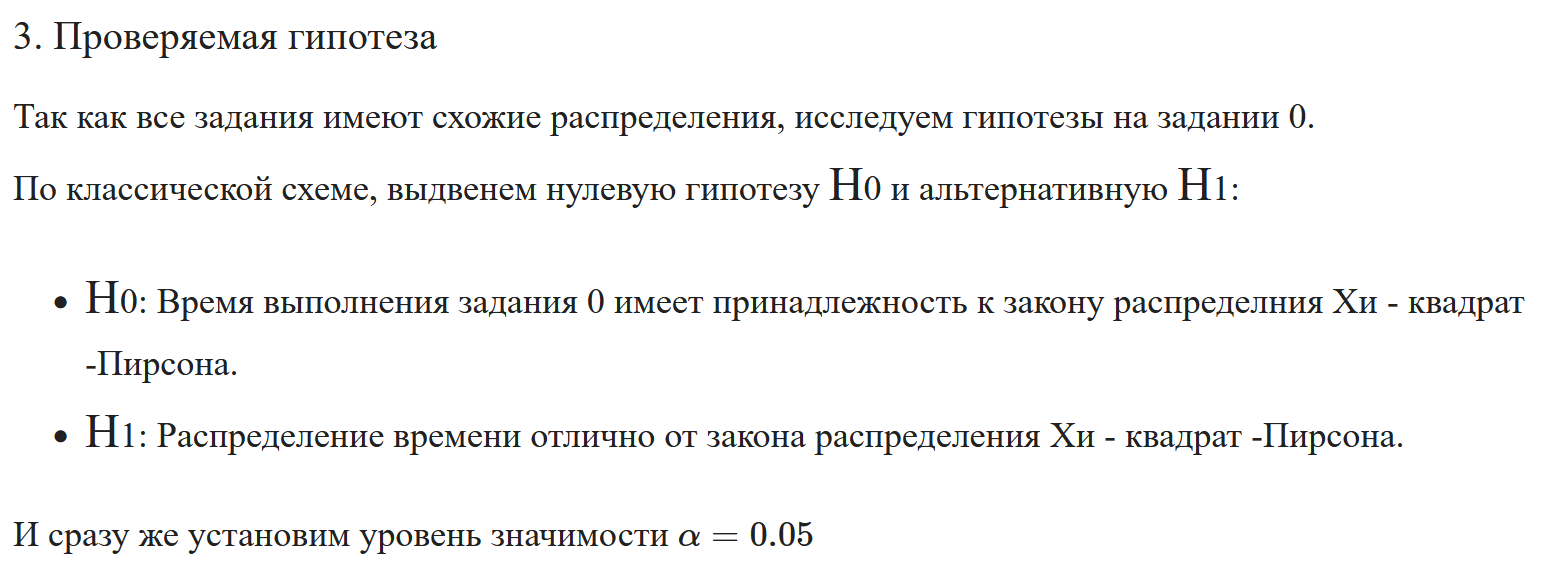


## Задание 2

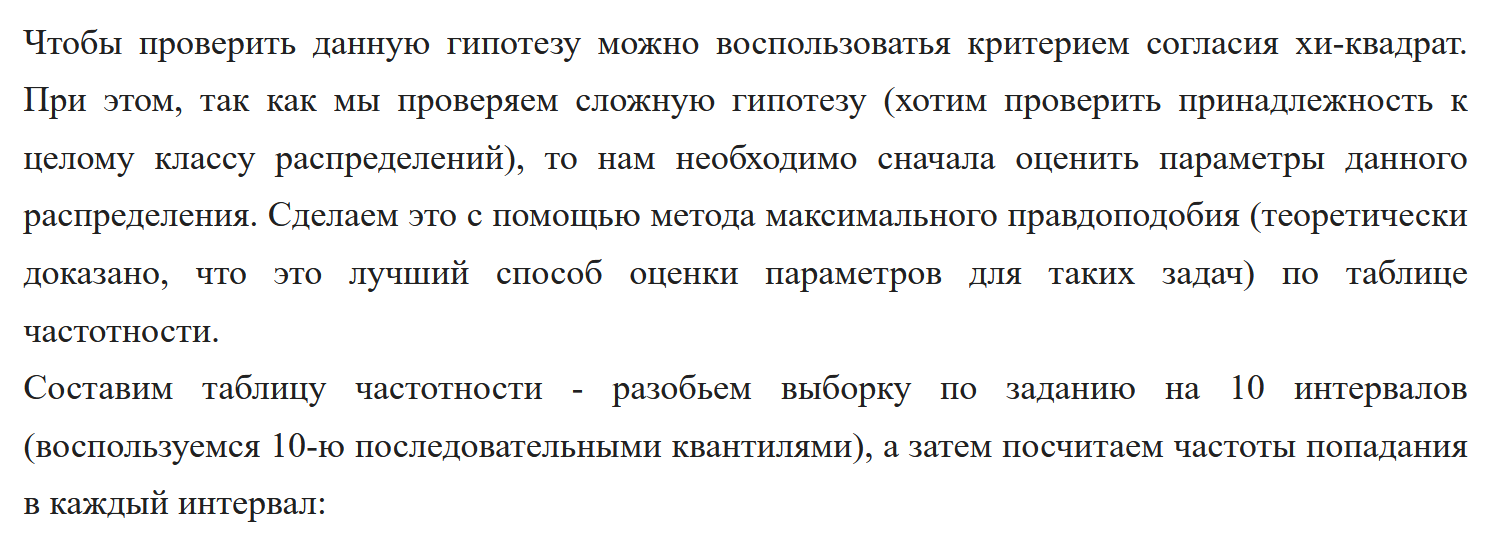


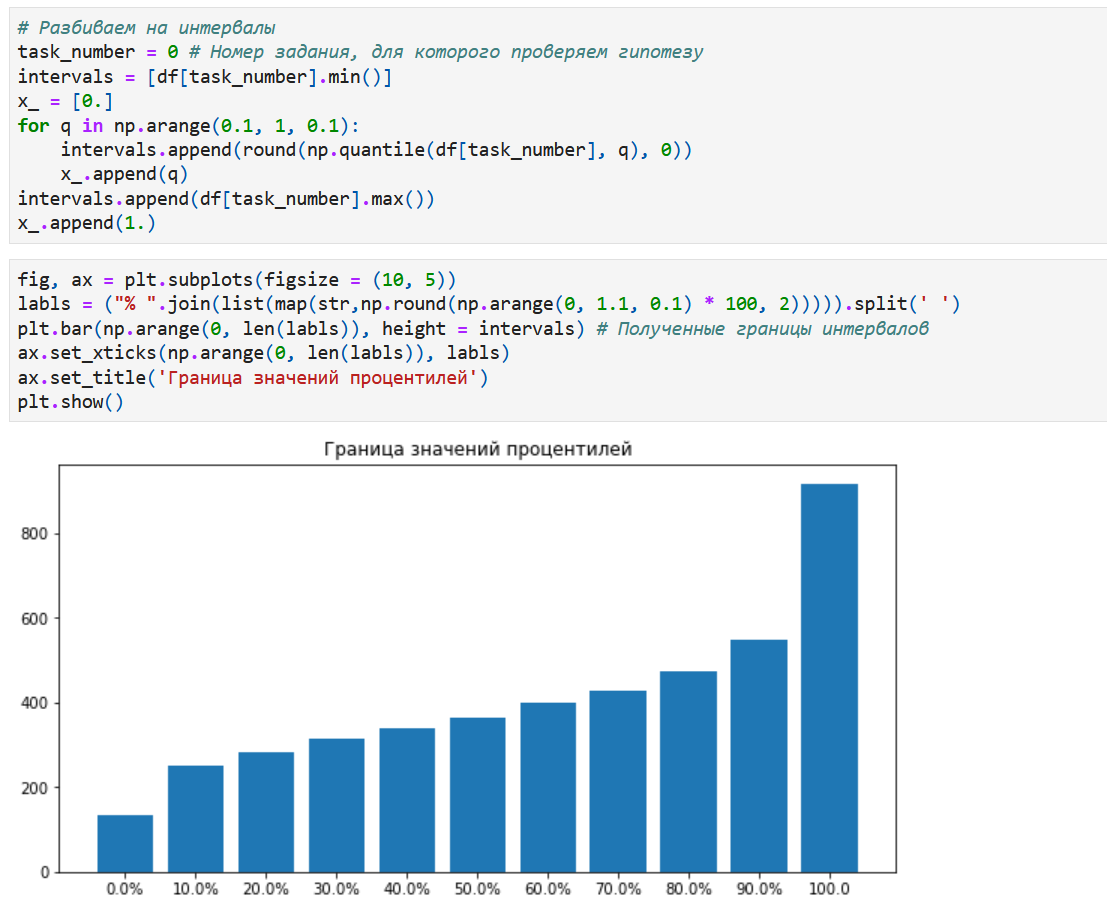
Можно заметить, что большинство распределений по форме отчетливо напоминают распределение хи-квадрат Пирсона.

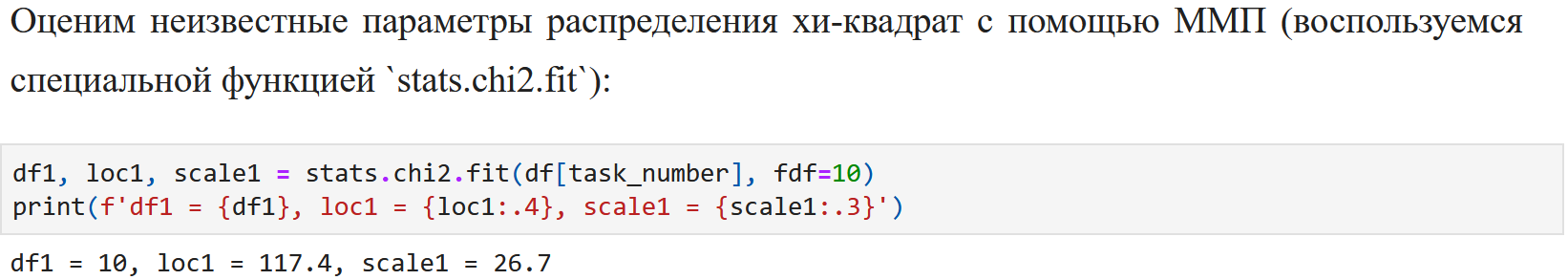
## Задание 3

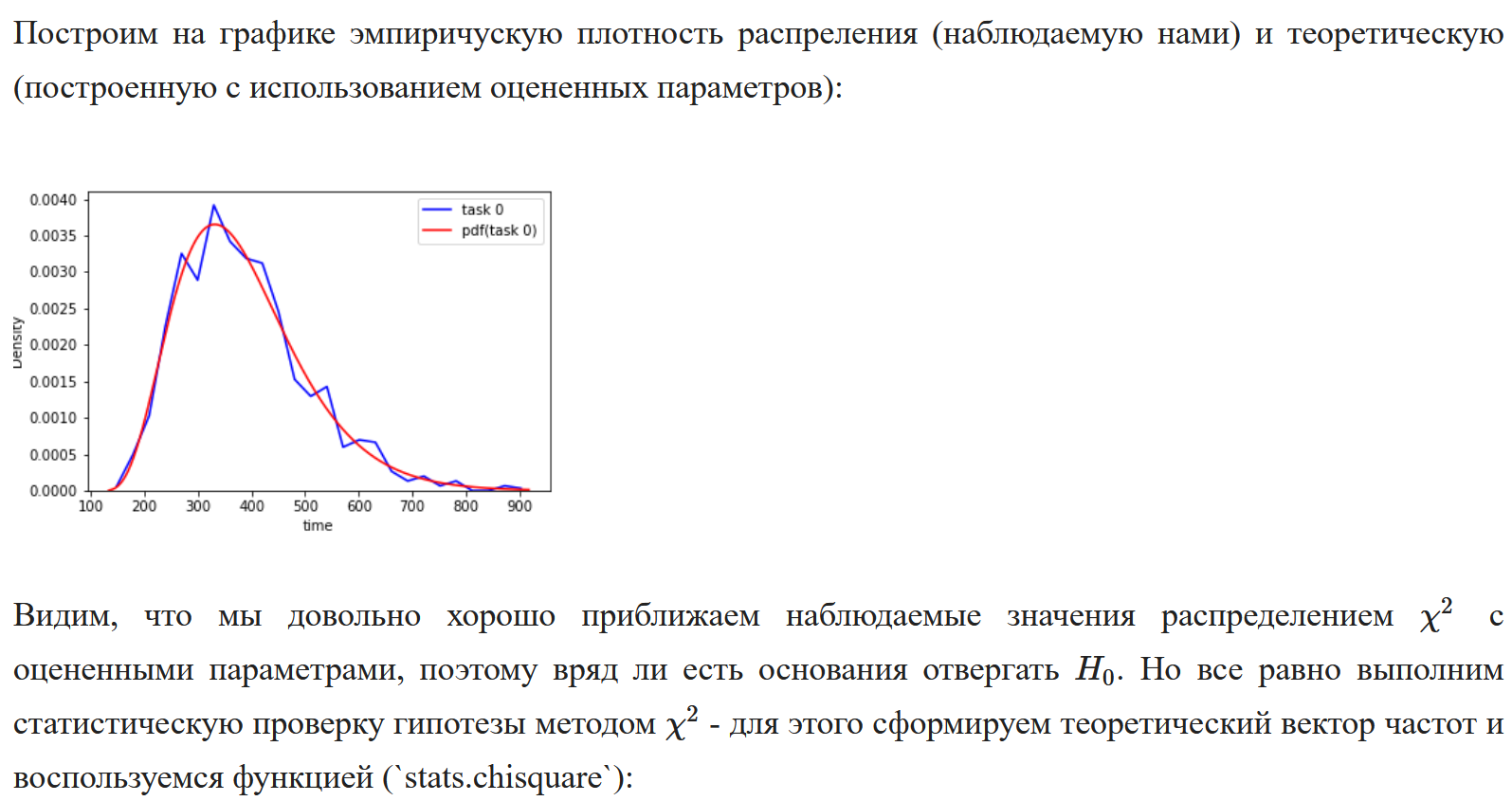


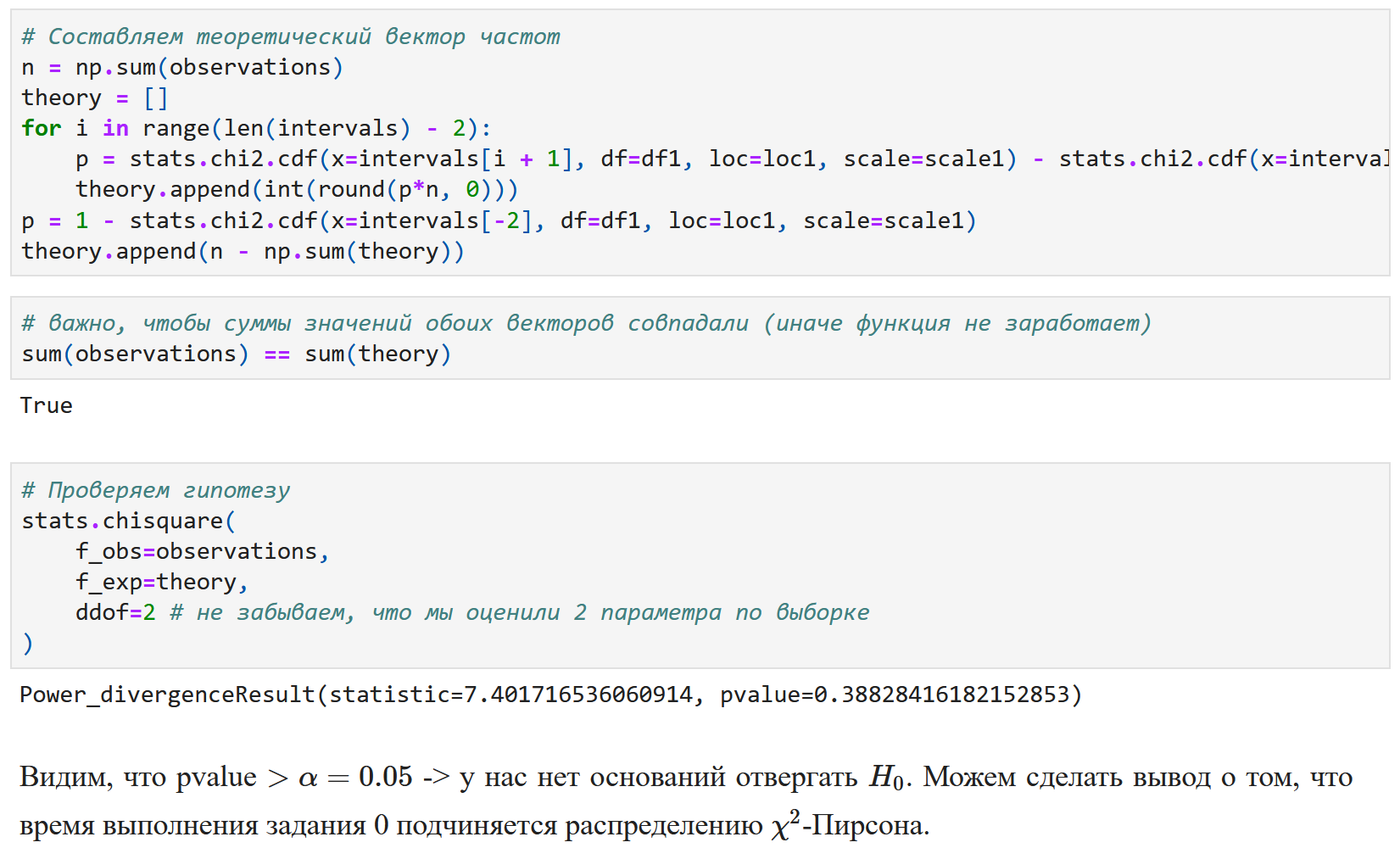
## Задание 4





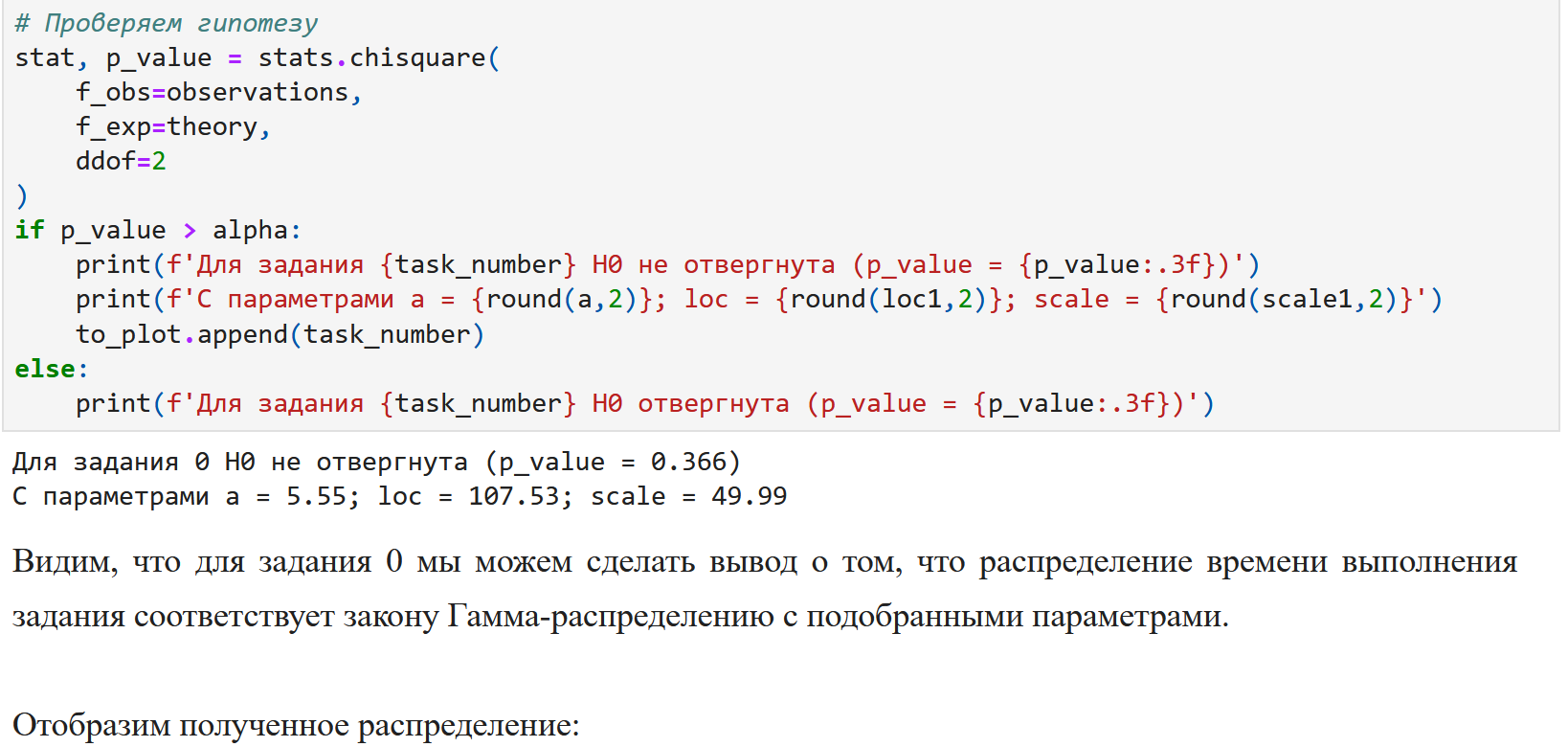


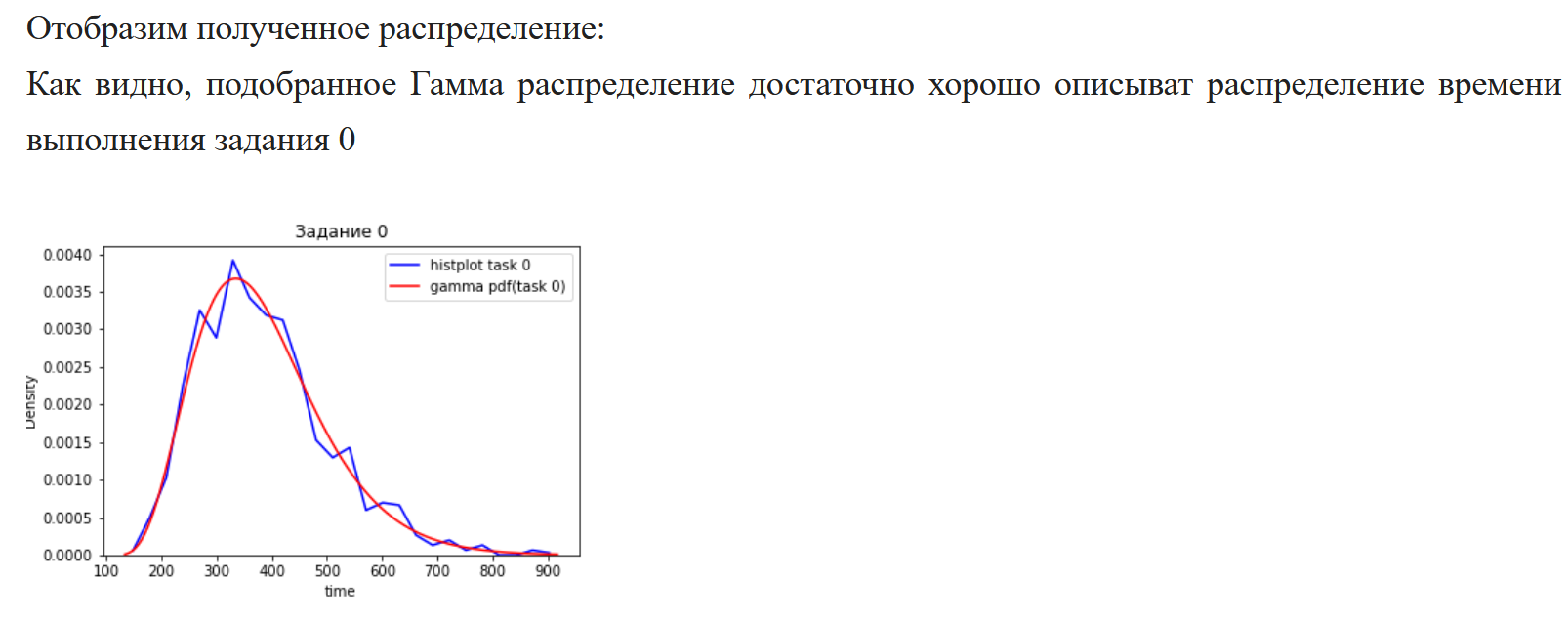




## Задание 5

## 





# Вывод

Таким образом, в данной курсовой работе был исследован набор данных из системы дистанционного обучения, имеющий структуру студент-задание-время. Отобразив гистограммы распределений времени выполнения заданий было предположено, что время выполнения задания всех студентов имеет закон распределения Хи – квадрат Пирсона и с помощью функции максимального правдоподобия нашли параметры распределения времени и построили по ним предполагаемое распределение. Моделирование показало, что время действительно имеет закон распределения Хи – квадрат Пирсона. Аналогичный алгоритм был проведен для закона Гамма – распределения, который тоже хорошо описывает распределение времени выполнения задания. Все результаты сопровождены иллюстрирующими графиками.

# Приложение

Код программ:

+\*In[1]:\*+

[source, ipython3]

----

import numpy as np

import pandas as pd

from scipy import stats

import seaborn as sns

import matplotlib.pyplot as plt

----

=== 1. Датасет

Строки - № задания (i = 1, 2, …) +

Стобцы - № студента (j = 1, 2, …) +

Ячейки - время выполнения i-го задания j-м студентом

Загрузим датасет:

+\*In[2]:\*+

[source, ipython3]

----

ds = pd.read\_csv('first\_task.csv', header=None)

ds

----

Транспонируем его для удобства

+\*In[3]:\*+

[source, ipython3]

----

ds = ds.transpose()

ds

----

=== 2. Гистограммы распределения

Построим гистграммы распределения времени по каждому из заданий:

+\*In[48]:\*+

[source, ipython3]

----

ds.hist(figsize=(10, 10), bins = 20);

----

=== 4. Проверка гипотезы

Чтобы проверить данную гипотезу можно воспользоватья критерием согласия

хи-квадрат. При этом, так как мы проверяем сложную гипотезу (хотим

проверить принадлежность к целому классу распределений), то нам

необходимо сначала оценить параметры данного распределения. Сделаем это

с помощью метода максимального правдоподобия (теоретически доказано, что

это лучший способ оценки параметров для таких задач) по таблице

частотности. +

Составим таблицу частотности - разобьем выборку по каждому заданию на 10

интервалов (воспользуемся 10-ю последовательными квантилями), а затем

посчитаем частоты попадания в каждый интервал:

+\*In[49]:\*+

[source, ipython3]

----

# Разбиваем на интервалы

task\_number = 0 # Номер задания, для которого проверяем гипотезу

intervals = [ds[task\_number].min()]

for q in np.arange(0.1, 1, 0.1):

intervals.append(round(np.quantile(ds[task\_number], q), 0))

intervals.append(ds[task\_number].max())

----

+\*In[51]:\*+

[source, ipython3]

----

# Соcтавляем вектор наблюдений (таблица частотности)

observations = []

for i in range(len(intervals) - 1):

observations.append(len(ds[task\_number][(ds[task\_number] >= intervals[i])&(ds[task\_number] <=intervals[i + 1])]))

observations

----

=== 5. Получение оценок параметров

Оценим неизвестные параметры распределения хи-квадрат с помощью ММП

(воспользуемся специальной функцией `stats.chi2.fit`):

+\*In[59]:\*+

[source, ipython3]

----

df1, loc1, scale1 = stats.chi2.fit(ds[task\_number], fdf=10)

print(f'df1 = {df1}, loc1 = {loc1:.4}, scale1 = {scale1:.3}')

----

+\*Out[59]:\*+

----

df1 = 10, loc1 = 117.4, scale1 = 26.7

----

Построим на графике эмпиричускую плотность распреления (наблюдаемую

нами) и теоретическую (построенную с использованием оцененных

параметров):

+\*In[60]:\*+

[source, ipython3]

----

fig, ax = plt.subplots()

sns.histplot(data=ds[task\_number], color='b', element='poly', fill=False, stat='density', label='task 0', ax=ax)

chi2\_rv1 = stats.chi2(df1, loc1, scale1)

x = np.linspace(min(intervals), max(intervals), 1000)

sns.lineplot(x=x, y=chi2\_rv1.pdf(x), color='r', label='pdf(task 0)', ax=ax)

ax.set\_xlabel('time');

----

+\*Out[60]:\*+

----

![png](output\_20\_0.png)

----

Видим, что мы довольно хорошо приближаем наблюдаемые значения

распределением latexmath:[$\chi^2$] с оцененными параметрами, поэтому

вряд ли есть основания отвергать latexmath:[$H\_0$]. +

Но все равно выполним статистическую проверку гипотезы методом

latexmath:[$\chi^2$] - для этого сформируем теоретический вектор частот

и воспользуемся функцией (`stats.chisquare`):

+\*In[14]:\*+

[source, ipython3]

----

# Составляем теоретический вектор частот

n = np.sum(observations)

theory = []

for i in range(len(intervals) - 2):

p = stats.chi2.cdf(x=intervals[i + 1], df=df1, loc=loc1, scale=scale1) - stats.chi2.cdf(x=intervals[i], df=df1, loc=loc1, scale=scale1)

theory.append(int(round(p\*n, 0)))

p = 1 - stats.chi2.cdf(x=intervals[-2], df=df1, loc=loc1, scale=scale1)

theory.append(n - np.sum(theory))

----

+\*In[15]:\*+

[source, ipython3]

----

# важно, чтобы суммы значений обоих векторов совпадали (иначе функция не заработает)

sum(observations) == sum(theory)

----

+\*Out[15]:\*+

----True----

+\*In[16]:\*+

[source, ipython3]

----

# Проверяем гипотезу

stats.chisquare(

f\_obs=observations,

f\_exp=theory,

ddof=1 # не забываем, что мы оценили параметр по выборке

)

----

+\*Out[16]:\*+

----Power\_divergenceResult(statistic=7.401716536060914, pvalue=0.493974120882411)----

Видим, что pvalue $ > =0.05$ -> у нас нет оснований отвергать

latexmath:[$H\_0$]. Можем сделать вывод о том, что время выполнения

задания 0 подчиняется распределению latexmath:[$\chi^2$]-Пирсона.

Проверим данную гипотезу для всех заданий:

+\*In[143]:\*+

[source, ipython3]

----

alpha = 0.05 # уровень значимости

to\_plot = []

statistics = []

for task in range(len(ds.transpose())):

intervals = [ds[task].min()]

for q in np.arange(0.1, 1, 0.1):

intervals.append(round(np.quantile(ds[task], q), 0))

intervals.append(ds[task].max())

# составляем вектор наблюдаемых частот

observations = []

for i in range(len(intervals) - 1):

observations.append(len(ds[task][(ds[task] >= intervals[i]) & (ds[task] <= intervals[i + 1])]))

# Определяем оценки параметров методом ММП

df1, loc1, scale1 = stats.chi2.fit(ds[task], fdf=15)

# Составляем теоретических ветор частот

n = np.sum(observations)

theory = []

for i in range(len(intervals) - 2):

p = stats.chi2.cdf(x=intervals[i + 1], df=df1, loc=loc1, scale=scale1) - stats.chi2.cdf(x=intervals[i], df=df1, loc=loc1, scale=scale1)

theory.append(int(round(p\*n, 0)))

p = 1 - stats.chi2.cdf(x=intervals[-2], df=df1, loc=loc1, scale=scale1)

theory.append(n - np.sum(theory))

# Проверяем гипотезу

stat, p\_value = stats.chisquare(

f\_obs=observations,

f\_exp=theory,

ddof=1

)

if p\_value > alpha:

print(f'Для задания {task} H0 не отвергнута (p\_value = {p\_value:.3f})')

else:

print(f'Для задания {task} H0 отвергнута (p\_value = {p\_value:.3f})')

to\_plot.append(task)

statistics.append(stat)

fig, ax = plt.subplots()

ax.set\_title(f'Задание №{task+1}')

sns.histplot(data=ds[task], color='b', element='poly', fill=False, stat='density', label=f'histplot task {task}', ax=ax)

chi = stats.chi2(df1, loc1, scale1)

x = np.linspace(min(intervals), max(intervals), 1000)

sns.lineplot(x=x, y=chi.pdf(x), color='r', label=f'$\chi^2$ pdf(task {task})', ax=ax)

ax.set\_xlabel('time');

----

+\*In[144]:\*+

[source, ipython3]

----

fig, ax = plt.subplots(1, 1)

df = 10

x = np.linspace(stats.chi2.ppf(0.001, df), stats.chi2.ppf(0.99, df), 100)

ax.plot(x, stats.chi2.pdf(x, df), 'g', lw=5, alpha=0.6, label='chi2 pdf')

plt.axvline(x=stats.chi2.ppf(0.95, df), color='red', lw=3)

ax.text(x=20.5, y=0.065, s='КРО', fontsize=12)

ax.text(x=20, y=0.0005, s=r'$\alpha$', fontsize=15)

ax.text(x=7.5, y=0.05, s=r'1-$\alpha$', fontsize=15)

ax.scatter(x=statistics, y=np.zeros(len(statistics)), edgecolors='black', s=50)

ax.set\_ylabel('Плотность вероятности', fontsize=12)

ax.set\_xlabel(r'Значения статистики критерия $\chi^2$', fontsize=12);

ax.set\_title(r'Критическая область критерия $\chi^2$', fontsize=12);

----

+\*Out[144]:\*+

----

![png](output\_28\_0.png)

----

Видим, что для всех заданий (кроме 7) мы можем сделать вывод о

latexmath:[$\chi^2$]-распределении времени их выполнения. +

Посмотрим на графики для 7 задания:

+\*In[67]:\*+

[source, ipython3]

----

task\_number = 7

intervals = [ds[task\_number].min()]

for q in np.arange(0.1, 1, 0.1):

intervals.append(round(np.quantile(ds[task\_number], q), 0))

intervals.append(ds[task\_number].max())

observations = []

for i in range(len(intervals) - 1):

observations.append(len(ds[task\_number][(ds[task\_number] >= intervals[i]) & (ds[task\_number] <= intervals[i + 1])]))

df1, loc1, scale1 = stats.chi2.fit(ds[task\_number], fdf=10)

n = np.sum(observations)

theory = []

for i in range(len(intervals) - 2):

p = stats.chi2.cdf(x=intervals[i + 1], df=df1, loc=loc1, scale=scale1) - stats.chi2.cdf(x=intervals[i], df=df1, loc=loc1, scale=scale1)

theory.append(int(round(p\*n, 0)))

p = 1 - stats.chi2.cdf(x=intervals[-2], df=df1, loc=loc1, scale=scale1)

theory.append(n - np.sum(theory))

fig, ax = plt.subplots()

sns.histplot(data=ds[task\_number], color='b', element='poly', fill=False, stat='density', label=f'task {task\_number}', ax=ax)

chi2\_rv1 = stats.chi2(df1, loc1, scale1)

x = np.linspace(min(intervals), max(intervals), 1000)

sns.lineplot(x=x, y=chi2\_rv1.pdf(x), color='r', label=f'pdf(task {task\_number})', ax=ax)

ax.set\_xlabel('time');

----

+\*Out[67]:\*+

----

![png](output\_30\_0.png)

----

Видим, что различия между 2 графиками заметно, поэтому неудивительно,

что мы отвергаем latexmath:[$H\_0$].

=== Гамма-распределение

Получим параметры подходящих гамма-распределений и проверим гипотезу

согласия наблюдаемых выборок полученным гамма-распределениям:

+\*In[147]:\*+

[source, ipython3]

----

alpha = 0.05 # уровень значимости

to\_plot = []

statistics = []

for task in range(len(ds.transpose())):

intervals = [ds[task].min()]

for q in np.arange(0.1, 1, 0.1):

intervals.append(round(np.quantile(ds[task], q), 0))

intervals.append(ds[task].max())

# составляем вектор наблюдаемых частот

observations = []

for i in range(len(intervals) - 1):

observations.append(len(ds[task][(ds[task] >= intervals[i]) & (ds[task] <= intervals[i + 1])]))

# Определяем оценки параметров Гамма-распределения методом ММП

a, loc1, scale1 = stats.gamma.fit(ds[task])

# Составляем теоретических ветор частот

n = np.sum(observations)

theory = []

for i in range(len(intervals) - 2):

p = stats.gamma.cdf(x=intervals[i + 1], a=a, loc=loc1, scale=scale1) - stats.gamma.cdf(x=intervals[i], a=a, loc=loc1, scale=scale1)

theory.append(int(round(p\*n, 0)))

p = 1 - stats.gamma.cdf(x=intervals[-2], a=a, loc=loc1, scale=scale1)

theory.append(n - np.sum(theory))

# Проверяем гипотезу

stat, p\_value = stats.chisquare(

f\_obs=observations,

f\_exp=theory,

ddof=2

)

if p\_value > alpha:

print(f'Для задания {task} H0 не отвергнута (p\_value = {p\_value:.3f}, stat={stat})')

else:

print(f'Для задания {task} H0 отвергнута (p\_value = {p\_value:.3f}, stat={stat})')

to\_plot.append(task)

statistics.append(stat)

Видим, что для всех заданий (кроме 5) мы можем сделать вывод о том, что

время их выполнения подчиняется Гамма-распределению с подобранными

параметрами.

Взглянем на этим распределения:

+\*In[195]:\*+

[source, ipython3]

----

for task\_number in to\_plot:

intervals = [ds[task\_number].min()]

for q in np.arange(0.1, 1, 0.1):

intervals.append(round(np.quantile(ds[task\_number], q), 0))

intervals.append(ds[task\_number].max())

observations = []

for i in range(len(intervals) - 1):

observations.append(len(ds[task\_number][(ds[task\_number] >= intervals[i]) & (ds[task\_number] <= intervals[i + 1])]))

a, loc1, scale1 = stats.gamma.fit(ds[task\_number])

n = np.sum(observations)

theory = []

for i in range(len(intervals) - 2):

p = stats.gamma.cdf(x=intervals[i + 1], a=a, loc=loc1, scale=scale1) - stats.gamma.cdf(x=intervals[i], a=a, loc=loc1, scale=scale1)

theory.append(int(round(p\*n, 0)))

p = 1 - stats.gamma.cdf(x=intervals[-2], a=a, loc=loc1, scale=scale1)

theory.append(n - np.sum(theory))

fig, ax = plt.subplots()

ax.set\_title(f'Задание №{task\_number+1}')

sns.histplot(data=ds[task\_number], color='b', element='poly', fill=False, stat='density', label=f'histplot task {task\_number}', ax=ax)

gam = stats.gamma(a, loc1, scale1)

x = np.linspace(min(intervals), max(intervals), 1000)

sns.lineplot(x=x, y=gam.pdf(x), color='r', label=f'gamma pdf(task {task\_number})', ax=ax)

ax.set\_xlabel('time');

Видим, что полученные распределения хорошо описывают наблюдаемые

выборки.

+\*In[146]:\*+

[source, ipython3]

----

fig, ax = plt.subplots(1, 1)

df = 10

x = np.linspace(stats.chi2.ppf(0.001, df), stats.chi2.ppf(0.99, df), 100)

ax.plot(x, stats.chi2.pdf(x, df), 'g', lw=5, alpha=0.6, label='chi2 pdf')

plt.axvline(x=stats.chi2.ppf(0.95, df), color='red', lw=3)

ax.text(x=20.5, y=0.065, s='КРО', fontsize=12)

ax.text(x=20, y=0.0005, s=r'$\alpha$', fontsize=15)

ax.text(x=7.5, y=0.05, s=r'1-$\alpha$', fontsize=15)

ax.scatter(x=statistics, y=np.zeros(len(statistics)), edgecolors='black', s=50)

ax.set\_ylabel('Плотность вероятности', fontsize=12)

ax.set\_xlabel(r'Значения статистики критерия $\chi^2$', fontsize=12);

ax.set\_title(r'Критическая область критерия $\chi^2$', fontsize=12);

----