

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 8](#_Toc104471599)

[Вывод 12](#_Toc104471600)

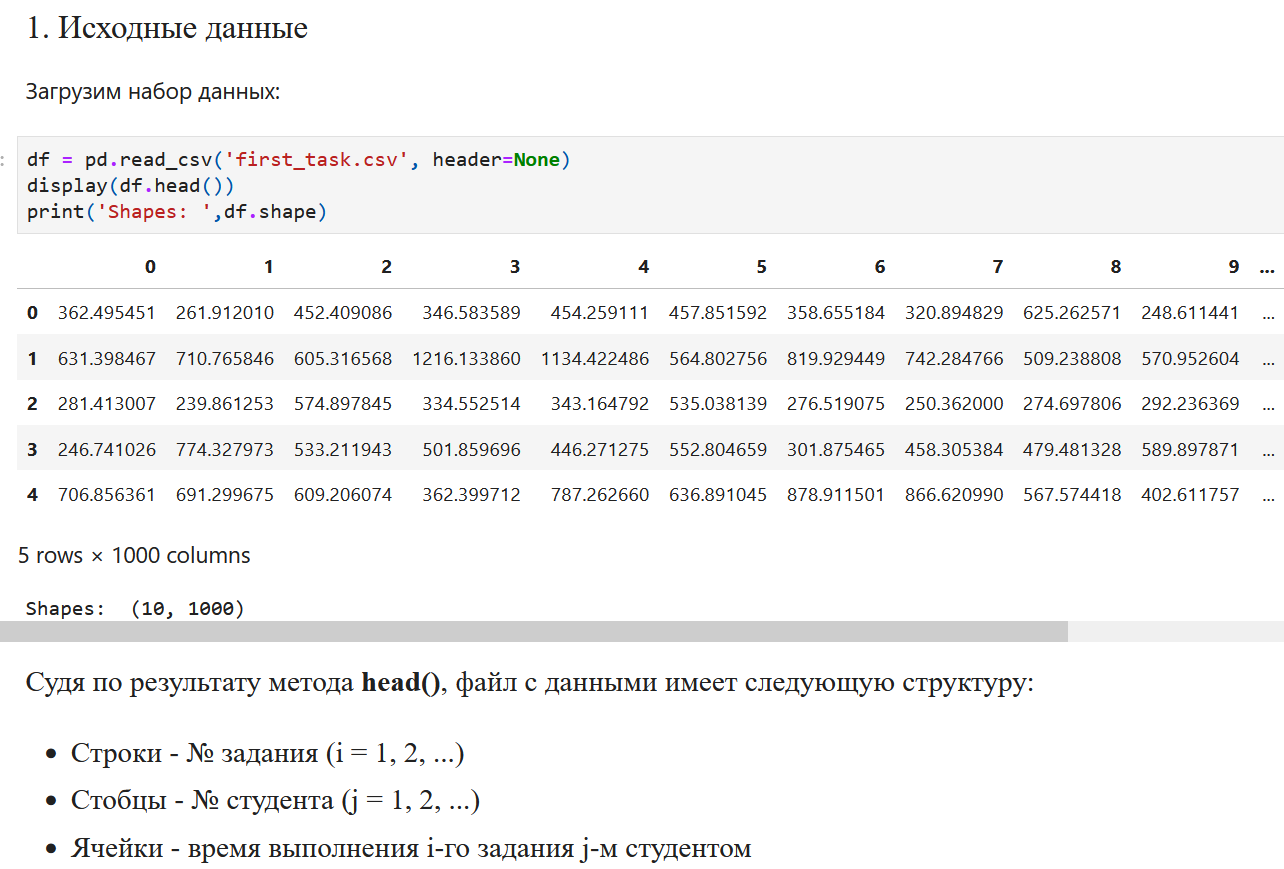
[Приложение 13](#_Toc104471601)

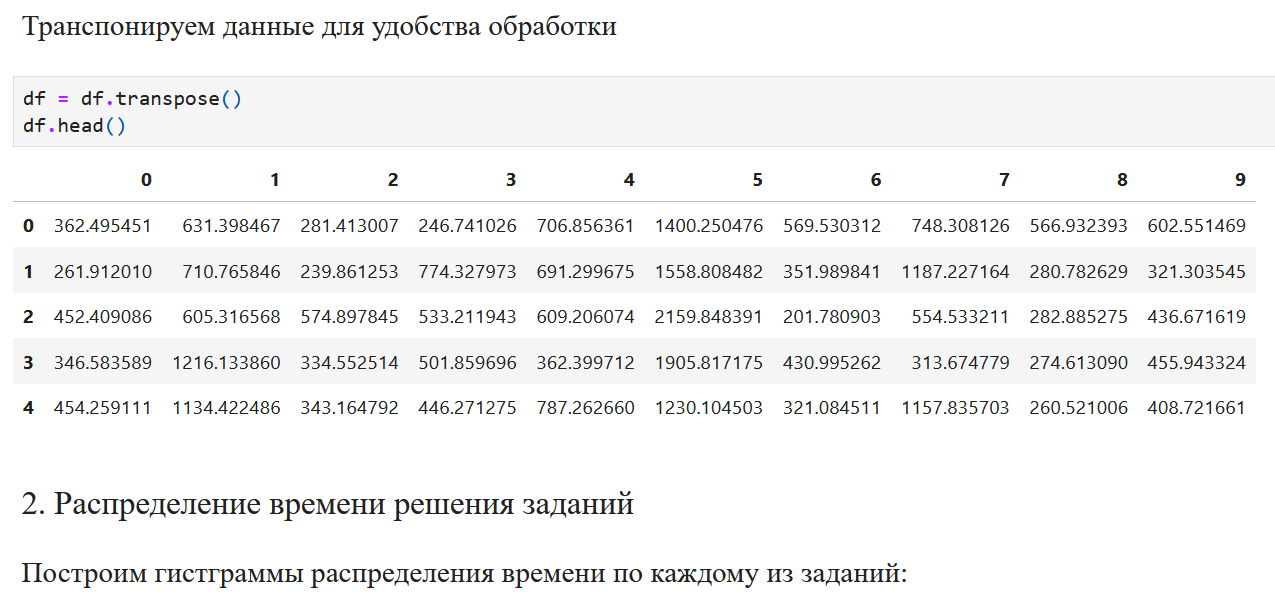
# Задание

1. Получить набор данных: 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.

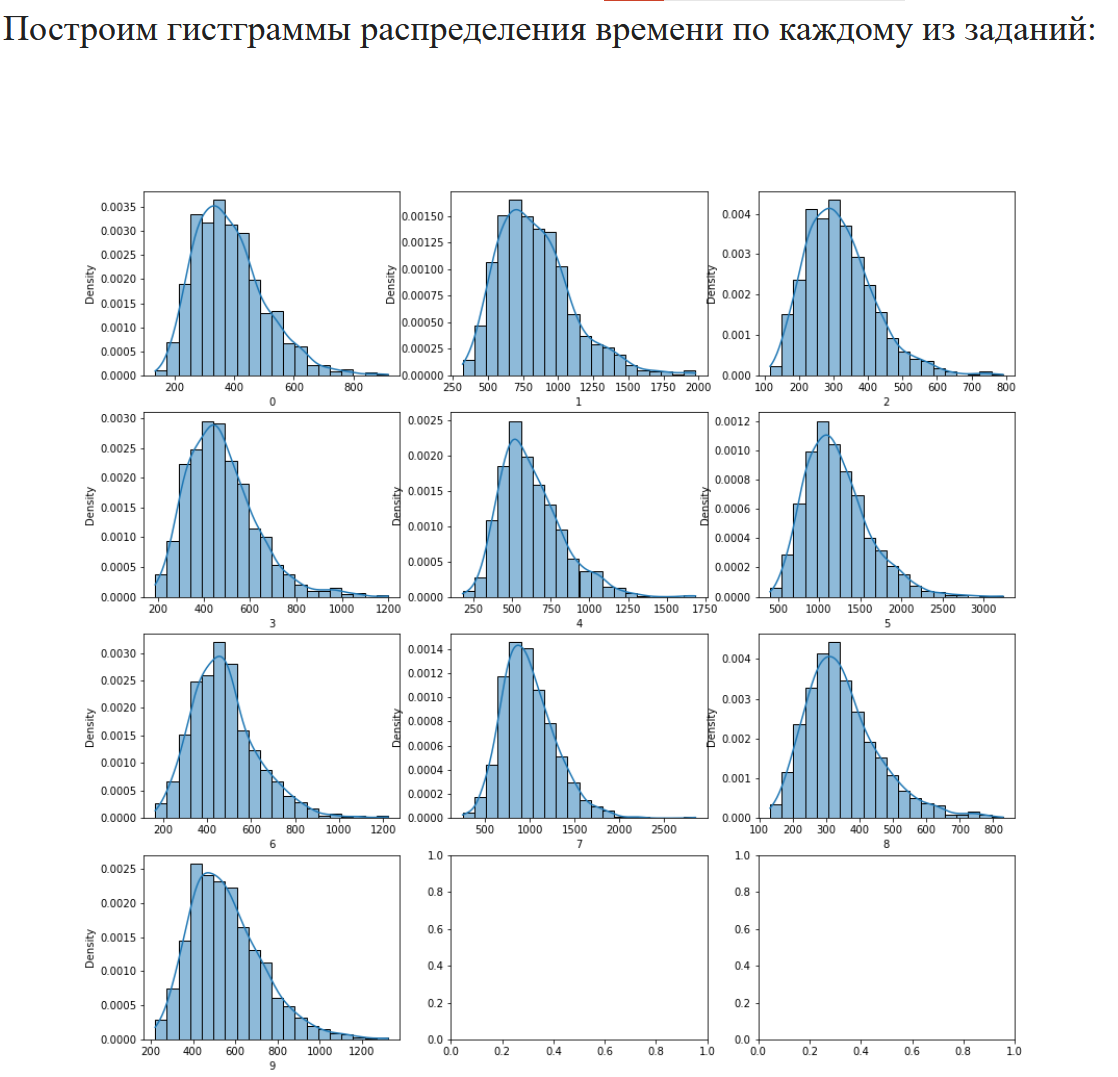
# Выполнение

## Задание 1



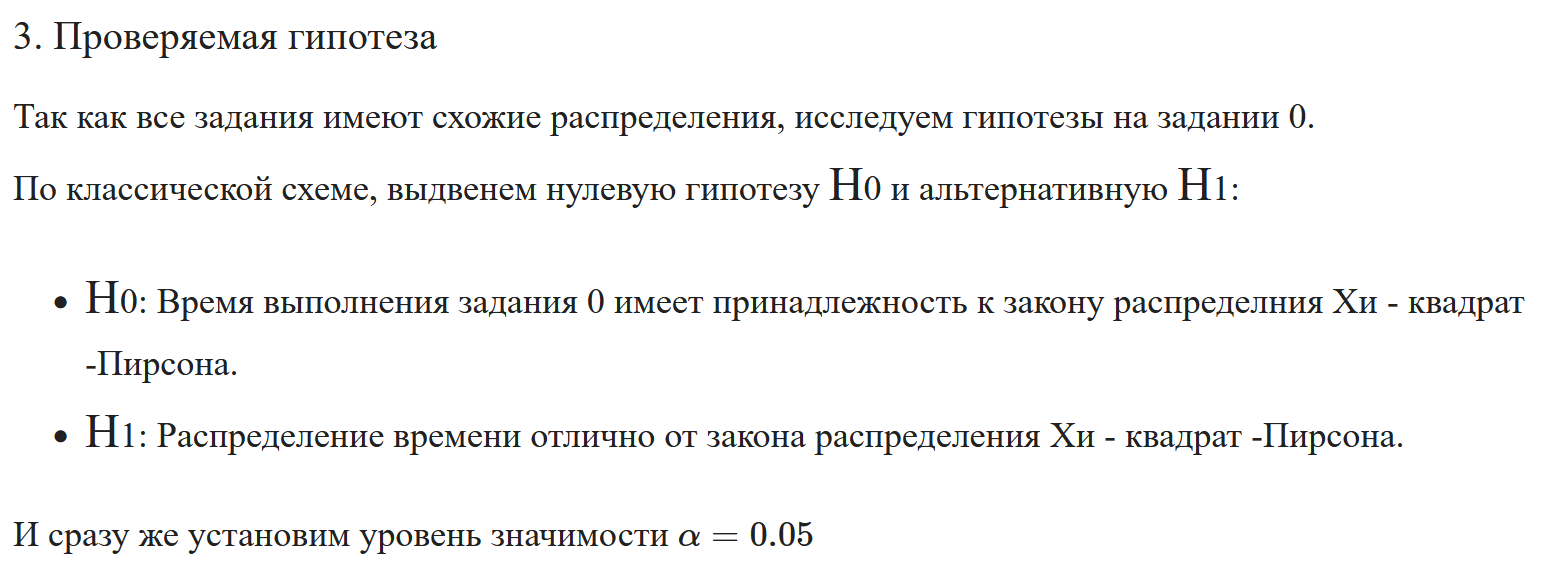


## Задание 2

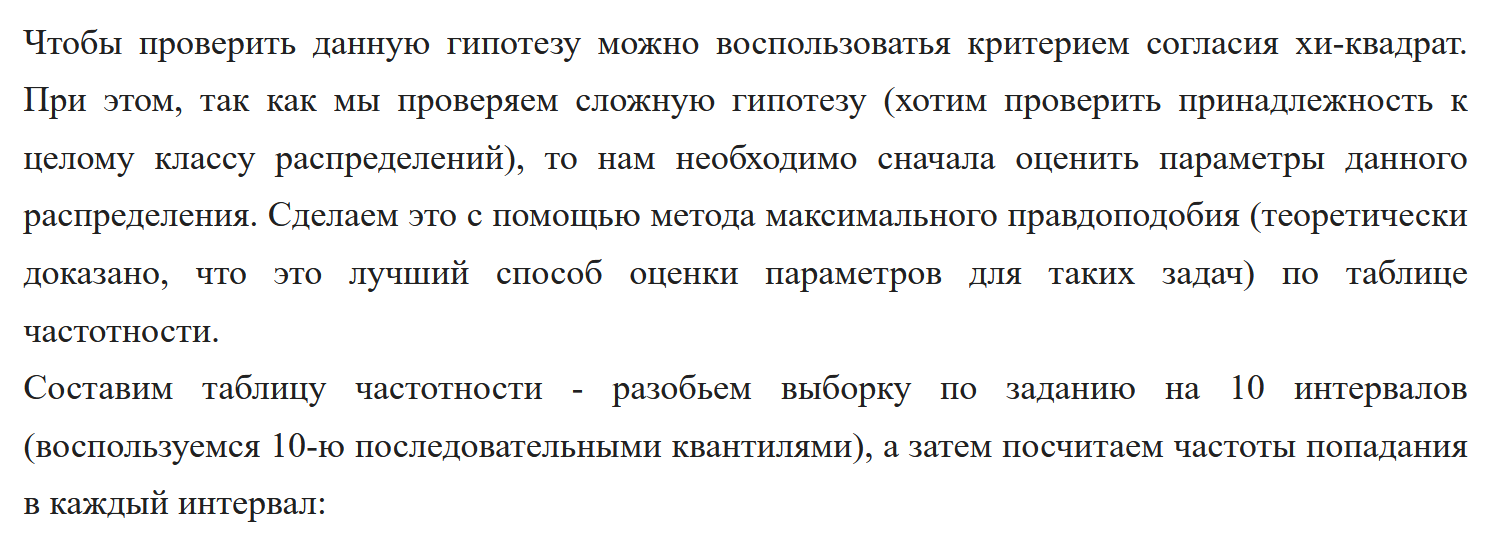


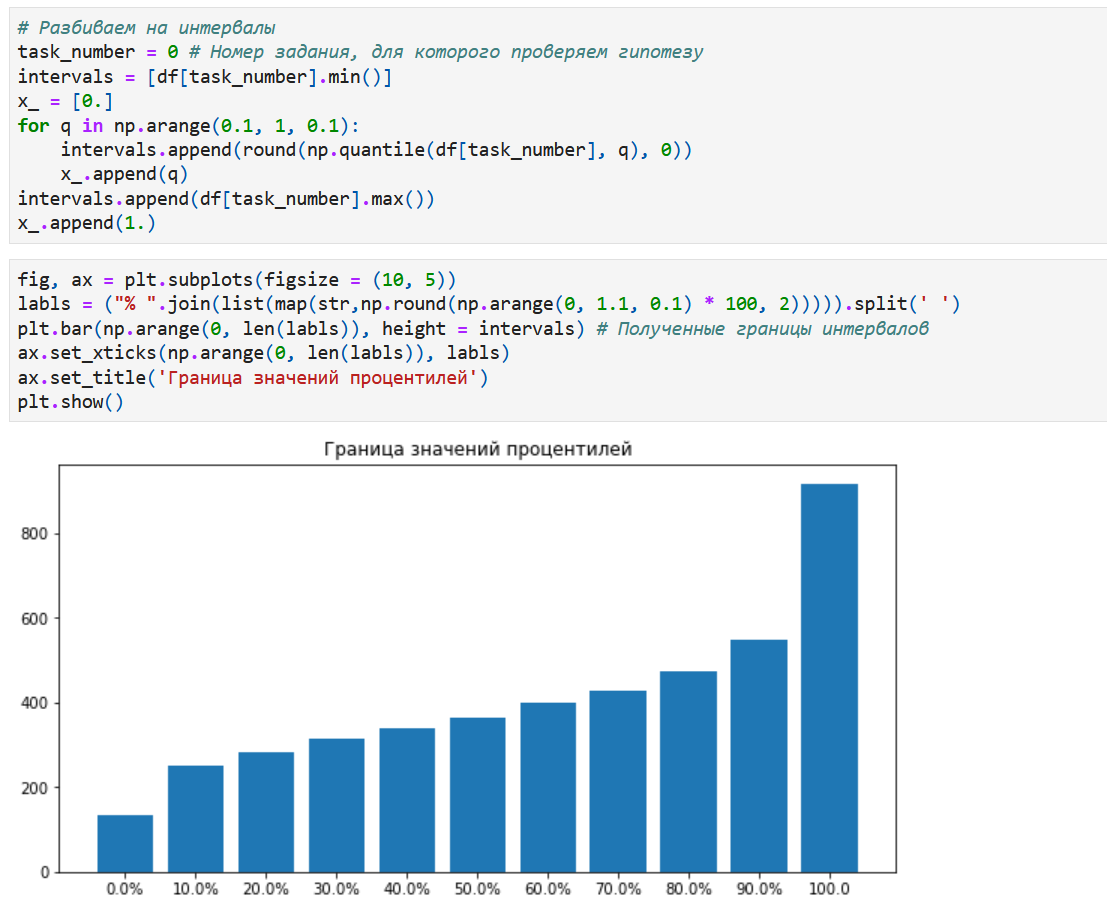
Можно заметить, что большинство распределений по форме отчетливо напоминают распределение хи-квадрат Пирсона.

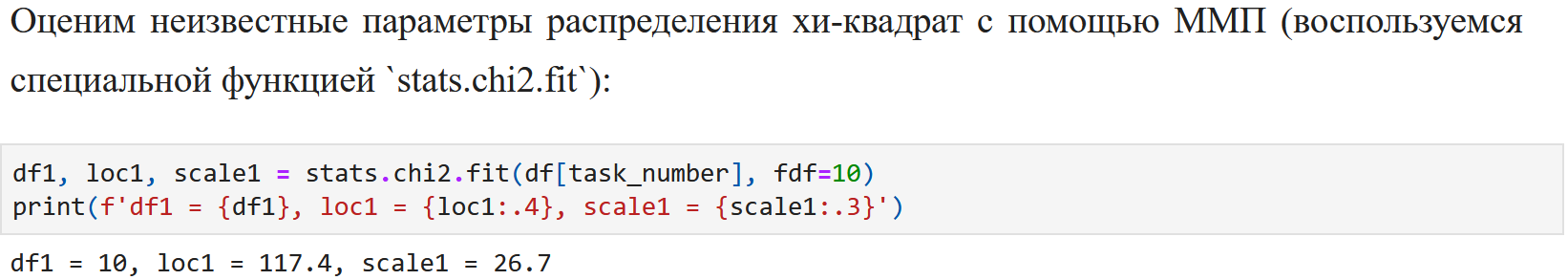
## Задание 3

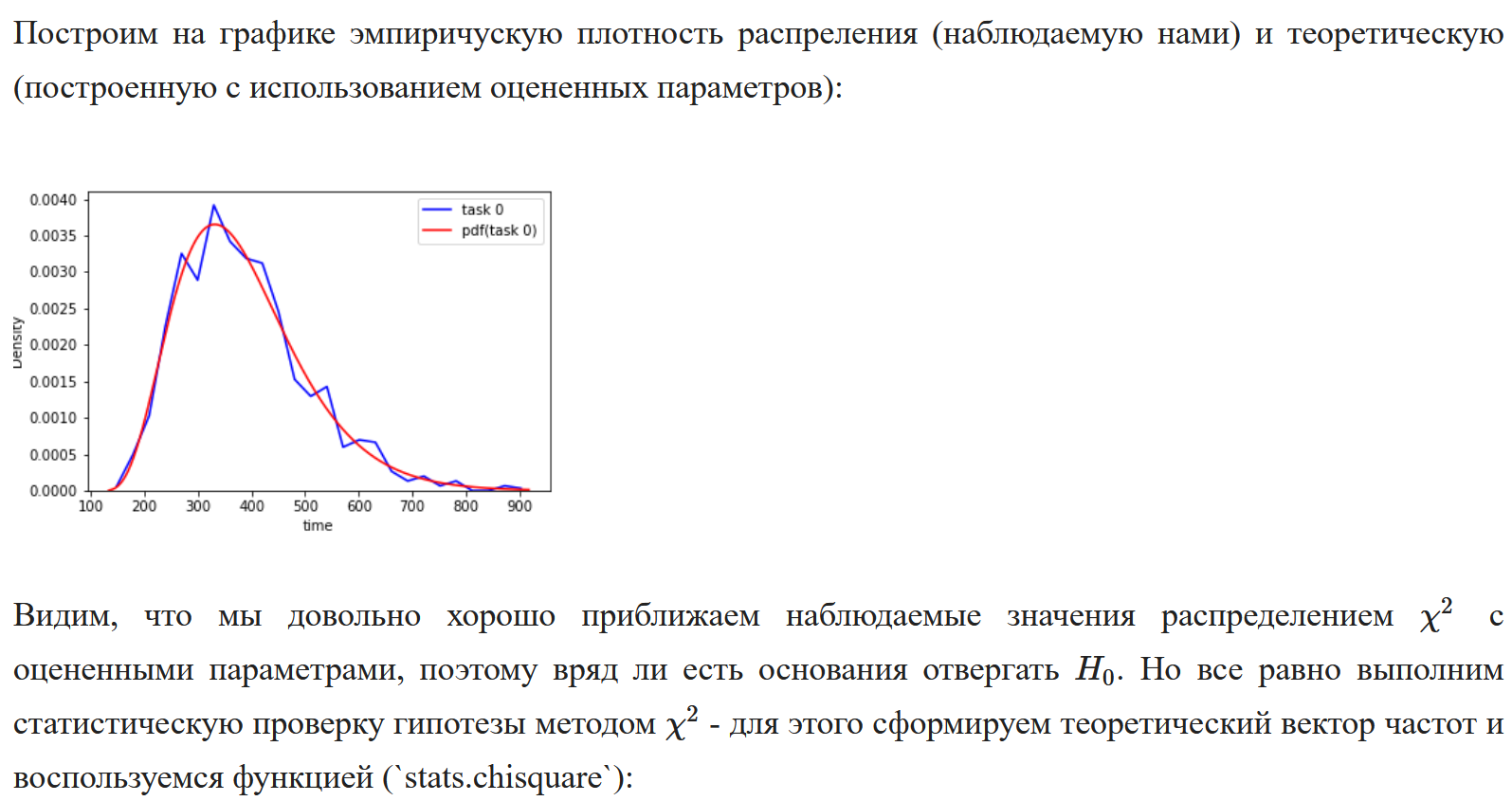


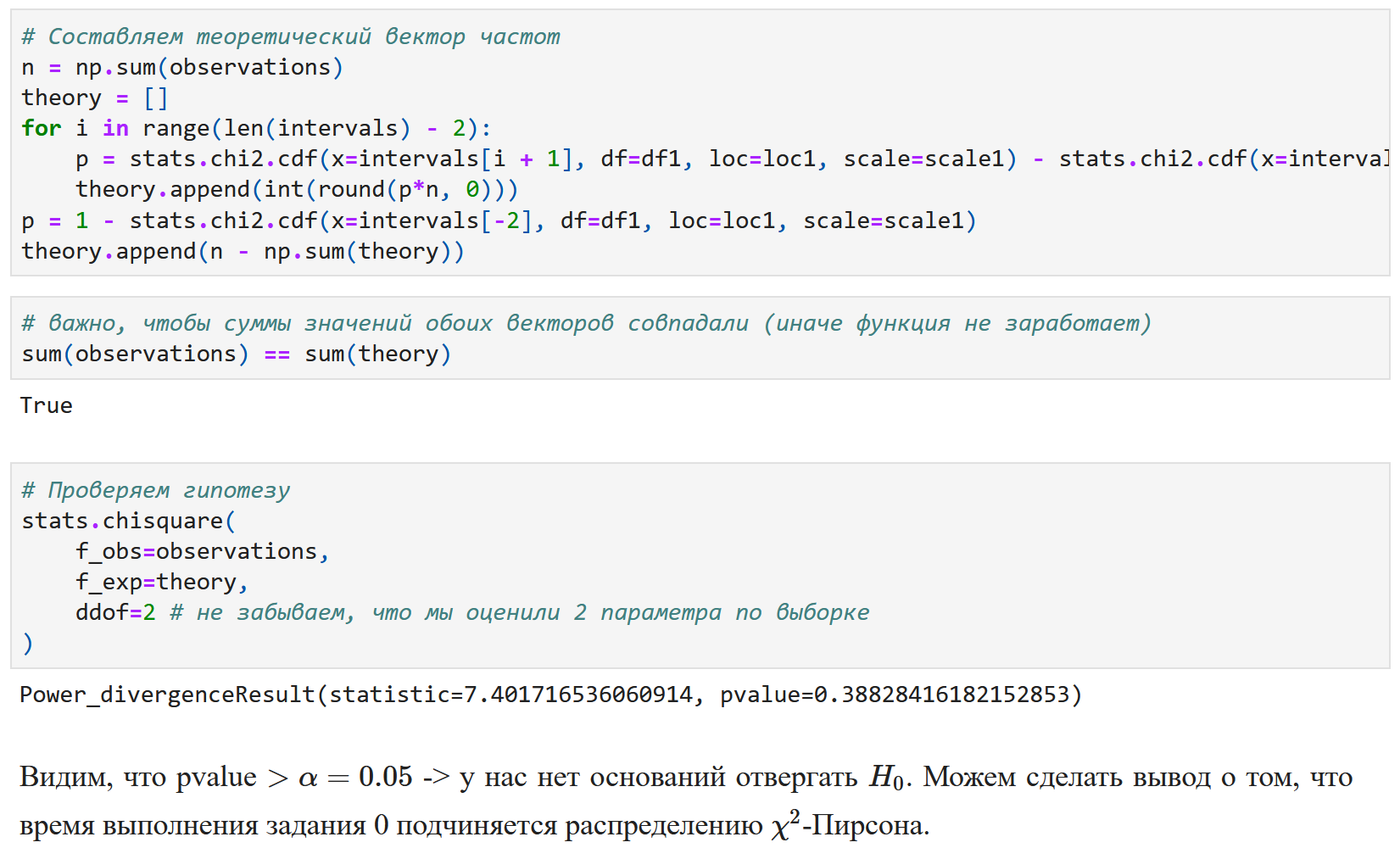
## Задание 4





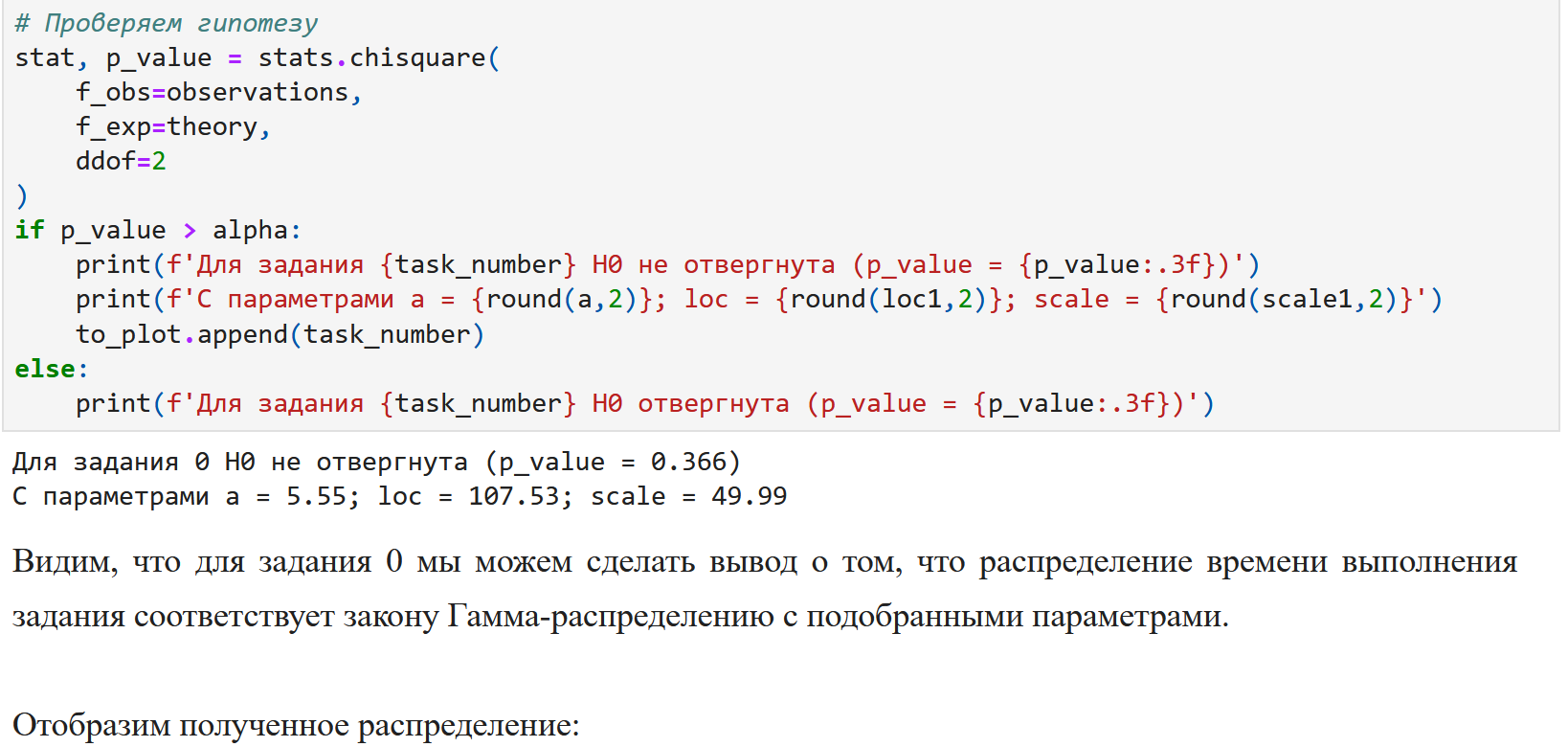


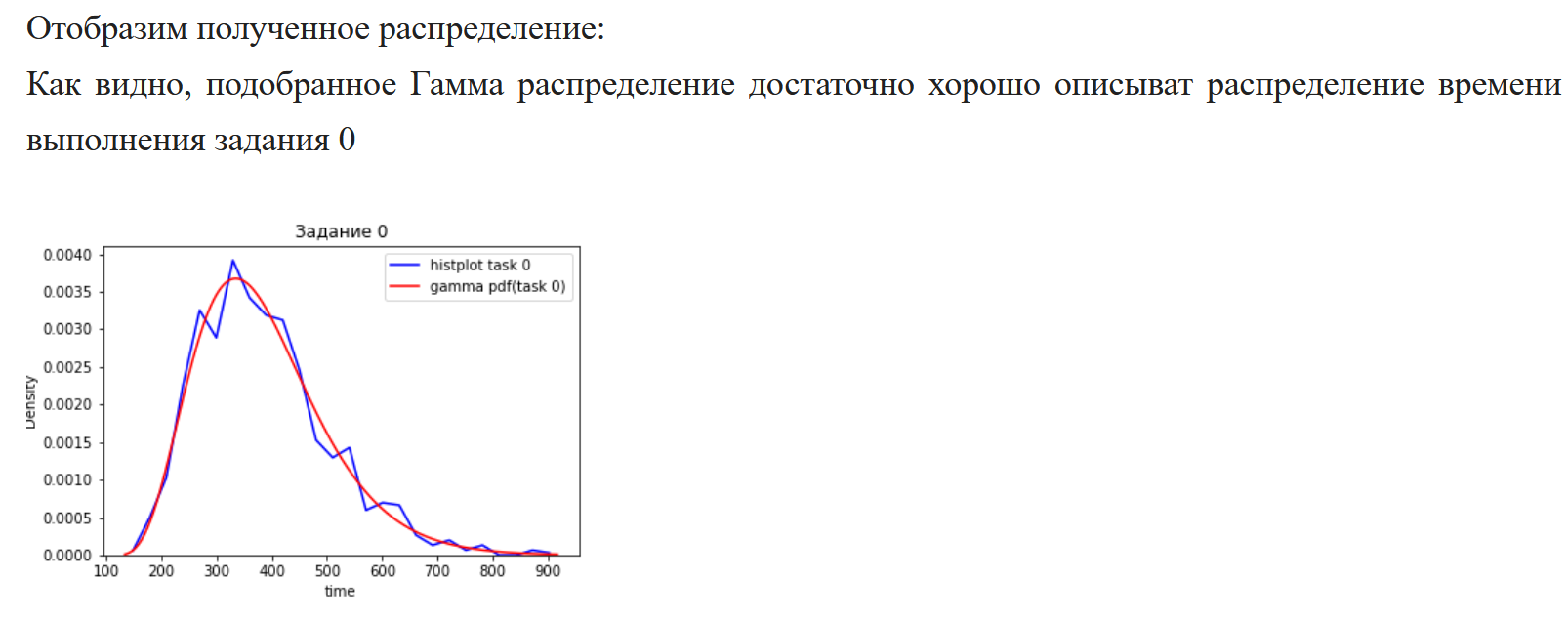




## Задание 5

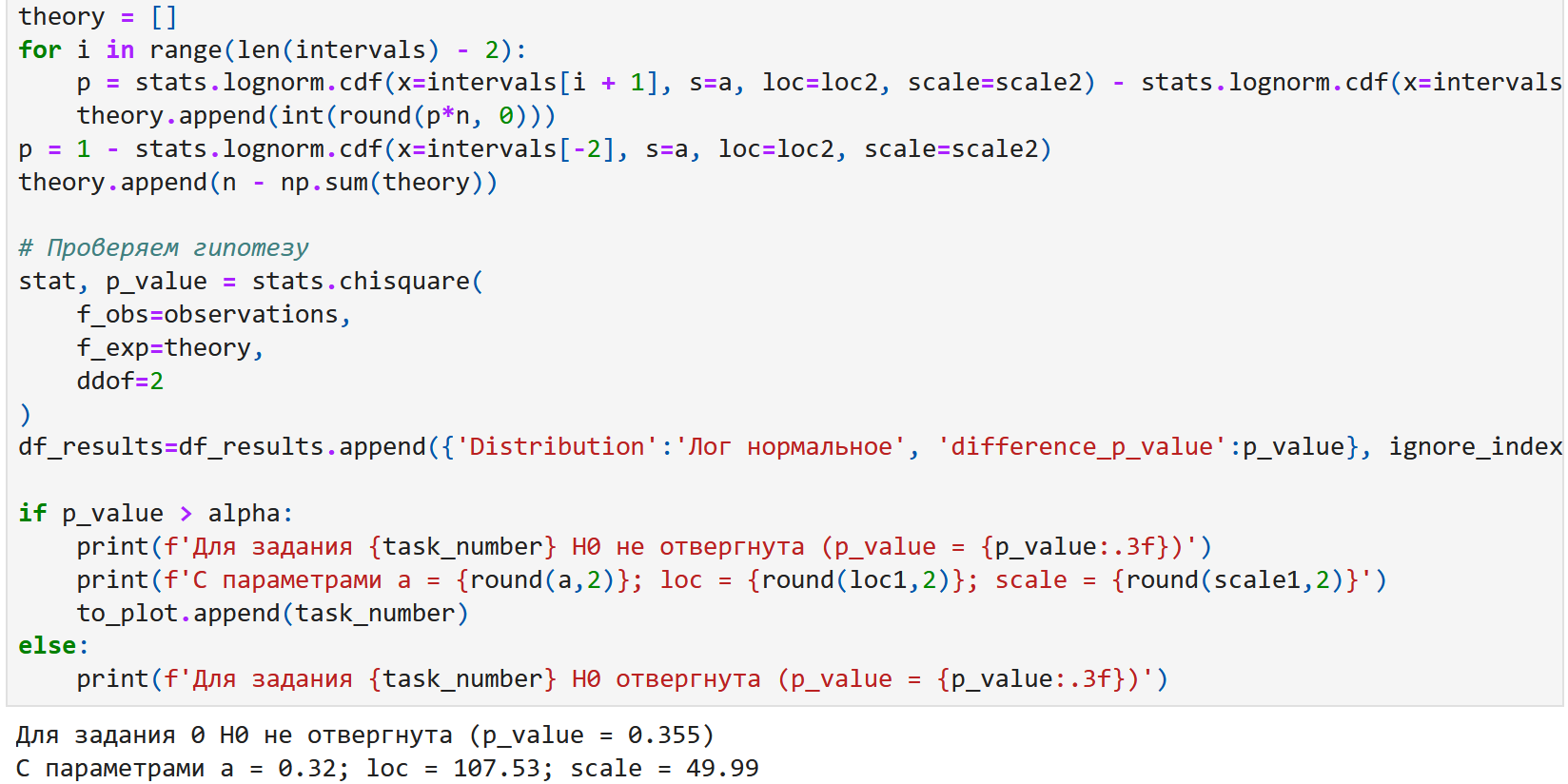
## 

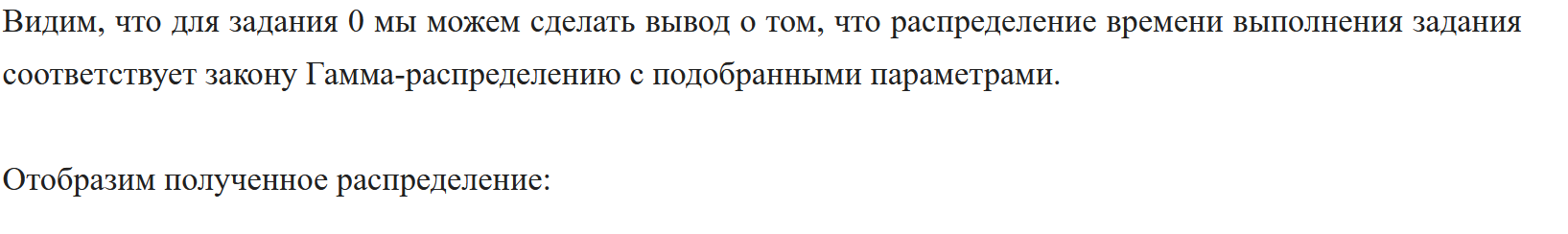


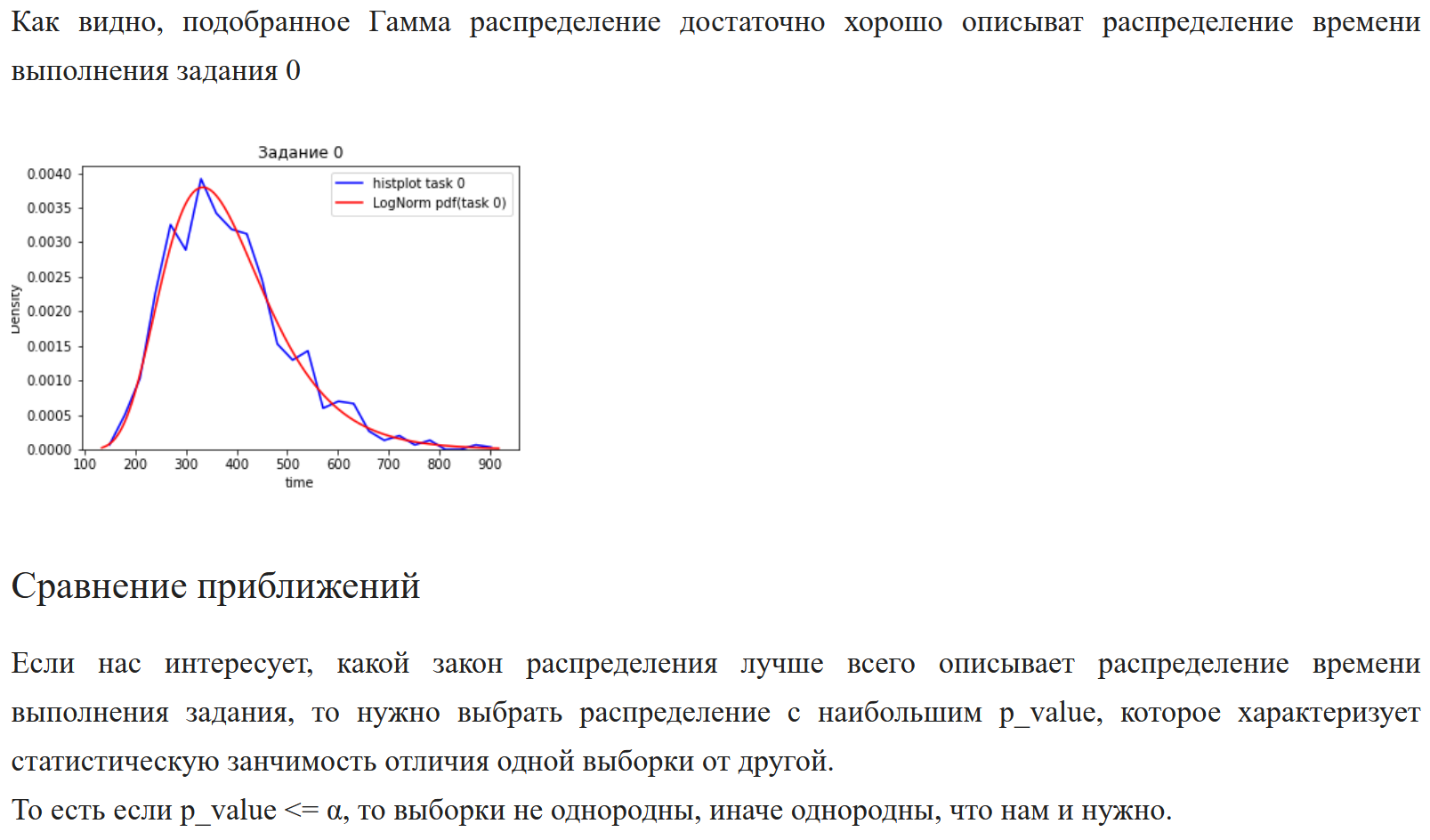


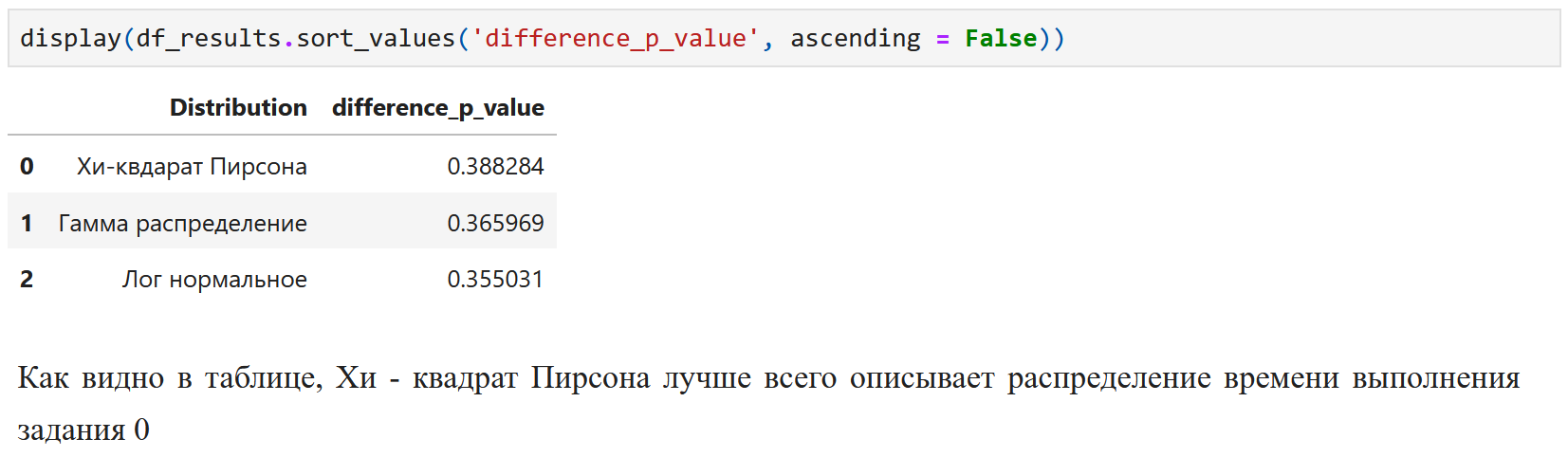
## Задание 5

## 









# 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 описывает 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 решения задания действительно хорошо описывает закон распределения Хи – квадрат Пирсона. Аналогичный алгоритм был проведен для закона Гамма – распределения и Логнормальное - распределения , которые тоже хорошо описывает распределение времени выполнения задания, но лучший результат – наибольшее статистическое различие, имеет приближение закона Хи – квадрат Пирсона. Все результаты сопровождены иллюстрирующими графиками.

# Приложение

# %%

import numpy as np

import pandas as pd

from scipy import stats

import seaborn as sns

import matplotlib.pyplot as plt

# %% [markdown]

# <h3><font face = 'Times New Roman'>1. Исходные данные</font></h3>

#

# %% [markdown]

# Загрузим набор данных:

# %%

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

display(df.head())

print('Shapes: ',df.shape)

# %% [markdown]

# <font size="4" face = 'Times New Roman'>

# Судя по результату метода <strong>head()</strong>, файл с данными имеет следующую структуру: <br>

#     <ul>

#     <li>Строки - № задания (i = 1, 2, ...)</li>

#     <li>Стобцы - № студента (j = 1, 2, ...)</li>

#     <li>Ячейки - время выполнения i-го задания j-м студентом</li>

#     </ul>

# </font>

# %% [markdown]

# <font size="4" face = 'Times New Roman'>Транспонируем данные для удобства обработки</font>

# %%

df = df.transpose()

df.head()

# %% [markdown]

# <h3><font face = 'Times New Roman'>2. Распределение времени решения заданий</font></h3>

# %%

#df.hist(figsize=(20, 20), bins = 25);

fig, ax = plt.subplots(4, 3, figsize = (15, 15))

n = len(df.columns.tolist())

cols = 0

for i in range(0, 4):

    for j in range(0, 3):

        if (cols < n):

            sns.histplot(df.iloc[:, cols], stat='density', common\_bins= False, common\_norm= False, kde = True, ax = ax[i][j], bins = 20)

        cols += 1

plt.savefig('distrib.png')

# %% [markdown]

# <div><font size="4" face = 'Times New Roman'>Построим гистграммы распределения времени по каждому из заданий:</font>

# <img src='https://github.com/JacKira/SDO/blob/main/distrib.png?raw=1' style = "width:70%; height:70%; padding:0; border:0" hspace = "0" vspace="0"></div>

# %% [markdown]

# <font size="3" face = 'Times New Roman'>Можно заметить, что большинство распределений по форме напоминают распределение хи-квадрат Пирсона.</font>

# %% [markdown]

# <h3><font  face = 'Times New Roman'>3. Проверяемая гипотеза</font></h3>

# <font size="4" face = 'Times New Roman'>Так как все задания имеют схожие распределения, исследуем гипотезы на задании 0.<br>

# По классической схеме, выдвенем нулевую гипотезу <font size = "5">H</font>0 и альтернативную <font size = "5">H</font>1: <br>

#     <ul>

#     <li><font size = "5">H</font>0: Время выполнения задания 0 имеет принадлежность к закону распределния Хи - квадрат -Пирсона.</li>

#     <li><font size = "5">H</font>1: Распределение времени отлично от закона распределения  Хи - квадрат -Пирсона.</li>

#     </ul>

#

# И сразу же установим уровень значимости $\alpha=0.05$

# <font>

# %% [markdown]

# <h3><font  face = 'Times New Roman'>4. Проверка гипотезы</font></h3>

# %% [markdown]

# <p style = "text-align:justify; margin-left:5%; margin-right:5%"><font size="4" face = 'Times New Roman'>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.  <br>

# Составим таблицу частотности - разобьем выборку по заданию  на 10 интервалов (воспользуемся 10-ю последовательными квантилями), а затем посчитаем частоты попадания в каждый интервал:

# </font>

# </p>

# %%

df\_results = pd.DataFrame({'Distribution':[], 'difference\_p\_value':[]})

df\_results.Distribution = df\_results.Distribution.astype(object)

df\_results.difference\_p\_value = df\_results.difference\_p\_value.astype(float)

# %%

# Разбиваем на интервалы

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

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

x\_ = [0.]

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

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

    x\_.append(q)

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

x\_.append(1.)

# %%

fig, ax = plt.subplots(figsize = (10, 5))

labls = ("% ".join(list(map(str,np.round(np.arange(0, 1.1, 0.1) \* 100, 2))))).split(' ')

plt.bar(np.arange(0, len(labls)), height = intervals) # Полученные границы интервалов

ax.set\_xticks(np.arange(0, len(labls)), labls)

ax.set\_title('Граница значений процентилей')

plt.show()

# %%

# Сотавляем вектор наблюдений (таблица частотности)

observations = []

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

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

observations

# %%

fig, ax = plt.subplots(figsize = (10, 5))

plt.bar(np.arange(0, len(observations)), height = observations)

ax.set\_title('Размер выборки в зависимости от интервала')

plt.show()

# %% [markdown]

# <h3><font  face = 'Times New Roman'> 5. Получение оценок параметров</font><h3>

# %% [markdown]

# <p style = 'text-align:justify'><font size = "4"  face = 'Times New Roman'>

# Оценим неизвестные параметры распределения хи-квадрат с помощью ММП (воспользуемся специальной функцией `stats.chi2.fit`):</font>

# </p>

# %%

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

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

# %%

fig, ax = plt.subplots()

sns.histplot(data=df[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');

plt.savefig('est\_plot.png')

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Построим на графике эмпиричускую плотность распреления (наблюдаемую нами) и теоретическую (построенную с использованием оцененных параметров):</font></p>

# <div>

# <img src='https://github.com/JacKira/SDO/blob/main/est\_plot.png?raw=1' style = "width:40%; height:40%; padding:0; border:0" hspace = "0" vspace="0"></div>

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Видим, что мы довольно хорошо приближаем наблюдаемые значения распределением $\chi^2$ с оцененными параметрами, поэтому вряд ли есть основания отвергать $H\_0$.

# Но все равно выполним статистическую проверку гипотезы методом $\chi^2$ - для этого сформируем теоретический вектор частот и воспользуемся функцией (`stats.chisquare`):

# </font>

# </p>

# %%

# 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))

# %%

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

sum(observations) == sum(theory)

# %%

# Проверяем гипотезу

res = stats.chisquare(

    f\_obs=observations,

    f\_exp=theory,

    ddof=2 # не забываем, что мы оценили 2 параметра по выборке

)

df\_results=df\_results.append({'Distribution':'Хи-квдарат Пирсона', 'difference\_p\_value':res[1]}, ignore\_index = True)

res

# %%

df\_results

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Видим, что pvalue $ > \alpha=0.05$ -> у нас нет оснований отвергать $H\_0$. Можем сделать вывод о том, что время выполнения задания 0 подчиняется распределению $\chi^2$-Пирсона.</font></p>

# %% [markdown]

# ### Гамма распределение

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Получим параметры подходящих гамма-распределений и проверим гипотезу согласия наблюдаемых выборок полученным гамма-распределениям:</font></p>

# %%

alpha = 0.05 # уровень значимости

to\_plot = []

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

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

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

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

# составляем вектор наблюдаемых частот

observations = []

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

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

# Определяем оценки параметров Гамма-распределения методом ММП

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

# 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

)

df\_results=df\_results.append({'Distribution':'Гамма распределение', 'difference\_p\_value':p\_value}, ignore\_index = True)

if p\_value > alpha:

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

    print(f'C параметрами a = {round(a,2)}; loc = {round(loc1,2)}; scale = {round(scale1,2)}')

    to\_plot.append(task\_number)

else:

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

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Видим, что для задания 0 мы можем сделать вывод о том, что распределение времени выполнения задания соответствует закону Гамма-распределению с подобранными параметрами.  <br><br>

# Отобразим полученное распределение:

# </font>

# </p>

# %%

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

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

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

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

observations = []

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

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

a, loc1, scale1 = stats.gamma.fit(df[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}')

sns.histplot(data=df[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');

plt.savefig('gamma\_est\_plot.png')

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Отобразим полученное распределение:<br>Как видно, подобранное Гамма распределение достаточно хорошо описыват распределение времени выполнения задания 0</font>

# <div>

# <img src='https://github.com/JacKira/SDO/blob/main/gamma\_est\_plot.png?raw=1' style = "width:40%; height:40%; padding:0; border:0" hspace = "0" vspace="0"></div></p>

# %% [markdown]

# ### Лог нормальное

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Получим параметры подходящих лог - нормальных распределений и проверим гипотезу согласия наблюдаемых выборок полученным лог-нормальным:</font></p>

# %%

alpha = 0.05 # уровень значимости

to\_plot = []

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

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

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

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

# составляем вектор наблюдаемых частот

observations = []

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

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

# Определяем оценки параметров Гамма-распределения методом ММП

a, loc2, scale2 = stats.lognorm.fit(df[task\_number])

# Составляем теоретических ветор частот

n = np.sum(observations)

theory = []

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

    p = stats.lognorm.cdf(x=intervals[i + 1], s=a, loc=loc2, scale=scale2) - stats.lognorm.cdf(x=intervals[i], s=a, loc=loc2, scale=scale2)

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

p = 1 - stats.lognorm.cdf(x=intervals[-2], s=a, loc=loc2, scale=scale2)

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

# Проверяем гипотезу

stat, p\_value = stats.chisquare(

    f\_obs=observations,

    f\_exp=theory,

    ddof=2

)

df\_results=df\_results.append({'Distribution':'Лог нормальное', 'difference\_p\_value':p\_value}, ignore\_index = True)

if p\_value > alpha:

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

    print(f'C параметрами a = {round(a,2)}; loc = {round(loc1,2)}; scale = {round(scale1,2)}')

    to\_plot.append(task\_number)

else:

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

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Видим, что для задания 0 мы можем сделать вывод о том, что распределение времени выполнения задания соответствует закону Гамма-распределению с подобранными параметрами.  <br><br>

# Отобразим полученное распределение:

# </font>

# </p>

# %%

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

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

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

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

observations = []

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

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

# Составляем теоретических ветор частот

n = np.sum(observations)

theory = []

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

    p = stats.lognorm.cdf(x=intervals[i + 1], s=a, loc=loc2, scale=scale2) - stats.lognorm.cdf(x=intervals[i], s=a, loc=loc2, scale=scale2)

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

p = 1 - stats.lognorm.cdf(x=intervals[-2], s=a, loc=loc2, scale=scale2)

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

fig, ax = plt.subplots()

ax.set\_title(f'Задание {task\_number}')

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

lnorm = stats.lognorm(s=a, loc=loc2, scale=scale2)

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

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

ax.set\_xlabel('time');

plt.savefig('lognorm\_est\_plot.png')

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Отобразим полученное распределение:<br>Как видно, подобранное Гамма распределение достаточно хорошо описыват распределение времени выполнения задания 0</font>

# <div>

# <img src='lognorm\_est\_plot.png' style = "width:40%; height:40%; padding:0; border:0" hspace = "0" vspace="0"></div></p>

# %% [markdown]

# <h3><font  face = 'Times New Roman'> Сравнение приближений</font></h3>

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Если нас интересует, какой закон распределения лучше всего описывает распределение времени выполнения задания, то нужно выбрать распределение с наибольшим p\_value, которое характеризует статистическую занчимость отличия одной выборки от другой. <br>То есть если  p\_value <= α, то выборки не однородны, иначе однородны, что нам и нужно.</font></p>

# %%

display(df\_results.sort\_values('difference\_p\_value', ascending = False))

# %% [markdown]

# <p style = 'text-align:justify'><font size = "3"  face = 'Times New Roman'>Как видно в таблице, Хи - квадрат Пирсона лучше всего описывает распределение времени выполнения задания 0</font></p>

# %%