Постановка задачи

Есть случайная величина, подчиняющаяся стандартному нормальному закону распределения $\xi \sim N(0,1)$

1. Необходимо получить k реализаций этой случайной величины

2. Построить гистограмму полученной выборки
3. Получить эмпирические характеристики данной выборки
4. С помощью критерия χ^2 проверить гипотезу о нормальном распределении

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as sps
```

Получение k реализаций случайной величины, распределенной по стандартному нормальному закону

```
k = int(input('Количество реализаций случайной величины: '))

played_selection = []
for i in range(k):
    number =  sum(np.random.uniform(0, 1, 12)) - 6    # сумма 12 случайных чисел (распр. по R[0,1]) минус мат.ожидание
    played_selection.append(number)   # добавляем полученное число в выборку

result = pd.Series(played_selection, index = [f'x{i}' for i in range(1, k+1)],
                   name = 'Разыгранная случайная величина с законом N(0, 1)')
result
```

```
Количество реализаций случайной величины: 1000

x1      -0.180691
x2       0.540281
x3      -1.427348
x4       0.154750
x5      -1.247620
           ...
x996    -1.637937
x997     0.081283
x998     1.102961
x999    -0.282373
x1000   -1.387810
```

```
Name: Разыгранная случайная величина с законом N(0, 1), Length: 1000,
dtype: float64
```

Разбиение интервала, построение группировочного ряда

```
variable = result.copy()
# строим вариационный ряд
variance_var = pd.Series(sorted(variable), index = [f'x*{i}' for i in
range(1, k+1)], name = 'Вариационный ряд')
print(variance_var, '\n', '-'*55)

# формула Стерджесса для определения числа интервалов
m_opt = round(1 + 3.322 * np.log10(k))
print('Оптимальное число интервалов = {}'.format(m_opt))
intervals = np.linspace(variance_var[0], variance_var[k-1], m_opt+1) #
интервалы
```

```
x*1      -2.915948
x*2      -2.830780
x*3      -2.815601
x*4      -2.652653
x*5      -2.256792
            ...
x*996     2.475760
x*997     2.493062
x*998     2.630220
x*999     2.957998
x*1000    3.319162
Name: Вариационный ряд, Length: 1000, dtype: float64
 -------------------------------------------------------
Оптимальное число интервалов = 11
```

Подсчитаем частоты для каждого интервала

```
interval_groups = pd.cut(variance_var, bins = intervals,
include_lowest=True) # разбиваем выборку по интервальным группам
interval_groups.name = 'Интервалы'
df = pd.concat([variance_var, interval_groups], axis=1)


interval_groups = interval_groups.value_counts(sort=False) #
подсчитываем количество элементов в каждом интервале
print(interval_groups, '-'*55, df, sep='\n')
```

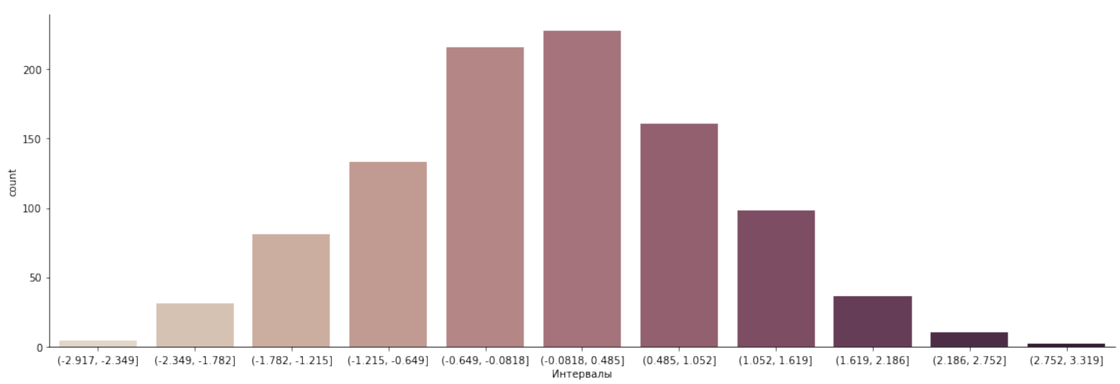
```
(-2.917, -2.349]       4
(-2.349, -1.782]      31
(-1.782, -1.215]      81
(-1.215, -0.649]     133
(-0.649, -0.0818]    216
(-0.0818, 0.485]     228
(0.485, 1.052]       161
(1.052, 1.619]        98
```

```
(1.619, 2.186]        36
(2.186, 2.752]        10
(2.752, 3.319]         2
Name: Интервалы, dtype: int64
-------------------------------------------------------
        Вариационный ряд         Интервалы
x*1            -2.915948  (-2.917, -2.349]
x*2            -2.830780  (-2.917, -2.349]
x*3            -2.815601  (-2.917, -2.349]
x*4            -2.652653  (-2.917, -2.349]
x*5            -2.256792  (-2.349, -1.782]
...                  ...               ...
x*996           2.475760    (2.186, 2.752]
x*997           2.493062    (2.186, 2.752]
x*998           2.630220    (2.186, 2.752]
x*999           2.957998    (2.752, 3.319]
x*1000          3.319162    (2.752, 3.319]

[1000 rows x 2 columns]
```

Гистограмма разыгранной выборки

```
# Построение гистограммы для эмпирического распределения
histogramm = sns.catplot(x="Интервалы", kind="count",
palette="ch:.25", data=df, height = 5, aspect=3)
```



Экспресс-анализ (дескриптивная статистика)

```
# найдем середины интервалов
middle_of_intrvls = []
temp_intervals = np.unique(df['Интервалы'].values)
for ti in temp_intervals:
    middle_of_intrvls.append(ti.mid)
emperic_distribution = pd.Series(interval_groups.values,
index=middle_of_intrvls,
                             name = 'Интервальное разбиение выборки')
print(emperic_distribution)

# описательные характеристики
emperic_mean = sum(interval_groups.values * middle_of_intrvls) / k
```

```
emperic_moda =
(emperic_distribution.index[emperic_distribution==max(emperic_distribu
tion)])[0] # середина интервала с наиб. частотой
emperic_median =
middle_of_intrvls[int(np.where(temp_intervals==df['Интервалы'][k//2])
[0])]
emperic_disp = sum(emperic_distribution * (middle_of_intrvls -
emperic_mean)**2) / k
emperic_deviation = emperic_disp**(1/2)
emperic_excess = sum(emperic_distribution * (middle_of_intrvls -
emperic_mean)**4) / (k * emperic_diviation**4) - 3
emperic_assim = sum(emperic_distribution * (middle_of_intrvls -
emperic_mean)**3) / (k * emperic_diviation**3)

# поместим характеристики в отдельную структуру
emperic_description = pd.Series([emperic_mean, emperic_disp,
emperic_deviation, emperic_excess,
                                emperic_assim, emperic_moda,
emperic_median],
                               index=['Выборочное среднее',
'Выборочная дисперсия', 'Выборочное отклонение',
                                     'Выборочный эксцесс', 'Выборочная
ассиметрия', 'Мода', 'Медиана'],
                              name='Описательные статистики')
emperic_description
```

```
-2.6330      4
-2.0655     31
-1.4985     81
-0.9320    133
-0.3654    216
 0.2016    228
 0.7685    161
 1.3355     98
 1.9025     36
 2.4690     10
 3.0355      2
Name: Интервальное разбиение выборки, dtype: int64
```

```
Выборочное среднее       0.001000
Выборочная дисперсия     0.975068
Выборочное отклонение    0.987455
Выборочный эксцесс      -0.662330
Выборочная ассиметрия    0.020791
Мода                     0.201600
Медиана                  0.201600
Name: Описательные статистики, dtype: float64
```

Теоретическое распределение

(найдем теоретические частоты для середин интервалов с помощью формулы $n_i = \frac{k * f(t_i) * h}{\sigma_{emperic}}$, где h - расстояние между серединами, k - общий объем выборки, $\sigma_{emperic}$ - эмпирическое отклонение, $f(\hat{t}_i)$ - значение плотности стандартно распределенной случайной величины в нормированной середине i-го интервала: $\hat{t}_i = \frac{t_i - a_{emperic}}{\sigma_{emperic}}$)

```
normal_frequency = sps.norm().pdf((middle_of_intrvls-
emperic_mean)/emperic_deviation)
normal_frequency *= k*(middle_of_intrvls[1]-
middle_of_intrvls[0])/emperic_deviation
sum(normal_frequency)
normal_distribution = pd.Series(normal_frequency, index =
middle_of_intrvls,
                                name = 'Теоретическое распределение
стандартной нормальной величины')
normal_distribution
```

```
-2.6330      6.535503
-2.0655     25.664777
-1.4985     72.379994
-0.9320    146.723938
-0.3654    214.023400
 0.2016    224.593357
 0.7685    169.502395
 1.3355     91.993034
 1.9025     35.904026
 2.4690     10.089976
 3.0355      2.040322
Name: Теоретическое распределение стандартной нормальной величины,
dtype: float64
```

Проверка статистической гипотезы

Есть случайная величина ψ, представленная выборкой, полученной на 1 этапе. Требуется проверить гипотезу о том, что данная величина подчинена стандартному нормальному закону распределения. Для проверки гипотезы будет использоваться критерий Пирсона.
$H_0 : \psi$ подчинена стандартному нормальному закону распределения $N(0,1)$
$H_1 : \psi$ не подчинена стандартному нормальному закону распределения $N(0,1)$

```
# зададим уровень значимости и найдем критическую область с помощью
alpha-квантиля распределения хи2
# с m_opt-params-1 степенями свободы
alpha = float(input('Уровень значимости: '))
```

```
critical_value = sps.chi2.isf(q=alpha, df=m_opt-2-1) # у норм.
величины два параметра
print('Критическая правосторонняя область: [{};
+inf]'.format(critical_value))

df3 = pd.concat([emperic_distribution, normal_distribution], axis=1)
df3 = df3.rename(columns = {'Интервальное разбиение выборки':
'Эмпирические частоты',
                     'Теоретическое распределение стандартной
нормальной величины': 'Теоретические частоты'})
df3
```

```
Уровень значимости: 0.05
Критическая правосторонняя область: [15.507313055865454;+inf]
```

	Эмпирические частоты	Теоретические частоты
-2.6330	4	6.535503
-2.0655	31	25.664777
-1.4985	81	72.379994
-0.9320	133	146.723938
-0.3654	216	214.023400
0.2016	228	224.593357
0.7685	161	169.502395
1.3355	98	91.993034
1.9025	36	35.904026
2.4690	10	10.089976
3.0355	2	2.040322

Проверка по критерию хи-квадрат Пирсона $\sum_{i=1}^{m_{opt}} \frac{\left(\hat{n}_i - n_i\right)^2}{n_i}$

```
value = sum(((df3['Эмпирические частоты'] - df3['Теоретические
частоты'])**2)/df3['Теоретические частоты'])
print('Наблюдаемое значение критерия = ', value)
if value > critical_value:
    print('С заданным уровнем значимости {} гипотеза о согласовании
эмпирического закона распределения со' \
    ' стандартным нормальным законом распределения
отклоняется'.format(alpha))
else:
    print('С заданным уровнем значимости {} гипотеза о согласовании
эмпирического закона распределения со' \
    ' стандартным нормальным  законом распределения
принимается'.format(alpha))
```

```
Наблюдаемое значение критерия =  5.293543779527519
С заданным уровнем значимости 0.05 гипотеза о согласовании
эмпирического закона распределения со стандартным нормальным  законом
распределения принимается
```