Российский Государственный Университет нефти и газа имени И.М.Губкина

Кафедра «Прикладной математики и компьютерного моделирования»

Лабораторная работа №3:

по предмету «Математическая статистика»

Выполнил работу:

студент гр. АМ-20-06

Коротченя И.С.

Проверил: Рыков В.В., Кузнецов П.В.

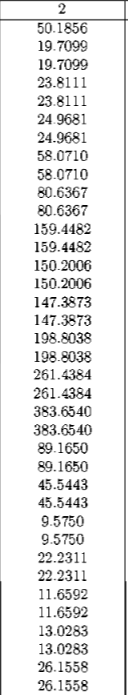
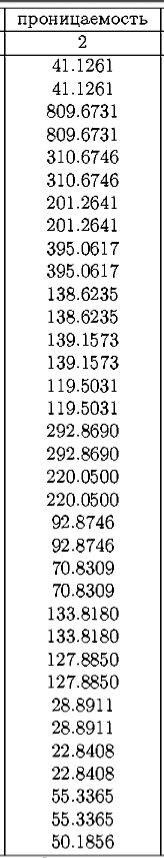
Москва   
2023 г.

**Задача лабораторной работы**

По данным о проницаемости скважин и сгенерированным выборкам проверить гипотезу о принадлежности выборки распределению по критерию хи-квадрат и по критерию Колмогорова.

Дополнительно сгенерировать выборку с распределением Колмогорова.

**Проницаемость скважин:**

****

**Критерий хи-квадрат**

Проверка гипотезы с использованием критерия типа предусматривает групировку наблюдений. Область определения случайной величины разбивают на непересекающихся интервалов

Где – нижняя граница области определения случайной величины; – верхняя грань.

В соответствии с заданным разбиением рассчитаем число выборочных значений, попавших в i-й интервал. Тогда соответсвующие частоты:

При проверке простой гипотезы известны как вид закона , так и все его параметры.

Вероятности попадания в интервал для предполагаемой (теоретической) функции распределения :

Согласно ЦПТ статистика Z является асимптотически нормальной:

Статистика критерия согласия Пирсона определяется соотношением

Тогда критерий согласия:

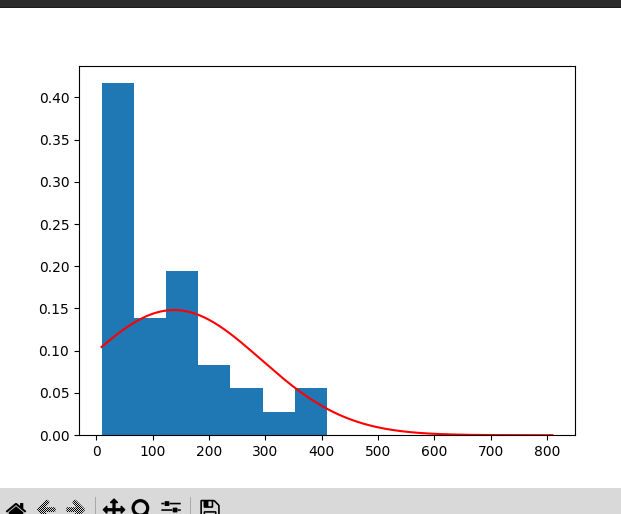
В случае проверки простой гипотезы, в пределе при эта статистика подчиняется - распределению с степенями свободы, если верна проверяемая гипотеза . Плотность - распределения, которое является частным случаем гамма-распределения, описывается формулой

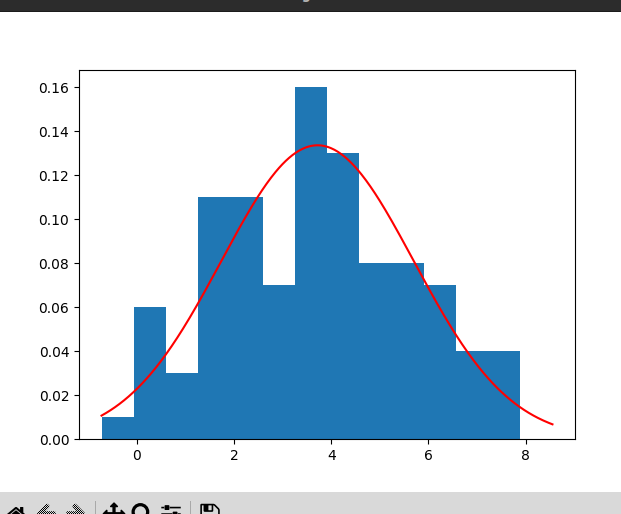
Проверим распределение проницаемости на соответствие нормальному распределению.

|  |  |  |  |
| --- | --- | --- | --- |
| Данные | Значение хи-квадрат | Квантиль хи-квадрат распределения | а |
| проницаемости | 76.47 | 30 | 0.95 |
| сгенерированные | 20.96 | 30.14 | 0.95 |

Нулевая гипотеза состоит в том, что рассматриваемую величину можно считать нормально распределенной. Так как хи-квадрат больше квантиля, нулевая гиптоеза о нормальном распределении отвергается.

Для наглядности построим гистограмму для случайной величины.



****

**Критерий Колмогорова**

Пусть эмпирическая функция распределения (ЭФР) , построенная по выборке , имеет вид:

где указывает, попало ли наблюдение в область :

Проверим, является ли выборка порождённой случайной величиной с функцией распределения .

Статистика критерия К:

Обозначим нулевую гипотезу , как гипотезу о том, что выборка подчиняется теоритическому распределению. Тогда по теореме Колмогорова для введённой статистики справедливо:

Критерий:

|  |  |  |
| --- | --- | --- |
| Данные | D | Квантиль |
| проницаемости | 0.195 | 0.016 |
| сгенерированные | 0.04 | 0.014 |

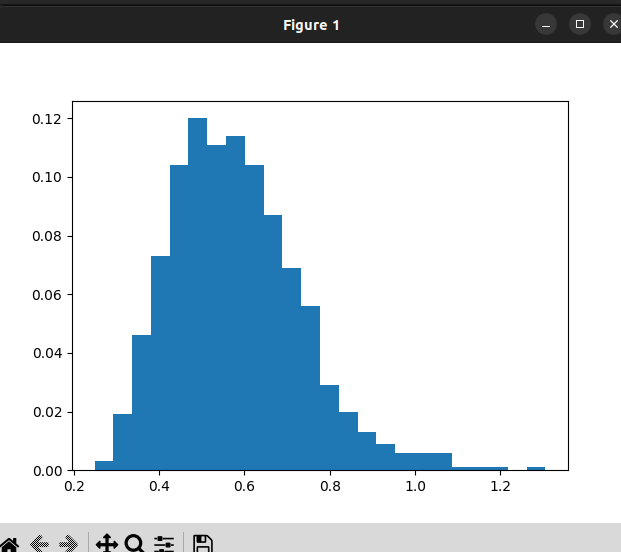
В нашем случае нулевая гипотеза состоит в том, что распределение случайной величины является нормальным. Так как D>квантиля, то гипотеза о нормальном распределении отвергается.

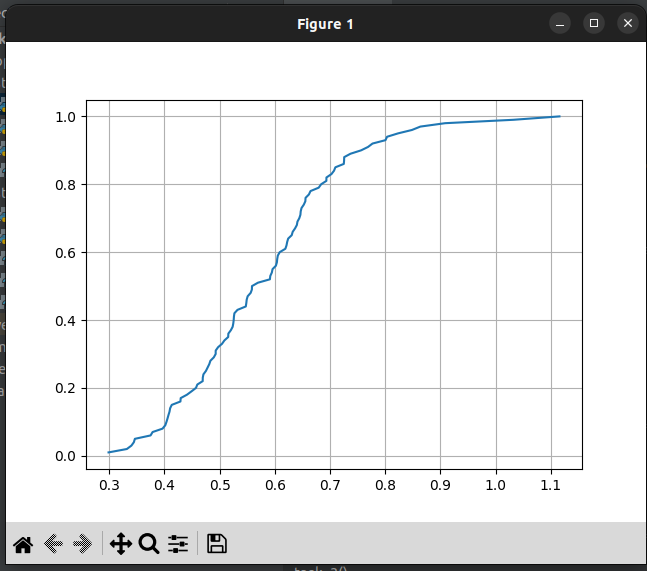
**Сгенерировать выборки с распределением Колмогорова**

Распределение Колмогорова – абсолютно непрерывное распределение. Случайная величина имеет распределение Колмогорова, если её функция распределения имеет вид:

Для того, чтобы получить случайную величину с распределением Колмогорова необходимо сгенерировать несколько случайных величин с непрерывной функцией распределения . Пусть – выборочная функция распределения.

Тогда величина имеет распределение Колмогорова.





Код для лабораторной работы на языке Python:

import math  
  
import numpy as np  
from scipy import stats  
import matplotlib.pyplot as plt  
  
  
data = '1968.492 41.1261 0.0857 60.7957 2216.473 41.1261 0.0857 41.6195 2216.722 809.6731 0.1788 43.5278 2221.238 809.6731 0.1788 34.2727 2221.242 310.6746 0.1813 34.2839 2230.852 310.6746 0.1813 49.5744 2230.862 201.2641 0.1654 49.6587 2240.280 201.2641 0.1654 48.5889 2240.289 395.0617 0.1730 48.6801 2249.526 395.0617 0.1730 63.8574 2249.535 138.6235 0.1413 63.9331 2251.817 138.6235 0.1413 73.2354 2251.819 139.1573 0.1586 73.2369 2260.172 139.1573 0.1586 73.9546 2260.180 119.5031 0.1353 73.9877 2269.059 119.5031 0.1353 78.7947 2269.068 292.8690 0.1453 78.8402 2277.819 292.8690 0.1453 88.6888 2277.828 220.0500 0.1566 88.6946 2286.453 220.0500 0.1566 70.0894 2286.462 92.8746 0.1440 70.1490 2294.964 92.8746 0.1440 80.3080 2294.972 70.8309 0.1499 80.2755 2303.363 70.8309 0.1499 71.7327 2303.372 133.8180 0.1567 71.6800 2311.662 133.8180 0.1567 73.2779 2311.670 127.8850 0.1502 73.2854 2319.864 127.8850 0.1502 84.1279 2319.872 28.8911 0.1270 84.2226 2321.135 28.8911 0.1270 99.8326 2321.136 22.8408 0.1323 99.8410 2329.262 22.8408 0.1323 92.3621 2329.270 55.3365 0.1447 92.3515 2337.311 55.3365 0.1447 90.0307 2337.319 50.1856 0.1411 90.0423 2345.281 50.1856 0.1411 91.4409 2345.289 19.7099 0.1337 91.4625 2353.179 19.7099 0.1337 90.7896 2353.187 23.8111 0.1328 90.7789 2361.009 23.8111 0.1328 98.0200 2361.016 24.9681 0.1285 97.9753 2361.660 24.9681 0.1285 94.2984 2361.660 58.0710 0.1401 94.2948 2366.887 58.0710 0.1401 80.8433 2366.893 80.6367 0.1548 80.8558 2374.759 80.6367 0.1548 63.0324 2374.767 159.4482 0.1680 63.0001 2382.580 159.4482 0.1680 81.9761 2382.587 150.2006 0.1592 82.0260 2390.350 150.2006 0.1592 49.0234 2390.358 147.3873 0.1607 49.0414 2398.074 147.3873 0.1607 66.3515 2398.082 198.8038 0.1425 66.3417 2405.749 198.8038 0.1425 79.2673 2405.757 261.4384 0.1503 79.2162 2413.385 261.4384 0.1503 44.0515 2413.392 383.6540 0.1799 43.9904 2420.979 383.6540 0.1799 69.4734 2420.987 89.1650 0.1239 69.5303 2421.380 89.1650 0.1239 73.3943 2421.380 45.5443 0.1278 73.3982 2425.361 45.5443 0.1278 120.4646 2425.365 9.5750 0.1079 120.4943 2433.004 9.5750 0.1079 115.9249 2433.011 22.2311 0.1111 115.9065 2440.719 22.2311 0.1111 112.1906 2440.727 11.6592 0.1126 112.1831 2448.403 11.6592 0.1126 113.2050 2448.411 13.0283 0.1096 113.1893 2456.067 13.0283 0.1096 113.2159 2456.075 26.1558 0.1200 113.1746 2463.703 26.1558 0.1200 106.1164'.split()  
  
res = np.array(sorted([float(data[i]) for i in range(len(data)) if i % 4 == 1]))  
  
  
def task\_1(penetration):  
 n\_bins = int(np.sqrt(len(penetration)))  
 bin\_edges = np.linspace(np.min(penetration), np.max(penetration), n\_bins)  
 our\_distribution\_n, \_ = np.histogram(penetration, bins=bin\_edges)  
 h\_n = np.array([i/len(penetration) for i in our\_distribution\_n])  
  
  
 #Оценка параметров  
 mu, std = stats.norm.fit(penetration)  
 n\_p\_i = stats.norm.pdf((bin\_edges[:-1]+bin\_edges[1:])/2, loc=mu, scale=std) \* len(penetration)  
  
 chi\_2 = sum((h\_n - n\_p\_i)\*\*2 / n\_p\_i)  
 df = len(penetration) - 1 # степени свободы  
 alpha = 0.95 # уровень значимости  
 c = stats.chi2.ppf(alpha, df)  
  
 print("Критерий:", chi\_2, "Квантиль:", c)  
  
 counts, bins, \_ = plt.hist(penetration, bins=15, color='skyblue', edgecolor='black') # подсчет частот значений  
 widths = np.diff(bins)  
 centers = (bins[:-1] + bins[1:]) / 2 # центры столбцов  
 p = stats.norm.pdf(centers, mu, std) \* len(penetration) \* widths[0] # добавление оценки плотности вероятности нормального распределения  
 plt.plot(centers, p, 'k', linewidth=2)  
 plt.xlabel('Значения')  
 plt.ylabel('Частота')  
 plt.title('Гистограмма и оценка нормального распределения')  
 plt.legend(['Нормальное распределение', 'Выборочные данные'])  
 plt.show()  
  
  
def task\_2(data):  
 mu, std = stats.norm.fit(data)  
 cdf = lambda x: stats.norm.cdf(x, mu, std)  
  
 n = len(data)  
 ecdf = lambda x: np.sum(np.array(data) <= x) / n  
  
 x = np.sort(data)  
 ecdf\_values = [ecdf(xi) for xi in x]  
 cdf\_values = [cdf(xi) for xi in x]  
  
 diff = np.max(np.abs(np.array(ecdf\_values) - np.array(cdf\_values)))  
 Dn = np.max(diff) # нахождение супремума этой разности  
 print('Статистика критерия Колмогорова:', Dn)  
 alpha = stats.ksone.ppf(0.95, n)  
 print('Квантиль:', alpha)  
  
  
def task\_3():  
 res = []  
 import random  
 for i in range(100):  
 data = np.random.normal(random.randint(1, 50), random.randint(1, 50), size=100)  
  
 mu, std = stats.norm.fit(data)  
 cdf = lambda x: stats.norm.cdf(x, mu, std)  
  
 n = len(data)  
 ecdf = lambda x: np.sum(np.array(data) <= x) / n  
  
 x = np.sort(data)  
 ecdf\_values = [ecdf(xi) for xi in x]  
 cdf\_values = [cdf(xi) for xi in x]  
  
 diff = np.max(np.abs(np.array(ecdf\_values) - np.array(cdf\_values)))  
 Dn = np.max(diff)  
 res.append(math.sqrt(n)\*Dn)  
  
 # создание столбчатой диаграммы  
 n\_bins = int(np.sqrt(len(res)))  
 bin\_edges = np.linspace(np.min(res), np.max(res), n\_bins)  
 our\_distribution\_n, \_ = np.histogram(res, bins=bin\_edges)  
 h\_n = np.array([i / len(res) for i in our\_distribution\_n])  
 plt.bar(range(len(h\_n)), h\_n)  
 plt.show()  
  
# task\_1(penetration)  
# data = np.random.normal(4, 2, size=100)  
# task\_1(data)  
# task\_2(penetration)  
# data = np.random.normal(4, 2, size=100)  
# task\_2(data)  
task\_3()