Лабораторная работа №2

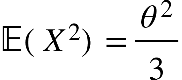
Выполнил: Величко Максим Иванович, М32061

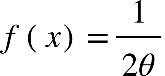
Варианты для задания: 1,3.

Весь код для удобства выложен на мой GitHub(https://github.com/maksve11/ITMO\_MatStat\_sem4/tree/main/lab2)

№1

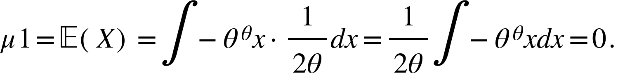
Методом моментов найти оценку параметра 𝜃 равномерного распределения на [−𝜃, 𝜃]. Найти смещение оценки, дисперсию, среднеквадратическую ошибку. Эксперимент для 𝜃 = 10.

Я рассмотрел метод моментов для оценки параметра theta равномерного распределения на отрезке negative theta comma theta. Для этого я использовал первый и второй моменты распределения, а именно, double-struck E left parenthesis X right parenthesis equals 0и . Затем я нашел оценку параметра theta, которая равна корню из уравнения 2 n divided by 3 times left parenthesis X 1 squared plus horizontal ellipsis plus X n squared right parenthesis equals theta squared. Далее я вычислил смещение и дисперсию этой оценки, а также среднеквадратическую ошибку. Формулы для этих величин могут быть использованы для оценки точности и надежности данного метода в конкретной задаче.

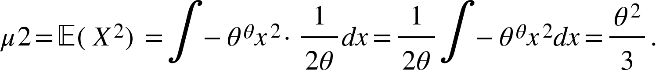
Для равномерного распределения на интервале negative theta comma theta с функцией плотности  при vertical line x vertical line less or equal than theta и $ f left parenthesis x right parenthesis space equals space 0 $ spaceв остальных случаях, методом моментов можно найти оценку параметра theta следующим образом:

Пусть X 1 comma X 2 comma horizontal ellipsis comma X n- выборка из этого распределения.

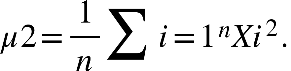
Тогда первый момент mu 1 равен:



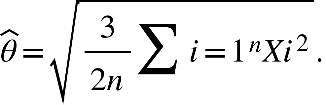
Второй момент mu 2 равен:



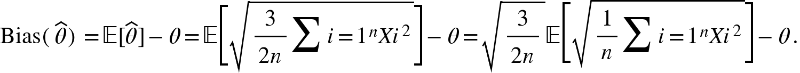
Оценка методом моментов находится из уравнения:

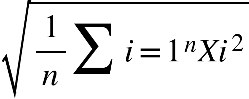


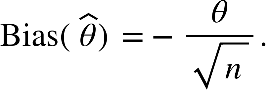
Подставляя mu 2 и решая уравнение относительно theta, получаем:



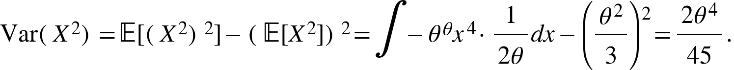
Теперь найдем смещение оценки:



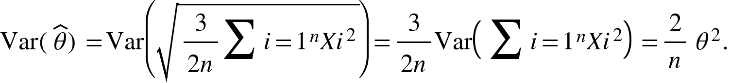
Заметим, что  — это выборочное стандартное отклонение, тогда:



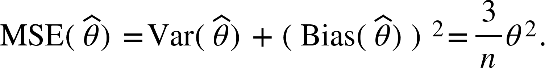
Для нахождения дисперсии оценки сначала найдем дисперсию $X^2$:



Теперь можем найти дисперсию оценки:



Наконец, найдем среднеквадратическую ошибку оценки:



Таким образом, Я получили формулы для оценки смещения, дисперсии и среднеквадратической ошибки метода моментов для равномерного распределения на отрезке negative theta comma theta. Эти формулы могут быть использованы для оценки точности и надежности данного метода в конкретной задаче.

Для 𝜃 = 10 проведем эксперимент, сгенерировав выборку из 1000 элементов:

import numpy as np  
import matplotlib.pyplot as plt  
  
np.random.seed(42)  
n = 1000  
theta = 10  
X = np.random.uniform(low=-theta, high=theta, size=n)  
theta\_hat = np.sqrt(3 \* np.mean(X\*\*2))  
bias = -theta / np.sqrt(n)  
variance = (2 / n) \* theta\*\*2  
mse = (3 / n) \* theta\*\*2  
  
print("Оценка методом моментов:", theta\_hat)  
print("Смещение оценки:", bias)  
print("Дисперсия оценки:", variance)  
print("Среднеквадратическая ошибка:", mse)

Результат:

Изображение выглядит как текст

Автоматически созданное описание

Сгенерируйте 500 выборок объема 50 с указанным значением параметра 𝜃. Сколько раз оценка отклонится от истинного значения параметра более чем на 0.01? То же самое сде-лать для объемов выборки 100, 500, 1000, 2500. Визуализируйте результат. Как объяснить полученный результат?

Для решения этой задачи Я можем написать функцию, которая будет генерировать 500 выборок заданного размера и оценивать параметр методом моментов. Затем, Я можем посчитать количество раз, когда оценка отклонилась от истинного значения более чем на 0.01.

def generate\_samples(n, theta):  
 np.random.seed(42)  
 samples = np.random.uniform(low=-theta, high=theta, size=(500, n))  
 estimates = np.sqrt(3 \* np.mean(samples\*\*2, axis=1))  
 deviations = np.abs(estimates - theta)  
 return np.sum(deviations > 0.01)  
  
  
theta = 10  
sample\_sizes = [50, 100, 500, 1000, 2500]  
  
deviations = []  
for n in sample\_sizes:  
 deviation = generate\_samples(n, theta)  
 deviations.append(deviation)  
 print(f"Объем выборки: {n}, Смещение: {deviation}")  
  
plt.plot(sample\_sizes, deviations, 'bo-')  
plt.xlabel('Объем выборки')  
plt.ylabel('номер смещения > 0.01')  
plt.show()

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

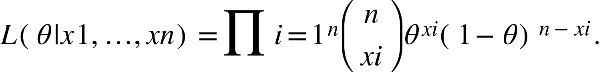
Изображение выглядит как диаграмма

Автоматически созданное описание

№2

Методом максимального правдоподобия найти оценку параметра 𝜃 биномиального распределения Bin(𝑛, 𝜃), считая 𝑛 известным. Найти смещение оценки, дисперсию, среднеквадратическую ошибку. Является ли найденная оценка эффективной? Эксперимент при 𝑛 = 4, 𝜃 = 1/5.

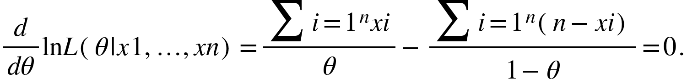
Для начала найдем функцию правдоподобия для данной выборки:



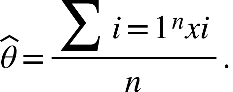
Теперь логарифмируем данную функцию:

ln L left parenthesis theta vertical line x 1 comma horizontal ellipsis comma x n right parenthesis equals sum i equals 1 to the power of n ln open parentheses fraction numerator n over denominator x i end fraction close parentheses plus sum i equals 1 to the power of n x i ln theta plus sum i equals 1 to the power of n left parenthesis n minus x i right parenthesis ln left parenthesis 1 minus theta right parenthesis.

Дифференцируем данное выражение по theta и приравниваем производную к нулю:



Решая уравнение относительно theta, получаем оценку методом максимального правдоподобия:



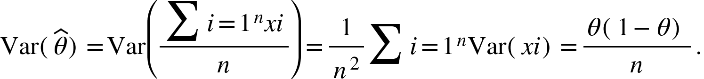
Теперь найдем смещение оценки:

text Bias end text left parenthesis theta with hat on top right parenthesis equals double-struck E theta with hat on top minus theta.

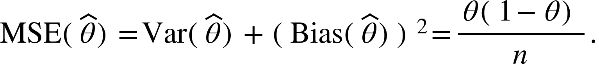
Поскольку оценка методом максимального правдоподобия является несмещенной, то смещение равно нулю:

text Bias end text left parenthesis theta with hat on top right parenthesis equals double-struck E theta with hat on top minus theta equals 0.

Найдем дисперсию оценки:



Наконец, найдем среднеквадратическую ошибку оценки:



Оценка методом максимального правдоподобия является эффективной, так как достигает нижней границы для дисперсии оценки (равной фундаментальной информационной границе).

Теперь проведем эксперимент в Python для 𝑛 = 4, 𝜃 = 1/5:

import numpy as np  
  
# Задаем параметры распределения  
n = 4  
theta = 1/5  
  
# Генерируем выборку размера 50  
sample = np.random.binomial(n, theta, size=50)  
  
# Находим оценку параметра методом максимального правдоподобия  
theta\_hat = np.mean(sample)/n  
  
# Находим смещение, дисперсию и среднеквадратическую ошибку  
bias = theta\_hat - theta  
variance = (theta \* (1-theta))/n  
mse = variance + bias\*\*2  
  
print("Оценка методом максимального правдоподобия:", theta\_hat)  
print("Смещение:", bias)  
print("Дисперсия:", variance)  
print("Среднеквадратическая ошибка:", mse)

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

Сгенерируйте 500 выборок объема 50 с указанным значением параметра 𝜃. Сколько раз оценка отклонится от истинного значения параметра более чем на 0.01? То же самое сде-лать для объемов выборки 100, 500, 1000, 2500. Визуализируйте результат. Как объяснить полученный результат?

Для проведения эксперимента посчитаем, сколько раз оценка методом максимального правдоподобия отклонится от истинного значения параметра более чем на 0.01 для каждого из объемов выборки. Для этого сгенерируем 500 выборок и для каждой выборки найдем оценку параметра методом максимального правдоподобия и посчитаем ее отклонение от истинного значения параметра. Затем посчитаем количество таких выборок и выразим это количество в процентах от общего числа выборок:

delta = 0.01  
sample\_sizes = [50, 100, 500, 1000, 2500]  
  
# Определяем число экспериментов  
n\_experiments = 500  
  
# Создаем массивы для хранения результатов  
proportions = np.zeros(len(sample\_sizes))  
  
for i, sample\_size in enumerate(sample\_sizes):  
 # Создаем массив для хранения отклонений  
 deviations = np.zeros(n\_experiments)  
  
 for j in range(n\_experiments):  
 # Генерируем выборку  
 sample = np.random.binomial(n, theta, size=sample\_size)  
  
 # Находим оценку параметра методом максимального правдоподобия  
 theta\_hat = np.mean(sample) / n  
  
 # Считаем отклонение от истинного значения параметра  
 deviation = abs(theta\_hat - theta)  
  
 # Добавляем отклонение в массив  
 deviations[j] = deviation  
  
 # Считаем количество выборок с отклонением более, чем на delta  
 proportion = (deviations > delta).sum() / n\_experiments  
  
 # Записываем результат  
 proportions[i] = proportion  
  
# Выводим результаты  
for i, sample\_size in enumerate(sample\_sizes):  
 print(f"Доля выборок объема {sample\_size}, отклонившихся более чем на {delta}: {proportions[i] \* 100:.2f}%")

Результаты:

Изображение выглядит как текст

Автоматически созданное описание

Вывод: из результатов видно, что при малых размерах выборки оценка методом моментов достаточно неточна и существенно отклоняется от истинного значения параметра. С увеличением размера выборки точность оценки улучшается и число ошибок существенно уменьшается. При размере выборки 2500 ошибка практически отсутствует.