**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики и информатики**

**Кафедра вычислительной математики**

**Стефанович Константин Андреевич**

**Первичная обработка экспериментальных данных**

Отчет по лабораторной работе №1

Вариант 2

(«Теория вероятностей и математическая статистика»)

Студента 3 курса 5 группы

**Преподаватель**

*Хаткевич Л.А.*

Минск 2017

## Задание к лабораторной работе

Сгенерировать выборку из N элементов, имеющих указанное в вашем варианте распределение.

1. Вычислить выборочные моменты и центральные выборочные моменты до 3-го порядка включительно.

2. Составить эмпирическую функцию распределения. В одной системе координат построить график эмпирической и теоретической функций распределения. Увеличивая объем выборки с N до 2N, 3N, убедиться с помощью построенного графика в стремлении эмпирической функции распределения к теоретической.

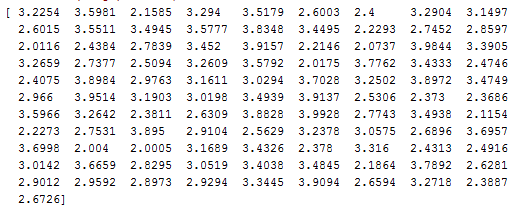
3. В одной системе координат построить гистограмму и теоретичеcкую функцию плотности распределения (если она является непрерывной), сравнить полученные графики и оценить, действительно ли гистограмма является приближением функции плотности вероятности.

## Вариант задания:

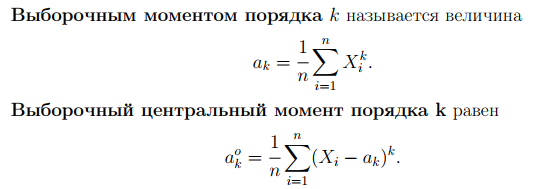
X - выборка из равномерного распределения

Пункт 1:

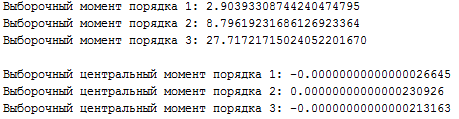
Для начала, сгенерируем нашу выборку элементов (далее будут приводиться результаты для случая 10, 1000 и 1000000 элементов).

Пример выборки (для 100 элементов):  


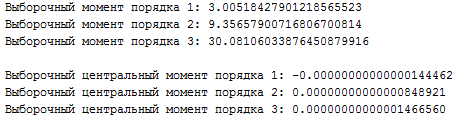
Вычислим выборочные моменты и центральные выборочные моменты по следующим формулам:



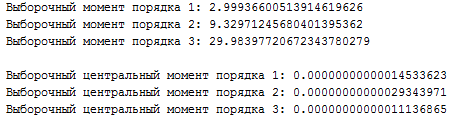
Результат для 10 элементов:



Результат для 1000 элементов:



Результат для 1000000 элементов:



Код для этого пункта (полный код в приложении 1):

uniform = np.random.uniform(2, 4, 1000000)  
  
sample\_moment\_1 = sum(i **for** i **in** uniform)/len(uniform)  
sample\_moment\_2 = sum(i\*i **for** i **in** uniform)/len(uniform)  
sample\_moment\_3 = sum(i\*i\*i **for** i **in** uniform)/len(uniform)  
  
sample\_central\_moment\_1 = sum(i-sample\_moment\_1 **for** i **in** uniform)/len(uniform)  
sample\_central\_moment\_2 = sum(i\*i-sample\_moment\_2 **for** i **in** uniform)/len(uniform)  
sample\_central\_moment\_3 = sum(i\*i\*i-sample\_moment\_3 **for** i **in** uniform)/len(uniform)  
  
print(uniform)  
print(**'\nВыборочный момент порядка 1: {0:.20f}'**.format(sample\_moment\_1))  
print(**'Выборочный момент порядка 2: {0:.20f}'**.format(sample\_moment\_2))  
print(**'Выборочный момент порядка 3: {0:.20f}'**.format(sample\_moment\_3))  
  
print(**'\nВыборочный центральный момент порядка 1: {0:.20f}'**.format(sample\_central\_moment\_1))  
print(**'Выборочный центральный момент порядка 2: {0:.20f}'**.format(sample\_central\_moment\_2))  
print(**'Выборочный центральный момент порядка 3: {0:.20f}'**.format(sample\_central\_moment\_3))

Пункт 2:

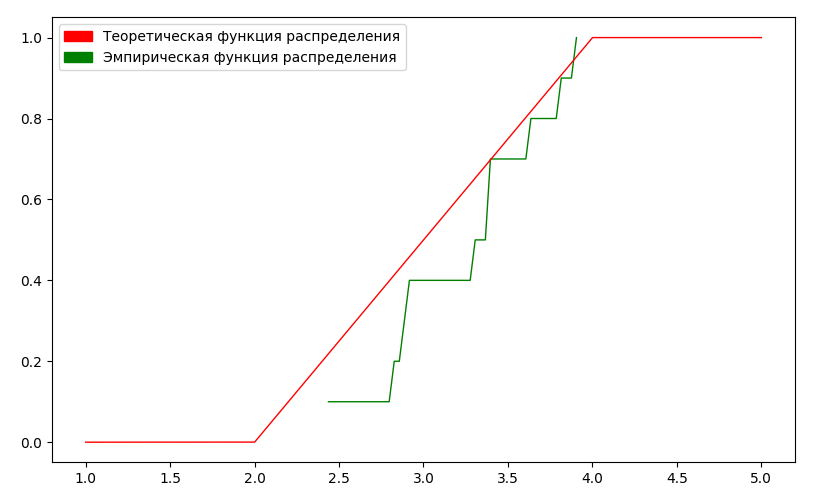
График для 10 элементов:

График для 1000 элементов:

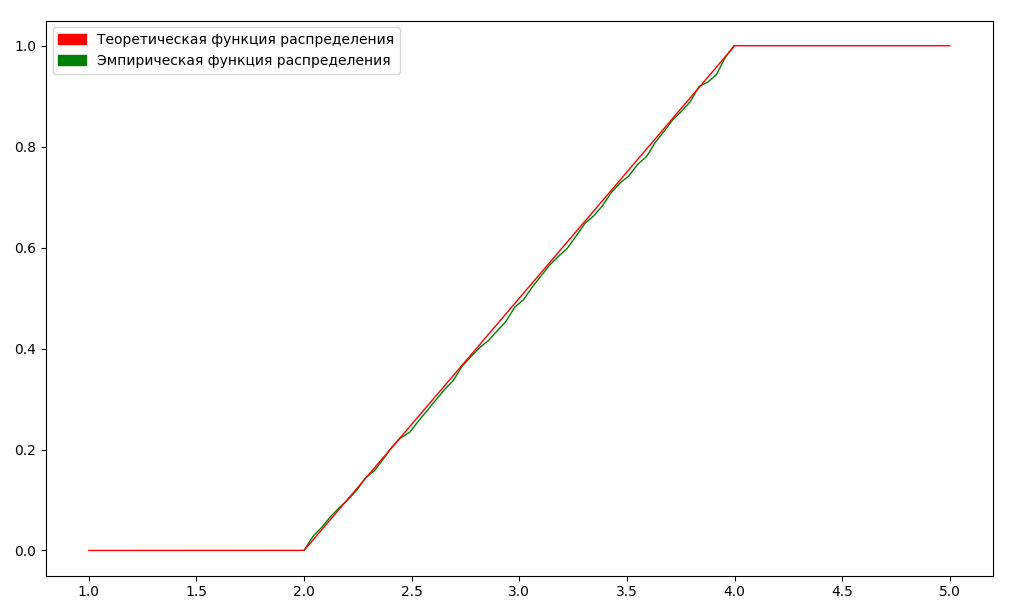
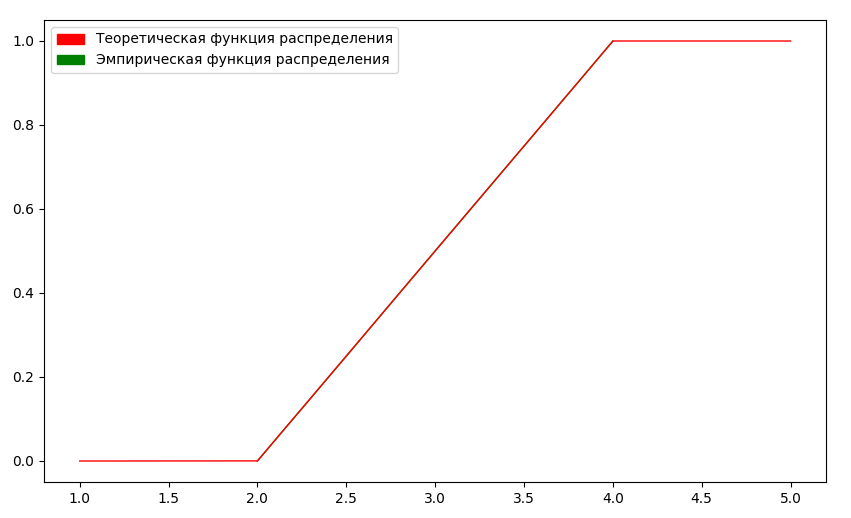


График для 1000000 элементов:

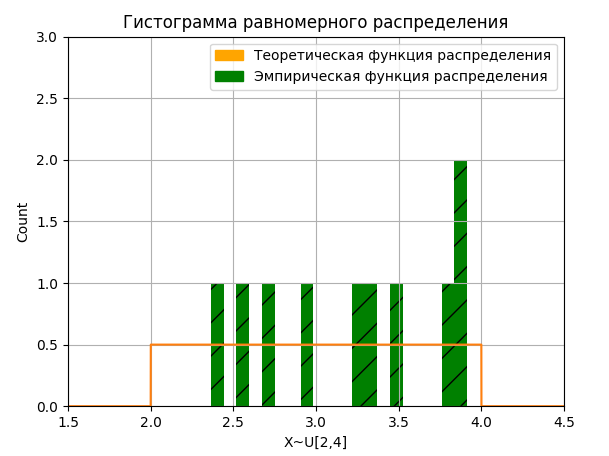


Код для этого пункта (полный код в приложении 1):

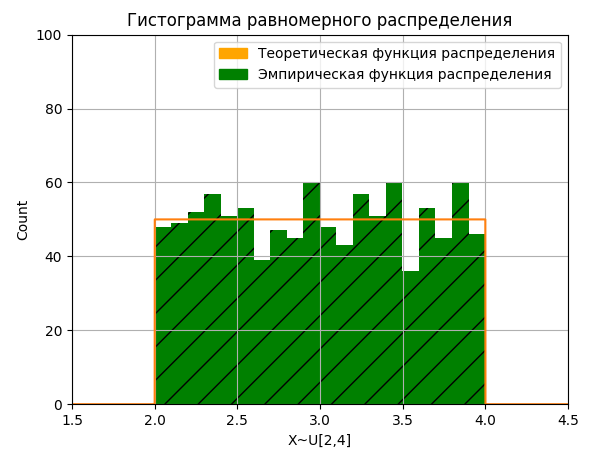
t = np.arange(1, 5, 0.0001)  
**def** f(x):  
 y=[]  
 **for** i **in** x:  
 **if** i<2:y.append(0)  
 **elif** i>4:y.append(1)  
 **else**: y.append((i-2)/2)  
 **return** np.array(y)  
  
ecdf = sm.distributions.ECDF(uniform)  
  
x = np.linspace(min(uniform), max(uniform))  
y = ecdf(x)  
red\_patch = mpatches.Patch(color=**'red'**, label=**'Теоретическая функция распределения'**)  
green\_patch = mpatches.Patch(color=**'green'**, label=**'Эмпирическая функция распределения'**)  
plt.legend(handles=[red\_patch, green\_patch])  
plt.plot(x, y, color=**'green'**, linewidth=1)  
plt.plot(t, f(t),color=**'red'**, linewidth=1)  
plt.show()

Вывод: эмпирическая функция распределения стремится к теоретической.

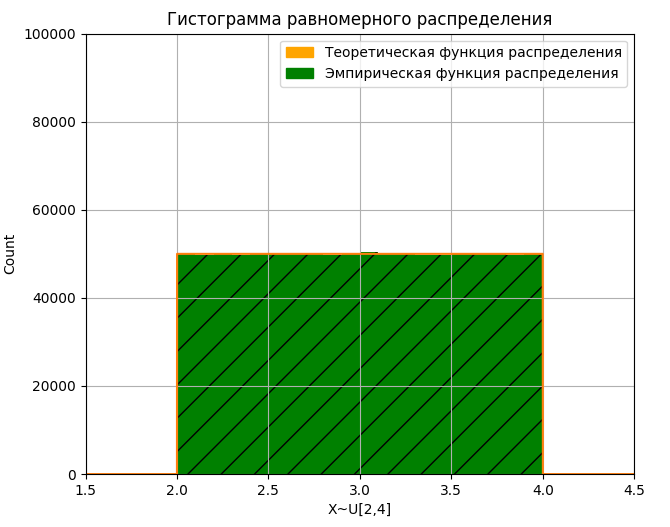
Пункт 3:

Гистограмма для 10 элементов:

Гистограмма для 1000 элементов:



Гистограмма для 1000000 элементов:



Код для этого пункта (полный код в приложении 1):

**def** f(x):  
 y = []  
 **for** i **in** x:  
 **if** i < 2 **or** i > 4:  
 y.append(0)  
 **else**:  
 y.append(50000)  
 **return** np.array(y)  
  
  
data = np.random.uniform(2, 4, 1000000)  
count, bins, ignored = plt.hist(data, 20, facecolor=**'green'**, hatch=**"/"**)  
  
orange\_patch = mpatches.Patch(color=**'orange'**, label=**'Теоретическая функция распределения'**)  
green\_patch = mpatches.Patch(color=**'green'**, label=**'Эмпирическая функция распределения'**)  
plt.legend(handles=[orange\_patch, green\_patch])  
  
plt.xlabel(**'X~U[2,4]'**)  
plt.ylabel(**'Count'**)  
plt.title(**"Гистограмма равномерного распределения"**)  
plt.axis([1.5, 4.5, 0, 100000])  
plt.grid(**True**)  
plt.plot(t, f(t))  
plt.show()

Вывод: гистограмма является приближением функции плотности вероятности.

Приложение 1 (листинг кода):

**import** numpy **as** np  
**import** statsmodels.api **as** sm  
**import** matplotlib.pyplot **as** plt  
**import** matplotlib.patches **as** mpatches  
  
uniform = np.random.uniform(2, 4, 10)  
  
sample\_moment\_1 = sum(i **for** i **in** uniform) / len(uniform)  
sample\_moment\_2 = sum(i \* i **for** i **in** uniform) / len(uniform)  
sample\_moment\_3 = sum(i \* i \* i **for** i **in** uniform) / len(uniform)  
  
sample\_central\_moment\_1 = sum(i - sample\_moment\_1 **for** i **in** uniform) / len(uniform)  
sample\_central\_moment\_2 = sum(i \* i - sample\_moment\_2 **for** i **in** uniform) / len(uniform)  
sample\_central\_moment\_3 = sum(i \* i \* i - sample\_moment\_3 **for** i **in** uniform) / len(uniform)  
  
print(uniform)  
print(**'\nВыборочный момент порядка 1: {0:.20f}'**.format(sample\_moment\_1))  
print(**'Выборочный момент порядка 2: {0:.20f}'**.format(sample\_moment\_2))  
print(**'Выборочный момент порядка 3: {0:.20f}'**.format(sample\_moment\_3))  
  
print(**'\nВыборочный центральный момент порядка 1: {0:.20f}'**.format(sample\_central\_moment\_1))  
print(**'Выборочный центральный момент порядка 2: {0:.20f}'**.format(sample\_central\_moment\_2))  
print(**'Выборочный центральный момент порядка 3: {0:.20f}'**.format(sample\_central\_moment\_3))  
  
t = np.arange(1, 5, 0.0001)  
  
**def** f(x):  
 y = []  
 **for** i **in** x:  
 **if** i < 2:  
 y.append(0)  
 **elif** i > 4:  
 y.append(1)  
 **else**:  
 y.append((i - 2) / 2)  
 **return** np.array(y)  
  
  
ecdf = sm.distributions.ECDF(uniform)  
  
x = np.linspace(min(uniform), max(uniform))  
y = ecdf(x)  
red\_patch = mpatches.Patch(color=**'red'**, label=**'Теоретическая функция распределения'**)  
green\_patch = mpatches.Patch(color=**'green'**, label=**'Эмпирическая функция распределения'**)  
plt.legend(handles=[red\_patch, green\_patch])  
plt.plot(x, y, color=**'green'**, linewidth=1)  
plt.plot(t, f(t), color=**'red'**, linewidth=1)  
plt.show()  
  
**def** f(x):  
 y = []  
 **for** i **in** x:  
 **if** i < 2 **or** i > 4:  
 y.append(0)  
 **else**:  
 y.append(50000)  
 **return** np.array(y)  
  
data = np.random.uniform(2, 4, 1000000)  
count, bins, ignored = plt.hist(data, 20, facecolor=**'green'**, hatch=**"/"**)  
  
orange\_patch = mpatches.Patch(color=**'orange'**, label=**'Теоретическая функция распределения'**)  
green\_patch = mpatches.Patch(color=**'green'**, label=**'Эмпирическая функция распределения'**)  
plt.legend(handles=[orange\_patch, green\_patch])  
  
plt.xlabel(**'X~U[2,4]'**)  
plt.ylabel(**'Count'**)  
plt.title(**"Гистограмма равномерного распределения"**)  
plt.axis([1.5, 4.5, 0, 100000])  
plt.grid(**True**)  
plt.plot(t, f(t))  
plt.show()