**ФГБОУ ВО**

**Национальный исследовательский университет**

**«МЭИ»**

**Лабораторная работа №2**

по курсу

«**Методы и средства защиты информации**»

**Разложение в ряд Фурье. Код проверки на чётность**

**Выполнил:**

студент группы А-07-18

Востриков Р.В.

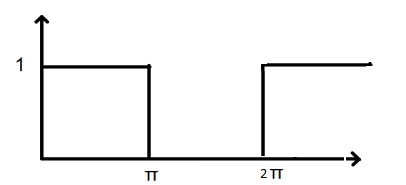
Вариант 8

**Москва, 2020**

1. **Постановка задачи**

**Задача №1**

Выполнить преобразование Фурье для функции



Период функции равен T = 2\*pi



1. Проверить формулы.

2. Составить программу, которая вычисляет разложение функции f(x) для различных m, определить максимальную разность между значениями функции f(x) и c помощью разложения Фурье.

3. Построить график при различных m, сравнить визуально схожесть графиков исходной функции и разложения Фурье.

**Задача №2**

Реализовать код проверки на чётность.

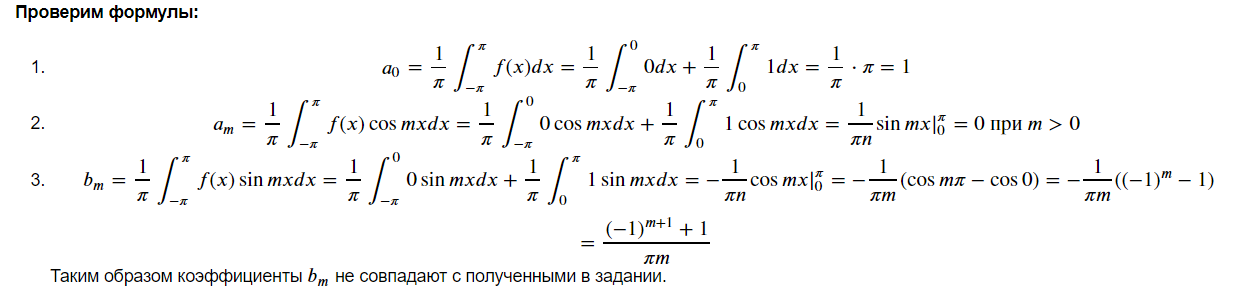
Пусть дано N-разрядное двоичное слово (или массив из N двоичных цифр 0 или 1). Проверочный (N+1) разряд вычисляется по формуле:

(A[0]+A[1]+...A[N-1]+ A[N] )% 2 = 0

A[N+1] выбирается таким образом, чтобы общая сумма равнялась нулю: если в N-разрядном двоичном слове чётное количество единиц, то он равен нулю, если нечётное, то единице.

1. **Описание результатов**

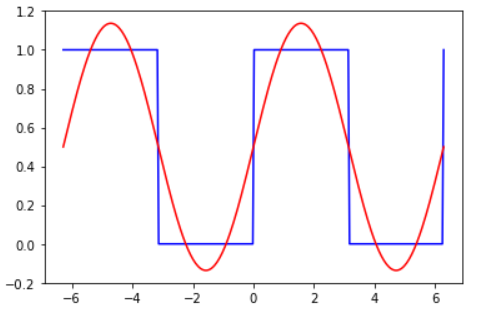
**Проверка формул в задаче №1**



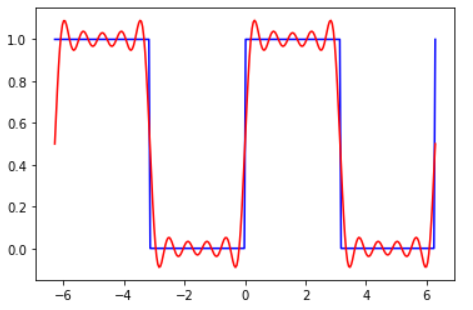
**Графики при различных m**

Синим изображена исходная функция, красным – ее разложение в ряд Фурье.

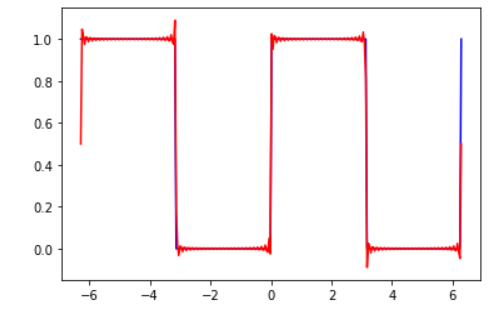
m = 1:



m = 10:



m = 100:



Как можно заметить, при увеличении значения m график суммы ряда все больше походит на график разложенной функции. Можно предположить, что при стремлении m к бесконечности график суммы разложения будет совпадать с графиком раскладываемой функции.

1. **Код программы**

**Задача №1**

def ft\_function(m, x):

result = 0.5 # <=> a0 = 1

for i in range(1,m+1):

result += (math.pow(-1,i+1)+1)/(math.pi\*i)\*math.sin(i\*x)

return result

m = 1000

points = 300 # количество точек для построения графика

# нахождение значений исходной функции и ее разложением в ряд Фурье на интервале [-2π,2π)

t = list(np.linspace(-2\*np.pi, 2\*np.pi, points))

y1 = list(map(original\_function, t))

y2 = list(map(lambda x: ft\_function(m,x), t))

plt.plot(t, y1, 'b')

plt.plot(t, y2, 'r')

plt.show()

# вычисление максимальной абсолютной погрешности

delta = []

for p in zip(y1[50:-50],y2[50:-50]): #отбрасываем несколько крайних точек для более достоверного результата

delta.append(abs(p[0]-p[1]))

print(max(delta))

**Задача №2**

# вычисление значения бита проверки на четность

def compute\_even\_bit(msg):

sum = 0

for digit in msg:

sum += int(digit)

return '0' if sum % 2 == 0 else '1'

# проверка на четность

# true если четность сохраняется <=> сообщение не содержит ошибок

# false в противном случае

def check\_even(msg):

expected = msg[-1]

actual = compute\_even\_bit(msg[0:-1])

return expected == actual

# искажение сообщения с вероятностью p для каждого бита

# искажение происходит в предположении, что ошибка не может возникнуть в бите проверки на четность

def add\_errors(msg, p):

errors = 0

new\_msg = ''

for digit in msg[0:-1]:

if random.random() <= p:

new\_msg += ('1' if digit == '0' else '0')

errors+=1

else:

new\_msg += digit

new\_msg += msg[-1]

return (new\_msg, errors)

1. **Выводы**

Было проведено разложение в ряд Фурье прямоугольного сигнала, была разработана программа для нахождения значения функции и наибольшего отклонения между суммой ряда и её исходным графиком.

Была разработана программа, позволяющая проверять наличие ошибок при передаче сообщения, состоящего из двоичного кода, путем реализации кода проверки на чётность. С использованием данного метода ошибки могут быть обнаружены, только если их количество нечётно.