МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Институт цифровых технологий, электроники и физики (ИЦТЭФ)

Кафедра вычислительной техники и электроники (ВТиЭ)

Лабораторная работа № 1

**Дискретизация, эффект "маскировки" частот и восстановление дискретного сигнала**

Выполнил студент 595 гр.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лаптев А.В

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Уланов П.Н.

Лабораторная работа защищена

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

Барнаул 2020

## Задание 1

Создать на основе непрерывного сигнала  с частотой  дискретный сигнал  с интервалом дискретизации *T*. Для этого нужно вычислить значения сигнала в определенные моменты времени (*T, 2T, 3T, …*). Определить частоту сигнала, который "маскируется" под данный сигнал при таком интервале дискретизации. Построить графики обоих сигналов и сравнить их.

## Решение

На вход программы подается сигнал . Частота поступающего сигнала Частота дискретизации следовательно, период дискретизации T=0.0032с. При выборе частоты дискретизации мы должны учесть, что половина частоты дискретизации должна быть больше частоты сигнала. В нашем случае данное условие выполняется, т.к., следовательно, частота маскирующегося сигнала .

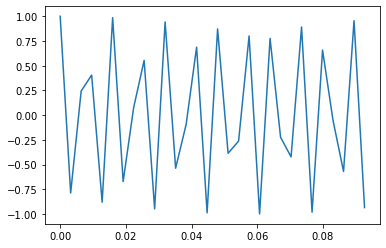


Рис.1 Косинусоида с частотой сигнала

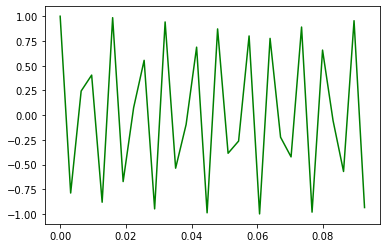


Рис. 2 Косинусоида с частотой маскирующего сигнала

Видим, что графики сигналов одинаковы.

# Задание 2

Уменьшить интервал дискретизации так, чтобы половина частоты дискретизации была больше, чем частоты обоих сигналов. Построить графики сигналов с новым интервалом дискретизации.

## Решение

Для того, чтобы исходный и маскирующийся сигналы были отличны друг от друга после оцифровки, достаточно увеличить частоту дискретизации так, чтобы её половина была больше частот входных косинусоид. Увеличиваем частоту дискретизации в 2 раза, теперь и частота не будет маскирующейся.

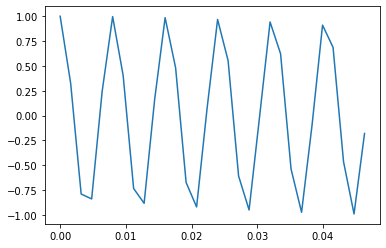


Рис. 3 Косинусоида с частотой сигнала и частотой дискретизации

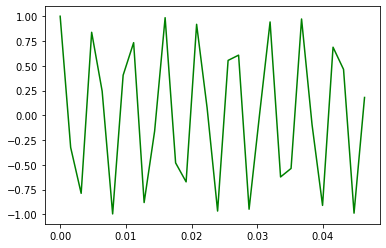


Рис. 4 Косинусоида с частотой сигнала и частотой дискретизации

Теперь графики исходных сигналов отличаются, и частота не будет маскирующейся.

# Задание 3

Теорема Котельникова утверждает, что если в спектре сигнала не содержится частот больших, чем половина частоты дискретизации, то исходный сигнал может быть однозначно восстановлен по своим дискретным отчетам следующим образом:

 (3.1).

Где *T* – интервал (период) дискретизации, - частота дискретизации.

Восстановить свой сигнал с помощью теоремы Котельникова, используя разное количество отсчетов (например, 10 и 50), постройте графики восстановленного сигнала. Обратить внимание, что прямое вычисление функции Котельникова  в некоторых точках (*t=nT*) приводит к делению на ноль. На самом деле, так как аргумент синуса тоже равен нулю, значение всей дроби следует положить равным 1 (поскольку (первый замечательный предел)).

## Решение

Числовые данные , .Сначала дискретизируем сигнал (сигнал задан функцией в программе). Затем восстанавливаем сигнал с помощью теоремы Котельникова по формуле (3.1). В некоторых точках знаменатель обращается в ноль. В этом случае по первому замечательному пределу значение всей дроби принимаем за 1. Строим графики восстановленного и идеального сигналов в одних осях (рис. 5).

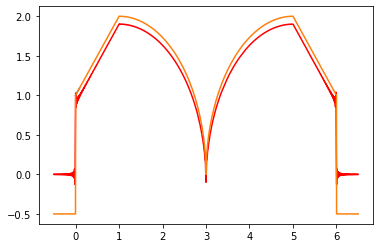


Рис. 5. Восстановленный сигнал по теореме Котельникова

## Вывод

В ходе выполнения лабораторной работы я научилcя дискретизировать сигнал, находить маскирующийся сигнал и восстанавливать сигнал по теореме Котельникова.

**Приложение к заданиям**

import math

import matplotlib.pyplot as plt

data = []

data1 = []

data2 = []

data3 = []

data4 = []

data5 = []

data6 = []

t0 = []

t1 = []

t2 = []

t3 = []

t4 = []

t5 = []

"""

Пункт 1.1

"""

f = float(input('Введите частоту сигнала: '))

fd = float(input('Введите частоту дискретизации: '))

PI = math.pi

Td = 1 / fd

for i in range(30):

t = i \* Td

t0.append(t)

g = math.cos(2\*PI\*f\*t)

data.append(g)

plt.plot(t0, data)

plt.show()

f1 = fd - f

PI = math.pi

Td = 1 / fd

for i in range(30):

t = i \* Td

t1.append(t)

g = math.cos(2\*PI\*f1\*t)

data1.append(g)

plt.plot(t1, data1, 'green')

plt.show()

"""

Пункт 1.2

"""

fd = 2 \* fd

PI = math.pi

Td = 1 / fd

for i in range(30):

t = i \* Td

t2.append(t)

g = math.cos(2\*PI\*f\*t)

data2.append(g)

plt.plot(t2, data2)

plt.show()

PI = math.pi

Td = 1 / fd

for i in range(30):

t = i \* Td

t3.append(t)

g = math.cos(2\*PI\*f1\*t)

data3.append(g)

plt.plot(t3, data3, 'green')

plt.show()

"""

Пункт 1.3

"""

PI = math.pi

Td = 1 / 100

Td2 = 1 / 1000

def f(t): # Исходная функция

if (t < 0):

return -0.5

elif (t <= 1):

return t + 1

elif (t <= 3):

return (4 - (t - 1) \*\* 2) \*\* 0.5

elif (t <= 5):

return (4 - (t - 5) \*\* 2) \*\* 0.5

elif (t <= 6):

return (-t + 5) + 2

else:

return -0.5

for i in range(10):

t = i \* Td

t5.append(t)

g = f(t)

data6.append(g)

plt.plot(t5,data6)

st = 1/1000

PI = math.pi

for j in range(-500,6500):# отрезок

G = 0

t2 = j \* st

t4.append(t2)

for i in range(600): # количество точек

arg = PI \* t2 / Td - i \* PI

if arg == 0:

dr = 1

else:

dr = math.sin(arg) / (arg)

t = i \* Td

g = f(t) -0.1

G += g \* dr

g = f(t2)

data5.append(g)

data4.append(G)

plt.plot(t4, data4,'red')

plt.plot(t4, data5)

plt.show()