Лабораторная работа №1

По курсу «Цифровая обработка сигналов»

Изучение спектров сигналов

Работу выполнил:

Студент группы Б14-501

Попцов П. А.

Работу проверил:

Ктитров С. В.

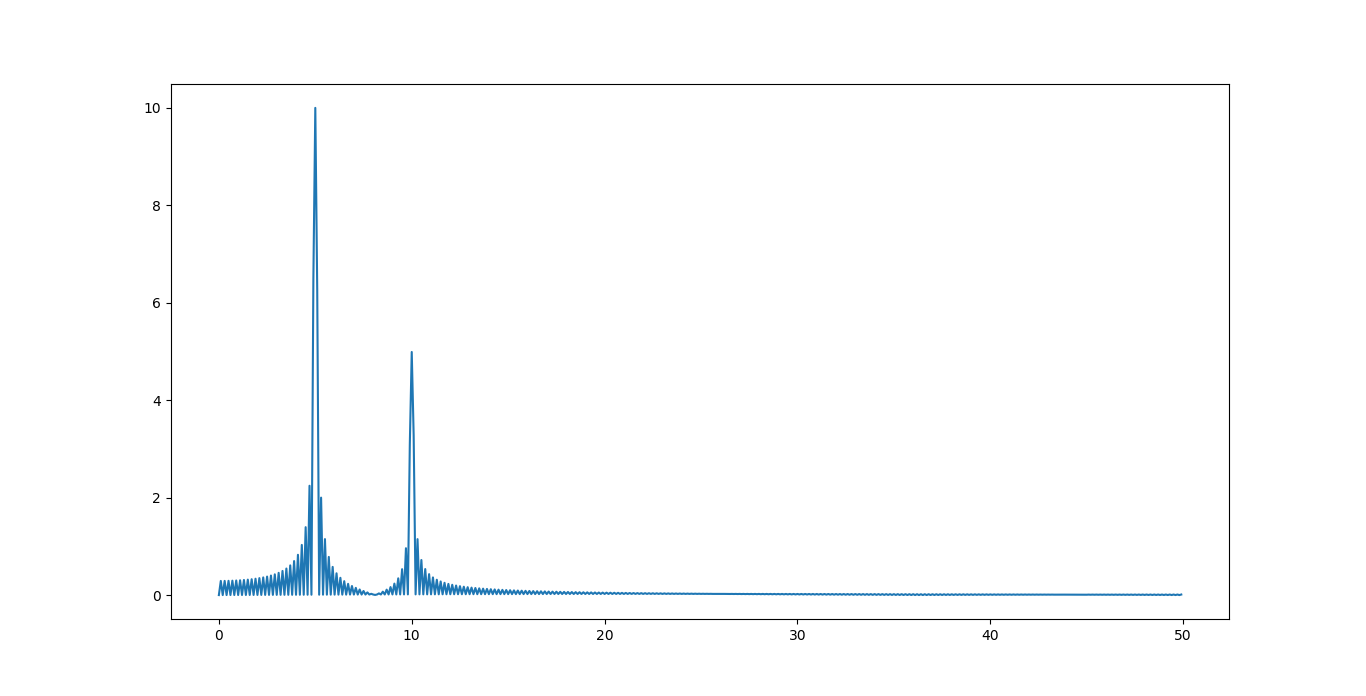
**1. Задание 1. Вычислить спектр непрерывного сигнала.**

Исходный сигнал: x1(t) = 10sin(2\*pi\*5\*t) + 5sin(2\*pi\*10\*t)

Время начала сигнала: t0 = 0.01

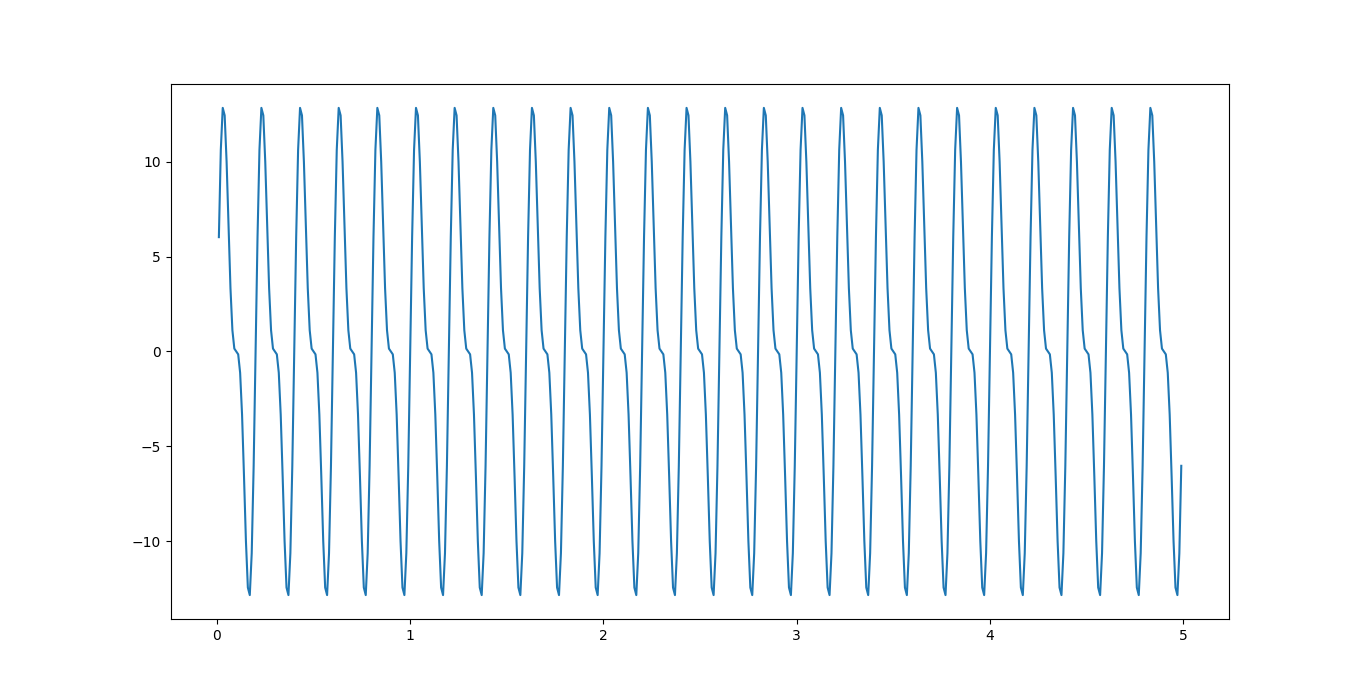
Время конца сигнала: t1 = 4.99

Период дискретизации: dt1 = 0.01

Рисунок 1. Спектр непрерывного сигнала x1(t).

Спектр получен с помощью реализованной функции *Sc,* которая вычисляет спектр непрерывного сигнала по определению с помощью метода интегрирования по частям.

На рисунке 1 приведён амплитудный спектр сигнала x1(t), по оси абсцисс отложены частоты в герцах, по оси ординат амплитуды, соответствующие своим частотам.

Рисунок 2: Дискретные отсчёты сигнала x1(t) для периода дискретизации dt1 = 0.01.

**2. Задание 2. Дискретизовать сигнал.**

Исходный сигнал: x1(t) = 10sin(2\*pi\*5\*t) + 5sin(2\*pi\*10\*t)

Время начала сигнала: t0 = 0.01

Время конца сигнала: t1 = 4.99

Период дискретизации: dt1 = 0.01

На рисунке 2 приведён график отсётов сигнала x1(t), полученный с помощью реализованной функции *SigcToSigd* и выполненный для 0.01 <= t <= 4.99 с периодом дискретизации dt1 = 0.01. По оси абсцисс отложено время t в секнудах, по оси ординат величина сигнала x1(t).

**3. Задание 3. Вычислить спектр дискретного сигнала по определению.**

Исходный сигнал: x1(t) = 10sin(2\*pi\*5\*t) + 5sin(2\*pi\*10\*t)

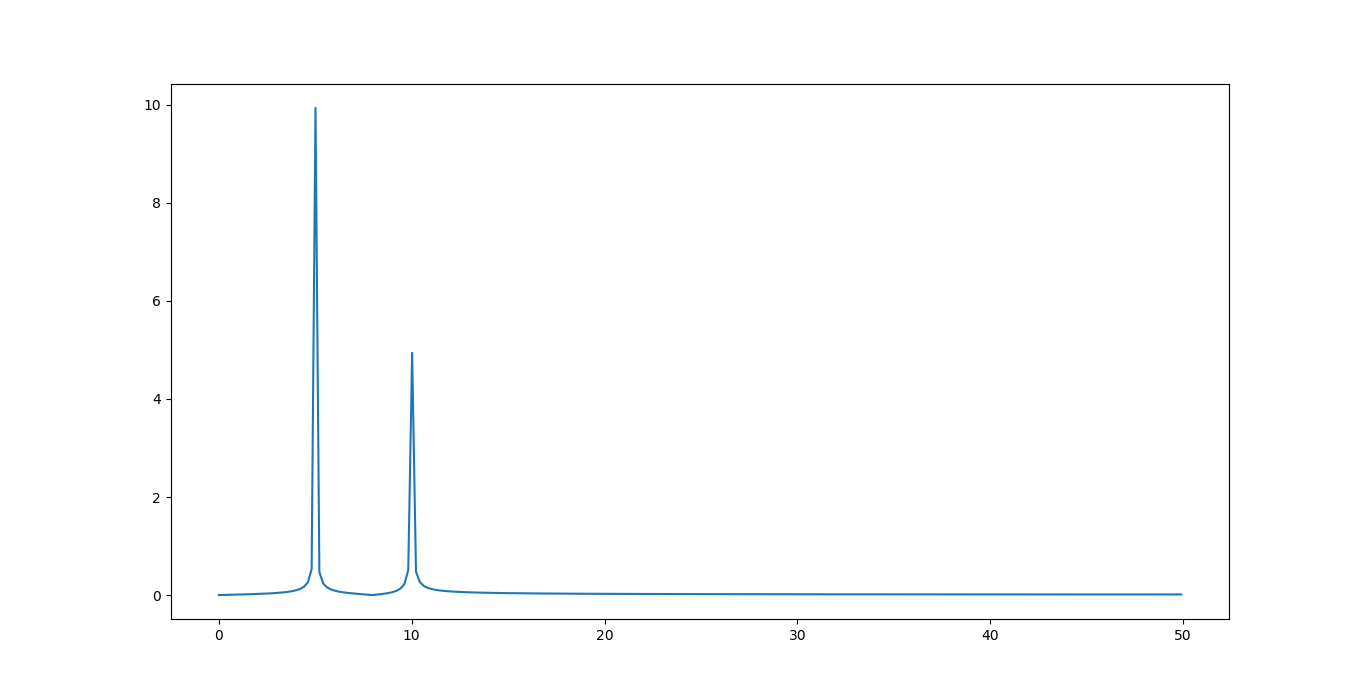
Время начала сигнала: t0 = 0.01

Время конца сигнала: t1 = 4.99

Период дискретизации: dt1 = 0.01

На рисунке 3 приведён амплитудный спектр дискретного сигнала x1(t), по оси абсцисс отложены частоты в герцах, по оси ординат амплитуды, соответствующие своим частотам. Спектр дискретного сигнала x1(t) получен с помощью реализованной функции *Sd.*

Рисунок 3: Спектр дискретного сигнала x1(t) вычисленный по определению спектра дискретного сигнала для периода дискретизации dt1 = 0.01.



Невооруженным глазом видно сходство непрерывного спектра на рисунке 1 и дискретного спектра на рисунке 3.

**4. Задание 4. Вычислить спектр дискретного сигнала через спектр непрерывного. Сравнить результат с заданием 3.**

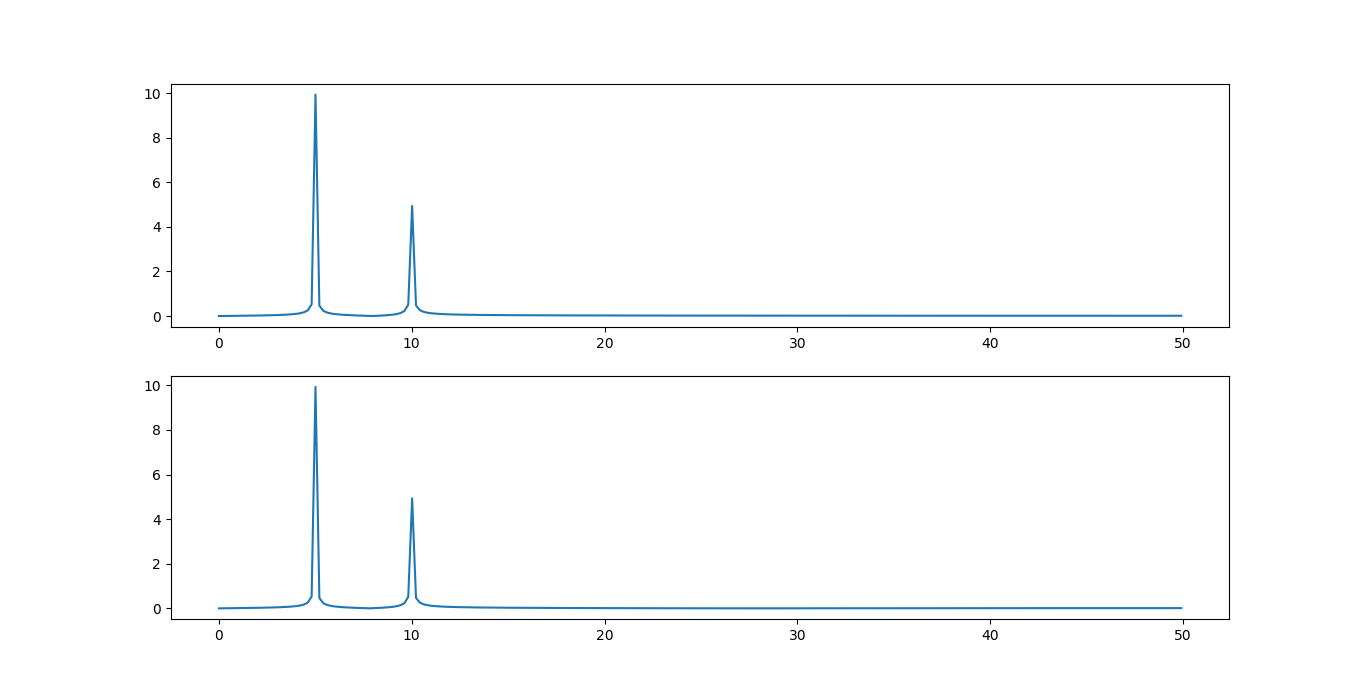
Исходный сигнал: x1(t) = 10sin(2\*pi\*5\*t) + 5sin(2\*pi\*10\*t)

Время начала сигнала: t0 = 0.01

Время конца сигнала: t1 = 4.99

Период дискретизации: dt1 = 0.01

На рисунке 4 представлено два амплитудных спектра. Верхний амплитудный спектра представляет собой дискретный амплитудный спектр, вычисленный по определению. Нижний — спектр, полученный из непрерывного спектра. На обоих графиках по оси абсцисс отложена частота в герцах, по оси ординат амплитуда сигнала. Из графиков видно, что амплитудный спектр дискретного сигнала, полученный двумя разными способами — совпадает.

Рисунок 4: Амплитудный спектр дискретного сигнала x1(t), полученный по определению спектра дискретного сигнала (сверху) и амплитудный спектр дискретного сигнала, полученного из спектра непрерывного сигнала x1(t) (снизу). Значения получены для периода дискретизации dt1 = 0.01.

**5. Задание 5. Выполнить задания 2-4 для другого шага дискретизации.**

Исходный сигнал: x1(t) = 10sin(2\*pi\*5\*t) + 5sin(2\*pi\*10\*t)

Время начала сигнала: t0 = 0.01

Время конца сигнала: t1 = 4.99

Период дискретизации: dt2 = 0.005 = dt1/2

На рисунках 5-7 приведены результаты выполнения заданий 2-4 соответственно для тех же данных, приведенных на рисунках 2-4, за исключением периода (частоты) дискретизации, которая была уменьшена в два раза: с 0.01 секунд до 0.005 секунд соответственно. Как видно, полученные графики на рисунках 5-7 не имеют принципиальных отличий от ранее полученных графиков на рисунках 2-4, так же новые графики не получились более детализированными, по сравнению с графиками для периода дискретизации dt1 = 0.01, хотя этот эффект является очевидным и естественным следствием уменьшения периода дискретизации (увеличения частоты дискретизации). Такой результат можно объяснить обратившись к виду сигнала x1(t), он содержит две гармоники с частотами 5 и 10 герц соответственно, но выбранный период дискретизации dt1 = 0.01 уже в пять раз меньше минимально возможного периода дискретизации из теоремы Котельникова (T = 1/(2\*f) => T1 = 1/20 = 0.05, T2 = 1/10 = 0.1), а второй выбранный период дискретизации dt2 = 0.005 уже в десять больше, чем величина T1 = 0.05, и в двадцать раз больше, чем величина T2 = 0.1.

**6. Задание 6. Задать спектр удовлетворяющий теореме Котельникова1, получить сигнал соответствующий данному спектру.**

Исходный спектр: S(w) = rect(w/(2\*a)), a = pi/2, где:

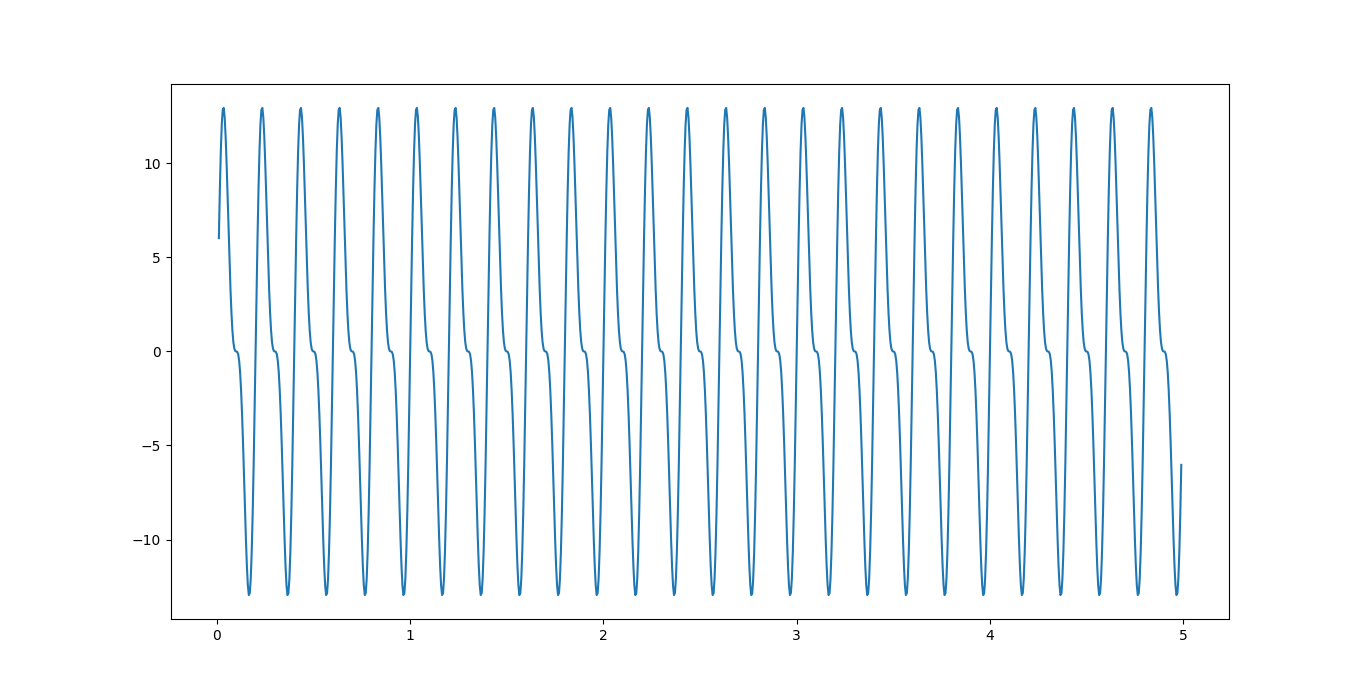
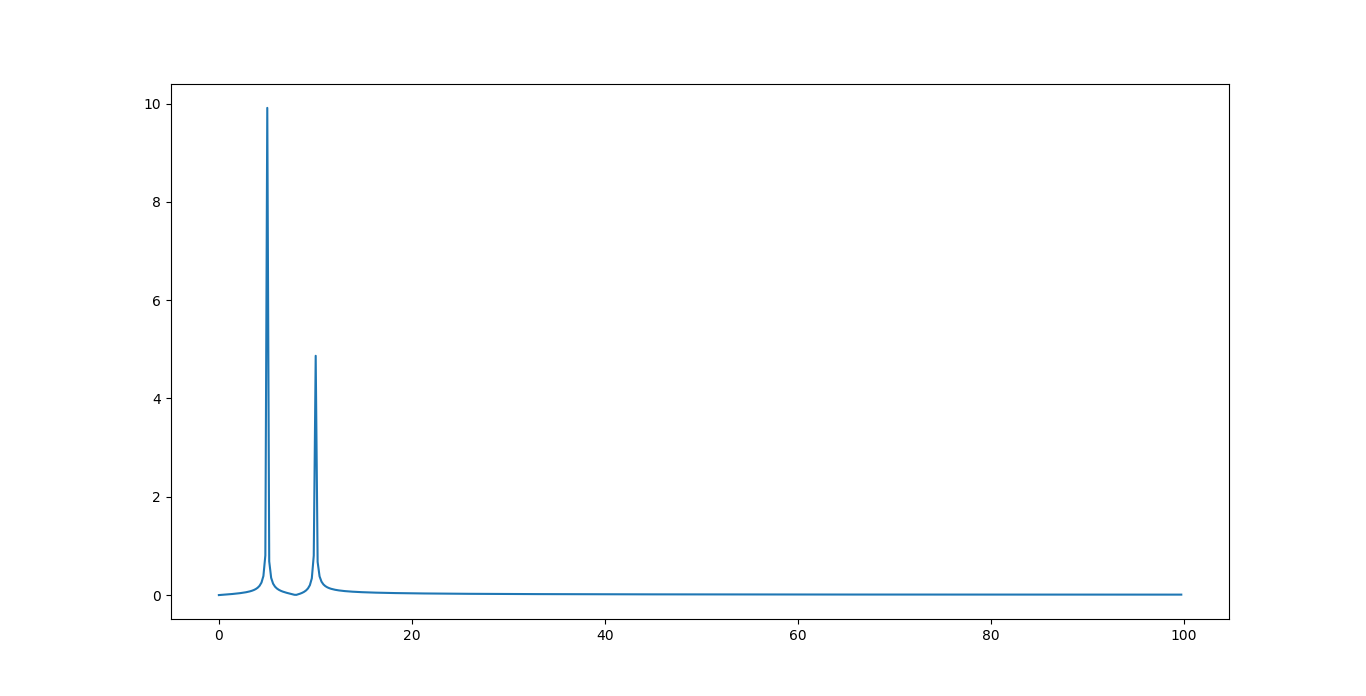
Рисунок 5: Дискретные отсчёты сигнала x1(t) для периода дискретизации dt2 = 0.005.

Рисунок 6: Спектр дискретного сигнала x1(t) вычисленный по определению спектра дискретного сигнала для периода дискретизации dt2 = 0.005.

| 0; |x| > 0.5

rect(x) = | 0.5; |x| = 0.5

| 1; |x| < 0.5

Начальная частота(w, рад/c): w0 = -pi

Конечная частота(w, рад/c): w1 = pi

Период дискретизации по циклической частоте: dw = pi/500

На рисунке 8 представлен график заданного спектра S(w), при это по оси абсцисс отложена циклическая частота w = 2\*pi\*f рад/c, а по оси ординат величина спектра S(w). Заданный спектр является действительной функцией действительной переменной.

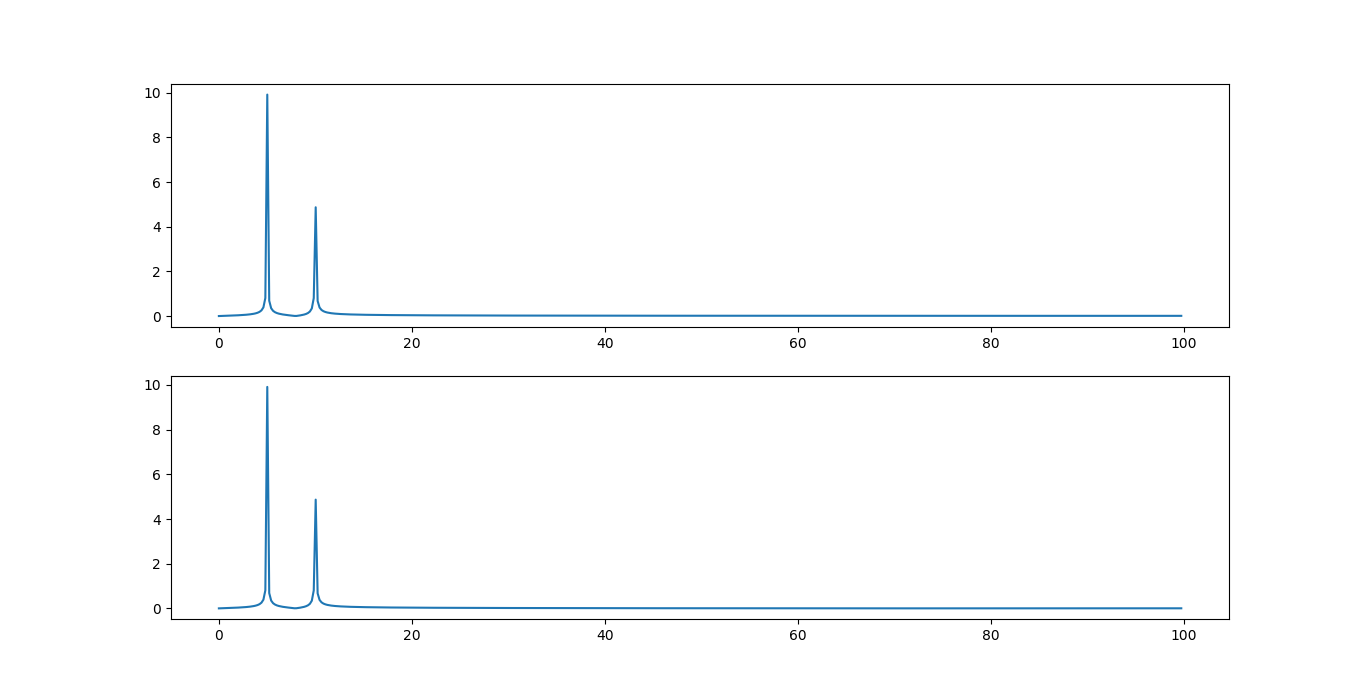
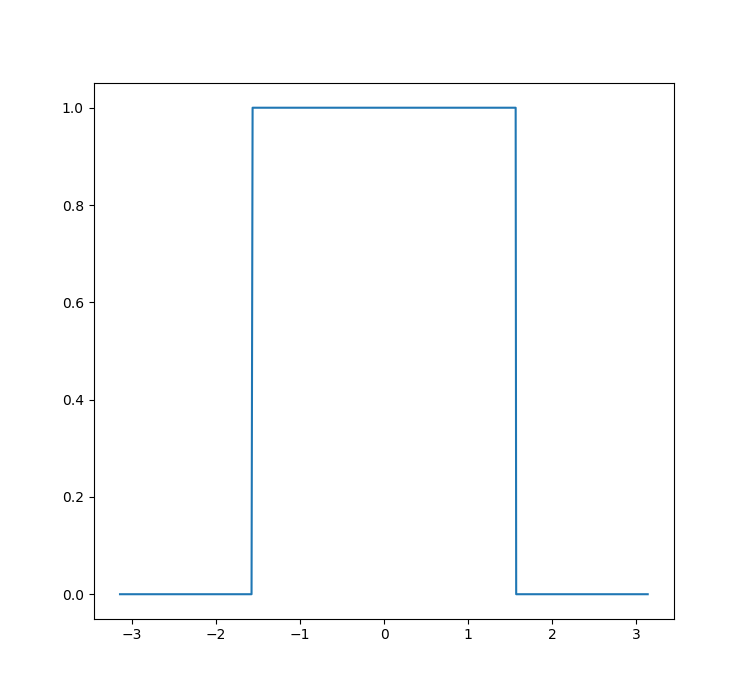
Рисунок 7: Амплитудный спектр дискретного сигнала x1(t), полученный по определению спектра дискретного сигнала (сверху) и амплитудный спектр дискретного сигнала, полученного из спектра непрерывного сигнала x1(t) (снизу). Значения получены для периода дискретизации dt2 = 0.005.

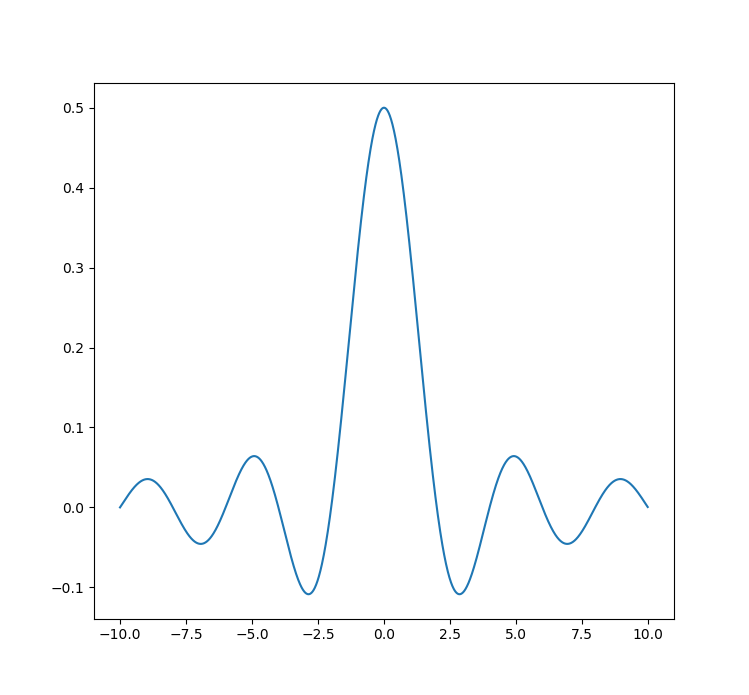
Рисунок 8: Заданный спектр S(w).

Для заданного спектра S(w) исходный сигнал y(t) можно найти аналитически, выполнив обратное преобразование Фурье для функции S(w). В результате получим искомый сигнал y(t), которому соответствует спектр S(w):

y(t) = a/pi\*sinc(a\*t), где a = pi/2 и sinc(x) = sin(pi\*x)/(pi\*x). Следовательно:

y(t) = 0.5\*sinc(0.5\*pi\*t).

На рисунке 9 приведен график сигнала y(t) на отрезке времени от -10 секунд до 10 секунд включительно.

Рисунок 9: График сигнала y(t) на отрезке времени -10 <= t <= 10.

**7. Задание 7. Дискретизовать сигнал, выбрав такты удовлетворяющими и неудовлетворяющими теореме Котельникова.**

Исходный сигнал: x2(t) = 10.0sin(2\*pi\*1\*t) + 10sin(2\*pi\*2\*t) + … + 10sin(2\*pi\*50\*t)

Первый эксперимент:

Начальное время: t1 = 0.1

Конечное время: t2 = 9.9

Период дискретизации: dt = 0.1

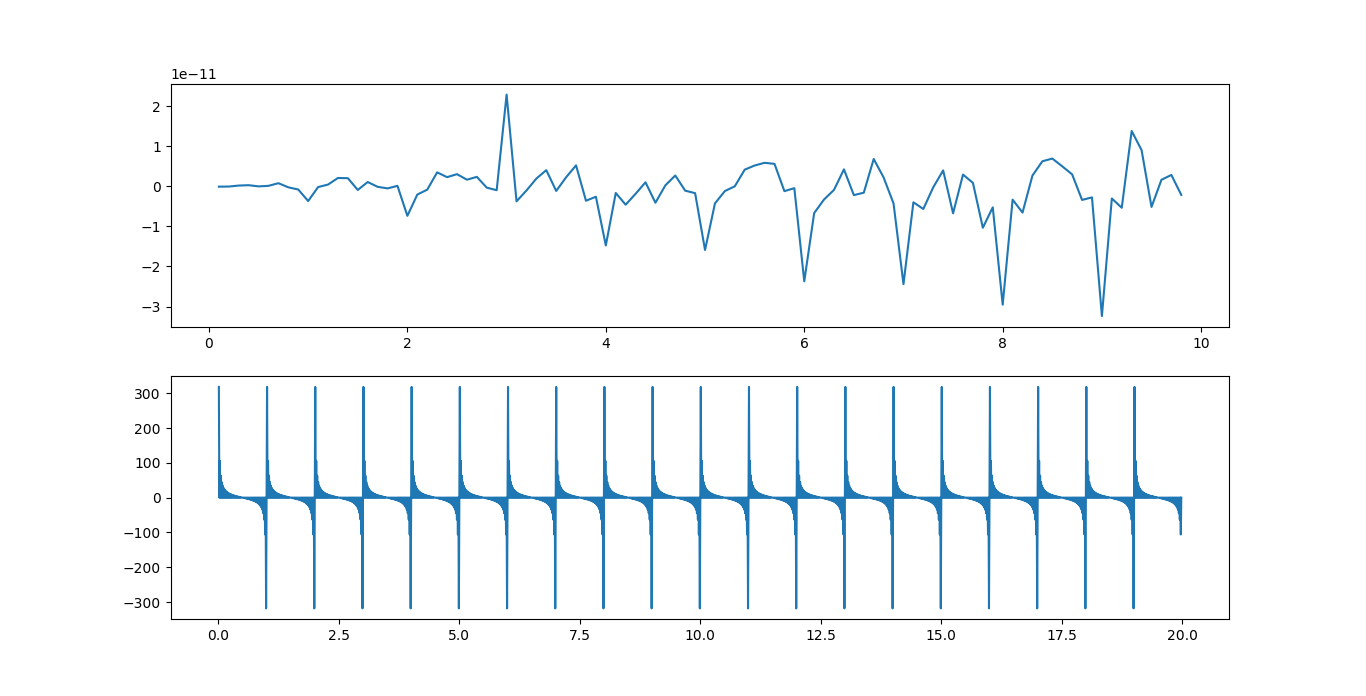
Второй эксперимент:

Начальное время: t1' = 0.01

Конечное время: t2' = 9.99

Период дискретизации: dt' = 0.01

По условию теоремы Котельникова: для восстановления сигнала период дискретизации T должен быть меньше или равен чем 1/(2\*50) = 0.01. Следовательно, в первом эксперименте исходные данные соответствуют случаю, когда условия теоремы Котельникова не удовлетворены, а во втором эксперименте данные соответствуют случаю, когда условия теоремы Котельникова удовлетворены. Для обоих экспериментов непрерывный сигнал x2(t) дискретизуется с помощью функции *SigcToSigd*.

Рисунок 10: Дискретизованный непрерывный сигнал x2(t) для данных первого эксперимента (сверху) и для данных второго эксперимента (снизу).

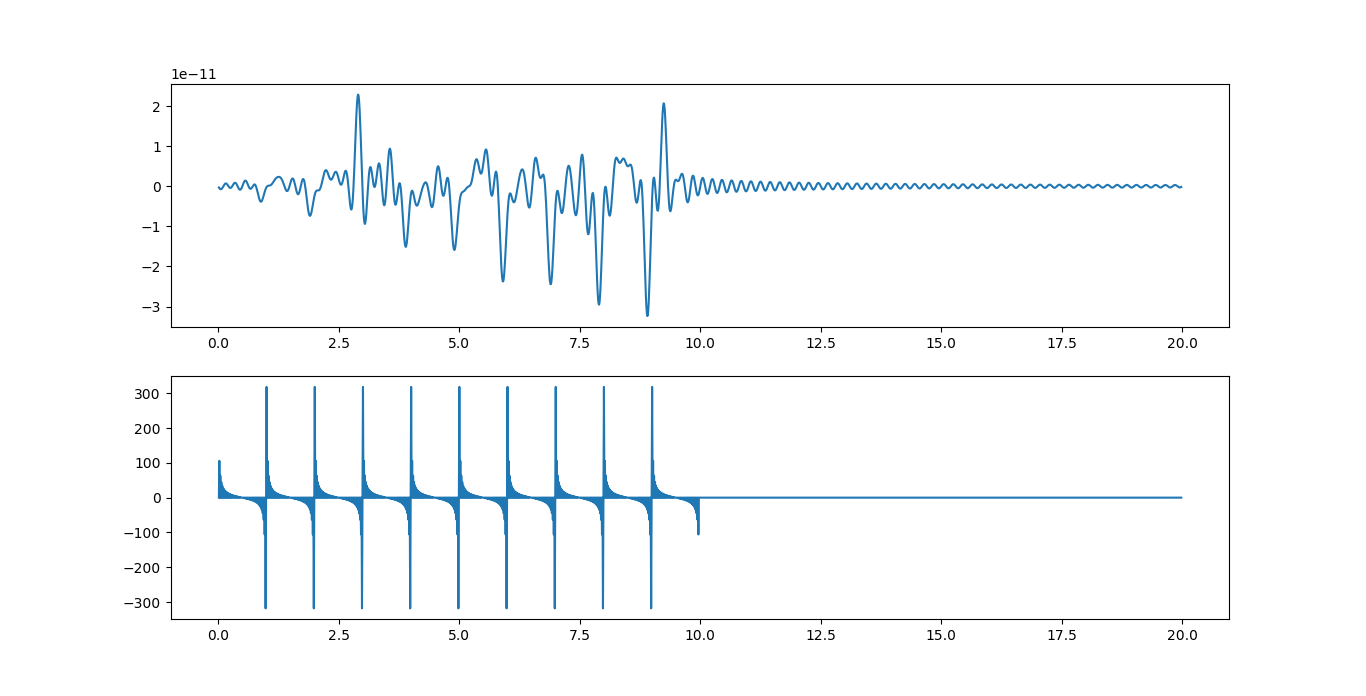
На рисунке 10 изображены два набора отсчётов непрерывного сигнала x2(t), которые соответствуют первому и второму эксперименту соответственно. По оси абсцисс на обоих подграфиках отложено время, а по оси ординат величина сигнала x2(t).

**8. Задание 8. Восстановить непрерывный сигнал по отсчётам полученным в задании 6.**

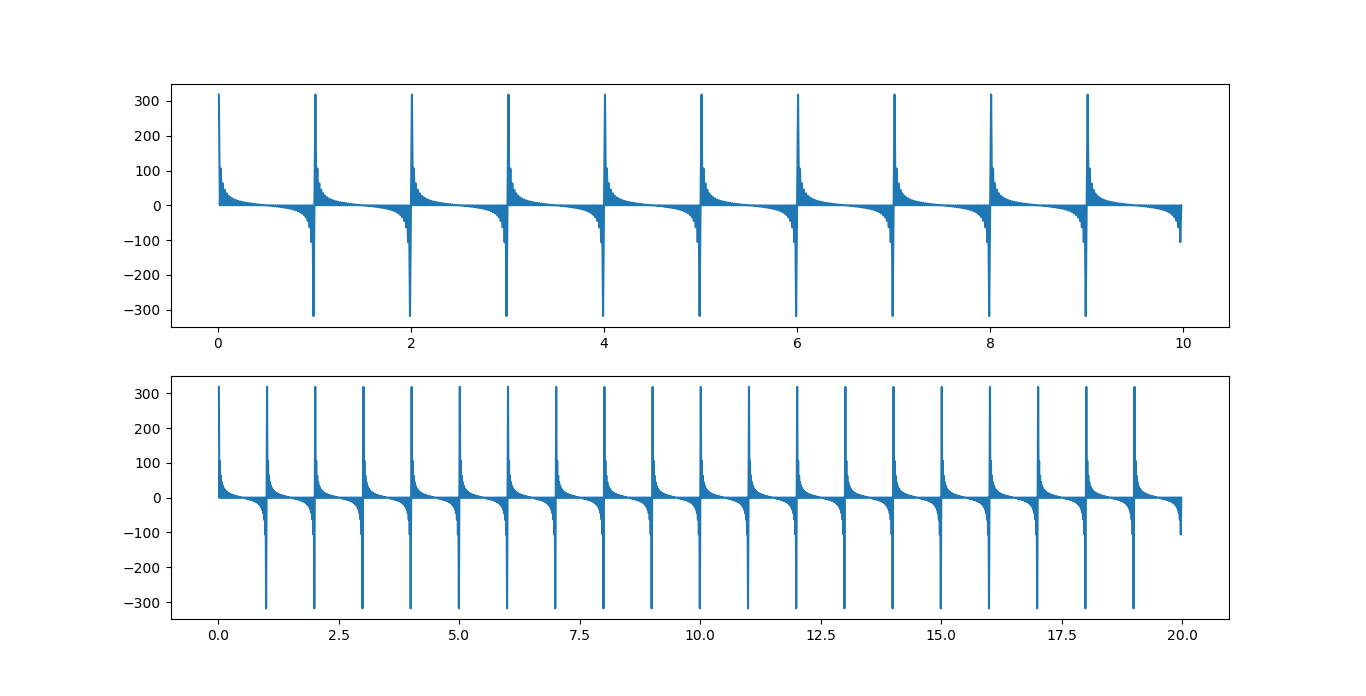
При помощи разработанной функции *TK,* отсчёты полученные в пункте 6 используются для восстановления непрерывного сигнала x2(t).

Восстановленный сигнал демонстрируется на отрезке времени от 0.1 секунды до 19.9 секунд в первом эксперименте и на отрезке времени от 0.01 секунды до 19.99 секунд во втором эксперименте. Графики восстановленных сигналов x2(t) приведены на рисунках 10 и 11. Рисунки 10 и 11 разделены на две половины — верхнюю и нижнюю. В верхней половине приводится результат восстановления непрерывного сигнала по его отсчётам, а в нижней половине приводится истинный сигнал x2(t). Как видно из рисунков 10 и 11 — для данных не удовлетворяющим условиям теоремы Котельникова восстановление непрерывного сигнала по отсчётам невозможна, восстановленная картина сильно искажена по сравнению с истинным ходом сигнала x2(t). Для условий удовлетворяющих теореме Котельникова ситуация прямо противоположная. Непрерывный сигнал был успешно восстановлен по дискретным отсчётам.

**9. Задание 9. Дискретизовать спектр, заданный в задании 6.**

Рисунок 11: Восстановленный сигнал x2(t) для первого эксперимента. Сверху восстановленный по теореме Котельникова сигнал, снизу оригинальный непрерывный сигнал для проверки.

Спектр S(w), заданный в задании 6 был успешно дискретизован с использованием разработанной функции S*igcToSigd,* в ходе выполнения задания 6. Полученный график спектра S(w) изображен на рисунке 8.

Рисунок 12: Восстановленный сигнал x2(t) для второго эксперимента. Сверху восстановленный по теореме Котельникова сигнал, снизу оригинальный непрерывный сигнал для проверки.

**10. Задание 10. Восстановить сигнал по дискретному спектру.**

**11. Исходный код программной реализации на языке python.**

**import** math

**import** cmath

**import** numpy **as** np

**import** os

**import** sys

**from** matplotlib **import** pyplot **as** plt

pi = math.pi

e = math.e

*#Функция Хевисайда*

**def** Theta(x):

**if** "array" **in** str(type(x)) **or** "list" **in** str(type(x)):

**return** np.array([ (1.0, 0.0)[value < 0] **for** value **in** x ])

**else**:

**if** x >= 0:

**return** 1.0

**else**:

**return** 0.0

*#тестовый сигнал, две гармоники и нормальный шум, чтобы не так скучно было*

**def** sig(t):

**return** 10\*np.sin(2\*pi\*5\*t) + 5\*np.sin(2\*pi\*10\*t) + np.random.normal(0.0, 1.0, (1, len(t))["array" **in** str(type(t\_a)) **or** "list" **in** str(type(t\_a))])

**def** sig2(t):

**return** 10\*np.sin(2\*pi\*5\*t) + 5\*np.sin(2\*pi\*10\*t)

*#начальное время наблюдения сигнала*

t\_0 = 0.0

*#конечное время наблюдения сигнала*

t\_f = 5.0

*#период дискретизации*

T = 0.01

t\_a = np.arange(t\_0, t\_f, T)

N = len(t\_a)

*#прямое ДПФ*

*#на вход отсчеты сигнала*

**def** FT(x):

**return** [ sum([ x[n]\*cmath.exp(-1j\*(2\*pi/len(x))\*k\*n) **for** n **in** range(0, len(x)) ]) **for** k **in** range(0, len(x)) ]

*#обратное ДПФ*

*#на вход отсчеты изображения Фурье сигнала*

**def** FT1(x):

**return** list(map(**lambda** t: t.conjugate()/len(x), FT(list(map(**lambda** t: t.conjugate(), x)))))

*#вычислить дискретный спектр, ну тут надо нормировку делать, я так в уире делал вроде*

*#вроде все правильно, но объяснить не могу почему так*

*#вернет амплитудный спектр, фазовый спектр и частоты в герцах для оси x спектра*

*#на вход отсчеты сигнала*

**def** Sd(x):

X = FT(x)

Ph = [ cmath.phase(z) **for** z **in** X ]

A = [ abs(z) **for** z **in** X ]

A = A[ 0 : round(len(A)/2) ]

Ph = Ph[ 0 : round(len(Ph)/2) ]

A = [ a/len(A) **for** a **in** A ]

F = [ f/(T\*len(x)) **for** f **in** np.arange(0, len(A), 1.0) ]

**return** [A, Ph, F]

*#посчитать спектр непрерывного сигнала*

*#ну тут надо численно проинтегрировать методом трапеций*

*#вроде результат сходится с дискретным спектром*

*#на вход: сигнал, начальное время наблюдения, конечное время наблюдения*

*#шаг сетки по времени для интрегрирования*

*#частоты для которых считем спектр*

**def** Sc(sg, t1, t2, ts, f):

t = np.arange(t1, t2, ts)

N = len(t)

w = [ 2\*pi\*fr **for** fr **in** f ]

tmp = []

**for** w0 **in** w:

x = sg(t)\*np.exp(-1j\*w0\*t)

tmp.append(sum( [ (x[i] + x[i + 1])\*0.5\*(t[i + 1] - t[i]) **for** i **in** range(0, N - 1) ] ))

**return** [ [ abs(z)/(ts\*round(N/2)) **for** z **in** tmp ], [ cmath.phase(z) **for** z **in** tmp ] ]

**def** Sinc(x):

epsilon = 1e-6

**if** abs(x) < epsilon:

**return** 1

**else**:

**return** math.sin(x)/x

*#восстанавливаем сигнал по теореме котельникова*

*#x - отсчёты*

*#t1, t2 - начальный и конечнй моменты времени*

*#ts - шаг по времени*

*#TT - период дискретизации*

**def** TK(x, t1, t2, ts, TT):

tmp = []

qwe = []

t = np.arange(t1, t2, ts)

NN = len(t)

K = len(x)

**for** ind, t0 **in** enumerate(list(t)):

qwe = []

qwe = [ x[k]\*Sinc(pi/TT\*(float(t0) - float(k)\*TT)) **for** k **in** range(0, K) ]

tmp.append(sum(qwe))

**return** tmp

*#получить дискретный спектр из непрерывного*

*#принимает:*

*#сам сигнал*

*#начальное, конечное время наблюдения и шаг дескритизации сигнала*

*#начальную и конечную частоты спектра*

*#период дескритизации дискретного спектра*

*#возвращает амлитудный и фазовый дискретный спектр сигнала*

**def** SdFromSc(sg, t1, t2, ts, f1, f2, df):

Ad = []

Pd = []

f = np.arange(f1, f2, df)

tmp = Sc(sg, t1, t2, ts, f)

**return** tmp

*#Получить дискретный спектр из непрерывного*

*#Вход: непрерывный спектр*

*#Начальная, конечная частоты и шаг дискретизации по частоте*

*#Выход: дискретный спектр*

**def** ScToSd(S\_func, f0, f1, df):

frs = np.arange(f0, f1, df)

**return** [ S\_func(i) **for** i **in** frs ]

*#Получить дискретный сигнал из непрерывного*

*#Вход: непрерывный сигнал*

*#Начальное, конечное время и шаг дискретизации по времени*

*#Выход: дискретный сигнал*

**def** SigcToSigd(sig\_func, t0, t1, dt):

ta = np.arange(t0, t1, dt)

*# return [ sig\_func(i) for i in ta ]*

**return** sig\_func(ta)

*#Восстановить сигнал по дискретному спектру*

*#A, Ph - амплитудный и фазовый дискретные спектры*

*#f - частоты спектра*

*#t0, t1, ts - начальное, конечное время и шаг по времени*

**def** SdToSig(A, Ph, f = [], t0 = 0, t1 = 0, ts = 0):

Tmp = [ cmath.rect(v\*len(A), Ph[i]) **for** i, v **in** enumerate(A) ]

Tmp2 = [ cmath.rect(v\*len(A), Ph[i]) **for** i, v **in** enumerate(A) ]

Tmp2.reverse()

Q = Tmp + np.conj(Tmp2).tolist()

Q.pop()

res = FT1(Q)

**return** res

*#Получить сигнал из непрерывного спектра*

*#Вход: функция спектра сигнала*

*#Начальная и конечная частота, шаг дескритизации по частоте и набор временных отсчётов*

*#Выход: сиганл соответствующий входному спектру*

**def** SigFromSc(sfunc, f1, f2, df, t):

fa = np.arange(f1, f2, df)

N = len(fa)

w = np.array([ 2\*pi\*fr **for** fr **in** fa ])

w = 2\*pi\*fa

TMP = []

*# print("N: ", N)*

**for** t0 **in** t:

*# print(t)*

x = [ x0/(2.0\*pi) **for** x0 **in** sfunc(fa)\*np.exp(1j\*w\*t0) ]

*# print(t0)*

TMP.append(sum( [ (x[i] + x[i + 1])\*0.5\*(w[i + 1] - w[i]) **for** i **in** range(0, N - 1) ] ))

**return** TMP

delta\_t = 0.1

t\_begin = 0.1

t\_end = 9.9

t\_begin\_restored = 0.01

t\_end\_restored = 19.99

delta\_t\_restored = 0.01

time\_array = np.arange(t\_begin, t\_end, delta\_t)

time\_array\_restored = np.arange(t\_begin\_restored, t\_end\_restored, delta\_t\_restored)

example\_signal\_1 = **lambda** t: np.sum([(1.0/i)\*np.sin(2\*pi\*sqrt(i)\*t) **for** i **in** range(1, 21)], axis = 0)

example\_signal\_1\_d = SigcToSigd(example\_signal\_1, t\_begin, t\_end, delta\_t)

example\_signal\_1\_restored = TK(example\_signal\_1\_d, t\_begin\_restored, t\_end\_restored, delta\_t\_restored, delta\_t)

example\_signal\_1\_check\_values = example\_signal\_1(np.arange(t\_begin\_restored, t\_end\_restored, delta\_t\_restored))

example\_signal\_2 = **lambda** t: np.sum([(10.0)\*np.sin(2\*pi\*i\*t) **for** i **in** range(1, 51) ], axis = 0)

example\_signal\_2\_d = SigcToSigd(example\_signal\_2, t\_begin, t\_end, delta\_t)

example\_signal\_2\_restored = TK(example\_signal\_2\_d, t\_begin\_restored, t\_end\_restored, delta\_t\_restored, delta\_t)

example\_signal\_2\_check\_values = example\_signal\_2(np.arange(t\_begin\_restored, t\_end\_restored, delta\_t\_restored))

delta\_t = 0.01

t\_begin = 0.01

t\_end = 4.99

time\_array = np.arange(t\_begin, t\_end, delta\_t)

T = delta\_t

example\_signal\_3 = **lambda** t: 10\*np.sin(2\*pi\*5\*t) + 5\*np.sin(2\*pi\*10\*t)

freq\_array = np.arange(0.0, 50.0, 0.1)

example\_signal\_3\_c\_spectrum = Sc(example\_signal\_3, t\_begin, t\_end, delta\_t, freq\_array)

example\_signal\_3\_d = SigcToSigd(example\_signal\_3, t\_begin, t\_end, delta\_t)

example\_signal\_3\_d\_spectrum = Sd(example\_signal\_3\_d)

example\_signal\_3\_d\_spectrum\_from\_c = Sc(example\_signal\_3, t\_begin, t\_end, delta\_t, example\_signal\_3\_d\_spectrum[-1])

delta\_t = 0.005

T = delta\_t

example\_signal\_3\_d\_ = SigcToSigd(example\_signal\_3, t\_begin, t\_end, delta\_t)

example\_signal\_3\_d\_spectrum\_ = Sd(example\_signal\_3\_d\_)

example\_signal\_3\_d\_spectrum\_from\_c\_ = Sc(example\_signal\_3, t\_begin, t\_end, delta\_t, example\_signal\_3\_d\_spectrum\_[-1])

f\_begin = -pi

f\_end = pi

delta\_f = 2\*pi/1000.0

freq\_array = np.arange(f\_begin, f\_end, delta\_f)

f\_begin = -3/(2\*pi)

f\_end = 6/(2\*pi)

delta\_f = 1/(2\*pi)

freq\_array = np.arange(f\_begin, f\_end, delta\_f)

**def** Delta(x):

epsilon = 10e-9

**if** "array" **in** str(type(x)) **or** "list" **in** str(type(x)):

**return** np.array([ (0.0, 1.0)[abs(value) <= epsilon] **for** value **in** x ])

**else**:

**if** abs(x) <= epsilon:

**return** 1.0

**else**:

**return** 0.0

example\_spectrum\_1 = **lambda** f: Theta(pi/2 - abs(f))\*np.sin(f + pi/2)

example\_spectrum\_1 = **lambda** f: Delta(f - 1/(2\*pi))

example\_spectrum\_1\_values = example\_spectrum\_1(freq\_array)

example\_signal\_4 = **lambda** t: sqrt(2.0/pi)\*np.cos(pi\*t/2.0)/(1 - t\*\*2)

t\_begin = 0.0

t\_end = 5.0

delta\_t = 0.01

T = delta\_t

time\_array = np.arange(t\_begin, t\_end, delta\_t)

example\_signal\_from\_spectrum\_1 = SigFromSc(example\_spectrum\_1, f\_begin, f\_end, delta\_f, time\_array)

*##############################################################*

**def** S(w):

**return** np.exp(-1j\*w)

**def** x(t):

epsilon = 10e-6

**if** abs(t - 1.0) <= epsilon:

**return** (pi/2.0)\*\*0.5

**else**:

**return** ((2.0/pi)\*\*0.5)\*(np.sin((t - 1)\*pi/2.0))/(t - 1)

w\_array = np.arange(-pi/2.0, pi/2.0, pi/200.0)

*##############################################################*

plt.figure(0)

freq0 = np.arange(0.0, 50.0, 0.1)

plt.plot(freq0, example\_signal\_3\_c\_spectrum[0])

plt.figure(1)

time0 = np.arange(0.01, 4.99, 0.01)

plt.plot(time0, example\_signal\_3\_d)

plt.figure(2)

plt.plot(example\_signal\_3\_d\_spectrum[-1], example\_signal\_3\_d\_spectrum[0])

plt.figure(3)

plt.subplot(2, 1, 1)

plt.plot(example\_signal\_3\_d\_spectrum[-1], example\_signal\_3\_d\_spectrum[0])

plt.subplot(2, 1, 2)

plt.plot(example\_signal\_3\_d\_spectrum[-1], example\_signal\_3\_d\_spectrum\_from\_c[0])

plt.figure(4)

time1 = np.arange(0.01, 4.99, 0.005)

plt.plot(time1, example\_signal\_3\_d\_)

plt.figure(6)

plt.plot(example\_signal\_3\_d\_spectrum\_[-1], example\_signal\_3\_d\_spectrum\_[0])

plt.figure(7)

plt.subplot(2, 1, 1)

plt.plot(example\_signal\_3\_d\_spectrum\_[-1], example\_signal\_3\_d\_spectrum\_[0])

plt.subplot(2, 1, 2)

plt.plot(example\_signal\_3\_d\_spectrum\_[-1], example\_signal\_3\_d\_spectrum\_from\_c\_[0])

plt.figure(8)

plt.plot(freq\_array, example\_spectrum\_1\_values)

plt.figure(9)

plt.plot(time\_array, example\_signal\_from\_spectrum\_1)

time2 = np.arange(0.1, 9.9, 0.1)

time3 = np.arange(0.01, 19.99, 0.01)

plt.figure(10)

plt.subplot(2, 1, 1)

plt.plot(time2, example\_signal\_1\_d)

plt.subplot(2, 1, 2)

plt.plot(time3, example\_signal\_1\_check\_values)

time2 = np.arange(0.1, 9.9, 0.1)

time3 = np.arange(0.01, 19.99, 0.01)

plt.figure(10)

plt.subplot(2, 1, 1)

plt.plot(time2, example\_signal\_2\_d)

plt.subplot(2, 1, 2)

plt.plot(time3, example\_signal\_2\_check\_values)

dt = 0.01

tb = 0.01

te = 9.99

tbr = 0.01

ter = 19.99

dtr = 0.01

xd = SigcToSigd(example\_signal\_2, tb, te, dt)

xdr = TK(xd, tbr, ter, dtr, dt)

xdc = example\_signal\_2(np.arange(tbr, ter, dtr))

plt.figure(11)

plt.subplot(2, 1, 1)

plt.plot(np.arange(tb, te, dt), xd)

plt.subplot(2, 1, 2)

plt.plot(time3, xdc)

plt.figure(12)

plt.subplot(2, 1, 1)

plt.plot(time3, example\_signal\_2\_restored)

plt.subplot(2, 1, 2)

plt.plot(time3, xdr)

**12. Заключение.**

В рамках выполненной лабораторной работы, при помощи специально разработанного программного обеспечения, были рассмотрены свойства различного рода (в том числе дискретных и непрерывных) сигналов и их спектров.

Созданное программное обеспечение способно дискретизовать непрерывные сигналы, а также вычислять непрерывные и дискретные спектры сигналов, представляя при этом полученные данные сигналов и их спектров в наглядной графической форме.

Также в рамкой данной лабораторной работы в условиях практики на некоторых тестовых сигналах была изучена важнейшая теорема цифровой обработки сигналов — теореме Котельникова. В частности были показаны возможности по восстановлению непрерывных сигналов в зависимости от того, какой была выбрана их частота дискретизации.