Федеральное агентство связи
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и
информатики» (СибГУТИ)

Отчет
по лабораторной работе №1
по дисциплине «Основы систем мобильной связи»
Тема: «Временная и частотная формы сигналов. Преобразования Фурье. Дискретизация сигналов»

Вариант 3

Выполнил:
студент гр. ИА-232
Багрей Анастасия Олеговна
GitHub: https://github.com/BagreyA/OCMC.git

Новосибирск 2023

Содержание

Цель работы

Получить представление о формах радиосигналов, их частотном и временном представлении, а также о преобразованиях Фурье и аналогово- цифровых преобразованиях сигналов, частоте дискретизации сигналов.

Задачи

Изучить влияние помех частоты дискретизации сигнала, выбранной при оцифровке, на его частотные характеристики, осуществляется использование дискретных преобразований Фурье для переходов между временным и частотным представлениями сигналов.

1. Необходимо сгенерировать и визуализировать непрерывный сигнал.
2. Определить максимальную частоту в спектре данного сигнала.
3. Определить минимальную необходимую частоту дискретизации полученного сигнала (теорема Котельникова).
4. Оцифровать сигнал с полученной частотой дискретизации, выбрав требуемое число отсчетов сигнала на длительности 1 секунда, сохранить полученные значения в массив, пока, не озадачиваясь разрядностью АЦП, просто выбранные с частотой дискретизации значения с теми уровнями, которые имеет функция в полученных точках.
5. Выполнить прямое дискретное преобразование Фурье для массива временных отсчетов сигнала оценить ширину данного спектра. А также объем памяти, требуемый для хранения данного массива.
6. Восстановить оригинальный аналоговый сигнал по массиву имеющихся у вас отсчетов, соединив их непрерывной линией и оценить визуальное сходство оригинального сигнала и восстановленного после оцифровки (увеличьте частоту дискретизации в 4 раза и проделайте задания из п.4-6.)
7. Записать аудиофайл со своим голосом. Проанализировать визуально спектр голоса. Определить максимальную частоту в спектре данного сигнала и выбрать требуемую для оцифровки частоту дискретизации. Проанализовать файл с запись голоса используя Matlab.
8. Определить частоту дискретизации, которая была использована при записи голоса на цифровой носитель. Сравнить вычисленное значение с тем, что выводит в Matlab для Fs.
9. Прорести полученный массив y с помощью функции downsample (иными словами – уменьшите частоту дискретизации) и воспроизвести полученный сигнал с помощью matlab-функции play()
10. Выполнить прямое дискретное преобразование Фурье для оригинального звучания и для прореженного сигнала, вывести на график амплитудный спектр сигнала, определитеь его ширину.
11. Оценить влияние разрядности АЦП на спектр сигнала. Для результирующего дискретного сигнала требуется выполнить прямое преобразование Фурье. Сравнить полученный спектр со спектром исходной синусоиды, отсчеты которой не подвергались квантованию по уровню. Вывести среднюю ошибку квантования для случаев, когда разрядность АЦП равна 3/4/5/6.
12. Составьте отчет.

Теоретические сведенья

Системы мобильной связи представляют собой сети связи, которые обеспечивают передачу голосовой информации, текстовых сообщений, данных и других типов информации между мобильными устройствами, такими как смартфоны, планшеты и ноутбуки, и сетью оператора связи. Эти системы позволяют пользователям оставаться подключенными к сети и обмениваться информацией в движении, находясь в различных местах.

Ключевыми устройствами, взаимодействующими между собой на радиоинтерфейсе, являются базовые станции (БС) и мобильные абонентские терминалы (АТ), представляющие собой совокупность приемопередающего оборудования, а также специализированного программного обеспечения (ПО).

Важным аспектом при взаимодействии этих устройств между собой является необходимость формирования, передачи и приема радиосигналов – электромагнитных колебаний. *Антенны* – это устройства, которые непосредственно преобразуют электрическую энергию в электромагнитные колебания или радиосигналы.

В рамках данного занятия студентам предстоит разобраться в следующих аспектах:

- что такое радиосигналы;
- частотное и временное представление сигналов и как помогают преобразования Фурье переходить из одной области в другую;
- как аналоговые сигналы записываются и обрабатываются цифровыми устройствами связи или что такое АЦП (ADC);
- что такое частота дискретизации сигналов (sample rate);
- что такое отсчет сигнала (sample);
- минимальный набор математического аппарата, необходимый для выполнения преобразований Фурье;
- что такое спектр радиосигналов и пр.

Начнем с радиосигналов. **Сигнал** – это некоторый физический процесс, несущий в себе информацию. На рисунке 1 показан пример зависимости амплитуды (напряжения) радиосигнала от времени – непрерывный, аналоговый сигнал. Вся современная техника радиосвязи, особенно мобильной, начиная со 2-го поколения – цифровая. Это значит, что элементы памяти таких устройств состоят из триггеров, которые могут принимать лишь 2 состояния – 0 или 1. Эти триггеры группируются в регистры, обычно по 8 бит (именно поэтому на 2 языке С/С++ нельзя создать переменную меньше 8 бит, так как это минимальный объем памяти, который можно выделить под хранение переменных или данных).

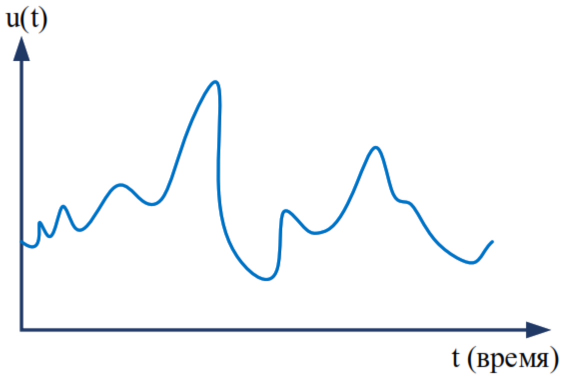


Рис. 1. Пример зависимости амплитуды (напряжения) радиосигнала от времени.

Закономерен вопрос, а как работать цифровому устройству с сигналом, представленным на рисунке 1? Как разложить его по ячейкам памяти для последующей обработки и извлечения из него информации?

Ответом на данный вопрос является процедура оцифровки аналогового сигнала или Аналого-цифровое преобразование (АЦП) – это элемент приемника радиосигнала, на вход которого поступает входное напряжение (например, как на рисунке 1), а на выходе – временные отсчеты данного сигнала – цифровые значения дискретизированной амплитуды сигнала, которые уже можно «сложить» в регистры памяти, например, в виде int16 или float значений, как показано на рисунке 2. То есть задача АЦП – превратить аналоговый сигнал в цифровой для последующей обработки цифровым устройством с целью извлечения из него данных.

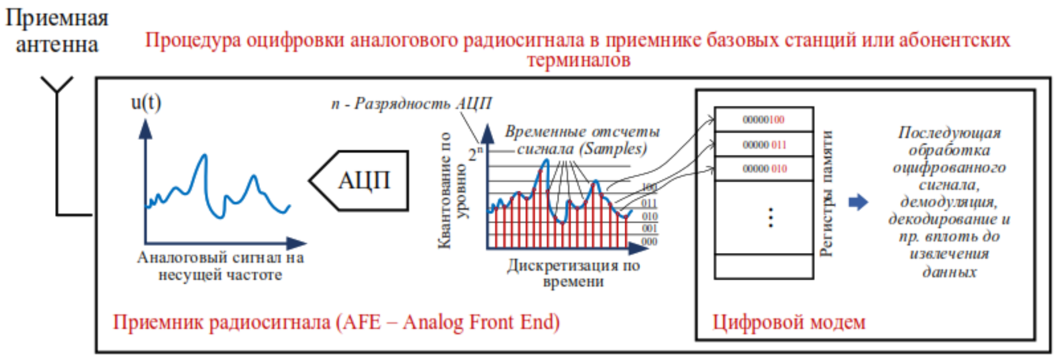


Рис. 2. Аналого-цифровое преобразование радиосигнала.

Таким образом, АЦП выполняют несколько функций: временная дискретизация («временная нарезка сигнала»), квантование по уровню (представление сигнала конечным числом уровней, округление его точных значений), кодирование (упаковка выхода АЦП с заданной разрядностью в выделенные ячейки памяти, размер которых кратен восьми). АЦП характеризуется следующими параметрами:

- разрядность АЦП – число бит, которым кодируется напряжение сигнала;
- частота дискретизации, задающаяся опорным генератором (скорость «нарезки» сигнала);
- диапазон входного сигнала – минимальные и максимальные значения напряжения сигнала на входе АЦП, при котором АЦП работает корректно;
- передаточная характеристика – зависимость числового эквивалента выходного двоичного кода от величины аналогового сигнала, имеет вид ступенчатой функцией.

АЦП может вносить искажения в сигнал в случае нелинейности его передаточной характеристики. Кроме того, стабильность опорного генератора, отвечающего за «нарезку» (дискретизацию) сигнала в строго определенные моменты времени, тоже порой может вызывать нарекания и приводить к тому, что отсчеты будут браться не совсем в предполагаемые моменты времени. Это вызывает *джиттер* – фазовый шум. Такой шум оказывает существенное влияние на быстро изменяющийся сигнал (с высокой частотой дискретизации, в широкой полосе частот).

Цифро-аналоговое преобразование (ЦАП) решает обратную задачу: на вход устройства подаются цифровые отсчеты, которые затем преобразуются в напряжение. Таким образом формируется ступенчатый и непрерывный сигнал, который затем может быть сглажен фильтром нижних частот.

Математические основы ЦОС

Дельта-функция

Основополагающим понятием в цифровой обработке сигналов (ЦОС) является дельта-функция, изображенная на рисунке 3, интегрируя которую на бесконечности, можно получить 1 (1.1). Рассмотрим некоторые полезные для дальнейших вычислений свойства этой функции. Во-первых, сдвиг функции по времени – рисунок 1 и выражение (1.2), а также стоит обратить внимание на произведение двух дельта-функций в 0 и в каком-то другом моменте времени, интеграл которого будет равен 0 (1.3).

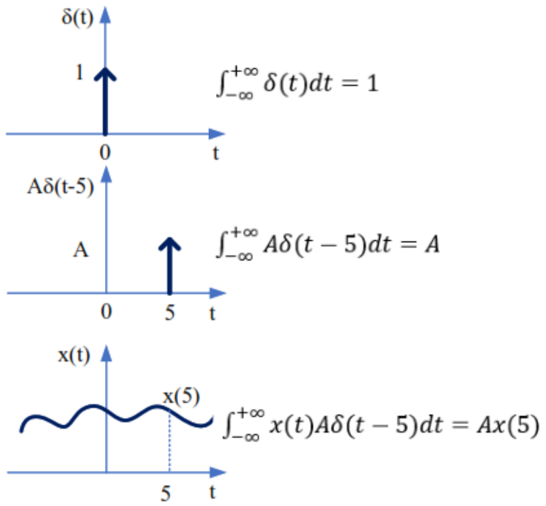


Рис. 3. Дельта-функция.

$$\int_{-\infty}^{+\infty} \delta(t)dt = 1 \qquad (1.1)$$
$$\int_{-\infty}^{+\infty} A\delta(t-5)dt = A \qquad (1.2)$$
$$\int_{-\infty}^{+\infty} A\delta(t)\delta(t-5)dt = 0 \qquad (1.3)$$

Если взять интеграл произведения непрерывной функции x(t) и дельтафункции, например, в точке t=5, то он будет равен произведению значения функции x(t) в t=5 на множитель, стоящий перед дельта-функцией (1.4).

$$\int_{-\infty}^{+\infty} x(t)A\delta(t-5)dt = A \cdot x(5) \qquad (1.4)$$

Это становится очевидным, если взглянуть на рисунок 3. Функция $A\delta(t-5)$ принимает ненулевое значение лишь в t=5. То есть при перемножении ее на x(t), получится $A \cdot x(5)$.

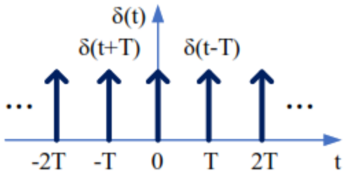
Сумма дельта-функций (1.5) показана на рисунке 4:

$$p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT). \qquad (1.5)$$

И еще один важный нюанс, касающийся дельта-функций – результат преобразования Фурье (позже рассмотрим подробнее) суммы дельта-функций – это также сумма дельта-функций (1.6). Прямое преобразование Фурье используется для перехода от временной к частотной форме сигнала.

$$p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \overset{\text{ППФ}}{\longleftrightarrow} P(j\omega) = \frac{2\pi}{T} \sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s), \quad (1.6)$$

где ω_s – это частота дискретизации, равная $\frac{2\pi}{T}$.



$$p(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT)$$

Рис. 4. Сумма дельта-функций.

В данном разделе столь пристальное внимание уделяется дельтафункциям, так как они используются в процессе оцифровки аналоговых сигналов в ходе временной дискретизации сигналов (выборка отсчетов сигналов). На рисунке 5 представлена процедура временной дискретизации аналогового сигнала в ходе АЦП. x(t) – это непрерывный аналоговый сигнал, который требуется записать в цифровое устройство для последующей обработки. Для этого x(t) перемножается с дельта-функциями, равномерно распределенными по оси времени и расстояние между которыми зависит от частоты дискретизации ωs (или $fs = \omega s/(2\pi)$).

На рисунке 5 для примера частота дискретизации выбрана равной 20 отсчетам в секунду (1/T = 1/0.05 = 20 Гц). Отсчеты отстоят друг от друга на величину T=0.05 секунд. Это означает, что чтобы сохранить/записать одну секунду такого сигнала нам потребуется выделить 20*(количество байт, занимаемых одним отсчетом) байт памяти. И речь не просто про память цифровых устройств, но и про требования к скорости обработки таких массивов данных, ведь одно дело, когда на обработку поступает 20 байт в секунду, а другое – 1 гигабайт. Разумеется, разработчики таких систем и ПО для них заинтересованы сохранять минимальное необходимое количество отсчетов для последующей обработки.

Прежде всего непрерывный сигнал нужно дискретизировать по времени – разбить ось времени на промежутки равной длины и сохранить значения амплитуд в эти моменты времени. Но закономерен вопрос – а как часто нужно брать эти отсчеты, чтобы не потерять данные, содержащиеся в этом сигнале при его дальнейшем восстановлении, какой должна быть частота дискретизации (количество отсчетов в секунду). Ответом на этот вопрос является **теорема Котельникова** (или Найквиста): непрерывный сигнал с ограниченным спектром можно точно восстановить по его дискретным отсчетам, если они были взяты с частотой дискретизации, превышающей максимальную частоту сигнала минимум **в два раза**.

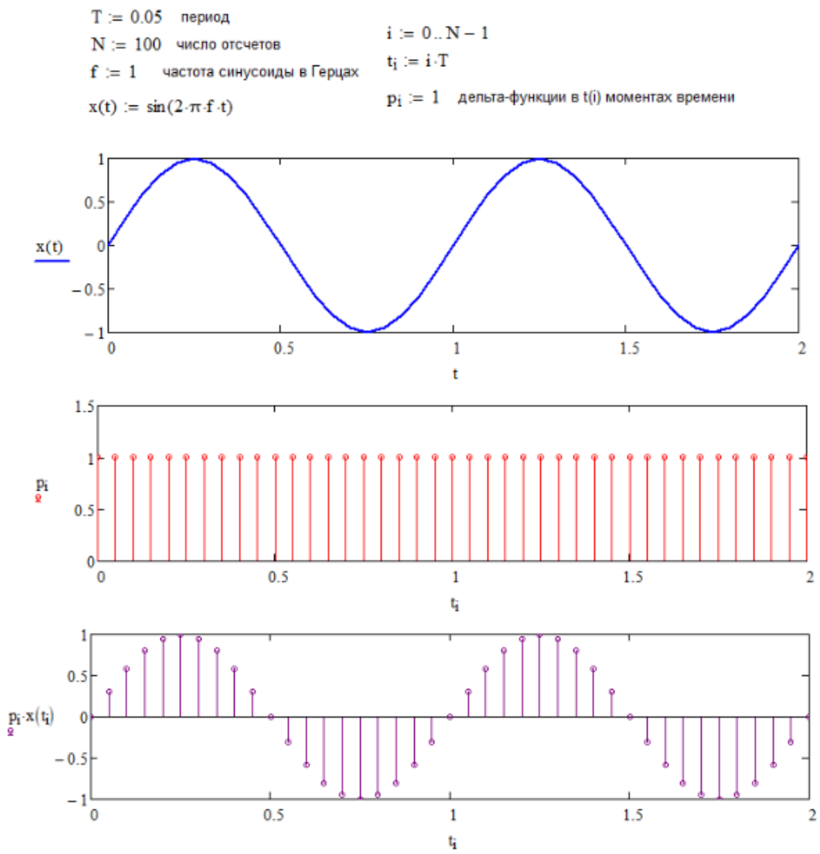


Рис. 5. Процедура временной дискретизации аналогового сигнала в ходе АЦП.

Рассмотрим на примере, что произойдет, если частота дискретизации $\omega s = 2\pi/T$ будет выбрана меньшей, чем требуется в соответствии с теоремой Котельникова (рисунок 6). Справа показаны спектры временных сигналов, полученные путем выполнения прямого преобразования Фурье (ППФ) над временными отсчетами.

Видно, что, если дискретизация выполняется недостаточно быстро, результирующая форма сигнала существенно искажается в частотной области, и соответственно во временной. Такой сигнал будет невозможно корректно восстановить.

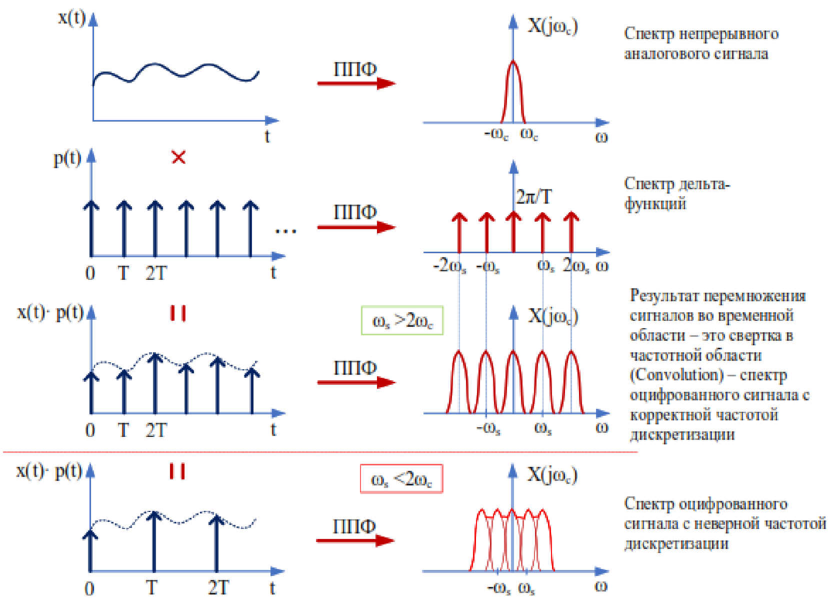


Рис. 6. Процедура временной дискретизации аналогового сигнала в ходе АЦП.

Итак, в результате был получен дискретный сигнал (1.7):

$$x[n] = x(t)|_{t=nT} = x(nT), \qquad (1.7)$$

где n – номер дискретного отсчета.

Преобразования Фурье

Важнейшим утверждением для цифровой обработки сигналов стало утверждение математика Ж.Б.Фурье о том, что любой сколь угодно сложный сигнал можно представить в виде суммы простых сигналов, например, синусоид. В самом деле, если нажать на клавишу фортепиано, мы услышим какой-то простой звук, на определенной частоте, но при одновременном нажатии нескольких клавиш, звуки суммируются и получается более сложное звучание. Такая же логика применима к сигналам, используемым в радиосвязи, достаточно сложным, но все же поддающимся декомпозиции на сумму простых колебаний.

Основываясь на этой логике, математические преобразования Фурье позволяют переключаться между различными эквивалентными представлениями сигналов – временным и частотным. Однако, для простых сигналов не составляет труда перейти от временной к частотной форме. Рассмотрим для примера периодические функции синуса и косинуса и их частотное представление. Для начала вспомним, что данные функции можно записать в комплексном виде следующим образом (тождества Эйлера):

$$\cos(x) = \frac{1}{2}\left(e^{jx} + e^{-jx}\right);\ \sin(x) = \frac{1}{2j}\left(e^{jx} - e^{-jx}\right), \qquad (1.8)$$

Если мы заменим аргумент x на $2\pi ft$, то получим:

$$\cos(2\pi ft) = \frac{1}{2}\left(e^{j2\pi ft} + e^{-j2\pi ft}\right); \sin(2\pi ft) = \frac{1}{2j}\left(e^{j2\pi ft} - e^{-j2\pi ft}\right), \qquad (1.9)$$

Также можно заметить, что

$$\frac{1}{2j} = -\frac{1}{2}j = \frac{1}{2}e^{-j\frac{\pi}{2}}. \qquad (1.10)$$

Данные преобразования (1.8)-(1.10) позволяют увидеть, что функции синуса и косинуса, состоят их суммы/разности двух радиус-векторов, вращающихся со скоростью/частотой f в противоположных направлениях (направление вращения определяется знаком перед частотой сигнала). Попробуем изобразить данные сигналы во временной и частотной областях (рисунок 7).

В частотной области можно отобразить зависимость амплитуды от частоты сигналов. В нашем случае у обоих сигналов есть две компоненты на частотах f и -f, с амплитудами, равными $\left(\frac{1}{2}^2+\frac{1}{2}^2\right)=1/2,$, которые показаны в виде дельта-функций. Если изображать лишь зависимость амплитуды от частоты, то может показаться, что сигналы идентичны, но ведь очевидно, что они отличаются по начальной фазе. Поэтому для корректного представления необходимо дополнять частотное представление еще и зависимостью начальной фазы от частоты (фазовый спектр).

Обратите еще раз внимание на наличие отрицательных частот в спектрах сигнала. Отрицательная частота не имеет физического смысла, лишь математический – показывает на направление вращения вектора по часовой стрелке (отрицательное приращение угла).

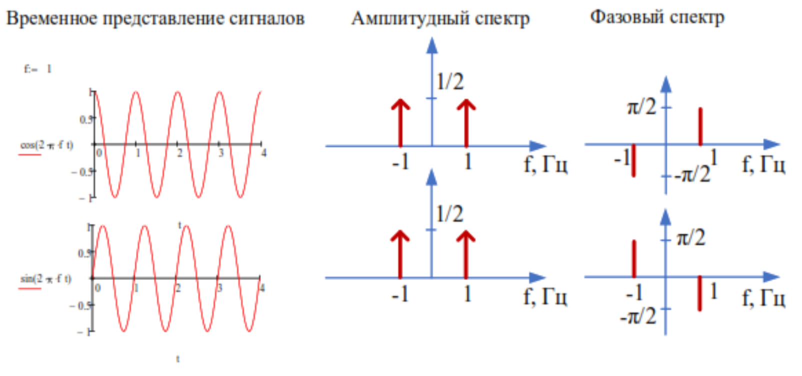


Рис. 7. Временное и частотное представление периодических сигналов *cos(2πft)* и *sin(2πft)*.

Если сравнить спектры периодических сигналов на рисунке 7 со спектрами реальных сигналов на рисунке 6, то можно сделать предположение, что просуммировав некоторое число синусоид с различными параметрами (частотами, амплитудами и фазами) можно получить любой сколь угодно сложный сигнал. Именно на этой идее и построены преобразования Фурье, которые будут рассмотрены ниже.

Задача прямого преобразования Фурье (ППФ) – получить частотное представление радиосигнала, имея его временные значения. И наоборот – обратное преобразование Фурье (ОПФ), зная частотную характеристику сигнала, восстанавливает его временной вид. Познакомиться более детально с преобразованиями Фурье можно и нужно в [1-3].

Существуют разновидности преобразования Фурье, называемые быстрым прямым преобразованием Фурье (БППФ или FFT) и обратным быстрым преобразованием Фурье (ОБПФ или IFFT), которые, как правило, чаще используются в цифровой обработке сигналов, благодаря меньшим вычислительным затратам на их осуществление. Алгоритмы быстрого преобразования Фурье основываются на том, что в вычислениях есть много периодически повторяющихся значений (в силу периодичности функций синуса и косинуса). БПФ группирует слагаемые с одинаковыми множителями, существенно уменьшая число умножений за счет исключения повторных вычислений. В результате быстродействие БПФ может в сотни раз превосходить быстродействие стандартного алгоритма.

БПФ и ОБПФ, используемые в модемах мобильных устройств, это прежде всего преобразования дискретного во времени сигнала, базирующиеся на алгоритме Кули-Тьюки, при котором время вычисления для N отсчетов будет порядка Nlog2N. Количество элементов (отсчетов временного сигнала, например), подаваемое на вход такой функции, должно быть равно N=2^k, где k=0,1,..,∞.

Уравнение для ППФ имеет вид (1.8):

$$F(x) = \sum_{n=0}^{N-1} f(n) e^{-j2\pi\left(x\frac{n}{N}\right)}, \qquad (1.8)$$

где N – размерность преобразования Фурье, $F(x)$ – сигнал в частотной области и $f(n)$ – дискретизированный сигнал во временной области.

ОПФ имеет вид (1.9):

$$f(n) = \frac{1}{N} \sum_{n=0}^{N-1} F(x) e^{j2\pi\left(x\frac{n}{N}\right)}. \qquad (1.9)$$

И еще один немаловажный факт, можно обратить внимание на то, что на вход преобразования Фурье могут идти временные отсчеты в комплексной форме и не всегда понятно, откуда появились комплексные числа, если временной сигнал выглядит как что-то вещественное. Если вкратце, то перед АЦП может стоять квадратурный демодулятор, домножающий вещественный входной сигнал на функции синуса и косинуса и выдавая на вход АЦП мнимую и реальную составляющую сигналов, с которыми и работает АЦП. Подробнее это рассматривается в курсе «Математические основы цифровой обработки сигналов» и в статьях [4-5], обязательных к прочтению для выполнения задания.

Исходные данные

Табл. 1. Варианты заданий.

№ варианта	**Непрерывная периодическая функция**
1	$y(t) = 2\sin\left(2\pi f t + \frac{\pi}{6}\right),$ $f = 5$
2	$y(t) = 2\cos\left(2\pi f t + \frac{\pi}{3}\right),$ $f = 4$
3	$y(t) = 4\sin\left(2\pi f t + \frac{\pi}{3}\right),$ $f = 6$

Этапы выполнения работы

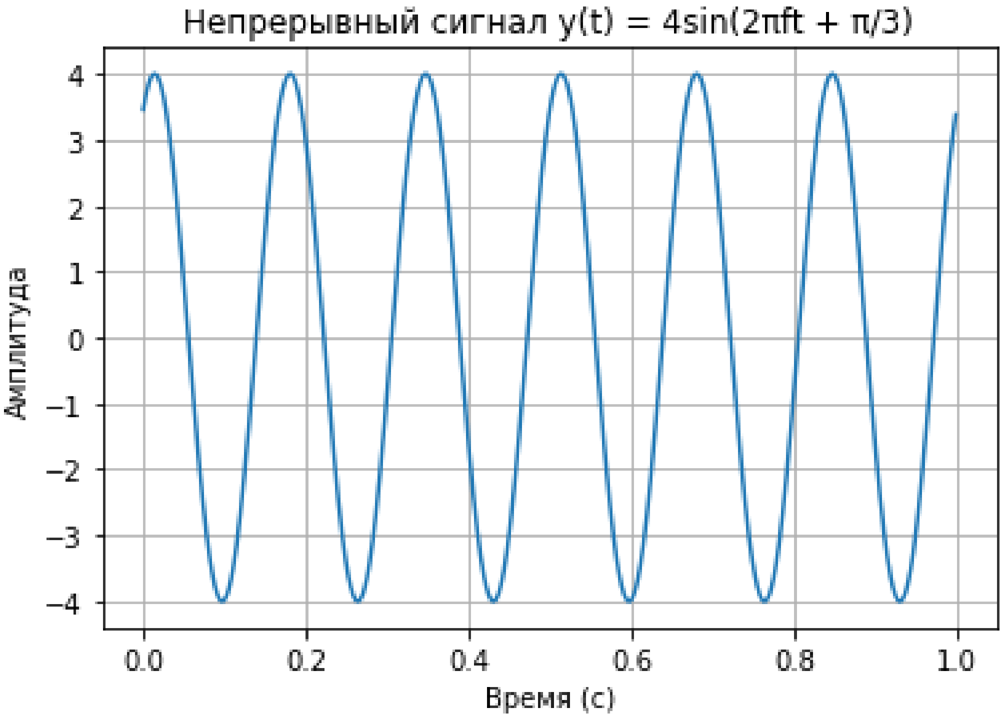
1) Нужно сгенерировать и визуализировать (вывести на график) непрерывный сигнал.

Работа будет проводиться на Python с использование библиотеки «matplotlib». Представленный код был предложен на дисциплине «Математические основы цифровой обработки сигналов» и подходит для этой задачи.

```
import numpy as np
import matplotlib.pyplot as plt

# Параметры сигнала
f = 6          # Частота в Гц
A = 4          # Амплитуда
phi = np.pi / 3 # Фаза

# Временные параметры
t = np.arange(0, 1, 0.001)
```

```
# Генерация сигнала
y = A * np.sin(2 * np.pi * f * t + phi)

plt.figure()
plt.plot(t, y)
plt.title('Непрерывный сигнал y(t) = 4sin(2πft + π/3)')
plt.xlabel('Время (с)')
plt.ylabel('Амплитуда')
plt.grid()
plt.axis('tight')
plt.show()
```

Программа вывела график с колебаниями непрерывного сигнала:



2) Определить максимальную частоту в спектре данного сигнала.

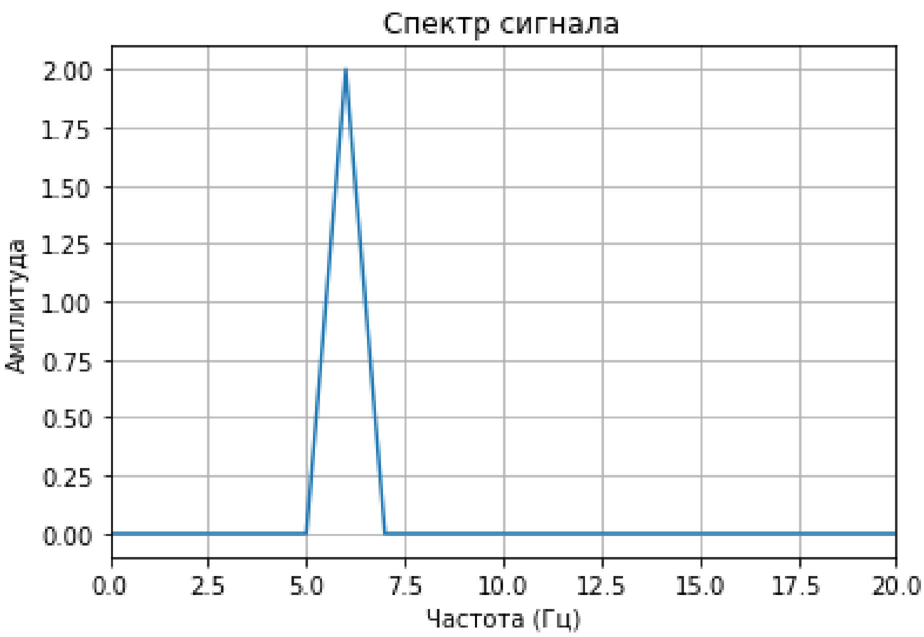
Поскольку сигнал является синусоидой, в его спектре присутствует одна единственная частота, равная f. Таким образом, максимальная частота в спектре сигнала y(t) равна 6 Гц.

Для уверенности можно воспользоваться методом преобразование Фурье, тогда в код добавляется эта часть:

```
import numpy as np
import matplotlib.pyplot as plt

*прошлый код*

# Преобразование Фурье
Y = np.fft.fft(y)
N = len(Y) # Количество точек
Y_magnitude = np.abs(Y) / N
```

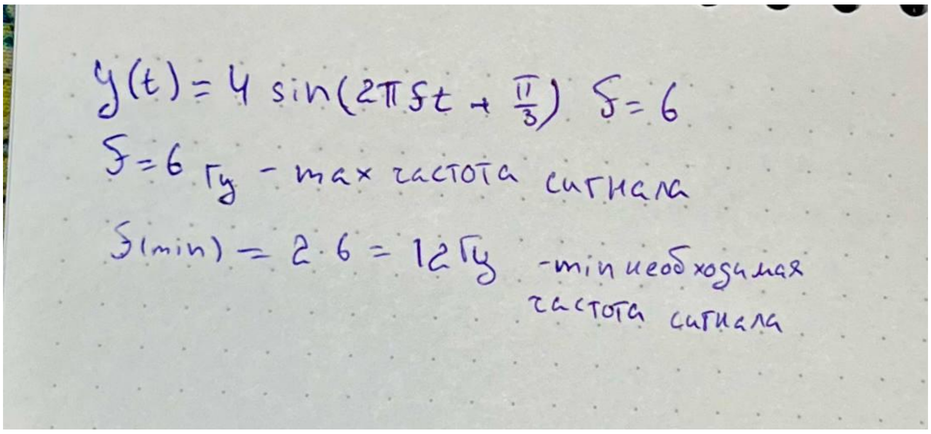
```
frequencies = np.fft.fftfreq(N, d=0.001)
# Визуализация спектра
plt.figure()
plt.plot(frequencies[:N // 2], Y_magnitude[:N // 2])
plt.title('Спектр сигнала')
plt.xlabel('Частота (Гц)')
plt.ylabel('Амплитуда')
plt.grid()
plt.xlim(0, 20)
plt.show()
```

Получаем график, который подтверждает изначальные выводы (при достижении 6 Гц график возрастает):



3) Определить минимальную необходимую частоту дискретизации полученного сигнала.

Воспользуемся теоремой Котельникова. Согласно теореме о дискретизации, минимальная частота дискретизации, необходимая для восстановления сигнала без искажения, должна быть как минимум в два раза больше самой высокой частоты в сигнале.

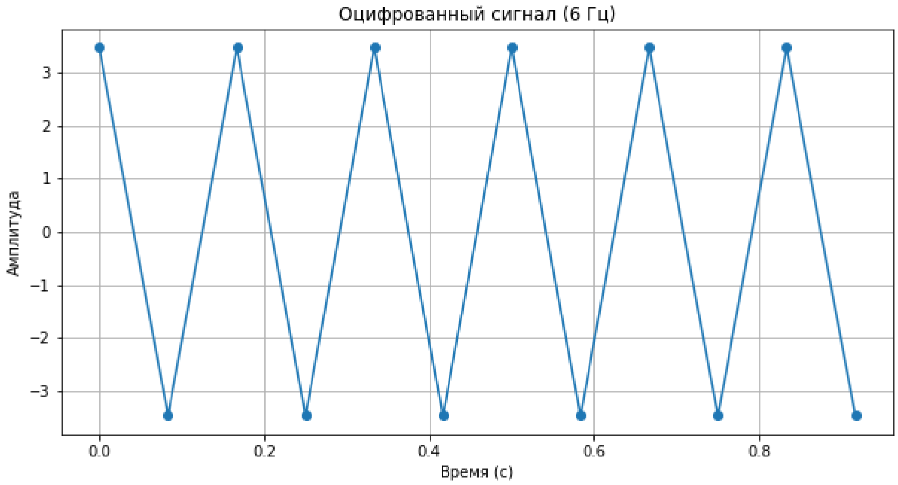


4) Оцифровать сигнал с полученной частотой дискретизации, выбрав требуемое число отсчетов сигнала на длительности 1 секунда, сохранить полученные значения в массив,

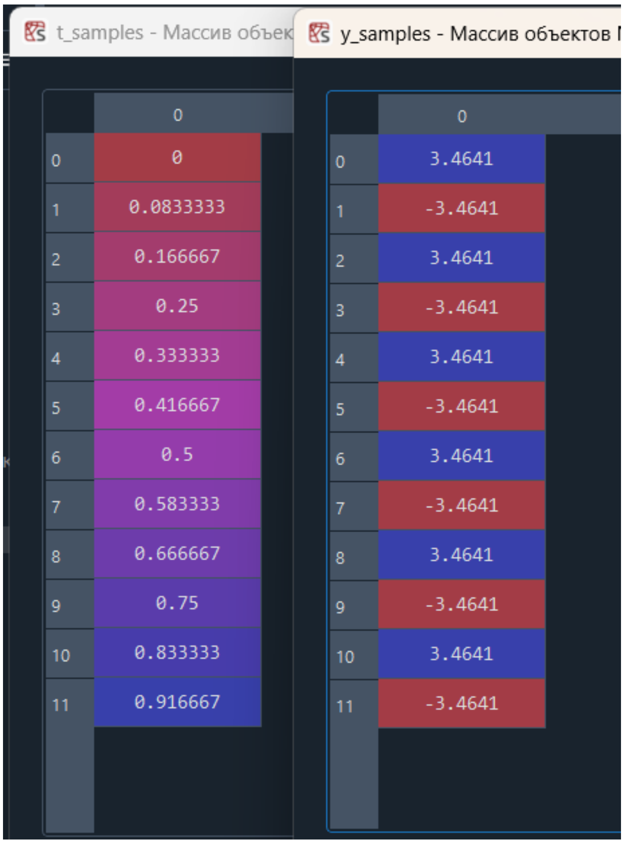
пока, не озадачиваясь разрядностью АЦП, просто выбранные с частотой дискретизации значения с теми уровнями, которые имеет функция в полученных точках.

Для этого нам также нужно внести изменения в изначальный код:

```
import numpy as np
import matplotlib.pyplot as plt

# Параметры сигнала
f = 6
A = 4
phi = np.pi / 3

# Параметры дискретизации
fs = 12
t_duration = 1
N = int(fs * t_duration)

# Генерация временных точек с частотой дискретизации 12 Гц
t_samples = np.linspace(0, t_duration, N, endpoint=False)
y_samples = A * np.sin(2 * np.pi * f * t_samples + phi)

# Сохранение в массиве (y_samples уже является массивом с оцифрованными значениями)
print("Временные точки (с):", t_samples)
print("Оцифрованные значения сигнала:", y_samples)
plt.figure(figsize=(10, 5))
plt.plot(t_samples, y_samples, marker='o', linestyle='-')
plt.title('Оцифрованный сигнал (6 Гц)')
plt.xlabel('Время (с)')
plt.ylabel('Амплитуда')
plt.grid()
plt.show()
```



Временные точки - t_samples; Оцифрованные значения сигнала - y_samples



t_samples - Массив объек

	0
0	0
1	0.0833333
2	0.166667
3	0.25
4	0.333333
5	0.416667
6	0.5
7	0.583333
8	0.666667
9	0.75
10	0.833333
11	0.916667

y_samples - Массив объектов

	0
0	3.4641
1	-3.4641
2	3.4641
3	-3.4641
4	3.4641
5	-3.4641
6	3.4641
7	-3.4641
8	3.4641
9	-3.4641
10	3.4641
11	-3.4641

Значения сигнала меняются между 3.464 и -3.464. Это происходит из-за того, что в пределах одного периода синусоидальной функции вы будете постоянно получать положительные и отрицательные значения:

- Для временных точек, соответствующих максимумам синусоиды (например, 0, 1/3 T, 2/3 T), вы получаете положительные значения.
- Для временных точек, соответствующих минимумам, вы получаете отрицательные значения.

5) Выполнить прямое дискретное преобразование Фурье для массива временных отсчетов сигнала оценить ширину данного спектра. А также объем памяти, требуемый для хранения данного массива.

Чтобы выполнить ДПФ для массива временных отсчетов сигнала и оценить ширину спектра, можем воспользоваться только библиотекой NumPy.

```
import numpy as np

# Параметры сигнала
f = 6           # Частота в Гц
A = 4           # Амплитуда
phi = np.pi / 3 # Фаза

# Параметры дискретизации
fs = 12
t_duration = 1
N = int(fs * t_duration)

# Генерация временных точек и оцифровка сигнала
t_samples = np.linspace(0, t_duration, N, endpoint=False)
y_samples = A * np.sin(2 * np.pi * f * t_samples + phi)

# Выполнение прямого дискретного преобразования Фурье
Y = np.fft.fft(y_samples)

# Оценка ширины спектра
frequencies = np.fft.fftfreq(N, d=1/fs)
```

```
# Определяем ширину спектра - максимальная частота
max_frequency = np.max(np.abs(Y))
Frequency_bin_width = frequencies[1] - frequencies[0]

# Объем памяти, занимаемый массивом
memory_usage = y_samples.nbytes
memory_usage_kb = memory_usage / 1024

# Вывод результатов
print("Дискретное преобразование Фурье (амплитуда):", np.abs(Y))
print("Ширина спектра (максимальная частота):", max_frequency)
print("Объем памяти, занятый массивом y_samples:", memory_usage_kb, "КБ")
```

```
Дискретное преобразование Фурье (амплитуда): [1.86517468e-14 6.9625
4828e-15 1.46482138e-14 1.49282687e-14
 9.60711850e-15 2.18832456e-14 4.15692194e+01 2.18832456e-14
 9.60711850e-15 1.49282687e-14 1.11616928e-14 6.96254828e-15]
Ширина спектра (максимальная частота): 41.569219381653
Объем памяти, занятый массивом y_samples: 0.09375 КБ
```

Дискретное преобразование Фурье (амплитуда): эти значения показывают амплитуды частот, присутствующих в сигнале, значения в середине массива значительно больше, чем остальные. Это связано с тем, что они представляют основные частоты вашего сигнала.

Ширина спектра (максимальная частота): это значение соответствует максимальной амплитуде в спектре, что связано с частотой сигнала, который был задан (6 Гц с амплитудой 4). Это значение отражает структуру сигнала и частоты, которые в него входят.

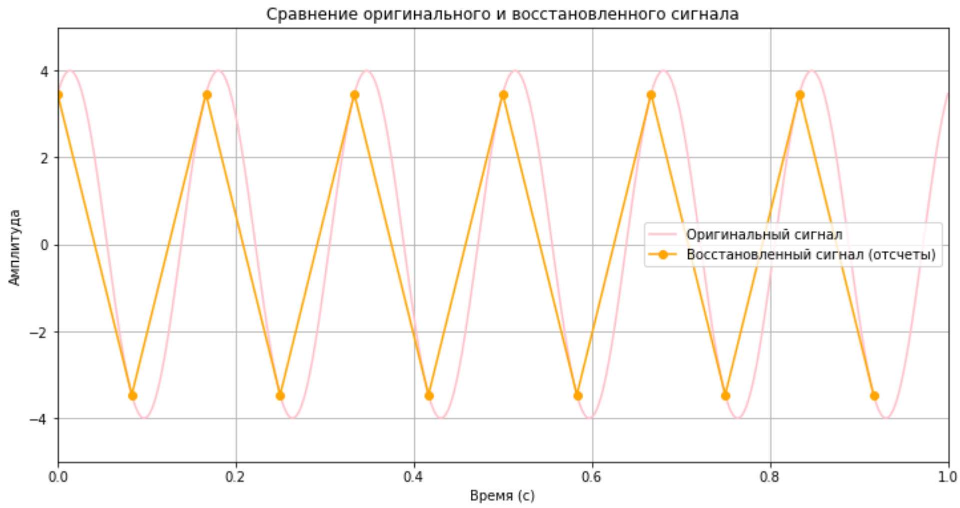
Объем памяти, занятый массивом y_samples: это значение соответствует количеству памяти, занимаемой вашим массивом, который содержит временные отсчеты вашего сигнала.

6) Восстановить оригинальный аналоговый сигнал по массиву имеющихся у вас отсчетов, соединив их непрерывной линией и оценить визуальное сходство оригинального сигнала и восстановленного после оцифровки (увеличьте частоту дискретизации в 4 раза и проделайте задания из п.4-6.)

Чтобы восстановить оригинальный сигнал из массива отсчетов, нужно интерполировать значения между этими точками.

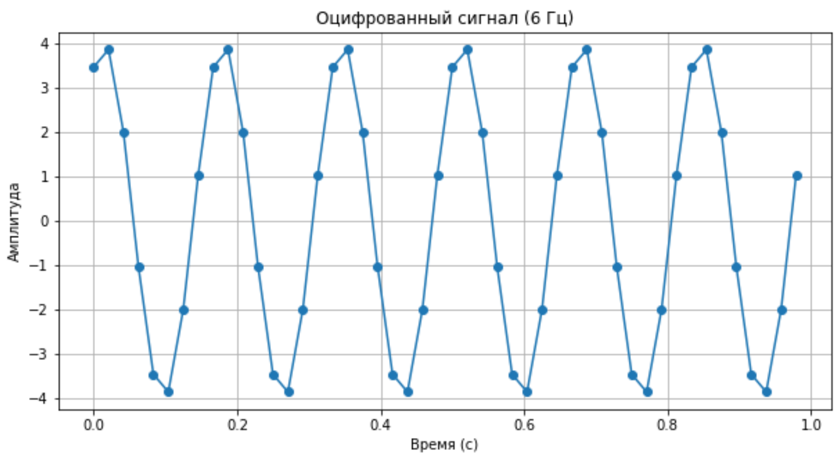
```
import numpy as np
import matplotlib.pyplot as plt

# Параметры сигнала
f = 6
A = 4
phi = np.pi / 3

fs = 12
t_duration = 1
N = int(fs * t_duration)

# Генерация временных точек и оцифровка сигнала
t_samples = np.linspace(0, t_duration, N, endpoint=False)
```

```
y_samples = A * np.sin(2 * np.pi * f * t_samples + phi)

# Восстановление оригинального сигнала с помощью интерполяции
t_fine = np.linspace(0, t_duration, 1000)
y_fine = A * np.sin(2 * np.pi * f * t_fine + phi)

# Визуализация
plt.figure(figsize=(12, 6))
plt.plot(t_fine, y_fine, label='Оригинальный сигнал', color='blue')
plt.plot(t_samples, y_samples, 'o-', label='Восстановленный сигнал (отсчеты)', color='orange')
plt.title('Сравнение оригинального и восстановленного сигнала')
plt.xlabel('Время (с)')
plt.ylabel('Амплитуда')
plt.legend()
plt.grid()
plt.xlim(0, t_duration)
plt.ylim(-5, 5)
plt.show()
```



7) Увеличьте частоту дискретизации в 4 раза и проделайте задания из п.4-6

4 задание:



t_samples - Масси

	0
0	0
1	0.0208333
2	0.0416667
3	0.0625
4	0.0833333
5	0.104167
6	0.125
7	0.145833
8	0.166667
9	0.1875
10	0.208333
11	0.229167
12	0.25
13	0.270833
14	0.291667
15	0.3125
16	0.333333
17	0.354167
18	0.375
19	0.395833
20	0.416667
21	0.4375
22	0.458333
23	0.479167
24	0.5
25	0.520833
26	0.541667
27	0.5625
28	0.583333
29	0.604167
30	0.625
31	0.645833
32	0.666667
33	0.6875
34	0.708333
35	0.729167
36	0.75
37	0.770833
38	0.791667
39	0.8125
40	0.833333
41	0.854167
42	0.875
43	0.895833
44	0.916667
45	0.9375
46	0.958333
47	0.979167

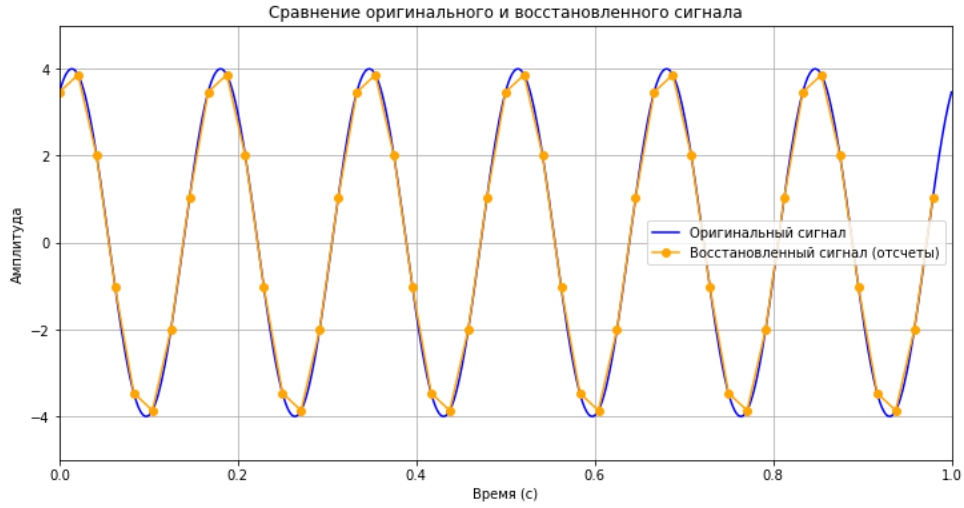
y_samples - Массив объ

	0
0	3.4641
1	3.8637
2	2
3	-1.03528
4	-3.4641
5	-3.8637
6	-2
7	1.03528
8	3.4641
9	3.8637
10	2
11	-1.03528
12	-3.4641
13	-3.8637
14	-2
15	1.03528
16	3.4641
17	3.8637
18	2
19	-1.03528
20	-3.4641
21	-3.8637
22	-2
23	1.03528
24	3.4641
25	3.8637
26	2
27	-1.03528
28	-3.4641
29	-3.8637
30	-2
31	1.03528
32	3.4641
33	3.8637
34	2
35	-1.03528
36	-3.4641
37	-3.8637
38	-2
39	1.03528
40	3.4641
41	3.8637
42	2
43	-1.03528
44	-3.4641
45	-3.8637
46	-2
47	1.03528

5 задание:

```
Дискретное преобразование Фурье (амплитуда): [5.01820807e-14 1.92054163e-14 1.68753900e-14
4.15244120e-14
 2.19340368e-14 9.69454068e-14 9.60000000e+01 7.54452833e-14
 3.30868814e-14 3.53902264e-14 5.54951617e-14 2.10552786e-14
 6.07883352e-14 1.40770190e-14 1.96054196e-14 1.88313315e-14
 1.17494961e-15 1.89321230e-14 2.56852569e-14 2.58031514e-14
 1.75767525e-14 3.63203564e-14 4.18433615e-14 1.41001450e-14
 3.24185123e-14 1.41001450e-14 3.55271368e-14 3.63203564e-14
 1.75767525e-14 2.58031514e-14 2.56752834e-14 1.89321230e-14
 1.17494961e-15 1.88313315e-14 1.80649858e-14 1.40770190e-14
 6.07883352e-14 2.10552786e-14 5.91590338e-14 3.53902264e-14
 3.30868814e-14 7.54452833e-14 9.60000000e+01 9.69454068e-14
 2.19340368e-14 4.15244120e-14 1.99561486e-14 1.92054163e-14]
Ширина спектра (максимальная частота): 95.99999999999999
Объем памяти, занятый массивом y_samples: 0.375 КБ
```

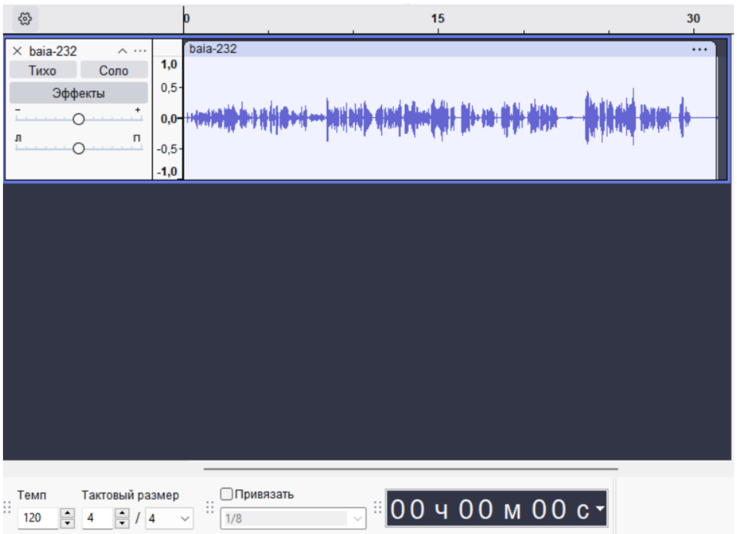
6 задание:



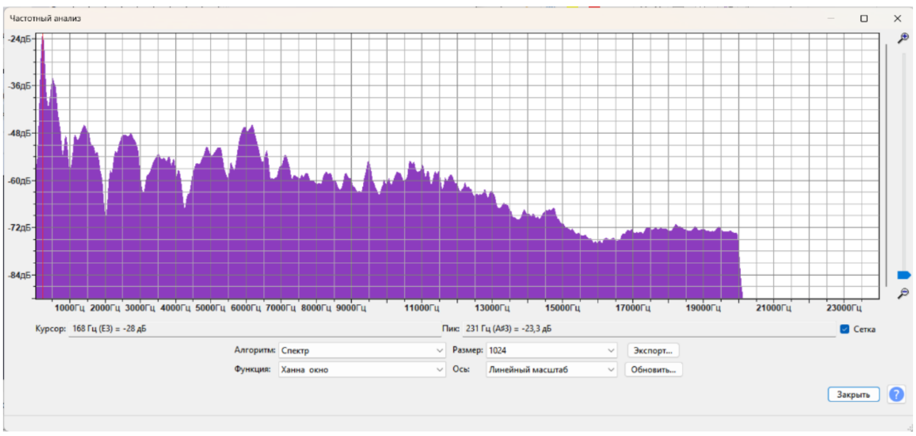
Теперь возвращется более полное представление спектра, поскольку увеличилась частота дискретизации, и можно лучше оценить содержание частот в вашем сигнале.

8) Записать аудиофайл со своим голосом. Проанализировать визуально спектр голоса, используя, например, приложение Audacity. Определить максимальную частоту в спектре данного сигнала и выберите требуемую для оцифровки частоту дискретизации.

Был записан файл «baia-232.wav» и перемещен в приложение Audacity:

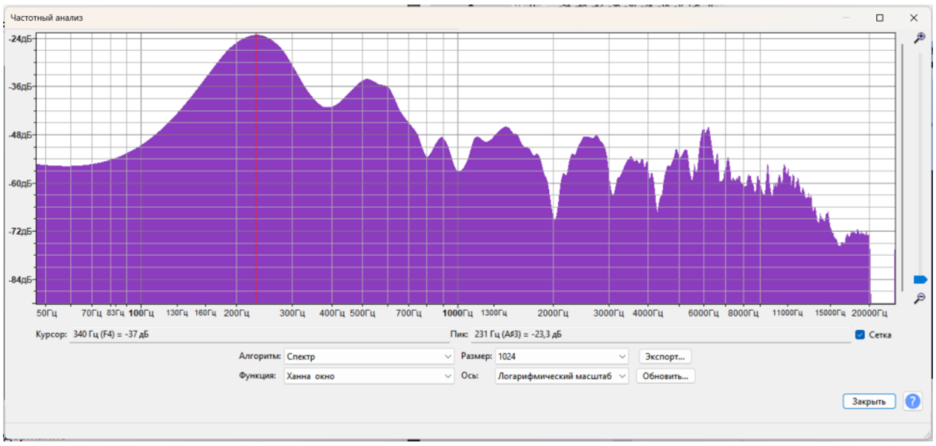


После анализа получался такие графики:



Линейный масштаб

Логарифмический масштаб



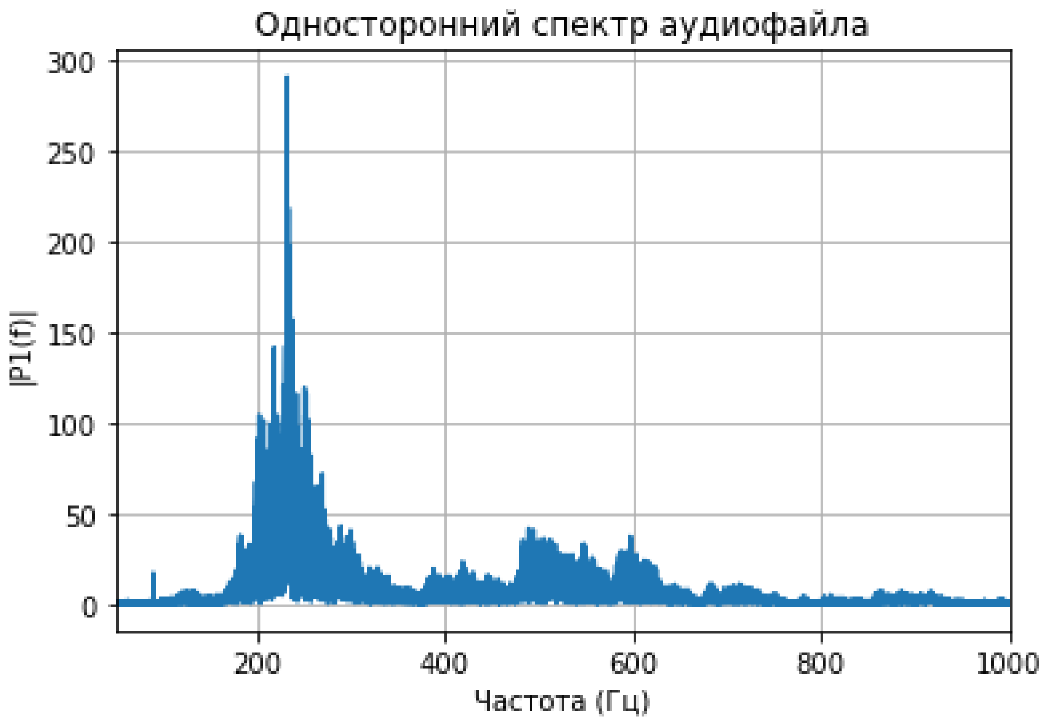
Максимальная частота в спектре: 232 Гц

Из максимальной частоты в спектре можно выбрать требуемую для оцифровки частоту дискретизации (главное условие: минимум в 2 раза больше): 468 Гц

9) Используя библиотеки работы со звуком Pyton проанализировать имеющуюся запись голоса.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

# Считываем файл
Fs, y = wavfile.read('baia-232.wav')

# Применяем БПФ
L = len(y)
Y = np.fft.fft(y)
P2 = np.abs(Y / L)
P1 = P2[:L//2+1]
P1[1:-1] = 2 * P1[1:-1]

# Вектор частот
f = Fs * np.arange(0, (L/2)+1) / L

# Строим график
plt.plot(f, P1)
plt.title('Односторонний спектр аудиофайла')
plt.xlabel('Частота (Гц)')
plt.ylabel('|P1(f)|')
plt.xlim(50, 1000)  # Например, ограничим до 500 Гц
plt.grid()
plt.show()
```

Получившийся график имеет схожесть с результатом Audacity, хорошо видна максимальная частота в спектре и колебания.



10) Определите частоту дискретизации, которая была использована при записи голоса на цифровой носитель. Для этого возьмите количество элементов в файле с записью и разделите на длительность записи в секундах. Например, если у вас файл состоит из 441 000 элементов (отсчетов) и имеет длительность 10 секунд, то частота дискретизации Fs будет равна 441 000/10 = 44 100 Гц. Сравните вычисленное значение с тем, что выводится в Matlab для Fs.
Файл состоит из 150602 элементов и длится 31,38: 150602/31,38 = 4799 Гц

Что выводит код. Разница есть, но она незначительная, время определено верно.

duration	float	1	31.38
f	Array of float64	(753121,)	[0.00000000e+00 3.18 2.39 ...
Fs	int	1	48000
L	int	1	1506240

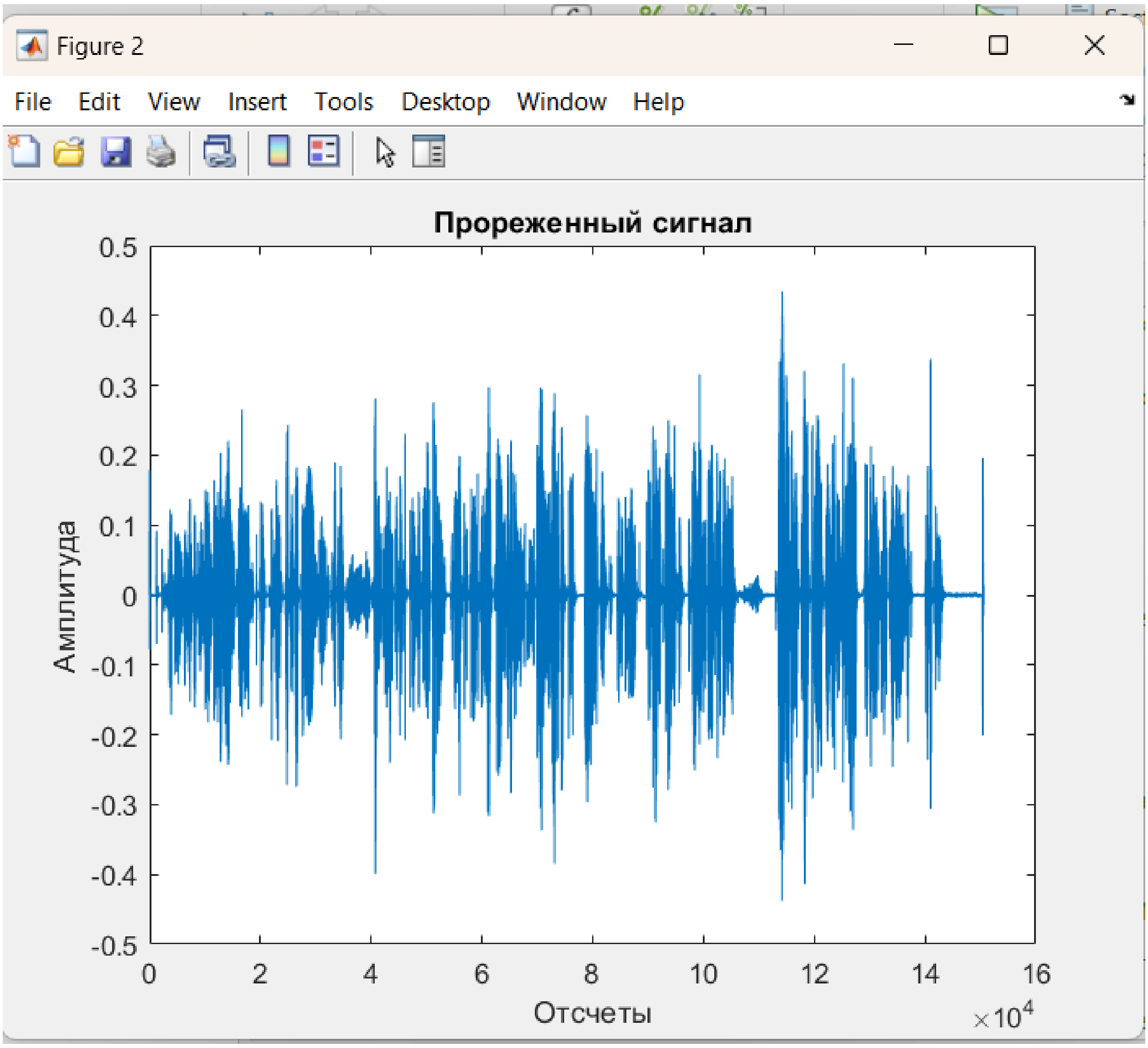
11) Проредите полученный массив y с помощью функции downsample (иными словами – уменьшите частоту дискретизации) и воспроизведите полученный сигнал.

С Pyton возникли трудности (воспроизведение файла так и не заработало), воспользуемся MATLAB.

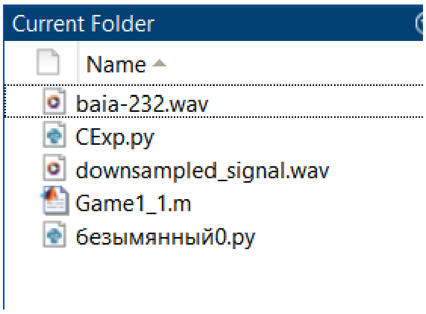
```
c[y, Fs] = audioread('baia-232.wav');

downsample_factor = 10;
y1 = downsample(y, downsample_factor);

zvuk = audioplayer(y1, Fs / downsample_factor);

play(zvuk);
```

```
figure;  % Создаем новую фигуру
plot(y1);
xlabel('Отсчеты');  % Подпись оси X
ylabel('Амплитуда'); % Подпись оси Y
title('Прореженный сигнал');

output_filename = 'downsampled_signal.wav';
audiowrite(output_filename, y1, Fs / downsample_factor);
```

В результате работы программы вывелся прореженный сигнал:



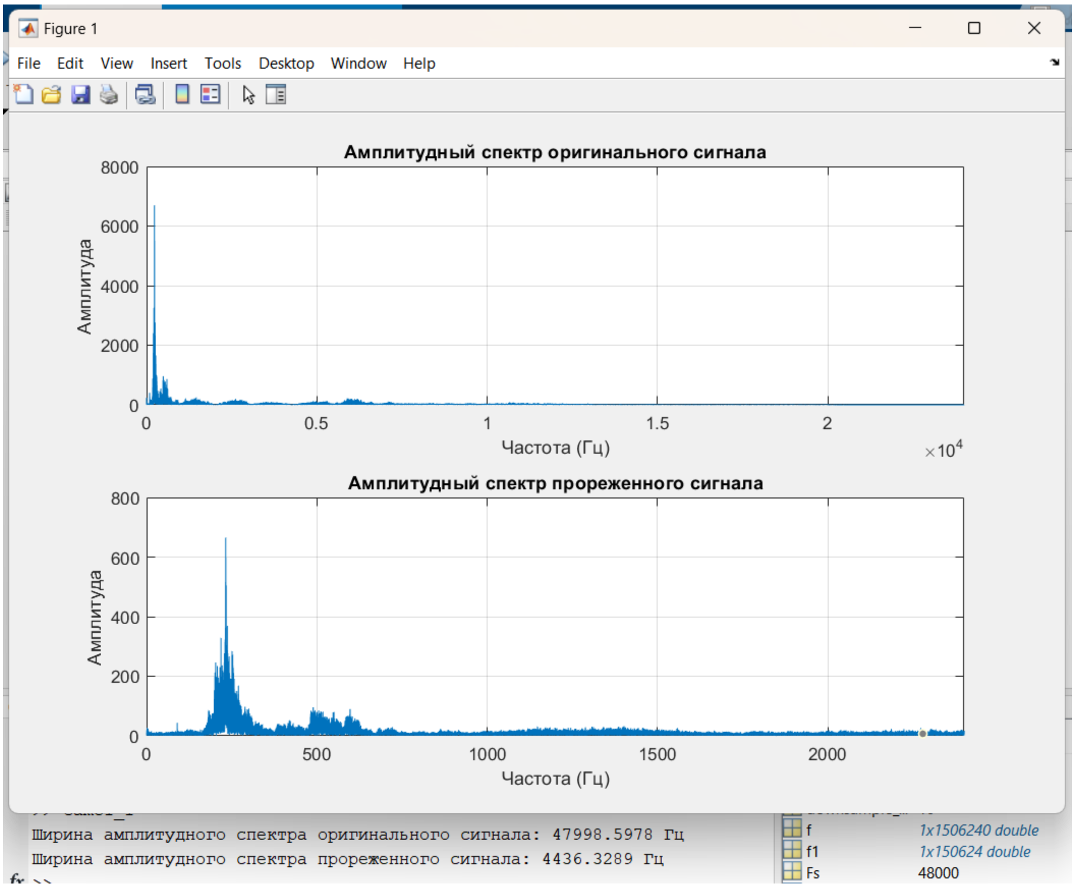
А также звуковой файл «downsampled_signal.wav» с прореженным массивом (файлы с начальным вариантом и конечным будут лежать в репозитории)

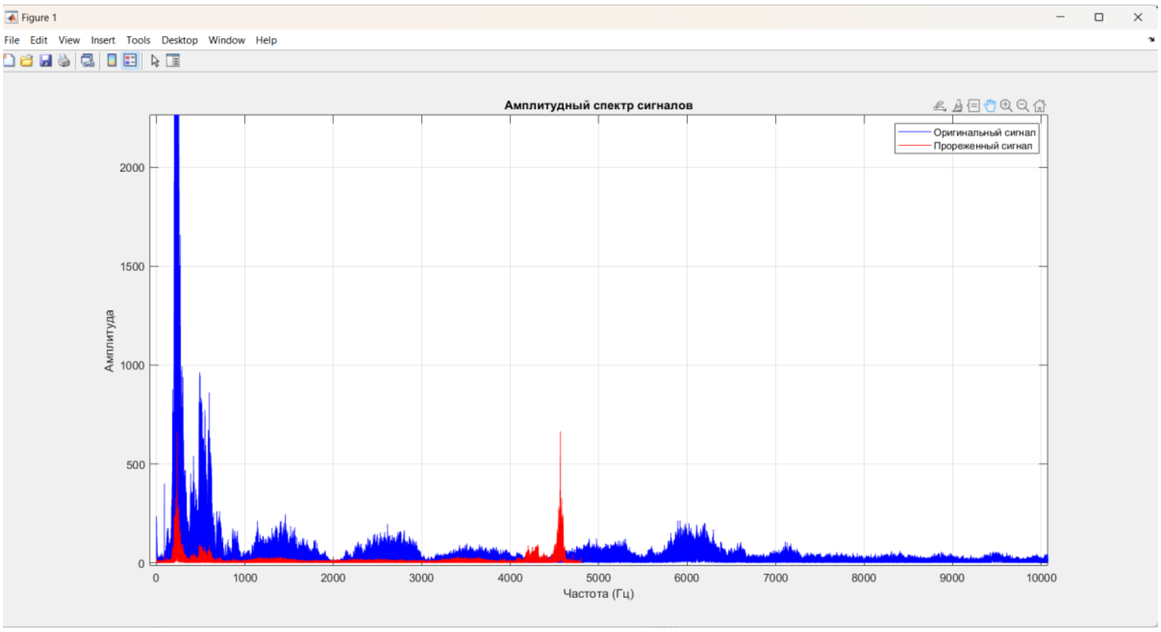


12) Выполните прямое дискретное преобразование Фурье для оригинального звучания и для прореженного сигнала, выведите на график амплитудный спектр сигнала, определите его ширину

```
[y, Fs] = audioread('baia-232.wav');

Y = fft(y);
N = length(Y);
f = (0:N-1)*(Fs/N);
amplitude_spectrum_Y = abs(Y);

downsample_factor = 10;
y1 = downsample(y, downsample_factor);
new_Fs = Fs / downsample_factor;

Y1 = fft(y1);
N1 = length(Y1);
f1 = (0:N1-1)*(new_Fs/N1);
amplitude_spectrum_Y1 = abs(Y1);

figure;

% Оригинальный сигнал
subplot(2, 1, 1);
plot(f, amplitude_spectrum_Y);
title('Амплитудный спектр оригинального сигнала');
xlabel('Частота (Гц)');
ylabel('Амплитуда');
xlim([0 Fs/2]); % Ограничиваем ось X до половины частоты дискретизации
grid on;

% Прореженный сигнал
subplot(2, 1, 2);
plot(f1, amplitude_spectrum_Y1);
title('Амплитудный спектр прореженного сигнала');
xlabel('Частота (Гц)');
ylabel('Амплитуда');
xlim([0 new_Fs/2]);
grid on;

threshold = 0.01 * max(amplitude_spectrum_Y); % Устанавливаем порог
width = f(find(amplitude_spectrum_Y > threshold, 1, 'last')) - ...
```

```
        f(find(amplitude_spectrum_Y > threshold, 1, 'first')); % Ширина для оригинального
сигнала

% Ширина для прореженного сигнала
width1 = f1(find(amplitude_spectrum_Y1 > threshold, 1, 'last')) - ...
        f1(find(amplitude_spectrum_Y1 > threshold, 1, 'first')); % Ширина для прореженного
сигнала

disp(['Ширина амплитудного спектра оригинального сигнала: ', num2str(width), ' Гц']);
disp(['Ширина амплитудного спектра прореженного сигнала: ', num2str(width1), ' Гц']);
```

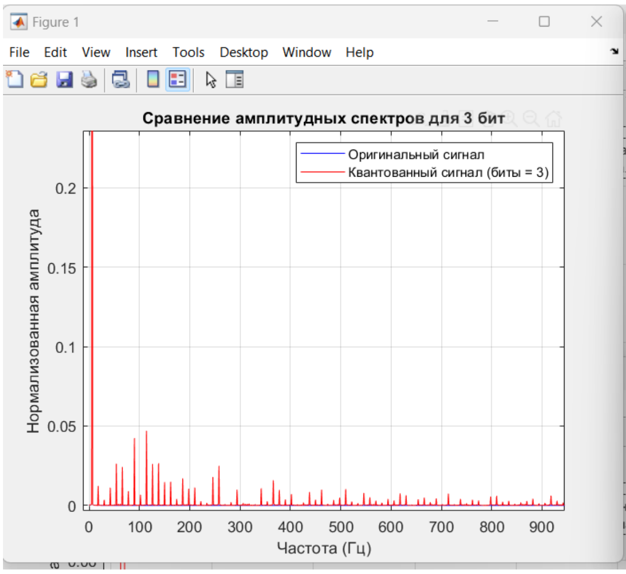


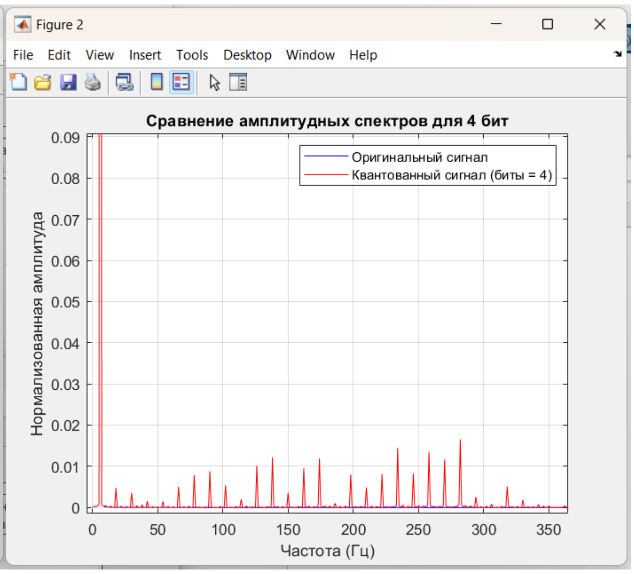


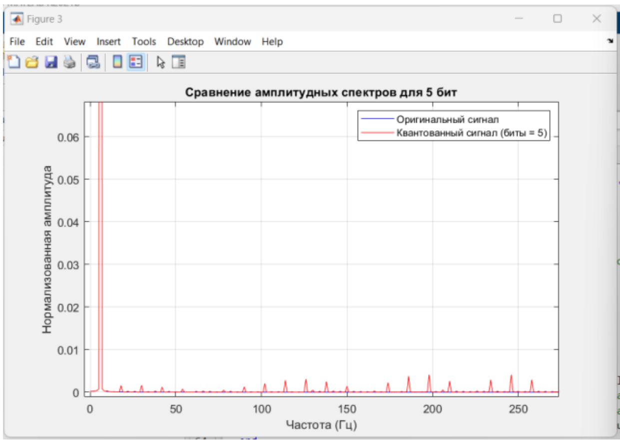
13) Оцените влияние разрядности АЦП на спектр сигнала. Для этого нужно написать функцию, которая бы округляла значения отсчетов сигнала, заданного в вашем варианте, до какого-то числа, определяемого разрядностью АЦП. Допустим если у АЦП всего 3 разряда, то диапазон возможных дискретных значений амплитуд временных отсчетов сигнала – это 0..7. Все значения больше 7 округляются до 7. Для результирующего дискретного сигнала требуется выполнить прямое преобразование Фурье. Сравнить полученный спектр со спектром исходной синусоиды, отсчеты которой не подвергались квантованию по уровню. Вывести среднюю ошибку квантования для случаев, когда разрядность АЦП равна 3/4/5/6.

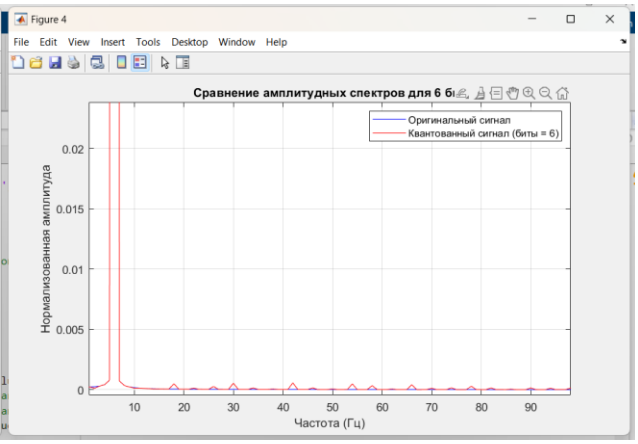
```
function quantization_effect()

    Fs = 8000;
    t = 0:1/Fs:1;
    f_signal = 6;
    original_signal = 4 * sin(2 * pi * f_signal * t + pi/3);

    bit_depths = [3, 4, 5, 6];

    for bits = bit_depths
        quantized_signal = quantize_signal(original_signal, bits);

        Y = fft(original_signal);
        Y_quantized = fft(quantized_signal);

        amplitude_spectrum = abs(Y);
        amplitude_spectrum_quantized = abs(Y_quantized);

        amplitude_spectrum = amplitude_spectrum / max(amplitude_spectrum);
        amplitude_spectrum_quantized = amplitude_spectrum_quantized /
max(amplitude_spectrum_quantized);

        quantization_error = original_signal - quantized_signal;
        mean_quantization_error = mean(abs(quantization_error));

        f = (0:length(original_signal)-1) * (Fs / length(original_signal));

        figure;
```

```
        plot(f(1:length(amplitude_spectrum)/2),
amplitude_spectrum(1:length(amplitude_spectrum)/2), 'b', 'DisplayName',
'Оригинальный сигнал');
        hold on;
        plot(f(1:length(amplitude_spectrum_quantized)/2),
amplitude_spectrum_quantized(1:length(amplitude_spectrum_quantized)/2), 'r',
'DisplayName', sprintf('Квантованный сигнал (биты = %d)', bits));
        hold off;
        title(sprintf('Сравнение амплитудных спектров для %d бит', bits));
        xlabel('Частота (Гц)');
        ylabel('Нормализованная амплитуда');
        xlim([0 Fs/2]);
        legend show;
        grid on;

        fprintf('Средняя ошибка квантования для %d бит: %.4f\n', bits,
mean_quantization_error);
    end
end

function quantized_signal = quantize_signal(signal, bits)

    max_value = 4;
    min_value = -4;

    levels = 2^bits;

    scaled_signal = (signal - min_value) / (max_value - min_value) * (levels - 1);
    scaled_signal = round(scaled_signal);
    scaled_signal(scaled_signal >= levels) = levels - 1;
    quantized_signal = scaled_signal / (levels - 1) * (max_value - min_value) +
min_value;
end
```










```
Средняя ошибка квантования для 3 бит: 0.2551

Средняя ошибка квантования для 4 бит: 0.1235

Средняя ошибка квантования для 5 бит: 0.0612

Средняя ошибка квантования для 6 бит: 0.0306
>>
```

Контрольные вопросы

1) Для чего используются прямое и обратное преобразование Фурье?

Задача прямого преобразования Фурье (ППФ) – получить частотное представление радиосигнала, имея его временные значения. И наоборот – обратное преобразование Фурье (ОПФ), зная частотную характеристику сигнала, восстанавливает его временной вид.

2) Что такое ошибка квантования и дискретизации?

Ошибкой квантований называют отклонение квантованного сигнала от аналогового.

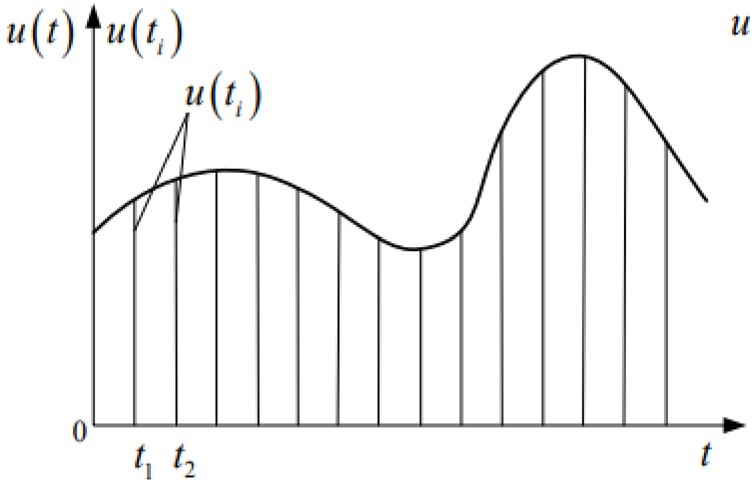
Ошибка дискретизации - это ошибка, возникающая в результате того факта, что функция непрерывной переменной представлена в компьютере конечным числом оценок.

3) Какое количество разрядов АЦП требуется, чтобы оцифровать голос?

Как правило в современных электроэнцефалографах используются АЦП разрядность 24 бита, однако в нейрогарнитурах или старых электроэнцефалографах разрядность АЦП может составлять 16 и менее бит.

4) Как математически получить дискретные отсчеты непрерывного сигнала?

При дискретизации непрерывная функция u(t) преобразуется в последовательность ее отсчетов $u(t_n)$



5) Какой спектр у периодического сигнала sin(10πt+π/2)?

5 Гц

6) Что такое быстрое преобразование Фурье?

Быстрое преобразование Фурье (Fast Fourier transform – FFT) — это более эффективный алгоритм, применяемый для вычисления дискретных сигналов или последовательности данных. Он существенно сокращает количество операций, необходимых для выполнения вычислений, что значительно ускоряет весь процесс расчётов.

7) Как определяется минимальная требуемая для оцифровки частота дискретизации сигнала?

По теореме Котельникова (Найквиста): непрерывный сигнал с ограниченным спектром можно точно восстановить по его дискретным отсчетам, если они были взяты с частотой дискретизации, превышающей максимальную частоту сигнала минимум в два раза

Вывод

В результате проведенного исследования и выполнения задач, связанных с оцифровкой и анализом сигналов, можно сделать следующие выводы:

1. **Генерация и визуализация непрерывного сигнала**: Мы сгенерировали исходный аналоговый сигнал, что позволило оценить его временные и частотные характеристики до оцифровки. Это дало представление о том, как сигнал выглядит в своем естественном (аналоговом) виде.
2. **Определение максимальной частоты в спектре сигнала**: Максимальная частота в спектре сигнала была определена с использованием преобразования Фурье. Это важный шаг, так как он дает понимание того, какие частоты присутствуют в сигнале и помогает выбрать оптимальную частоту дискретизации для его оцифровки.
3. **Минимальная частота дискретизации (по теореме Котельникова)**: Согласно теореме Котельникова, минимальная частота дискретизации должна быть в два раза больше максимальной частоты спектра исходного сигнала. Мы рассчитали и использовали эту частоту для дальнейших шагов оцифровки.
4. **Оцифровка сигнала**: Исходный сигнал был оцифрован с рассчитанной частотой дискретизации, и полученные дискретные значения были сохранены. Это

позволило перевести сигнал в цифровую форму, пригодную для обработки на компьютере.

5. **Дискретное преобразование Фурье и анализ спектра**: Прямое дискретное преобразование Фурье показало ширину спектра оцифрованного сигнала. Было также оценено, сколько памяти требуется для хранения массива дискретных отсчетов. Этот шаг позволил убедиться, что дискретизация корректно сохраняет информацию о частотах сигнала.
6. **Восстановление аналогового сигнала**: При восстановлении аналогового сигнала по дискретным отсчетам были выявлены отличия от исходного сигнала, что связано с потерей информации из-за конечной частоты дискретизации. Однако увеличение частоты дискретизации в 4 раза позволило значительно улучшить качество восстановленного сигнала.
7. **Анализ аудиофайла с голосом**: Был записан аудиофайл, содержащий голос, и его спектр был визуализирован и проанализирован. Определена максимальная частота голоса и выбрана частота дискретизации, которая обеспечивала сохранение основных частотных компонентов сигнала.
8. **Сравнение частоты дискретизации**: Частота дискретизации, использованная при записи голоса на цифровой носитель, была определена и сопоставлена с выводом MATLAB. Вычисленные значения совпали с результатами MATLAB, что подтвердило корректность оцифровки и анализа.
9. **Прореживание сигнала**: С помощью функции downsample частота дискретизации была уменьшена, что привело к снижению качества звучания при воспроизведении сигнала. Это продемонстрировало, как понижение частоты дискретизации ухудшает качество сигнала.
10. **Анализ спектра прореженного сигнала**: Прореженный сигнал и его амплитудный спектр были сопоставлены с исходным. Ширина спектра прореженного сигнала оказалась меньше, что обусловлено потерей высокочастотных компонентов сигнала.
11. **Влияние разрядности АЦП на спектр**: Влияние разрядности АЦП на спектр сигнала было оценено с помощью преобразования Фурье. Уменьшение разрядности АЦП приводит к увеличению ошибок квантования и искажению спектра. Средняя ошибка квантования была рассчитана для разрядностей 3, 4, 5 и 6 бит. Чем меньше разрядность, тем больше искажения в сигнале.