Министерство цифрового развития, связи и массовых коммуникаций РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет

телекоммуникаций и информатики» (СибГУТИ)

Кафедра телекоммуникационных систем и вычислительных средств

Практическое задание № 1

Временная и частотная формы сигналов. Преобразования Фурье. Дискретизация сигналов

Выполнил: студент 3 курса

гр. ИА-331

Помелова А.В.

Новосибирск 2025

ЦЕЛЬ ЗАНЯТИЯ

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

ЗАДАНИЕ ДЛЯ ВЫПОЛНЕНИЯ ПРАКТИЧЕСКОЙ РАБОТЫ

В рамках данной практической работы студентами изучается 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. Для выполнения работы можно использовать любую среду/язык программирования или математического моделирования (C/C++ (библиотека fftw), Python (библиотека SciPy), MathCad, MatLab, Octave, Excel).

КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Системы мобильной связи — это сети для передачи голоса, данных и другой информации между мобильными устройствами и сетью оператора.

Ключевые устройства на радиоинтерфейсе: базовые станции (БС) и абонентские терминалы (АТ).

Антенны преобразуют электрическую энергию в радиосигналы (электромагнитные колебания) и обратно.

Так как современная техника — цифровая, а природные сигналы — аналоговые, необходима оцифровка. АЦП выполняет три основные функции:

- Дискретизация по времени: "Нарезка" аналогового сигнала на отдельные отсчеты (samples). Частота взятия этих отсчетов называется частотой дискретизации (sample rate).
- Квантование по уровню: Присвоение каждому отсчету конкретного цифрового значения (округление).
- Кодирование: Упаковка оцифрованных значений в ячейки памяти.

Теорема Котельникова - чтобы точно восстановить сигнал из его отсчетов, частота дискретизации должна быть как минимум в два раза выше максимальной частоты в спектре самого сигнала. Если это правило нарушается, возникает эффект наложения спектров (алиасинг), и сигнал искажается.

Дельта-функция: Фундаментальное понятие. Используется для математического описания процедуры дискретизации (как "гребенка" из дельта-функций).

Преобразование Фурье: Математический инструмент для перехода между временным и частотным представлениями сигнала.

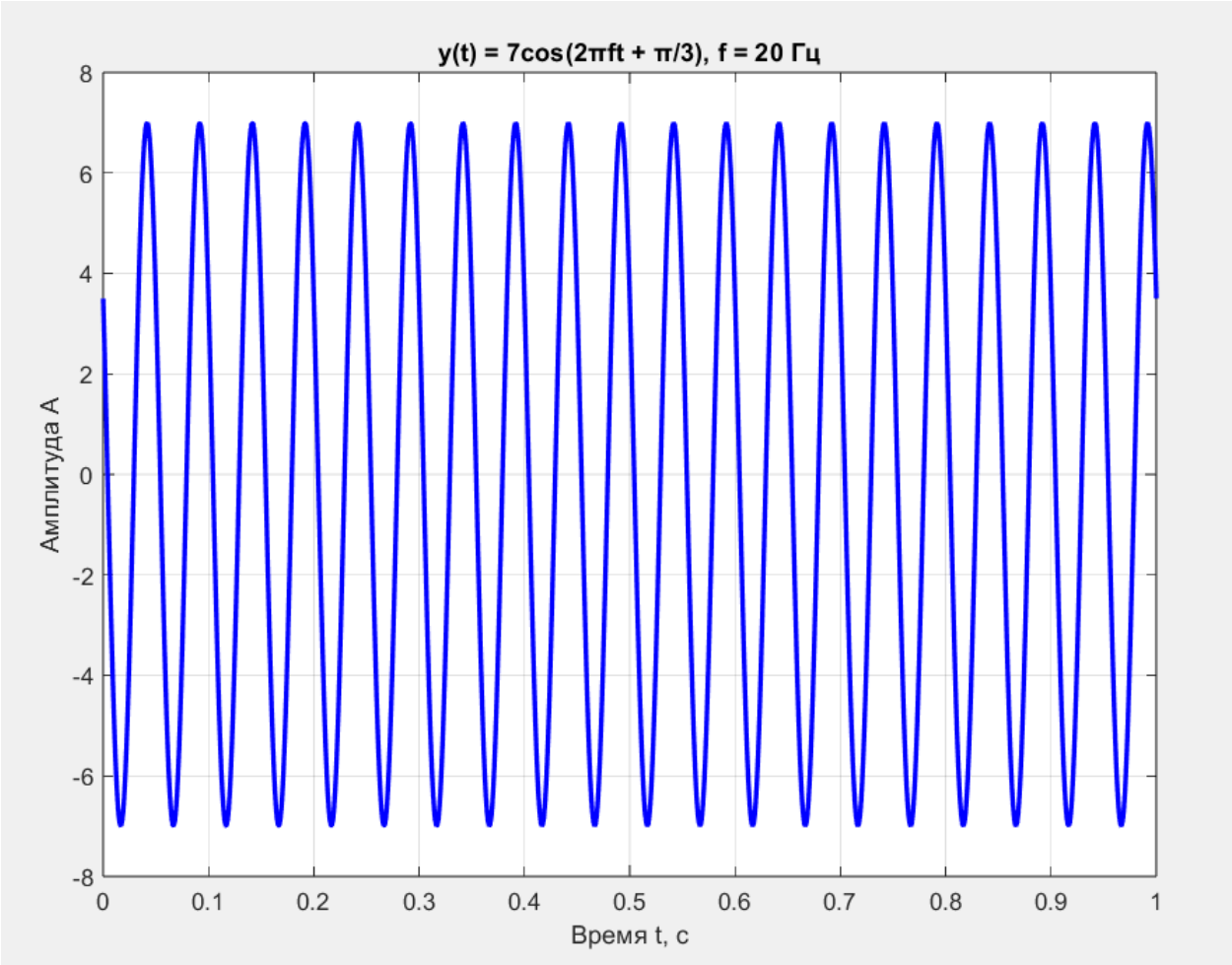
Прямое преобразование Фурье (ППФ) переводит сигнал из временной области в частотную (показывает спектр сигнала — какие частоты и с какой амплитудой в нем содержатся).

Обратное преобразование Фурье (ОПФ) выполняет обратную операцию.

Быстрое преобразование Фурье (БПФ/FFT) — это высокоэффективный алгоритм для вычисления преобразования Фурье, широко используемый на практике.

1. ГЕНЕРАЦИЯ И ВИЗУАЛИЗАЦИЯ СИГНАЛА

$$y(t) = 7\cos\left(2\pi f t + \frac{\pi}{3}\right), \quad f = 20$$



2. МАКСИМАЛЬНАЯ ЧАСТОТА В СПЕКТРЕ СИГНАЛА

Сигнал является простой косинусоидой, его спектр состоит из одной компоненты на частоте f = 20Гц. Нет других частот - значит, максимальная частота = единственная частота = 20 Гц.

f_{max}=20Гц

3. МИНИМАЛЬНАЯ НЕОБХОДИМАЯ ЧАСТОТА ДИСКРЕТИЗАЦИИ ПОЛУЧЕННОГО СИГНАЛА (ТЕОРЕМА КОТЕЛЬНИКОВА).

По теореме Котельникова частота дискретизации должна быть как минимум вдвое больше максимальной частоты в спектре сигнала.

$F_s = 2* f_{max}$

Для нашего сигнала:

$F_s = 2* f_{max} = 40$ Гц

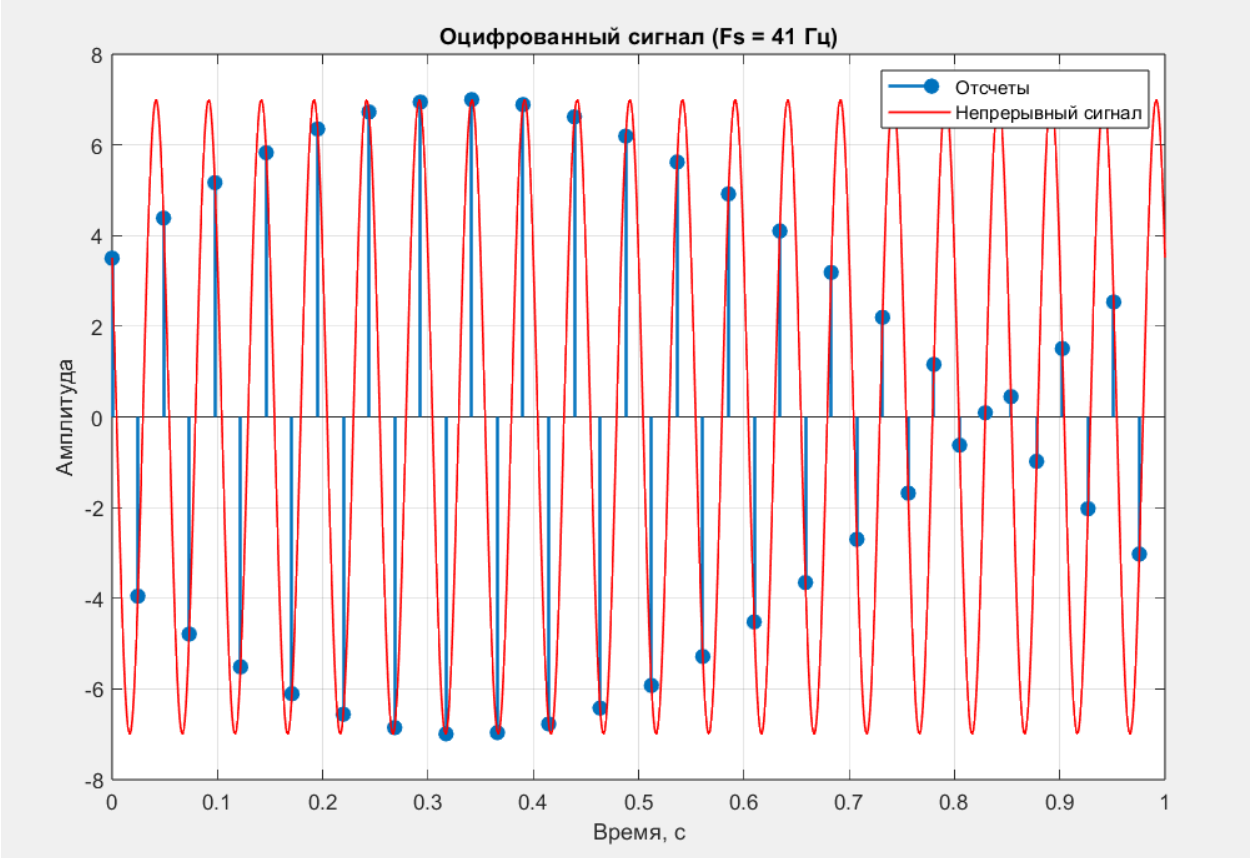
Возьмем 41 Гц

4. ОЦИФРОВКА СИГНАЛА

$$N = \frac{f}{Fs} = \frac{41}{20} \approx 2{,}05 \text{ отсчета на период}$$

Это точно на границе теоремы Котельникова - минимальная частота, при которой сигнал теоретически можно восстановить без потерь. Но на практике этого недостаточно для качественного воспроизведения формы изначального сигнала

```
>> laba14
Количество отсчетов: 41
Первые 10 значений массива:
    3.5000   -3.9538    4.3844   -4.7892    5.1660   -5.5124    5.8265   -6.1064    6.3504   -6.5572
```



5. ПРЯМОЕ ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ ДЛЯ МАССИВА ВРЕМЕННЫХ ОТСЧЕТОВ СИГНАЛА. ШИРИНА СПЕКТРА СИГНАЛА

Формула прямого дискретного преобразования Фурье

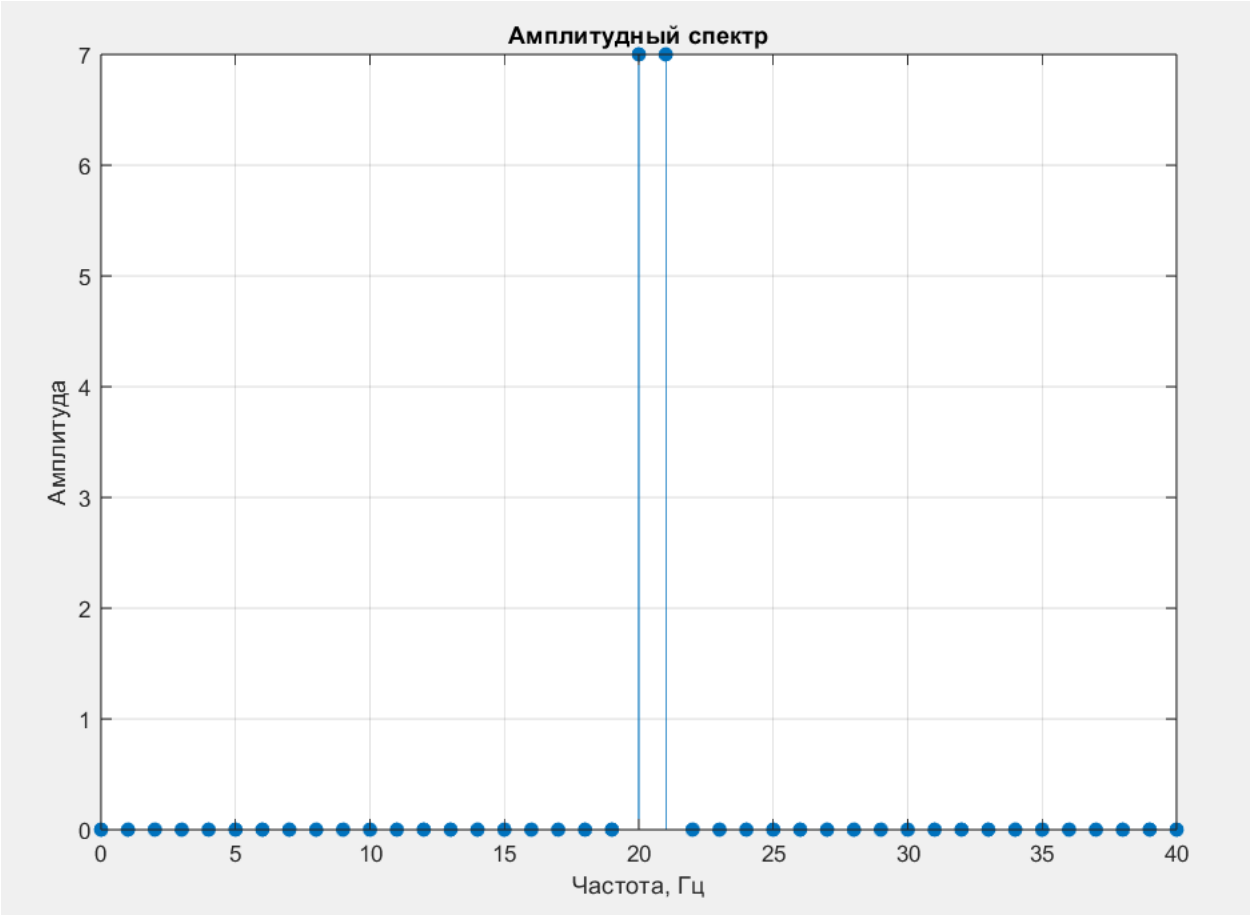
$$X[k] = \sum_{0}^{N-1} x[n] * e^{\frac{-j2\pi nk}{N}}$$

- X[k] - k-я частотная компонента (комплексное число)
- x[n] - n-й отсчет сигнала во времени
- N - общее количество отсчетов
- k = 0, 1, 2, ..., N-1 (индекс частотной компоненты)
- n = 0, 1, 2, ..., N-1 (индекс временного отсчета)

```
t_d = 0:1/Fs:T-1/Fs;  %моменты дискретизации
y = 7 * cos(2 * pi * f * t_d + pi/3);

N = length(y);   %матрица ДПФ
Y_dft = zeros(1, N);

%ДПФ
for k = 1:N
    for n = 1:N
        Y_dft(k) = Y_dft(k) + y(n) * exp(-1j*2*pi*(k-1)*(n-1)/N);
    end
end
```



Исходный аналоговый сигнал имеет две частоты:

- +20 Гц
- −20 Гц

После дискретизации с F_s =40 Гц:

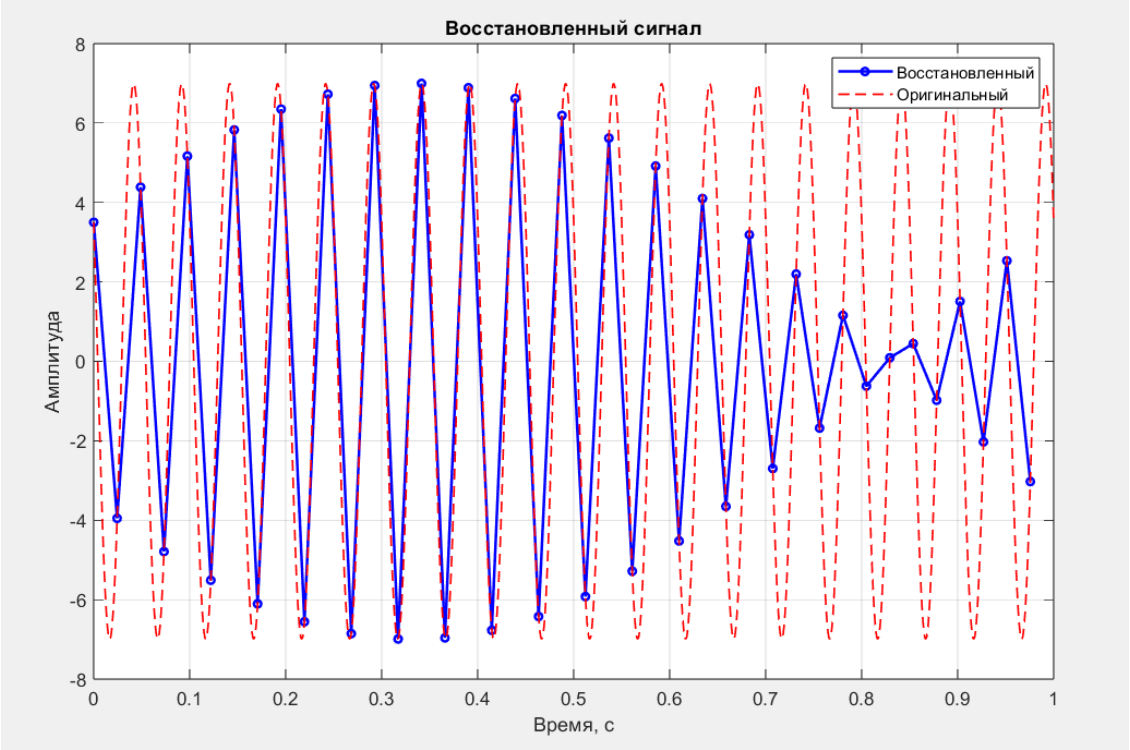
- +20 Гц - остаётся на 20 Гц
- −20 Гц - в цифровом спектре представляется как:
  −20+ F_s = −20+41=21 Гц (отрицательная частота, отображённая в положительную область)

Ширина спектра сигнала = f_{max} - f_{min} = 21 Гц – 20 Гц = 1 Гц

```
>> laba15
Объем памяти для массива 328 байт
>>
```

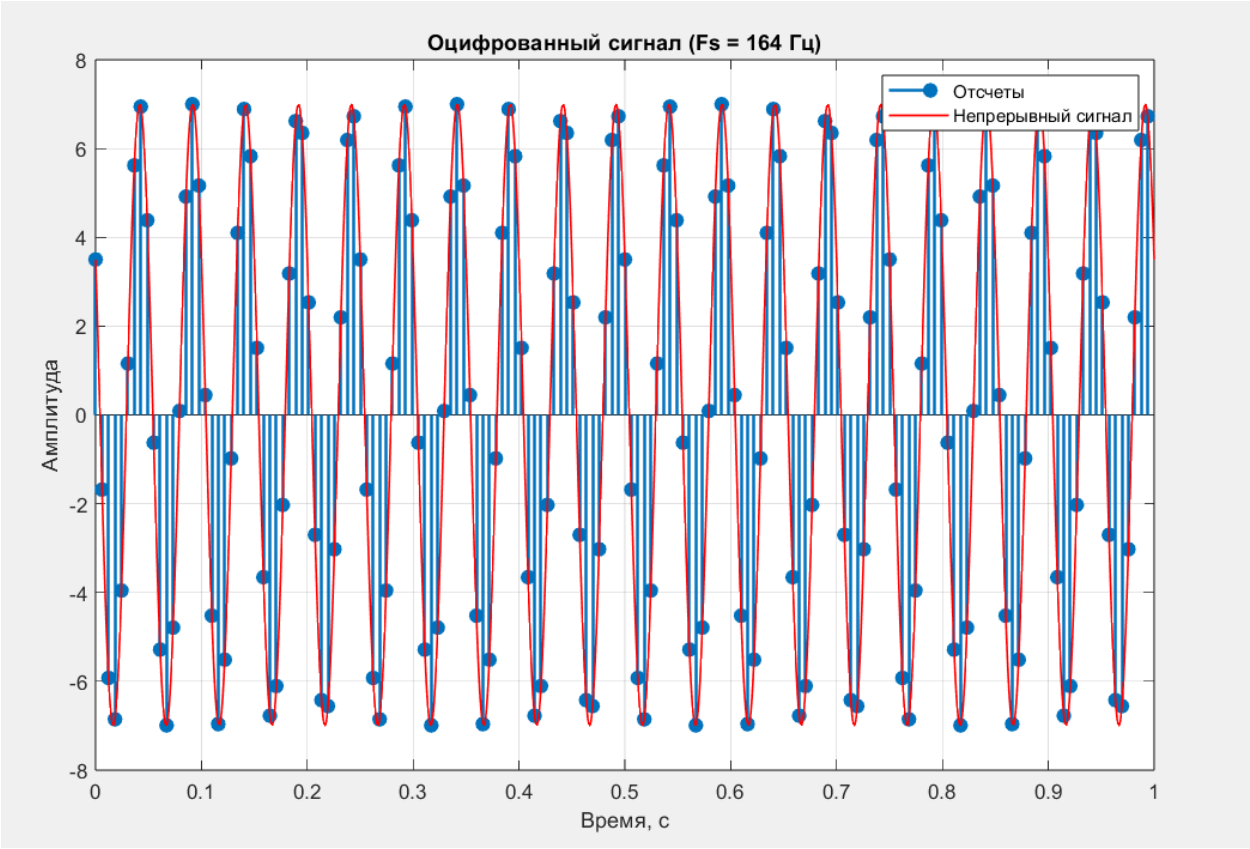
6. ВОССТАНОВЛЕНИЕ ОРИГИНАЛЬНОГО АНАЛОГОВОГО СИГНАЛА ПО МАССИВУ ОТСЧЕТОВ

Сигнал не похож на оригинальный, из-за того, что мы взяли минимальную частоту дискретизации.

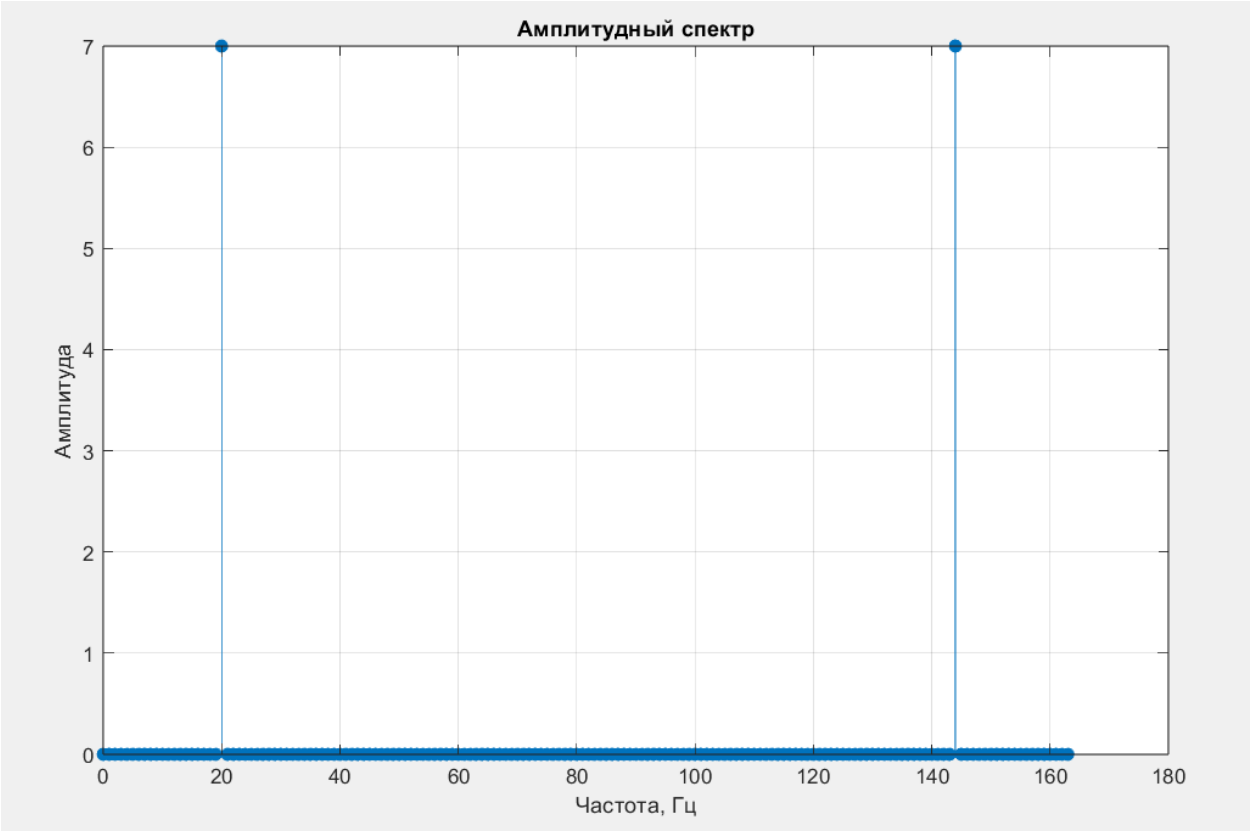


7. УВЕЛИЧЕНИЕ ЧАСТОТЫ ДИСКРЕТИЗАЦИИ В 4 РАЗА

Повторим пункты 4-6 с частотой дискретизации $F_s = 4 * 41$ Гц = 164 Гц, 8,2 отсчетов на период.




```
Количество отсчетов: 164
Первые 10 значений массива:
    3.5000   -1.6819   -5.9237   -6.8544   -3.9538    1.1568    5.6208    6.9430    4.3844   -0.6249
```



Исходный аналоговый сигнал имеет две частоты:

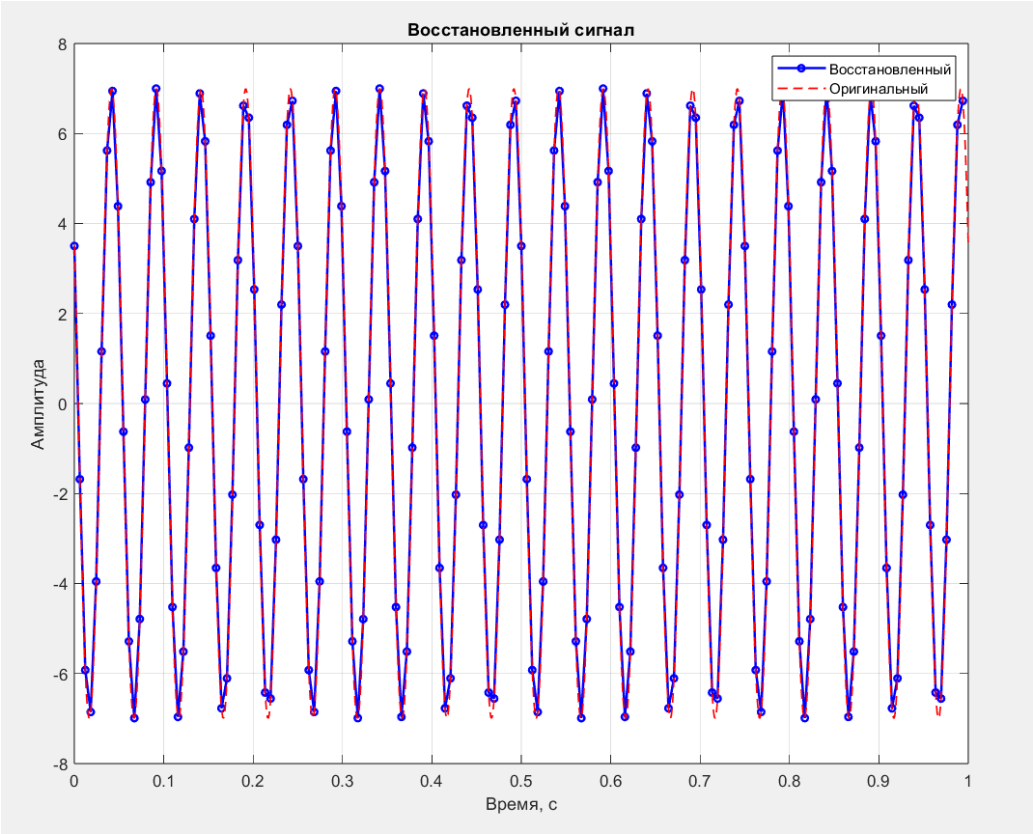
- +20 Гц
- −20 Гц

После дискретизации с F_s =164 Гц:

- +20 Гц - остаётся на 20 Гц
- −20 Гц - в цифровом спектре представляется как:
  −20+ F_s = −20+164=144 Гц (отрицательная частота, отображённая в положительную область)

Объем памяти для хранения: 164 * 8 байт = 1312 байт

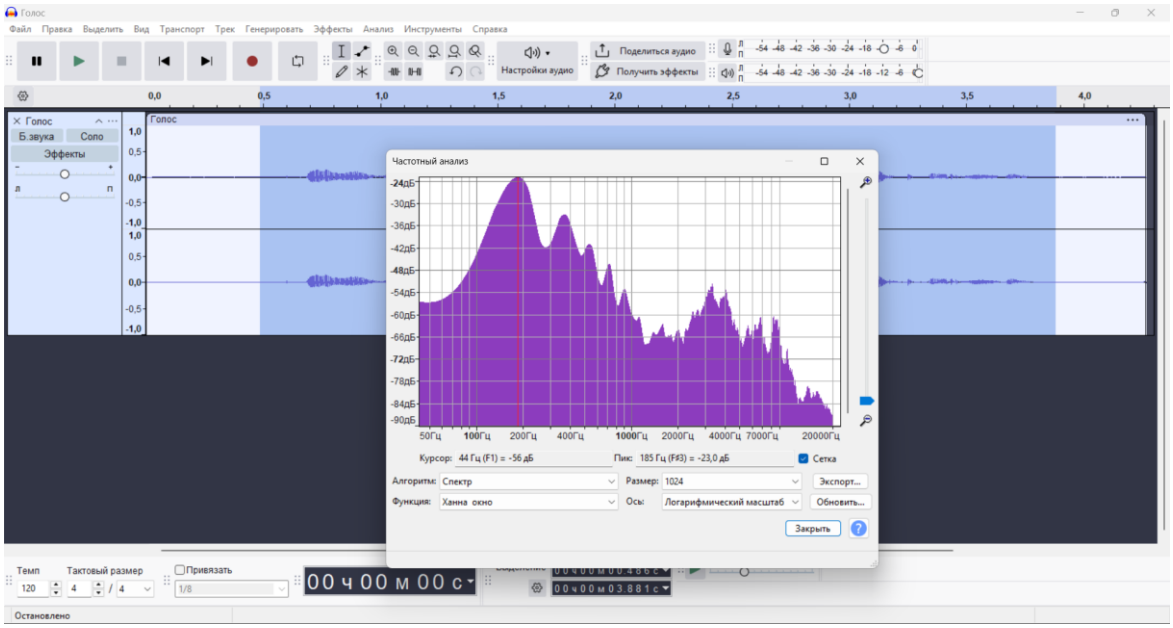
```
>> laba15
Объем памяти для массива 1312 байт
```

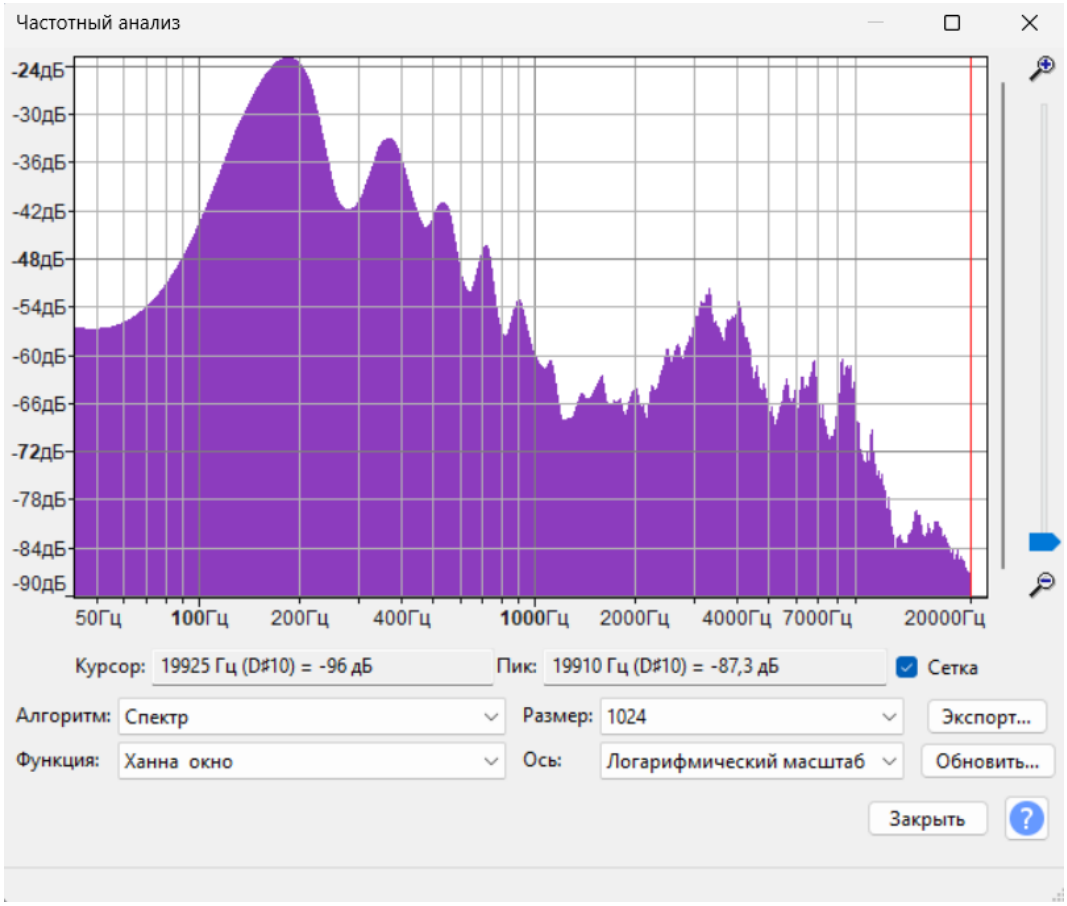


Восстановленный сигнал практически совпадает с оригиналом, просто не такой плавный.

8. АНАЛИЗ СПЕКТРА ГОЛОСА

График спектра построен в приложении Audacity





Максимальная частота значимого сигнала находится около 19,9 кГц (пик на 19910 Гц = -87 дБ)

По теореме Котельникова:

$F_s > 2* f_{max} = 2 \cdot 19910$ Гц = 39820 Гц

Если округлить, то минимальная требуемая частота дискретизации = 40 кГц.

9. СЧИТЫВАНИЕ ЗАПИСИ ГОЛОСА

Используем встроенные функции MATLAB

```
[y, Fs] = audioread('Голос.wav');   % y - массив отсчетов, Fs - частота дискретизации
```

f	*1x188160 double*
Fs	44100

Количество отчетов = 188160

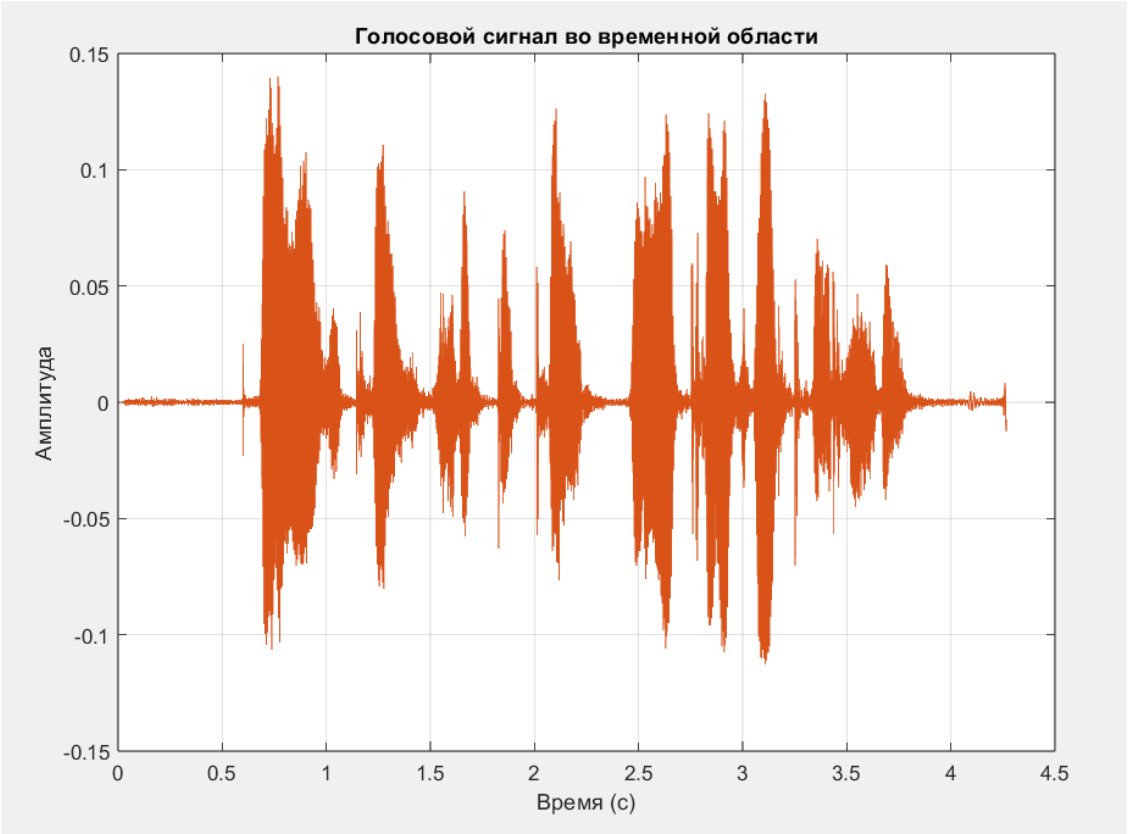
Частота дискретизации = 44100 Гц

10. ОПРЕДЕЛЕНИЕ ЧАСТОТЫ ДИСКРЕТИЗАЦИИ

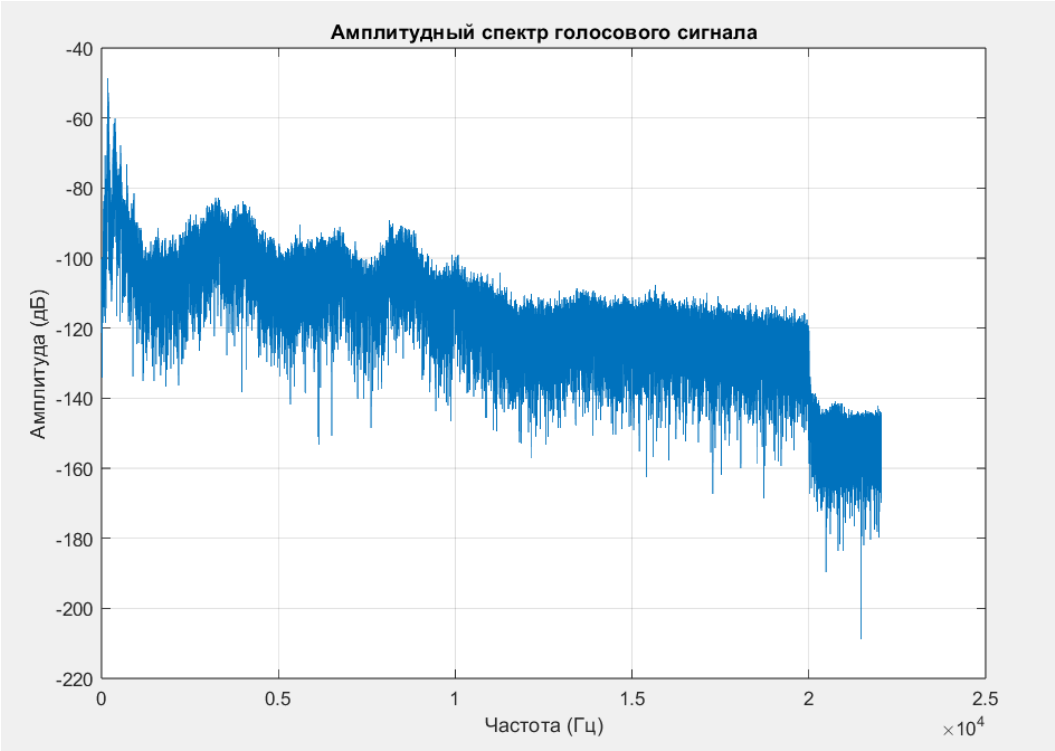
$$F_s = \frac{N}{T} = \frac{188160}{4.2667} = 44100 \text{ Гц}$$

Вычисленная частота дискретизации совпадает частотой дискретизации аудиофайла

```
Количество отсчетов 188160
Длительность записи 4.2667 сек
Частота дискретизации (из файла) 44100 Гц
Частота дискретизации (расчет) 44100 Гц
fx >>
```



Показывает, как изменяется громкость голоса во времени



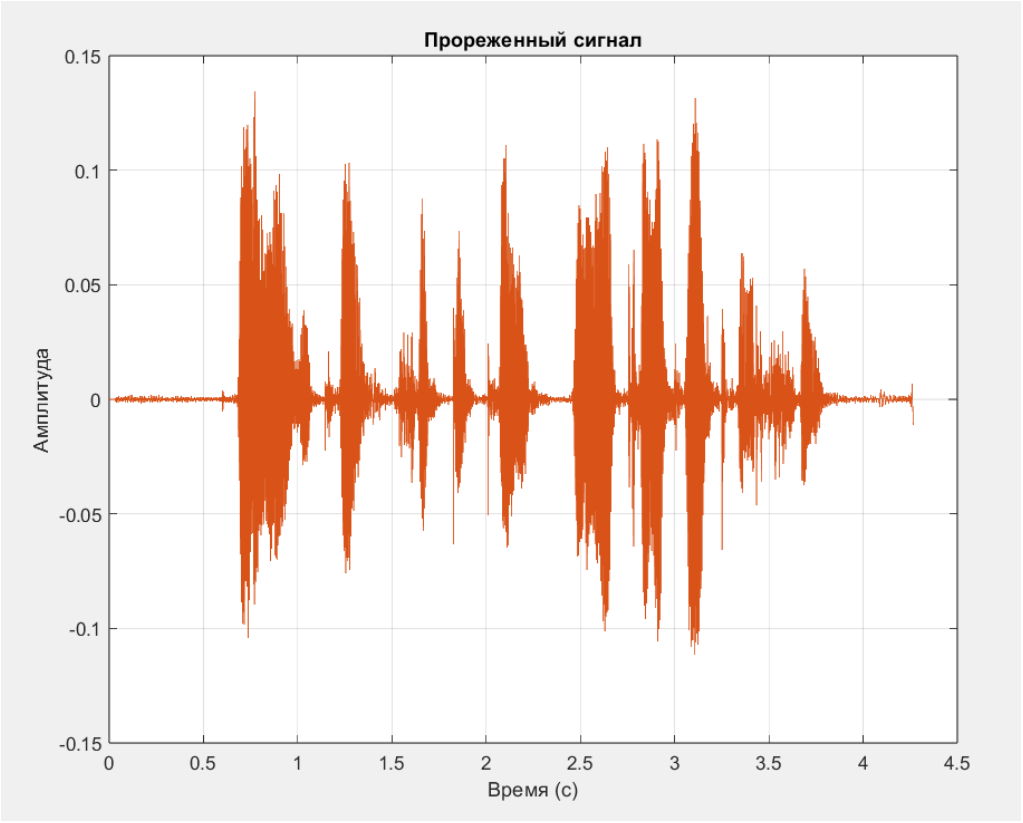
11. ПРОРЕЖИВАНИЕ МАССИВ ОТЧЕТОВ И ИСКАЖЕНИЕ ЗВУКА

Убираем каждый 10 отчет, F_s уменьшается в 10 раз.

```
y1 = downsample(y, 10);
Fs_new = Fs / 10;

zvuk = audioplayer(y1, Fs_new);
play(zvuk);
```

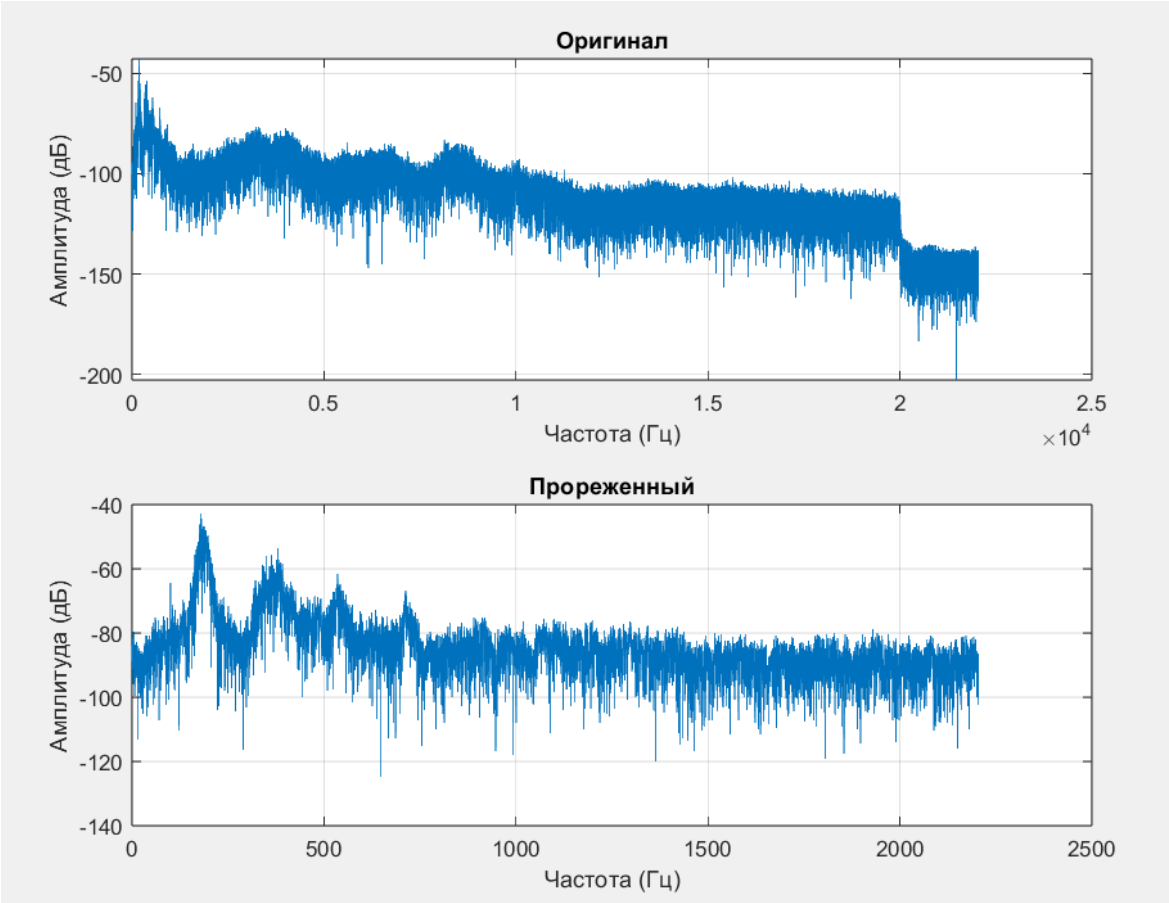
Качество звука ухудшилось, но слова все еще различимы.



12. ПРЯМОЕ ДИСКРЕТНОЕ ПРЕОБРАЗОВАНИЕ ФУРЬЕ ДЛЯ ОРИГИНАЛЬНОГО ЗВУЧАНИЯ И ДЛЯ ПРОРЕЖЕННОГО СИГНАЛА. ШИРИНА СПЕКТРА

Проведем ДПФ для оригинального и прореженного сигнала и построим графики амплитудного спектра

```
%ДПФ для оригинального и прореженного сигнала
Y_original = fft(y);
Y_downsampled = fft(y1);
```



Верхний график — спектр полного аудиосигнала широкий, с распределённой энергией, без ярких пиков.
Нижний график — спектр прореженного сигнала:

- Ярко выраженный пик на ~178 Гц
- Общий уровень амплитуды выше, чем в оригинале — потому что после прореживания сигнал нормирован не так, как оригинал
- Видны мелкие пики и шумы - остатки высокочастотных компонент, которые не были отфильтрованы перед прореживанием - эффект алиасинга ("шум" и аномалии, вызванные тем, что высокие частоты "перешли" вниз)

Ширина спектра:

1. Полная ширина спектра — от 0 Гц до частоты Найквиста (максимально возможная полоса);
   - Для оригинального сигнала (F_s=44.1 кГц): $f_N=2F_s$=22050 Гц
   - Для прореженного сигнала (F_{s_new}=4.41 кГц): f_{N_new}=24410=2205 Гц

2. Эффективная (значимая) ширина спектра – диапазон частот, на которых сигнал имеет значительную амплитуду

Без ввода значимого значения (визуально по графику):

Для прореженного сигнала основная энергия сосредоточена в диапазоне от 99 Гц до 711 Гц.

Значимая ширина амплитудного спектра = 711 – 99 = 612 Гц

13) ОЦЕНИТЕ ВЛИЯНИЕ РАЗРЯДНОСТИ АЦП НА СПЕКТР СИГНАЛА

Нужно смоделировать работу аналого-цифрового преобразователя (АЦП) с разной разрядностью (3, 4, 5 и 6 бит) и посмотреть, как это влияет на спектр сигнала.

АЦП - устройство, которое превращает аналоговый сигнал в цифровой. Разрядность АЦП - сколько «ступенек» он использует для описания амплитуды сигнала:

3 бита - всего 8 уровней - грубое приближение (много ошибок)

6 бит - 64 уровня - точнее (меньше ошибок)

Частота берется не целая, чтобы сигнал не был строго периодическим - чтобы ошибка квантования выглядела как шум, а не как дополнительные пики.

1. **Моделируется АЦП с разной разрядностью**

Для каждой разрядности (3, 4, 5, 6 бит):

а) Сигнал масштабируется в диапазон АЦП:

Например, при 3 битах - в диапазон 0-7.

При 6 битах - диапазон 0-63.

б) Выполняется округление (все значения «притягиваются» к ближайшему разрешённому уровню)

в) Сигнал возвращается в исходный масштаб (чтобы можно было сравнить с оригиналом)

2. **Вычисляется спектр (ДПФ)**

И для исходного, и для квантованного сигнала применяется быстрое преобразование Фурье (БПФ).

Спектр показывает, какие частоты есть в сигнале и с какой амплитудой.

3. **Строятся графики**

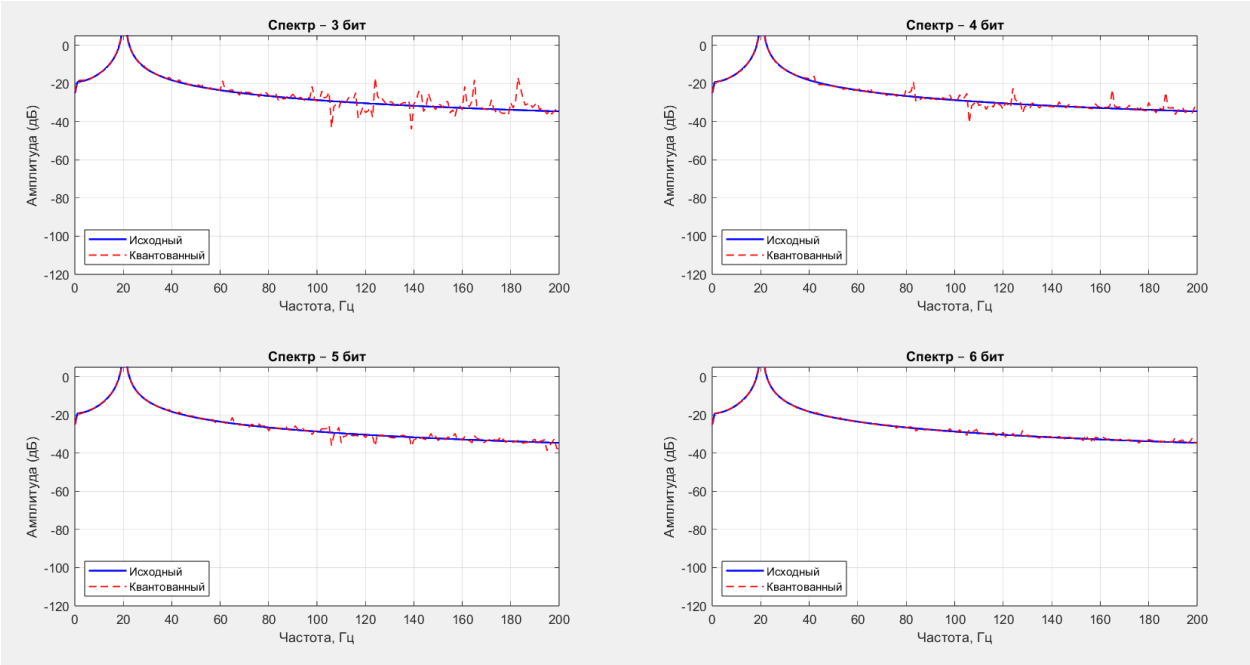
- Синяя линия - спектр идеального сигнала: один чистый
- Красная пунктирная линия - спектр после квантования.

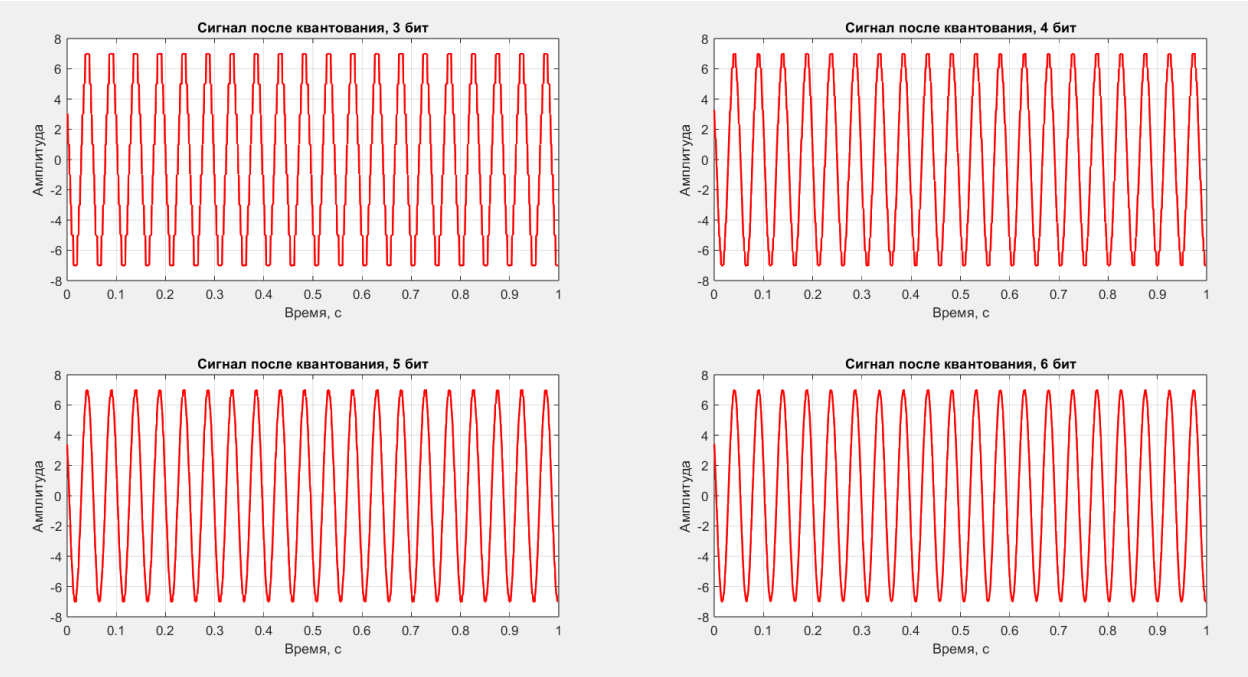
График в децибелах, чтобы видеть даже очень слабые компоненты (шум).

4. **Считается и выводится ошибка квантования**

ошибка = среднее (|исходный – квантованный|)

Чем выше разрядность - тем меньше ошибка.





Фигура 1- спектр (частотная область)

- Частотное содержание сигнала
- Виден шум квантования, уменьшающийся с ростом бит

Фигура 2 - сигнал во времени (временная область)

- При 3 битах чёткие ступеньки
- При 6 битах почти идеальная синусоида

Чем больше бит, тем меньше шум - тем чище спектр.

```
Средние абсолютные ошибки квантования:
   3 бит: 0.446370
   4 бит: 0.216876
   5 бит: 0.106424
   6 бит: 0.052799
>>
```

При 3 битах - всего 8 уровней, шаг квантования 14/8=1.75 - ошибка до ±0.875, средняя ошибка ~0.45

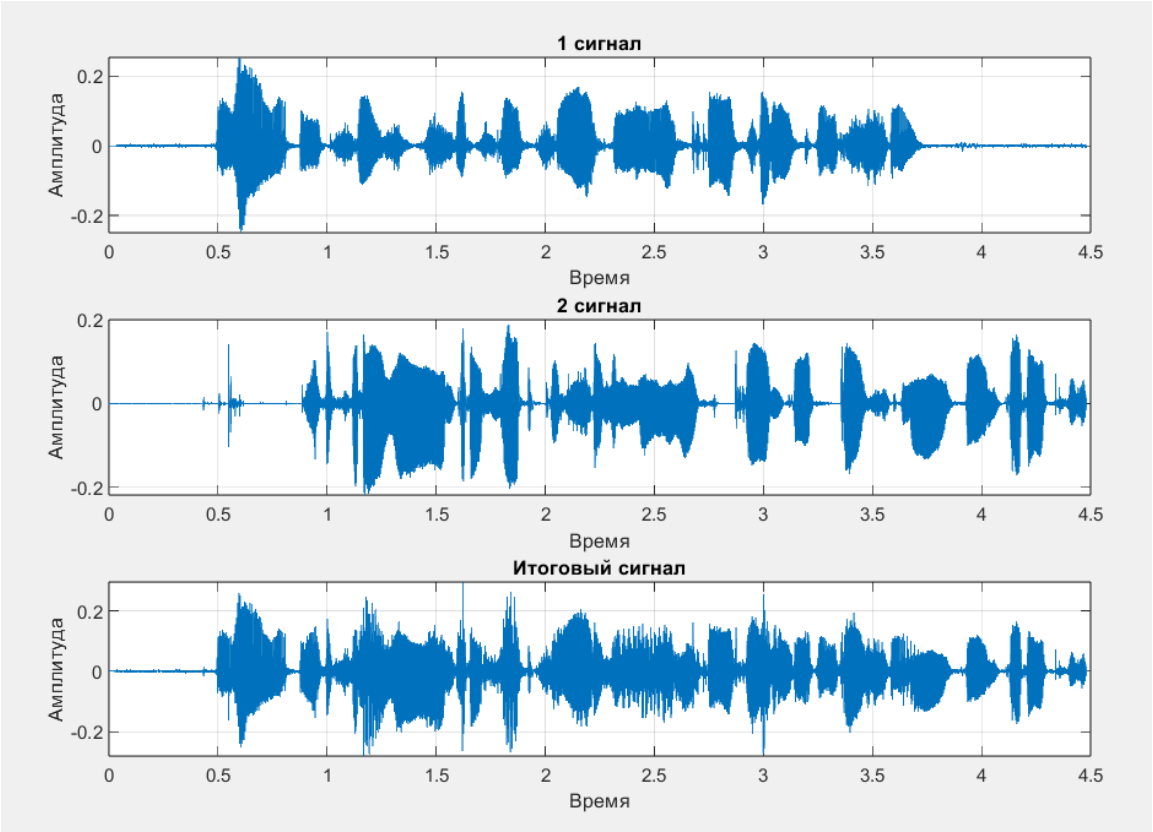
При 6 битах - 64 уровня, шаг 14/64=0.21875 - ошибка до ±0.109, средняя ошибка ~0.06

ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ 1

Взять 2 аудиозаписи разложить их на отчёты по временной области, сложить отчёты и воспроизвести полученную запись

Загружаем аудиофайлы, приводим их к одной длине и просто складываем каждые отсчеты

На полученном аудиофайле 2 голоса звучат параллельно, словно люди говорят одновременно.

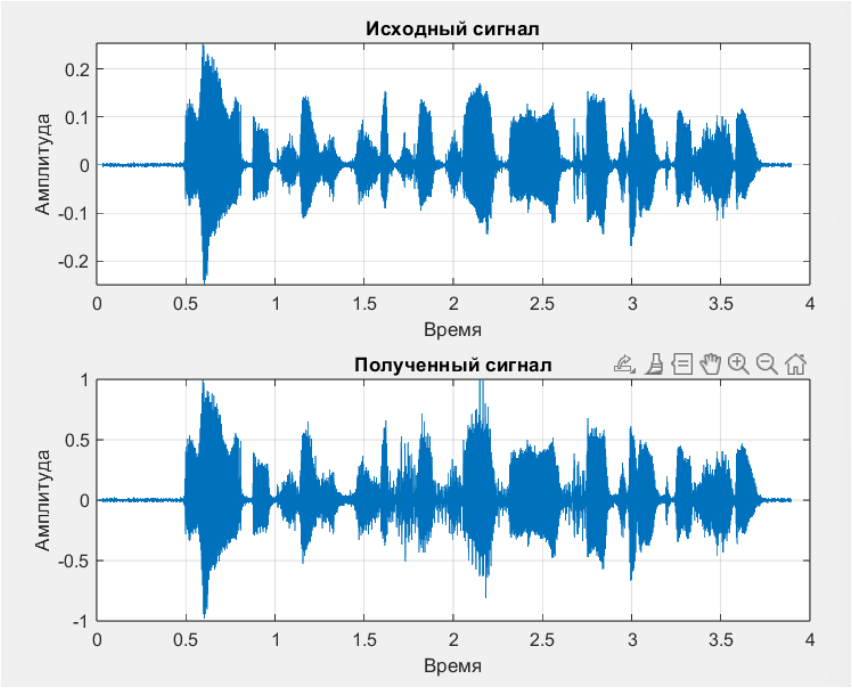


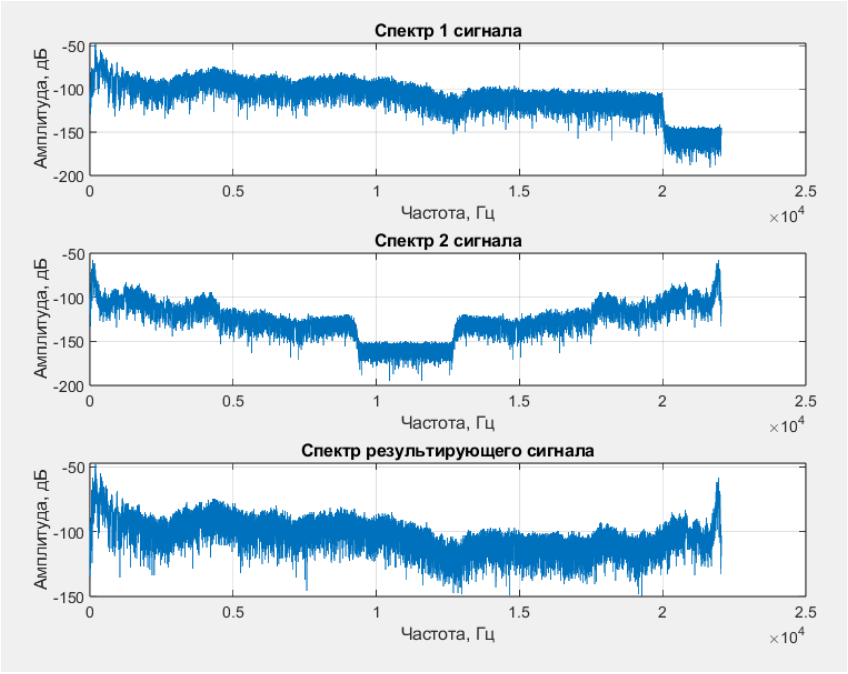
ДОПОЛНИТЕЛЬНОЕ ЗАДАНИЕ 2

Разложить аудиозапись на отчеты, считать отчеты другой аудиозаписи из файла, сложить отчёты в частотной области, перевести во временную и воспроизвести полученную запись

Складываем спектр голоса с аудиозаписи со спектром из файла в частотной области – получаем комбинированный спектр, который звучит как голос + шум

```
Размер y1: 197568x1
Прочитано 85872 строк
Размер y2: 171744x1
Размер y3: 171744x1
```





ИТОГ

Сложение во временной области = сложение в частотной области.

Преобразование Фурье линейно, поэтому:

ifft(fft(y1) + fft(y2)) ≡ y1 + y2

Оба варианта звучат как смешанный звук.

ВЫВОДЫ

1. Теорема Котельникова

Подтверждено, что для корректного восстановления сигнала частота дискретизации должна быть не менее удвоенной максимальной частоты в спектре. Соблюдение этого условия гарантирует отсутствие наложения спектров.

2. Преобразование Фурье

Прямое и обратное преобразования Фурье являются эффективным инструментом для анализа частотного состава сигналов и преобразования между временной и частотной областями.

3. Влияние разрядности АЦП

С увеличением разрядности АЦП ошибка квантования значительно уменьшается:

- 3 бита - значительные искажения спектра
- 6 бит - спектр практически соответствует исходному сигналу

4. Ошибка дискретизации возникает при нарушении теоремы Котельникова. Ошибка квантования зависит от разрядности АЦП и уменьшается с её увеличением

5. Спектральный анализ

Показано, что периодические сигналы (например, $\cos(10\pi t)$) имеют дискретный спектр с четко выраженными гармониками, в отличие от сигналов после квантования, где появляются дополнительные спектральные компоненты.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1) Для чего используются прямое и обратное преобразование Фурье?

- Прямое преобразование Фурье - переводит сигнал из временной области в частотную. Позволяет анализировать, какие частотные компоненты присутствуют в сигнале и с какой амплитудой.

- Обратное преобразование Фурье - переводит сигнал из частотной области обратно во временную. Позволяет синтезировать сигнал по известным частотным компонентам.

2) Что такое ошибка квантования и дискретизации?

- Ошибка дискретизации - возникает из-за недостаточно высокой частоты дискретизации (нарушение теоремы Котельникова), приводит к наложению спектров (алиасингу).
- Ошибка квантования - разница между истинным значением аналогового сигнала и ближайшим уровнем квантования. Возникает из-за конечной разрядности АЦП.

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

- Телефонная связь 8 бит
- Стандартное качество 12-16 бит
- Студийное качество 16-24 бита

4) Как математически получить дискретные отсчеты непрерывного сигнала?

Дискретные отсчеты получаются путем умножения непрерывного сигнала x(t) на последовательность дельта-функций:

$x[n] = x(nT) = x(t) \cdot \sum\delta(t - nT)$,

где T - период дискретизации, n - номер отсчета.

5) Какой спектр у периодического сигнала $\sin(10\pi t+\pi/2)$?

Сигнал $\sin(10\pi t + \pi/2) = \cos(10\pi t)$

Частота $f = 10\pi/(2\pi) = 5$ Гц

Спектр - одна спектральная линия на частоте 5 Гц (дельта функция)

6) Что такое быстрое преобразование Фурье (БПФ)?

БПФ - алгоритм для эффективного вычисления дискретного преобразования Фурье. Снижает вычислительную сложность с $O(N^2)$ до $O(N \cdot \log N)$, где N - количество отсчетов.

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

Согласно теореме Котельникова, минимальная частота дискретизации должна быть как минимум в 2 раза выше максимальной частоты сигнала:

$Fs \geq 2*f_{max}$

где Fs - максимальная частота в спектре сигнала.

КОД

1 пункт

```
f = 20;       %частота
A = 7;       %амплитуда
phi = pi/3;     %фаза

t0 = 0; %начальное время
t1 = 1; %конечное время

t = linspace(t0, t1, 1000);

y = A * cos(2*pi*f*t + phi);

figure;
plot(t, y, 'b-', 'LineWidth', 2);
grid on;
title('y(t) = 7cos(2πft + π/3), f = 20 Гц');
xlabel('Время t, с');
ylabel('Амплитуда A');
xlim([t0, t1]);
```

4 пункт

```
f = 20;       %частота
A = 7;       %амплитуда
phi = pi/3;     %фаза
fs = 40;
T = 1;
f_max = f;
Fs = 2 * f_max;
N = Fs * T;  %количество отсчетов
t_dig = 0:1/Fs:T-1/Fs;  %моменты дискретизации
y_dig = 7 * cos(2 * pi * f * t_dig + pi/3);

t_1 = 0:0.001:T;  % более частые точки для плавного графика
y_1 = A * cos(2 * pi * f * t_1 + phi);

fprintf('Количество отсчетов: %d\n', N);
fprintf('Первые 10 значений массива:\n');
disp(y_dig(1:10));

figure;
```

```
stem(t_dig, y_dig, 'filled', 'LineWidth', 1.5);
hold on;
plot(t_1, y_1, 'r-', 'LineWidth', 1);
xlabel('Время, с');
ylabel('Амплитуда');
title('Оцифрованный сигнал (Fs = 40 Гц)');
legend('Отсчеты', 'Непрерывный сигнал');
grid on;
```

6 пункт

```
f = 20;       %частота
A = 7;       %амплитуда
phi = pi/3;     %фаза
fs = 40;
T = 1;
f_max = f;
Fs = 2 * f_max;

t_1 = 0:0.001:T;
y_1 = A * cos(2 * pi * f * t_1 + phi);

t_dig = 0:1/Fs:T-1/Fs;  %моменты дискретизации
y_dig = 7 * cos(2 * pi * f * t_dig + pi/3);

N = length(y_dig);   %матрица ДПФ
Y_dft = zeros(1, N);

for k = 1:N
    for n = 1:N
        Y_dft(k) = Y_dft(k) + y_dig(n) * exp(-1j*2*pi*(k-1)*(n-1)/N);
    end
end

figure;
plot(t_dig, y_dig, 'b-o', 'LineWidth', 1.5, 'MarkerSize', 4);
hold on;
plot(t_1, y_1, 'r--', 'LineWidth', 1);
xlabel('Время, с');
ylabel('Амплитуда');
title('Восстановленный сигнал');
legend('Восстановленный', 'Оригинальный');
grid on;
```

10 пункт

```
clear; clc; close all;

[y, Fs] = audioread('Голос.wav');   % y - массив отсчетов, Fs - частота дискретизации
N = size(y, 1);                % количество отсчетов
T = N / Fs;              % длительность записи в секундах

t = (0:N-1)/Fs;            % временная ось
figure;
plot(t, y);
xlabel('Время (с)');
ylabel('Амплитуда');
title('Голосовой сигнал во временной области');
grid on;
```

```
Y = fft(y);          % БПФ сигнала
L = length(Y);
f = (0:L-1)*(Fs/L);      % ось частот
P = abs(Y)/L;           % нормированный спектр

figure;
plot(f(1:floor(L/2)), 20*log10(P(1:floor(L/2))));
xlabel('Частота (Гц)');
ylabel('Амплитуда (дБ)');
title('Амплитудный спектр голосового сигнала');
grid on;

Fs_calc = N / T;     % пересчитанная частота дискретизации

disp(['Количество отсчетов ', num2str(N)]);
disp(['Длительность записи ', num2str(T), ' сек']);
disp(['Частота дискретизации (из файла) ', num2str(Fs), ' Гц']);
disp(['Частота дискретизации (расчет) ', num2str(Fs_calc), ' Гц']);
```

11 пункт

```
clear; clc; close all;

[y, Fs] = audioread('Голос.wav');   % y - массив отсчетов, Fs - частота дискретизации
N = size(y, 1);                % количество отсчетов
T = N / Fs;               % длительность записи в секундах

y1 = downsample(y, 30);
Fs_new = Fs / 30;

zvuk = audioplayer(y1, Fs_new);
play(zvuk);

figure;
plot((0:length(y1)-1)/Fs_new, y1);
xlabel('Время (с)');
ylabel('Амплитуда');
title('Прореженный сигнал');
grid on;
```

12 пункт

```
clear; clc; close all;

[y, Fs] = audioread('Голос.wav');
if size(y,2) > 1, y = y(:,1); end
y = y(:);

% Прореживание
M = 10;
y1 = downsample(y, M);
Fs1 = Fs / M;

% ДПФ
Y = fft(y);         Y1 = fft(y1);
N = length(y);      N1 = length(y1);

% Амплитудные спектры (односторонние)
```

```
P = abs(Y(1:N/2+1)) / N;      P1 = abs(Y1(1:N1/2+1)) / N1;
P(2:end-1) = 2*P(2:end-1);    P1(2:end-1) = 2*P1(2:end-1);

% Частотные оси
f = (0:N/2) * Fs/N;
f1 = (0:N1/2) * Fs1/N1;

% Графики
figure;
subplot(2,1,1); plot(f, 20*log10(P + eps));
xlabel('Частота (Гц)'); ylabel('Амплитуда (дБ)'); title('Оригинал'); grid on;

subplot(2,1,2); plot(f1, 20*log10(P1 + eps));
xlabel('Частота (Гц)'); ylabel('Амплитуда (дБ)'); title('Прореженный'); grid on;

thr = -200;
db = 20*log10(P + eps);
db1 = 20*log10(P1 + eps);


% Оригинал
idx = db >= thr;
w_orig = idx(end) * f(find(idx,1,'last')) - idx(1) * f(find(idx,1,'first'));
if ~any(idx), w_orig = f(db == max(db)); end

% Прореженный
idx1 = db1 >= thr;
w_down = idx1(end) * f1(find(idx1,1,'last')) - idx1(1) * f1(find(idx1,1,'first'));
if ~any(idx1), w_down = f1(db1 == max(db1)); end

% Вывод
fprintf('Ширина спектра (оригинал): %.1f Гц\n', w_orig);
fprintf('Ширина спектра (прореженный): %.1f Гц\n', w_down);
```

13 пункт

```
function adc_analysis_full()
    clear; clc; close all;

    fs = 1000;
    t = 0:1/fs:1-1/fs;  % ровно 1 секунда
    f0 = 20.37;         % нецелая частота - шум квантования
    x = 7 * cos(2*pi*f0*t + pi/3);

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

    [f_orig, X_orig] = calculate_spectrum(x, fs);
    errors = zeros(size(bit_depths));
    x_quantized_all = cell(1, length(bit_depths));  % для хранения сигналов

    figure('Name', 'Спектры квантованного сигнала', 'Position', [100, 100, 1000,
800]);
    for i = 1:length(bit_depths)
        bits = bit_depths(i);

        % Квантование
        x_quantized = quantize_signal(x, bits);
        x_quantized_all{i} = x_quantized;

        % Ошибка
```

```
        errors(i) = mean(abs(x - x_quantized));

        % Спектр
        [f_quant, X_quant] = calculate_spectrum(x_quantized, fs);

        % График спектра
        subplot(2, 2, i);
        plot(f_orig, 20*log10(abs(X_orig) + eps), 'b-', 'LineWidth', 1.5); hold on;
        plot(f_quant, 20*log10(abs(X_quant) + eps), 'r--', 'LineWidth', 1);
        xlim([0, 200]);
        ylim([-120, 5]);
        title(['Спектр – ', num2str(bits), ' бит'], 'FontSize', 11);
        xlabel('Частота, Гц');
        ylabel('Амплитуда (дБ)');
        legend('Исходный', 'Квантованный', 'Location', 'southwest');
        grid on;
    end

    fprintf('Средние абсолютные ошибки квантования:\n');
    for i = 1:length(bit_depths)
        fprintf('  %d бит: %.6f\n', bit_depths(i), errors(i));
    end

    figure('Name', 'Сигналы после квантования (временная область)', ...
       'Position', [100, 100, 1000, 800]);
for i = 1:length(bit_depths)
    bits = bit_depths(i);
    subplot(2, 2, i);  % <-- теперь 2x2
    plot(t, x_quantized_all{i}, 'r-', 'LineWidth', 1.5);
    title(['Сигнал после квантования, ', num2str(bits), ' бит'], 'FontSize', 11);
    xlabel('Время, с');
    ylabel('Амплитуда');
    grid on;
    xlim([0, 1]);  % показываем всю секунду
end
end

function x_quantized = quantize_signal(x, bits)
    max_level = 2^bits - 1;
    x_min = min(x);
    x_max = max(x);
    x_scaled = (x - x_min) / (x_max - x_min) * max_level;
    x_quantized = round(x_scaled);
    x_quantized = max(0, min(max_level, x_quantized));
    x_quantized = x_quantized / max_level * (x_max - x_min) + x_min;
end

function [f, X] = calculate_spectrum(x, fs)
    N = length(x);
    X = fft(x);
    X = X / N;
    if mod(N, 2) == 0
        X(2:end-1) = 2 * X(2:end-1);
    else
        X(2:end) = 2 * X(2:end);
    end
    f = (0:N-1) * fs / N;
    if mod(N, 2) == 0
        X = X(1:N/2+1);
        f = f(1:N/2+1);
    else
        X = X(1:(N+1)/2);
```

```
        f = f(1:(N+1)/2);
    end
end

adc_analysis_full();
```

Доп.задание 1

```
clear; clc; close all;

[y1, Fs1] = audioread('Голос.wav');   % y - массив отсчетов, Fs - частота
дискретизации
[y2, Fs2] = audioread('voice.wav');

N = min(length(y1), length(y2));
t = (0:N-1) / Fs1;
y1 = y1(1:N);
y2 = y2(1:N);

y3 = y1 + y2;

zvuk = audioplayer(y3, Fs1);
play(zvuk);

figure;

subplot(3,1,1);
plot(t, y1);
xlabel('Время');
ylabel('Амплитуда');
title('1 сигнал');
grid on;

subplot(3,1,2);
plot(t, y2);
xlabel('Время');
ylabel('Амплитуда');
title('2 сигнал');
grid on;

subplot(3,1,3);
plot(t, y3);
xlabel('Время');
ylabel('Амплитуда');
title('Итоговый сигнал');
grid on;
```

Доп.задание 2

```
clear; clc; close all;

[y1,Fs1] = audioread('Голос.wav');
fprintf('Размер y1: %dx%d\n', size(y1));

f = fopen('sample.txt', 'r');
if f == -1
    error('Не получается открыть sample.txt');
```

```
end

try
    lines = {};
    while ~feof(f)
        line = fgetl(f);
        lines{end+1} = line;
    end
    fclose(f);

    fprintf('Прочитано %d строк\n', length(lines));

    real1 = [];
    imag1 = [];
    real2 = [];
    imag2 = [];

    for i = 1:length(lines)
        line = lines{i};
        parts = strsplit(line); %делим по пробелам
        numbers = [];

        for j = 1:length(parts)
            part = strtrim(parts{j});
            if ~isempty(part)
                clean_part = strrep(part, 'i', '');  %удаляем 'i' и '+'
                clean_part = strrep(clean_part, '+', '');
                num = str2double(clean_part); %строку в число
                if ~isnan(num)
                    numbers(end+1) = num;
                end
            end
        end

        if length(numbers) >= 4
            real1(end+1) = numbers(1);
            imag1(end+1) = numbers(2);
            real2(end+1) = numbers(3);
            imag2(end+1) = numbers(4);
        end
    end

     %комплексные массивы
    left = real1(:) + 1i * imag1(:);
    right = real2(:) + 1i * imag2(:);

catch ME
    fclose(f);
    error('Не получается прочитать данные из sample.txt');
end

y2 = (left + right)/2;
y2 = [y2; y2]; %удваиваем сигнал

fprintf('Размер y2: %dx%d\n', size(y2));

N = min(length(y1), length(y2));

if N == 0
    error('Один из сигналов пустой');
end
```

```
y1 = y1(1:N);
y2 = y2(1:N);

y2_time = real(ifft(y2));
y2_time = y2_time/ max(abs(y2_time));

Y1 = fft(y1);
Y3 = y2 + Y1;

y3 = real(ifft(Y3));  %убираем мнимую часть
y3 = y3 / max(abs(y3));  %нормализация

fprintf('Размер y3: %dx%d\n', size(y3));

zvuk = audioplayer(y3, Fs1);
play(zvuk);
%(y3, Fs1);
%(y2_time, Fs1);

figure;

subplot(2,1,1);
t = (0:N-1)/Fs1;
plot(t,y1);
title('Исходный сигнал');
xlabel('Время');
ylabel('Амплитуда');
grid on;

subplot(2,1,2);
plot(t,y3);
title('Полученный сигнал');
xlabel('Время');
ylabel('Амплитуда');
grid on;

figure;

subplot(3,1,1);
F = (0:N-1)*Fs1/N;
P1 = abs(Y1)/N;
plot(F(1:floor(N/2)),  20*log10(P1(1:floor(N/2))));
title('Спектр 1 сигнала');
xlabel('Частота, Гц');
ylabel('Амплитуда, дБ');
grid on;

subplot(3,1,2);
F = (0:N-1)*Fs1/N;
P2 = abs(y2)/N;
plot(F(1:floor(N/2)), 20*log10(P2(1:floor(N/2))));
title('Спектр 2 сигнала');
xlabel('Частота, Гц');
ylabel('Амплитуда, дБ');
grid on;

subplot(3,1,3);
F = (0:N-1)*Fs1/N;
P3 = abs(Y3)/N;
plot(F(1:floor(N/2)), 20*log10(P3(1:floor(N/2))));
title('Спектр результирующего сигнала');
xlabel('Частота, Гц');
```

```
ylabel('Амплитуда, дБ');
grid on;
```