

ГУАП

КАФЕДРА № 21

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

профессор, д-р техн. наук

должность, уч. степень, звание

подпись, дата

Л.Б.Фридман

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2

Преобразование Фурье. Спектры сигналов.

по курсу:

Цифровая обработка сигналов

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № _____ 2315

подпись, дата

Тараканов Д.Д.

инициалы, фамилия

Санкт-Петербург 2025

Вариант 1

Задание 1

```
function result = delta(n)
result = (n == 0);
end
function result = unit_step(n)
result = (n >= 0);
end
a1 = 1.2; b1 = 4; n1 = 1:20;
```

```
figure
x1 = a1 * delta(n1 - b1);
stem(x1, "m", "filled")
grid("on")
title('a1 * delta(n1 - b1));
```

```
ampspec = fft(x1);
```

```
figure
plot(abs(ampspec));
```

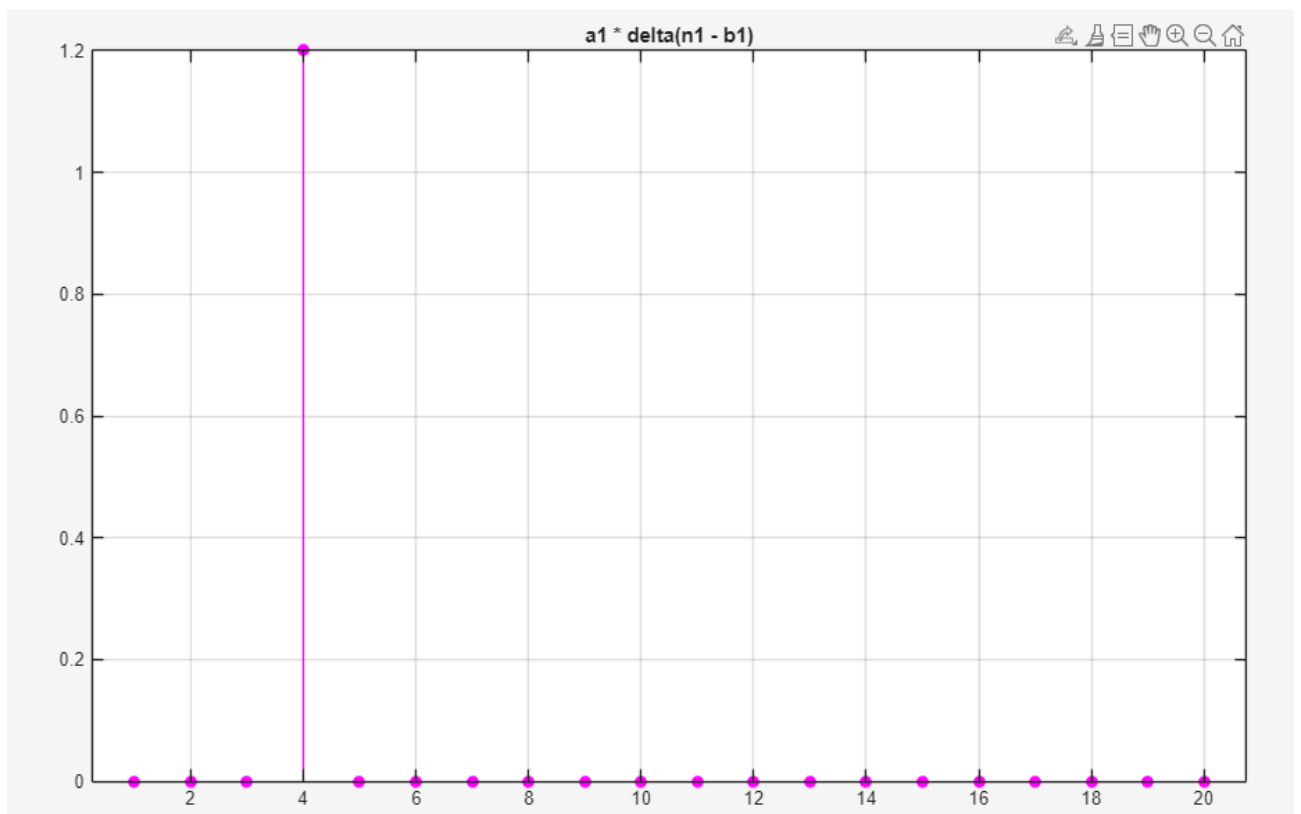


Рисунок 1 - График дельта-импульса

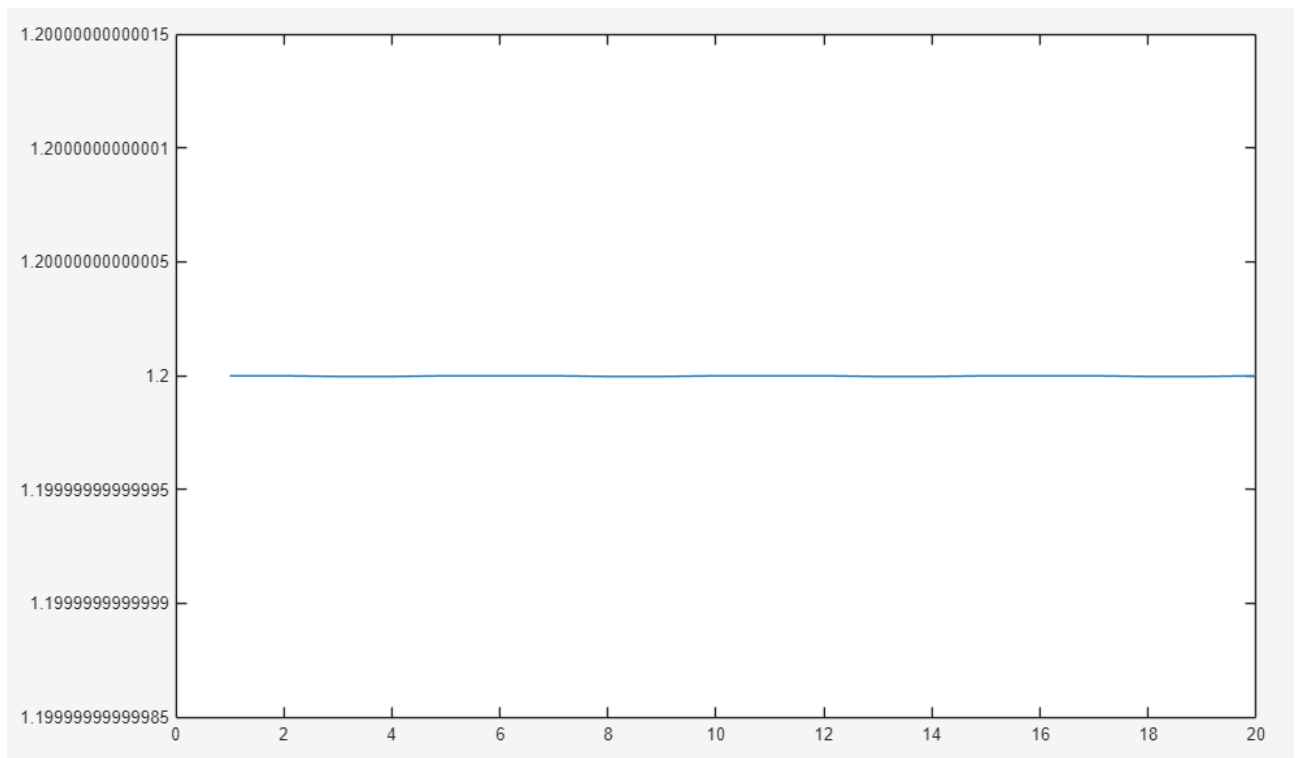


Рисунок 2 - График амплитудного спектра дельта-импульса

Задание 2

% Параметры, Вариант 1

t0_us = 5; % мкс

T_imp_us = 2; % мкс

Td_ns = 200; % нс

t_range_us = [0, 20]; % мкс

% Преобразование в секунды для согласованности

t0 = t0_us * 1e-6;

T_imp = T_imp_us * 1e-6;

Td = Td_ns * 1e-9;

% Временная ось

t_start = t_range_us(1) * 1e-6;

t_end = t_range_us(2) * 1e-6;

t = t_start:Td:t_end;

x = zeros(size(t)); % Инициализация сигнала нулями

% Установка значения 1 в интервале [t0, t0 + T_imp]

x(t >= t0 & t < (t0 + T_imp)) = 1;

figure;

plot(t * 1e6, x, 'LineWidth', 2);

title('График сигнала x(t)');

```

xlabel('Время, мкс');
ylabel('Амплитуда');
grid on;
ylim([-0.1, 1.1]);

```

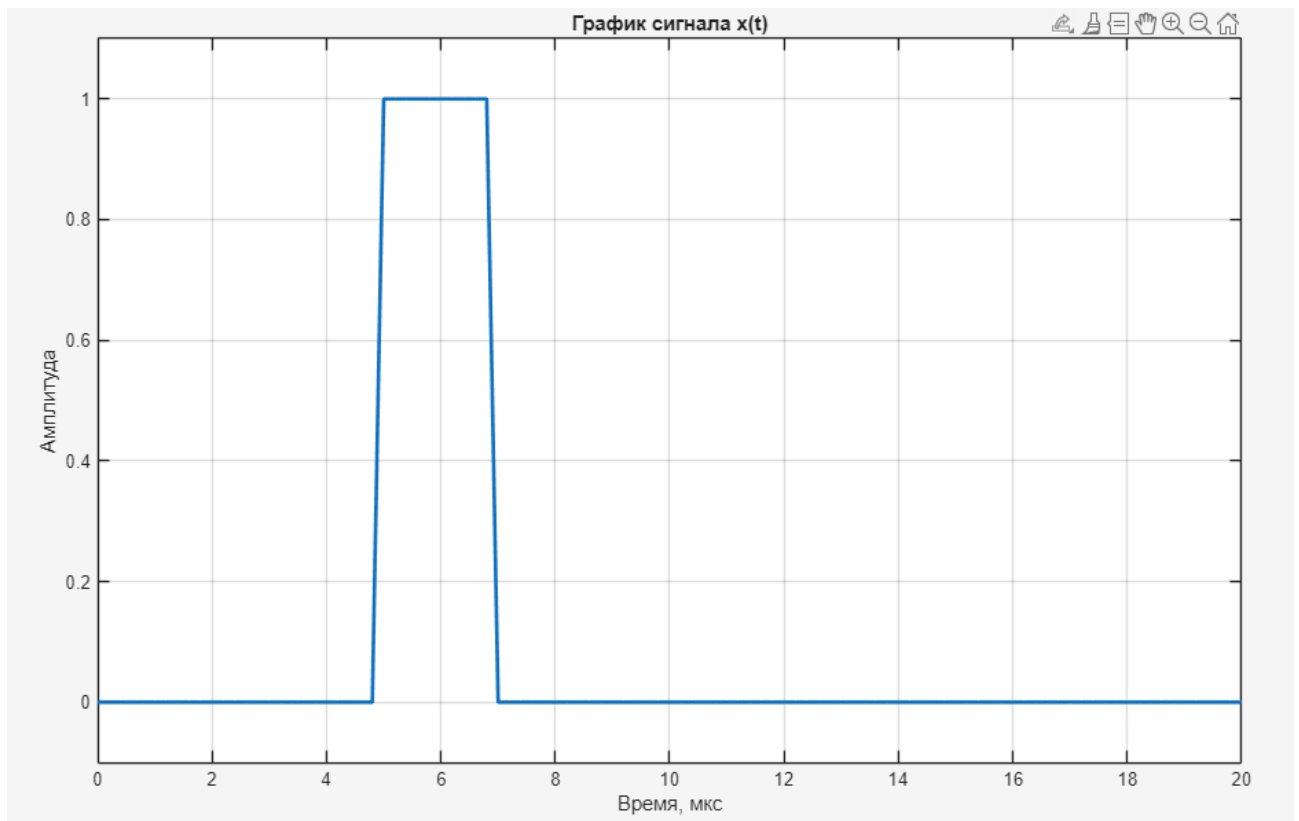


Рисунок 3 - График прямоугольного импульса.

```

N = length(x); % Количество отсчетов
X_f = fftshift(fft(x)) / N; % Вычисление БПФ и нормализация
frequencies = (-N/2:N/2-1) * (1/(N*Td)); % Частотная ось (Гц)

amplitude_spectrum = abs(X_f); % Амплитудный спектр

figure;
plot(frequencies / 1e6, amplitude_spectrum, 'LineWidth', 2);
title('Амплитудный спектр сигнала |X(f)|');
xlabel('Частота, МГц');
ylabel('Модуль амплитуды');
grid on;
xlim([-10/T_imp*1e-6, 10/T_imp*1e-6]); % Ограничение оси X для лучшего
отображения основного лепестка

```

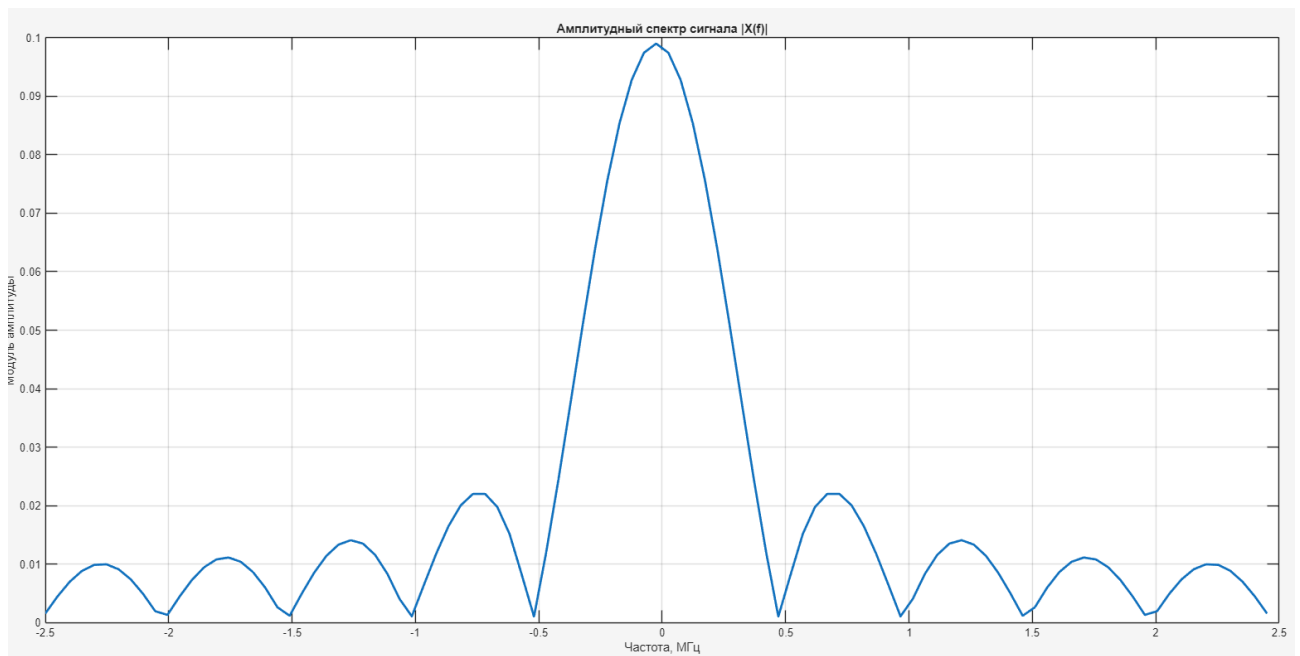


Рисунок 4 - Амплитудный спектр заданного сигнала.

Задание 3

```
f_signal_kHz = 0.8;    % Частота сигнала, кГц
phi_deg = 45;         % Начальная фаза, градусы
amplitude = 40;       % Нормализованная амплитуда, ед.
fs_kHz = 8;           % Частота дискретизации, кГц
T_start_ms = 0;       % Начальное время, мс
T_end_ms = 12.5;      % Конечное время, мс

% Преобразование единиц измерения в СИ (секунды, герцы, радианы)
f_signal = f_signal_kHz * 1e3;
fs = fs_kHz * 1e3;
T_start = T_start_ms * 1e-3;
T_end = T_end_ms * 1e-3;
phi_rad = deg2rad(phi_deg); % Перевод градусов в радианы

% 1. Формирование сигнала
dt = 1/fs; % Период дискретизации
% Создаем временной вектор от начального до конечного времени с шагом dt
t = T_start:dt:T_end;

% Формируем синусоидальный сигнал
% Уравнение:  $A * \sin(2\pi * f * t + \phi)$ 
signal = amplitude * sin(2 * pi * f_signal * t + phi_rad);

% Построение графика сигнала во временной области (опционально для
```

```

проверки)
figure;
plot(t*1e3, signal); % Время в мс для читаемости
title('Исходный гармонический сигнал');
xlabel('Время (мс)');
ylabel('Амплитуда');
grid on;

```

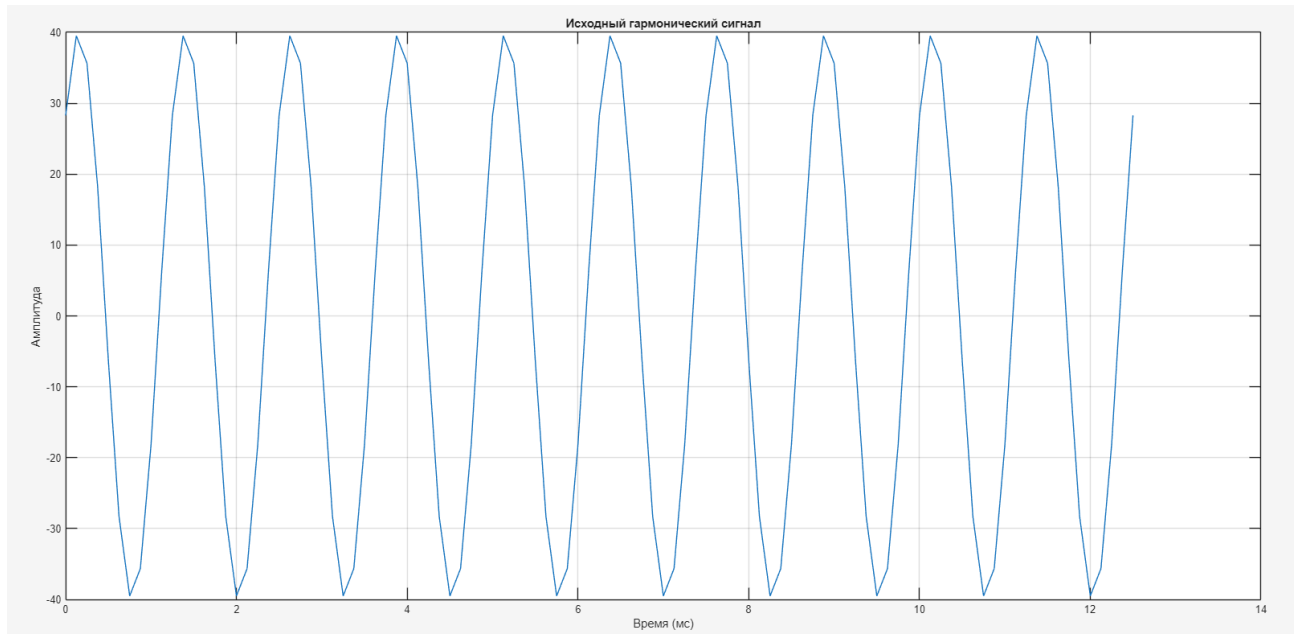


Рисунок 5 – График исходного сигнала

% 2. Вычисление спектра

```
X_freq = fft(signal);
```

```
% Расчет вектора частот для корректного отображения спектра
```

```
N = length(signal);
```

```
% f_step - шаг по частоте
```

```
f_step = fs / N;
```

```
% Создаем вектор частот от 0 до fs/2
```

```
frequencies = 0:f_step:(fs/2 - f_step);
```

```
% Амплитудный спектр: берем модуль (abs) и нормализуем (делим на N)
```

```
amplitude_spectrum = abs(X_freq) / N;
```

```
% Так как спектр симметричен, берем только первую половину (до частоты Найквиста)
```

```
half_spectrum = amplitude_spectrum(1:N/2);
```

```
% Отображение на графике амплитудного спектра
```

```

figure;
% Отображаем частоты в кГц для удобства
plot(frequencies/1e3, half_spectrum);
title(['Амплитудный спектр (N=', num2str(N), ' точек)']);
xlabel('Частота (кГц)');
ylabel('Нормализованная амплитуда');
grid on;
% Ограничиваем ось X до частоты Найквиста fs/2, если необходимо
xlim([0 fs/2000]);

```

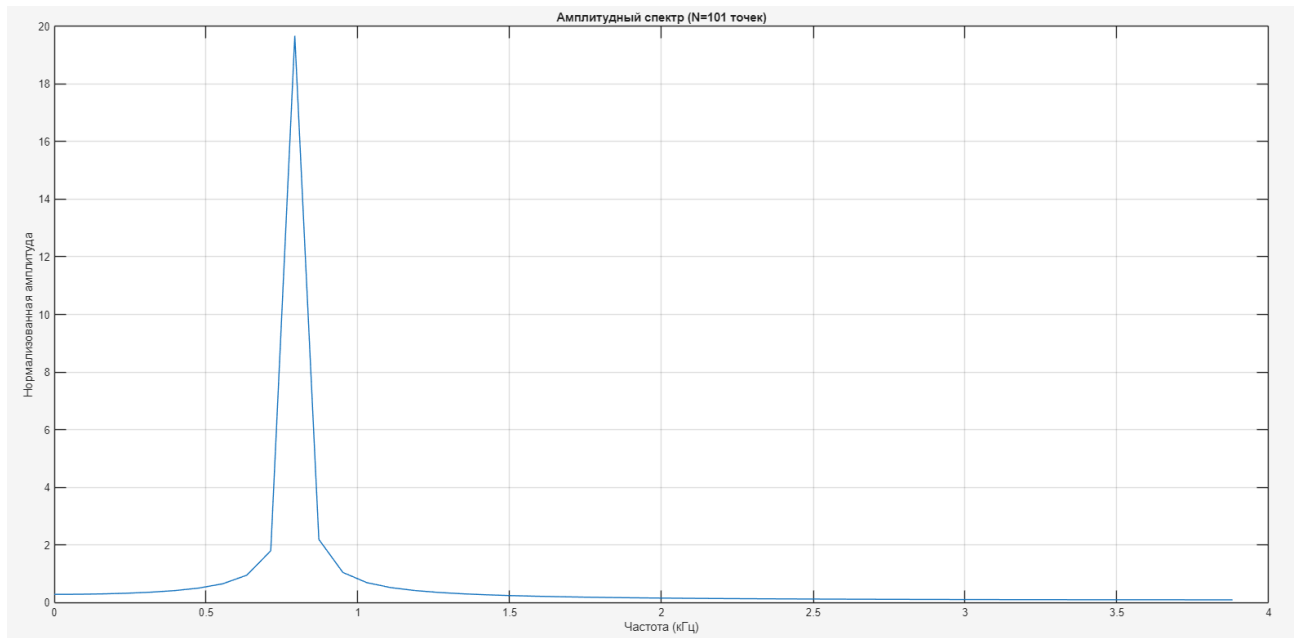


Рисунок 6 – Амплитудный спектр сигнала.

```

% Дополнение количества точек нулями до 1000
N_target = 1000;
% Используем fft с вторым аргументом, чтобы выполнить дополнение
автоматически
X_freq = fft(signal, N_target);

% Расчет вектора частот для корректного отображения спектра
% N - фактическое количество точек после дополнения (1000)
N = length(X_freq);
% f_step - шаг по частоте
f_step = fs / N;
% Создаем вектор частот от 0 до fs/2
frequencies = 0:f_step:(fs/2 - f_step/2);

% Амплитудный спектр: берем модуль (abs) и нормализуем (делим на N)
amplitude_spectrum = abs(X_freq) / N;

% Так как спектр симметричен, берем только первую половину (до частоты

```

Найквиста)

```
half_spectrum = amplitude_spectrum(1:N/2);
```

% Отображение на графике амплитудного спектра

```
figure;
```

% Отображаем частоты в кГц для удобства

```
plot(frequencies/1e3, half_spectrum);
```

```
title(['Амплитудный спектр (N=', num2str(N), ' точек)']);
```

```
xlabel('Частота (кГц)');
```

```
ylabel('Нормализованная амплитуда');
```

```
grid on;
```

% Ограничиваем ось X до частоты Найквиста $f_s/2$, если необходимо

```
xlim([0 fs/2000]);
```

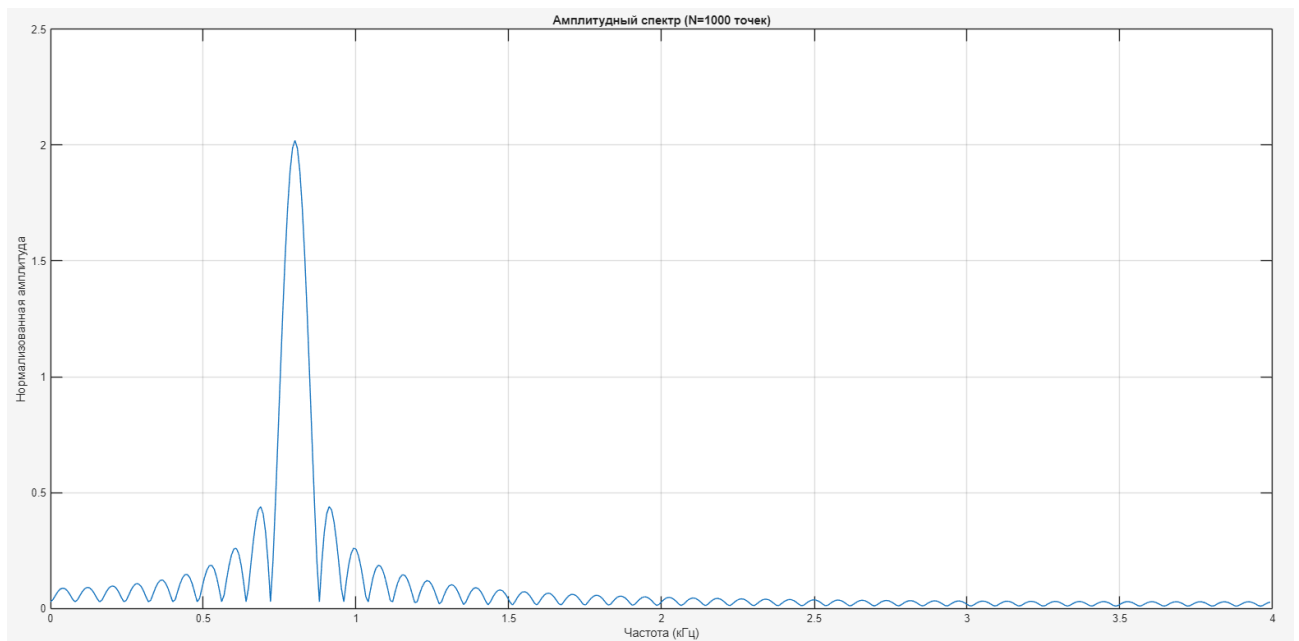


Рисунок 7 - Амплитудный спектр сигнала дополненный до 1000 точек

%То же самое но для второй таблицы

```
f_signal_kHz = 0.84;
```

```
f_signal = f_signal_kHz * 1e3;
```

```
signal = amplitude * sin(2 * pi * f_signal * t + phi_rad);
```

```
figure;
```

```
plot(t*1e3, signal); % Время в мс для читаемости
```

```
title('Исходный гармонический сигнал');
```

```
xlabel('Время (мс)');
```

```
ylabel('Амплитуда');
```

```
grid on;
```

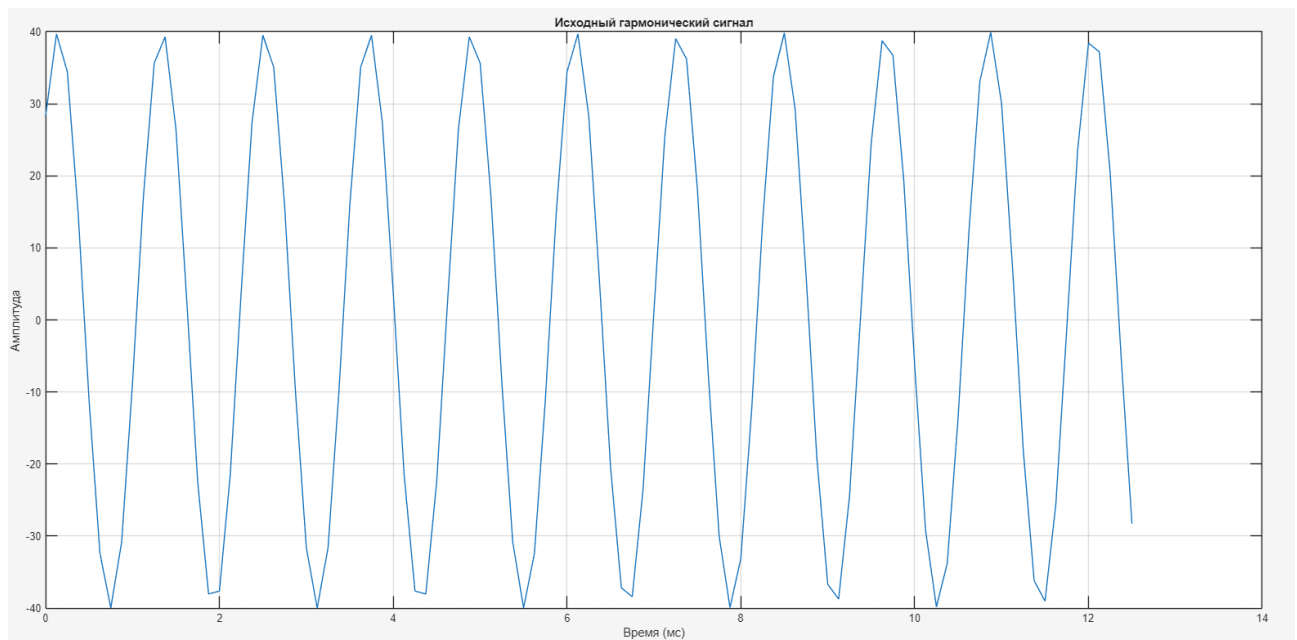



Рисунок 8 – График второго заданного сигнала

```
X_freq = fft(signal);
```

```
% Расчет вектора частот для корректного отображения спектра
```

```
N = length(signal);
```

```
% f_step - шаг по частоте
```

```
f_step = fs / N;
```

```
% Создаем вектор частот от 0 до fs/2
```

```
frequencies = 0:f_step:(fs/2 - f_step);
```

```
% Амплитудный спектр: берем модуль (abs) и нормализуем (делим на N)
```

```
amplitude_spectrum = abs(X_freq) / N;
```

```
% Так как спектр симметричен, берем только первую половину (до частоты Найквиста)
```

```
half_spectrum = amplitude_spectrum(1:N/2);
```

```
% 3. Отображение на графике амплитудного спектра
```

```
figure;
```

```
% Отображаем частоты в кГц для удобства
```

```
plot(frequencies/1e3, half_spectrum);
```

```
title(['Амплитудный спектр (N=', num2str(N), ' точек)']);
```

```
xlabel('Частота (кГц)');
```

```
ylabel('Нормализованная амплитуда');
```

```
grid on;
```

```
% Ограничиваем ось X до частоты Найквиста fs/2, если необходимо
```

```
xlim([0 fs/2000]);
```

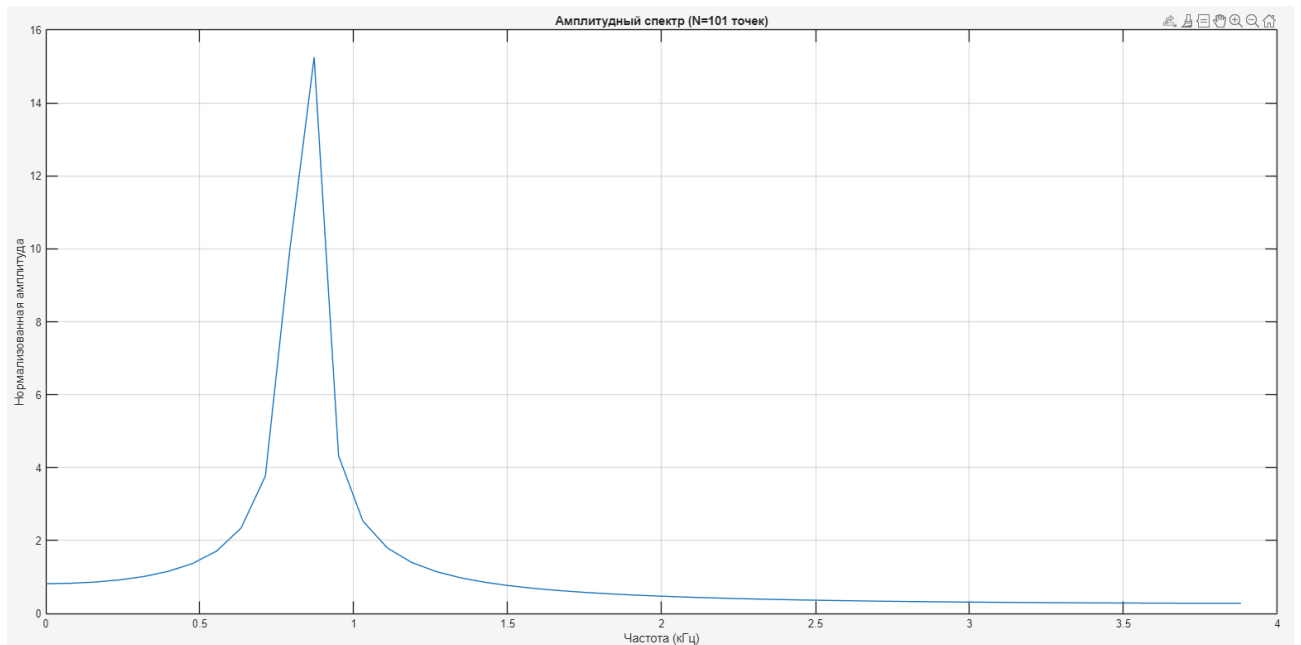


Рисунок 9 – Амплитудный спектр второго сигнала

```
X_freq = fft(signal, N_target);
N = length(X_freq);
f_step = fs / N;
frequencies = 0:f_step:(fs/2 - f_step/2);
amplitude_spectrum = abs(X_freq) / N;
half_spectrum = amplitude_spectrum(1:N/2);

figure;
plot(frequencies/1e3, half_spectrum);
title(['Амплитудный спектр (N=', num2str(N), ' точек)']);
xlabel('Частота (кГц)');
ylabel('Нормализованная амплитуда');
grid on;
xlim([0 fs/2000]);
```

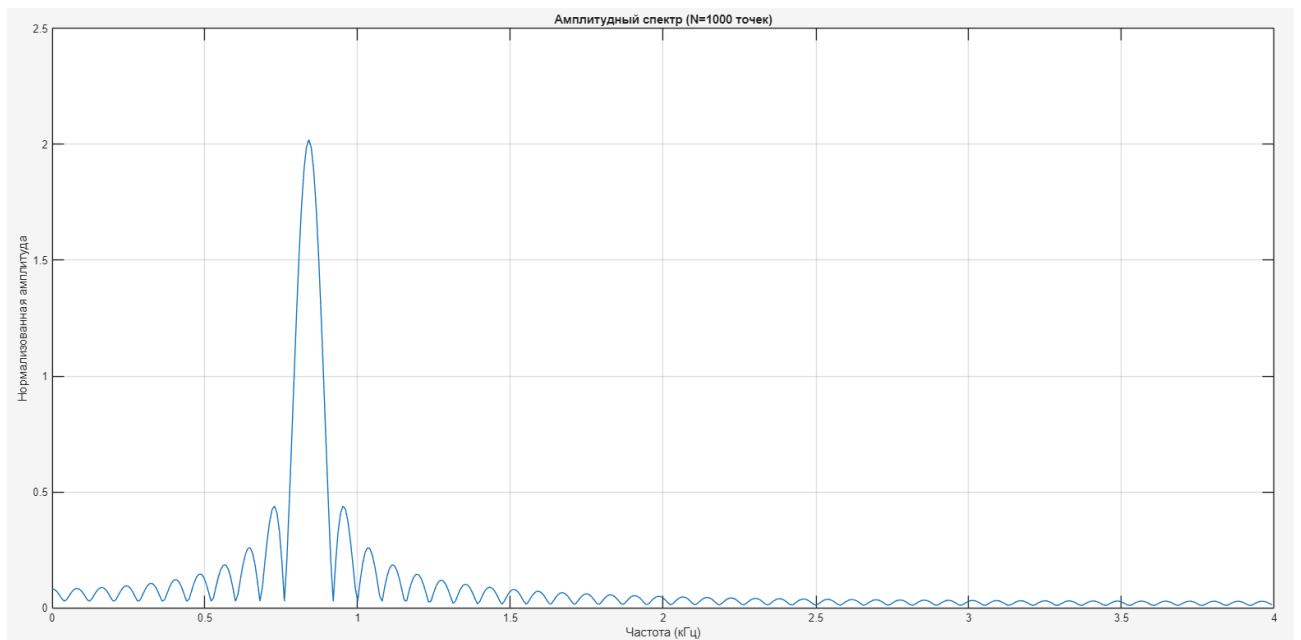


Рисунок 10 – Амплитудный спектр второго сигнала дополненный до 1000 точек.

Вывод: В ходе данной работы я научился строить амплитудные спектры сигналов, а так же проводить преобразование Фурье в программе matlab.