|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Калужский филиал федерального государственного автономного образовательного учреждения высшего образования**  **«Московский государственный технический университет  имени Н.Э. Баумана  (национальный исследовательский университет)»**  **(КФ МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ ИУК «Информатика и управление

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологи»

**ЛАБОРАТОРНАЯ РАБОТА**

**«Дискретное преобразование Фурье»**

**по дисциплине: «Цифровая обработка сигналов»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил: студент группы ИУК4-72Б | |  |  | И. А. Петроченков | |
|  | | (Подпись) |  | (И.О. Фамилия) | |
| Проверил(-а): | |  |  | О. И. Чурилин | |
|  | | (Подпись) |  | (И.О. Фамилия) | |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | | | | |
|  | - Балльная оценка:  - Оценка: | | | |

Калуга, 2025

**Целью** выполнения лабораторной работы является формирование практических навыков анализа спектра дискретных сигналов с помощью дискретного преобразования Фурье (ДПФ).

**Основными задачами** выполнения лабораторной работы являются:

1. Используя ДПФ, построить АЧХ сигналов: заданного и отфильтрованного;
2. С помощью АЧХ проверить правильность процедуры фильтрации, при необходимости скорректировать параметры фильтра.

**Вариант задания:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **№** | **Значения частот** | | | **Вид фильтра и составляющие сигнала, подлежащие фильтрации для двух видов сигнала (верхняя строка для сигнала , нижняя строка для )** | | | |
|  |  |  | **Баттерворта** | **Чебышева 1 рода** | **Чебышева 2 рода** | **Эллиптический** |
| 19 | 25 | 40 | 60 | ФВЧ, | РФ, | ФНЧ, | ПФ, |
|  |  |  | ПФ, | ФНЧ, | РФ, | ФВЧ, |

**Порядок выполнения работы:**

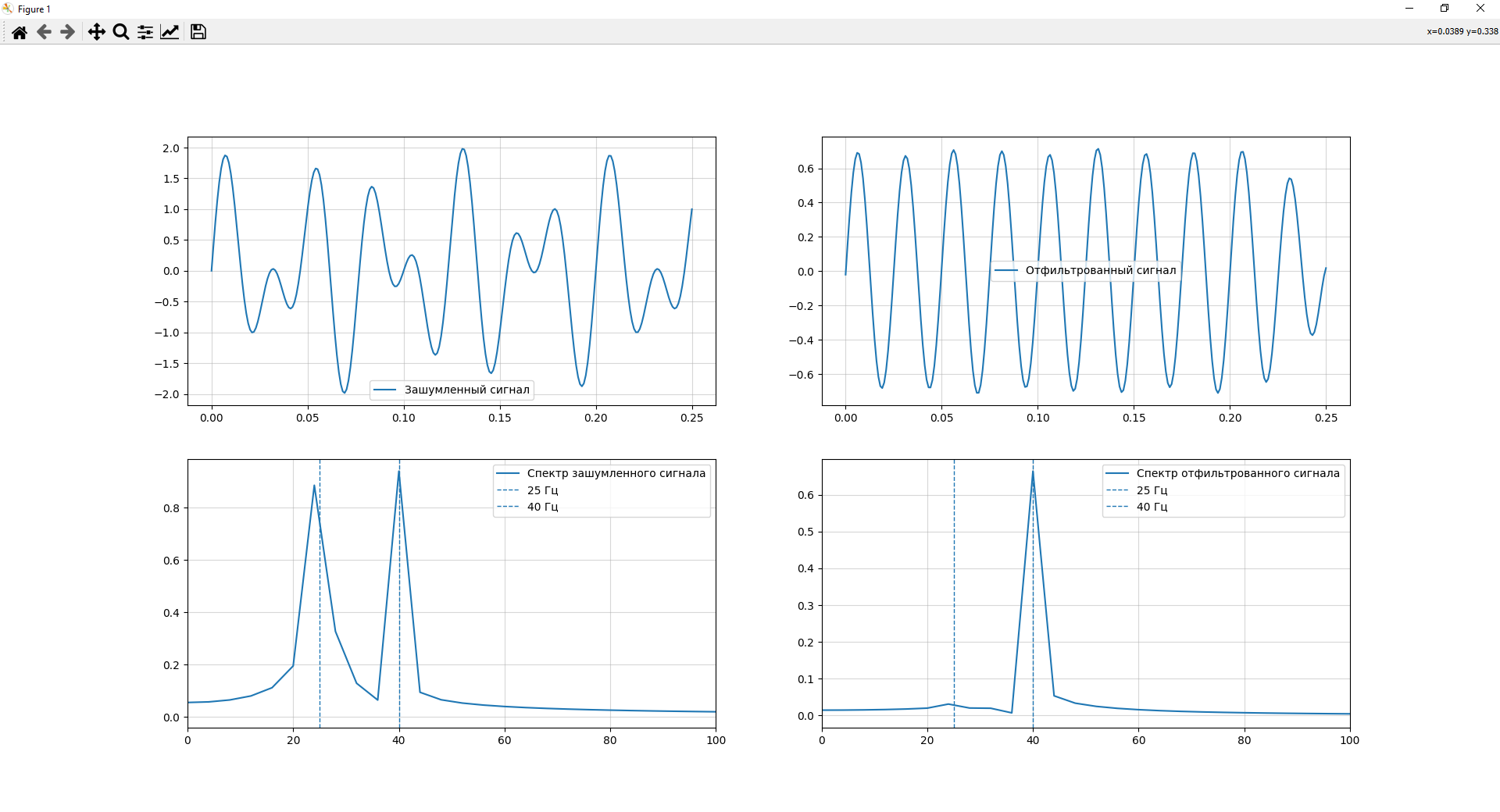


Рис. 1 Анализ фильтрации сигнала с помощью фильтра Баттерворта

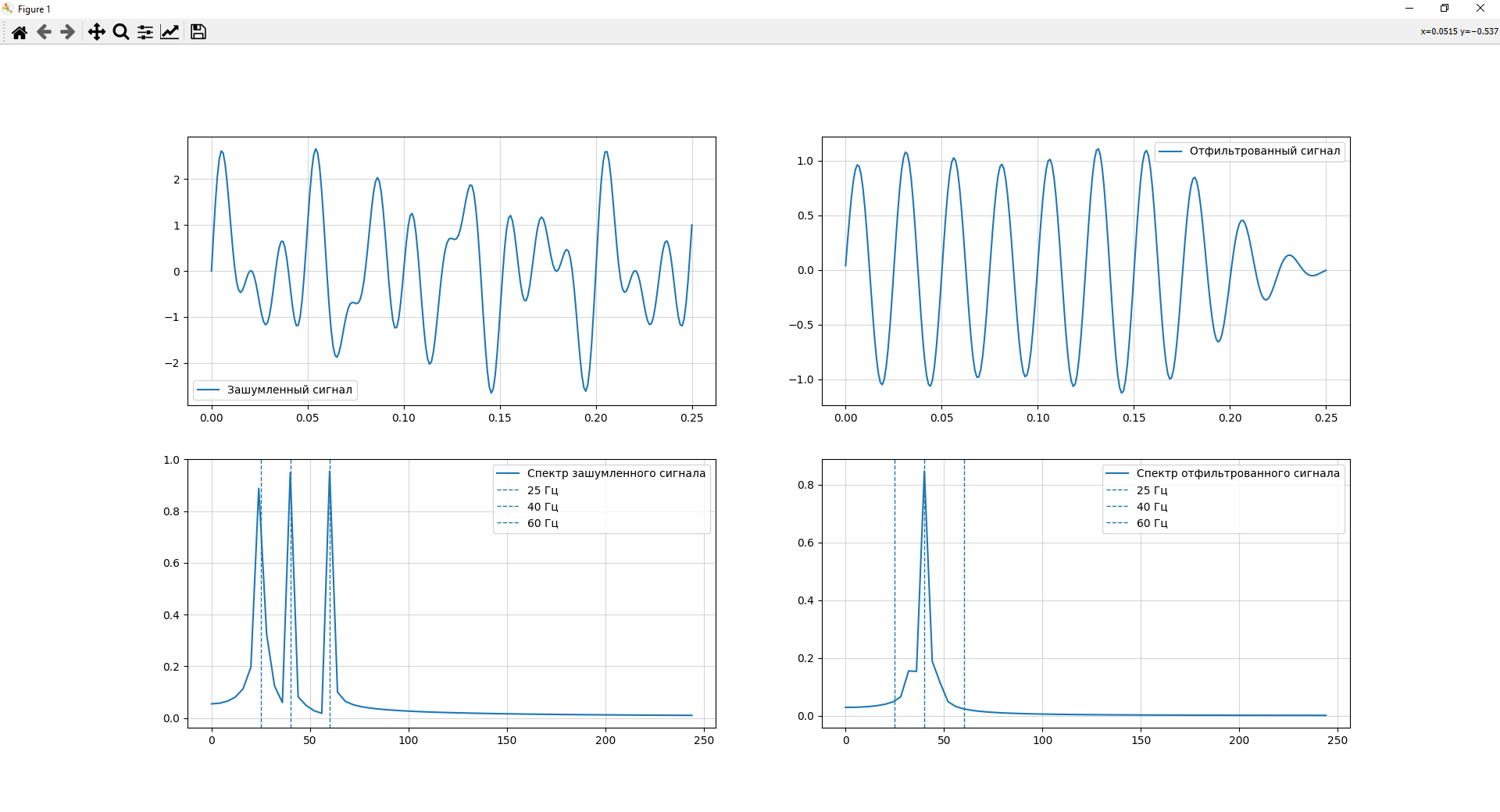


Рис. 2 Анализ фильтрации сигнала с помощью фильтра Баттерворта

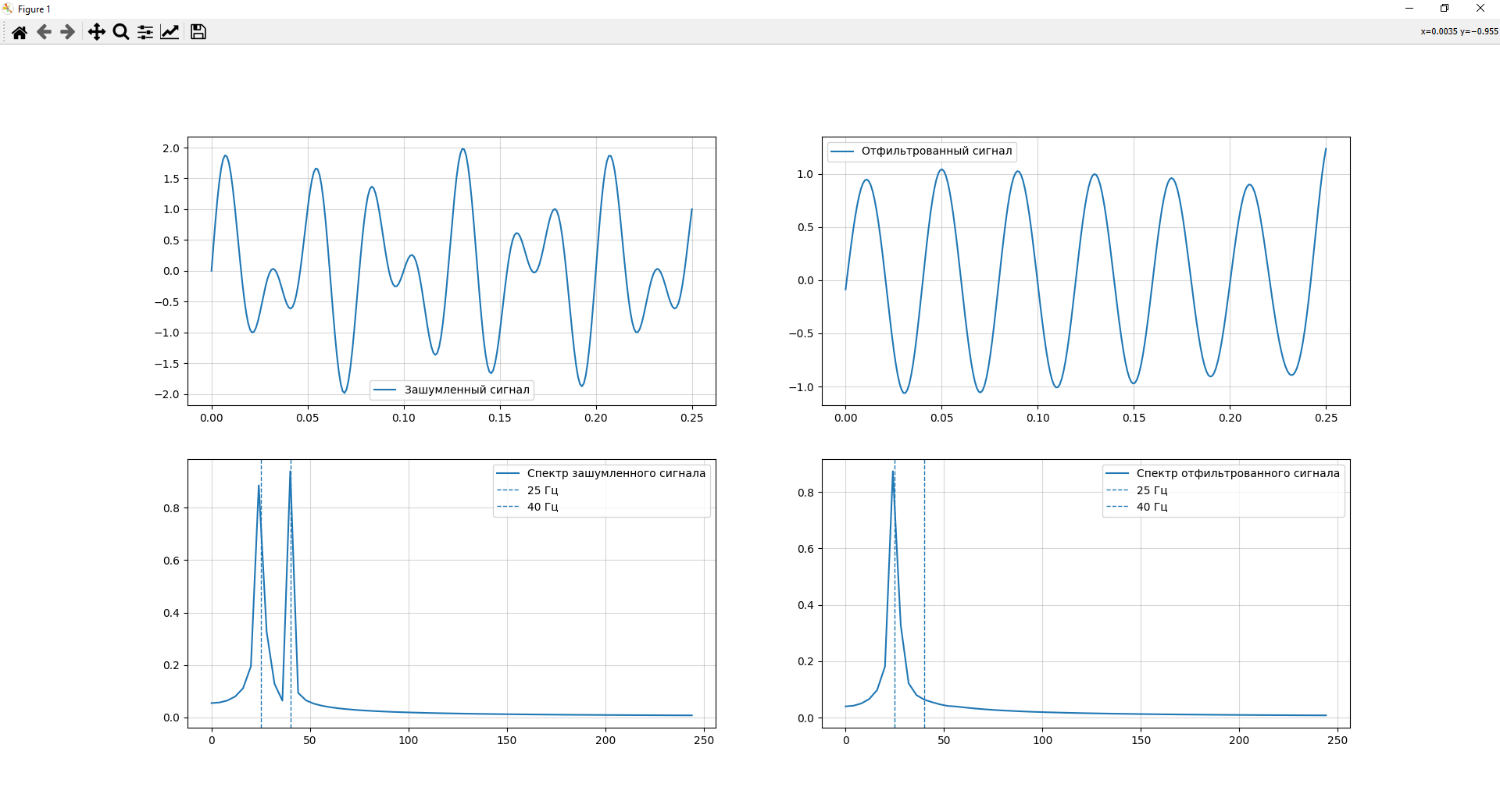


Рис. 3 Анализ фильтрации сигнала с помощью фильтра Чебышева 1 рода

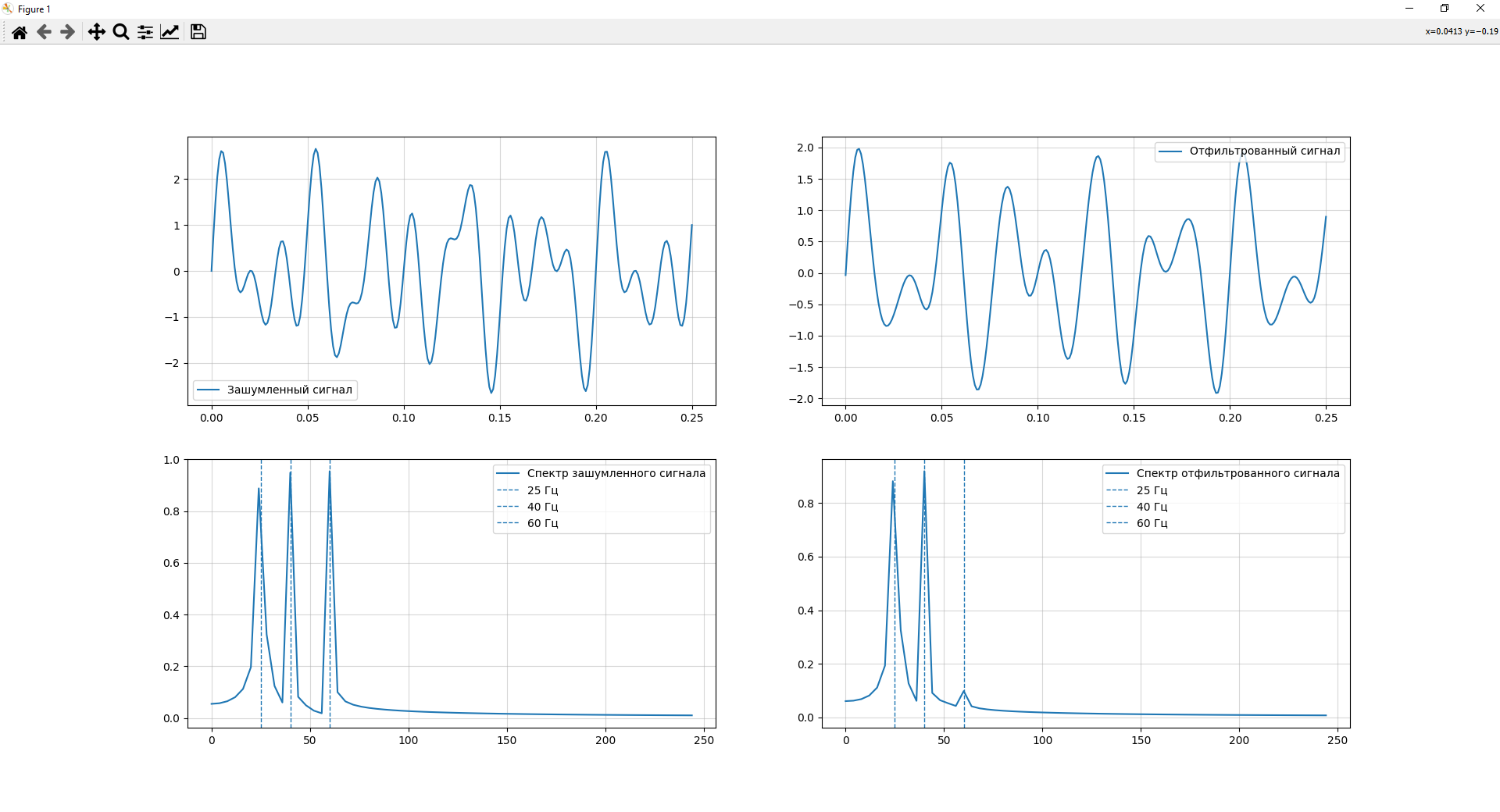


Рис. 4 Анализ фильтрации сигнала с помощью фильтра Чебышева 1 рода

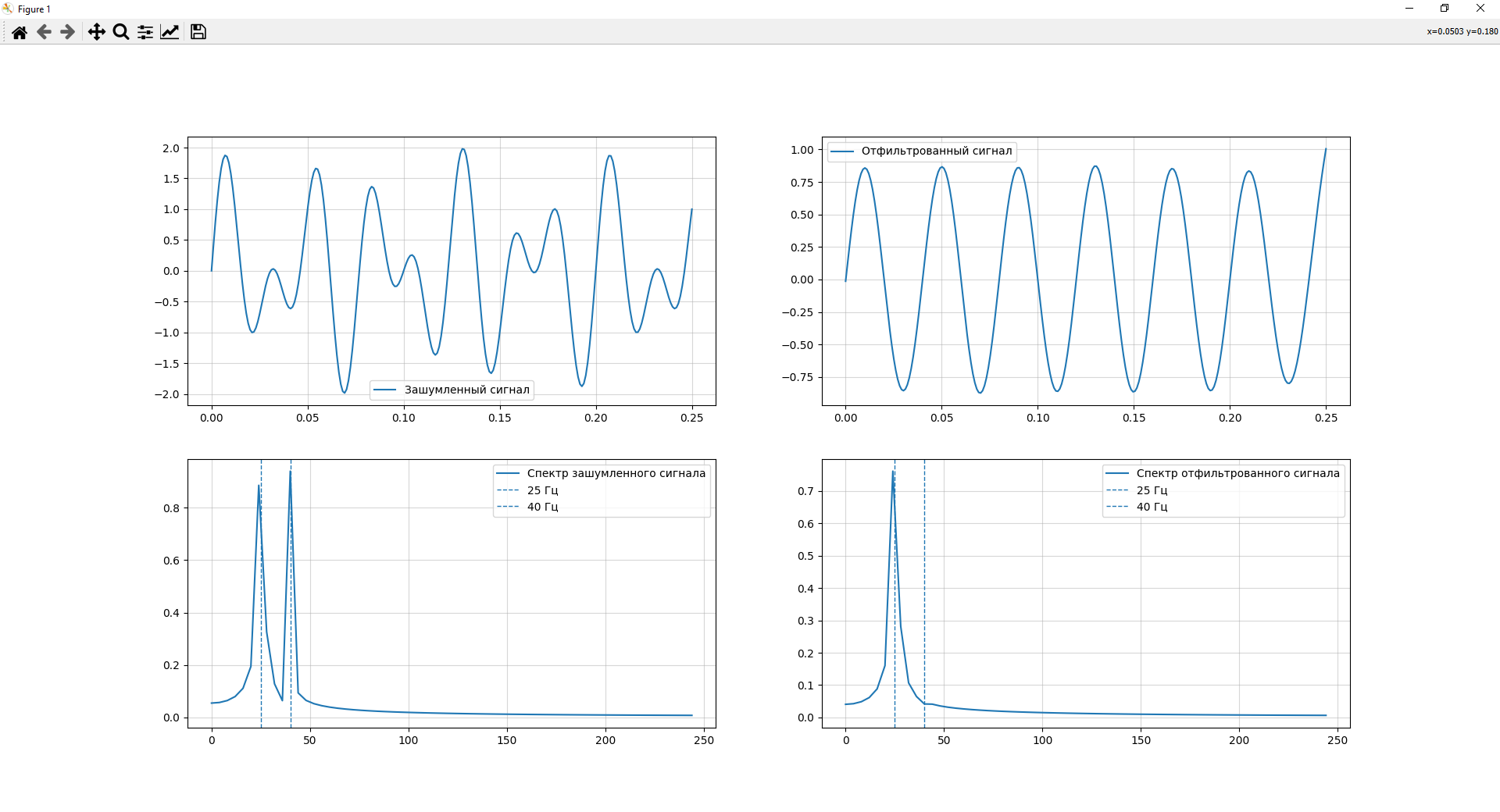


Рис. 5 Анализ фильтрации сигнала с помощью фильтра Чебышева 2 рода

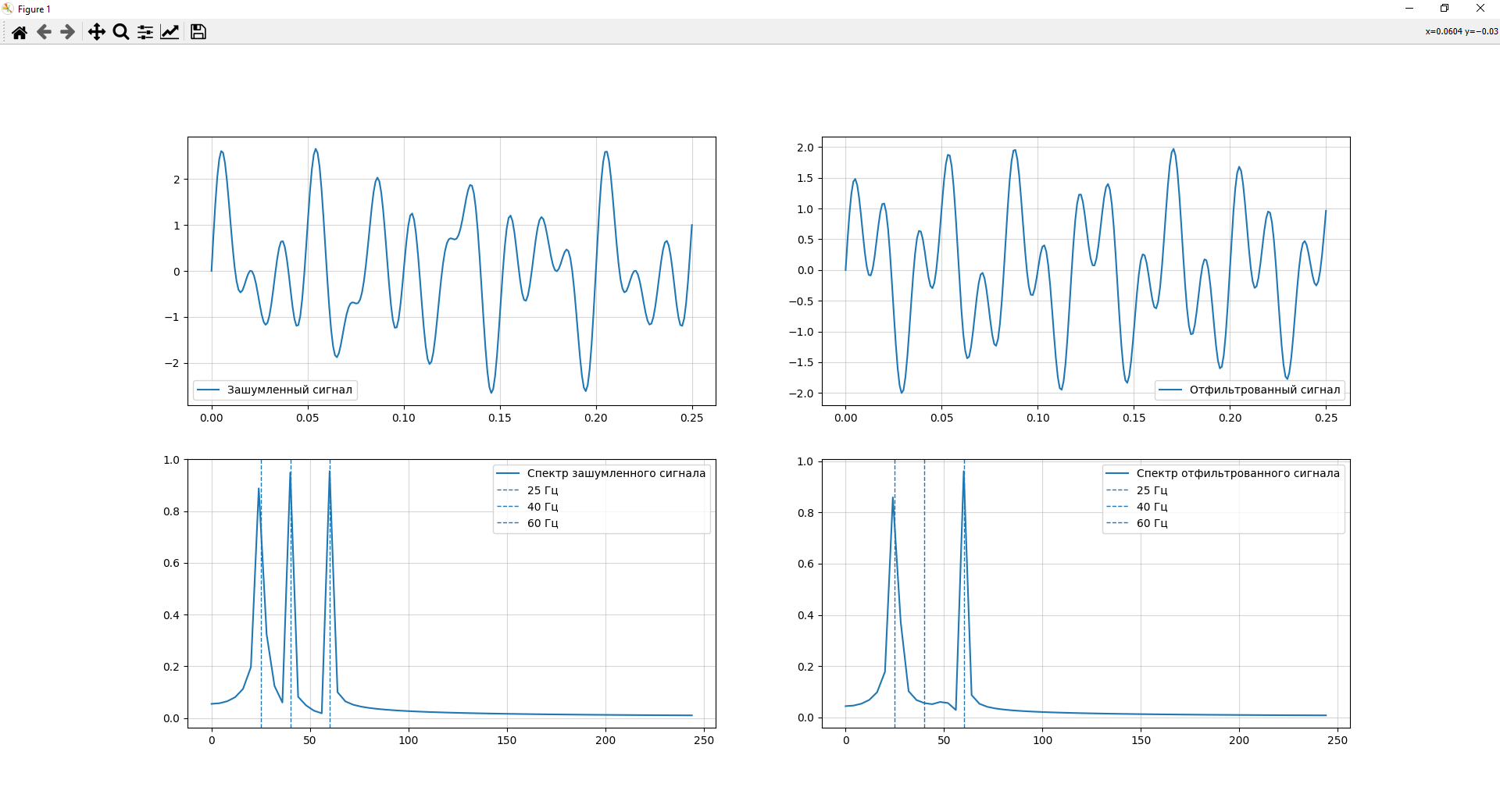


Рис. 6 Анализ фильтрации сигнала с помощью фильтра Чебышева 2 рода

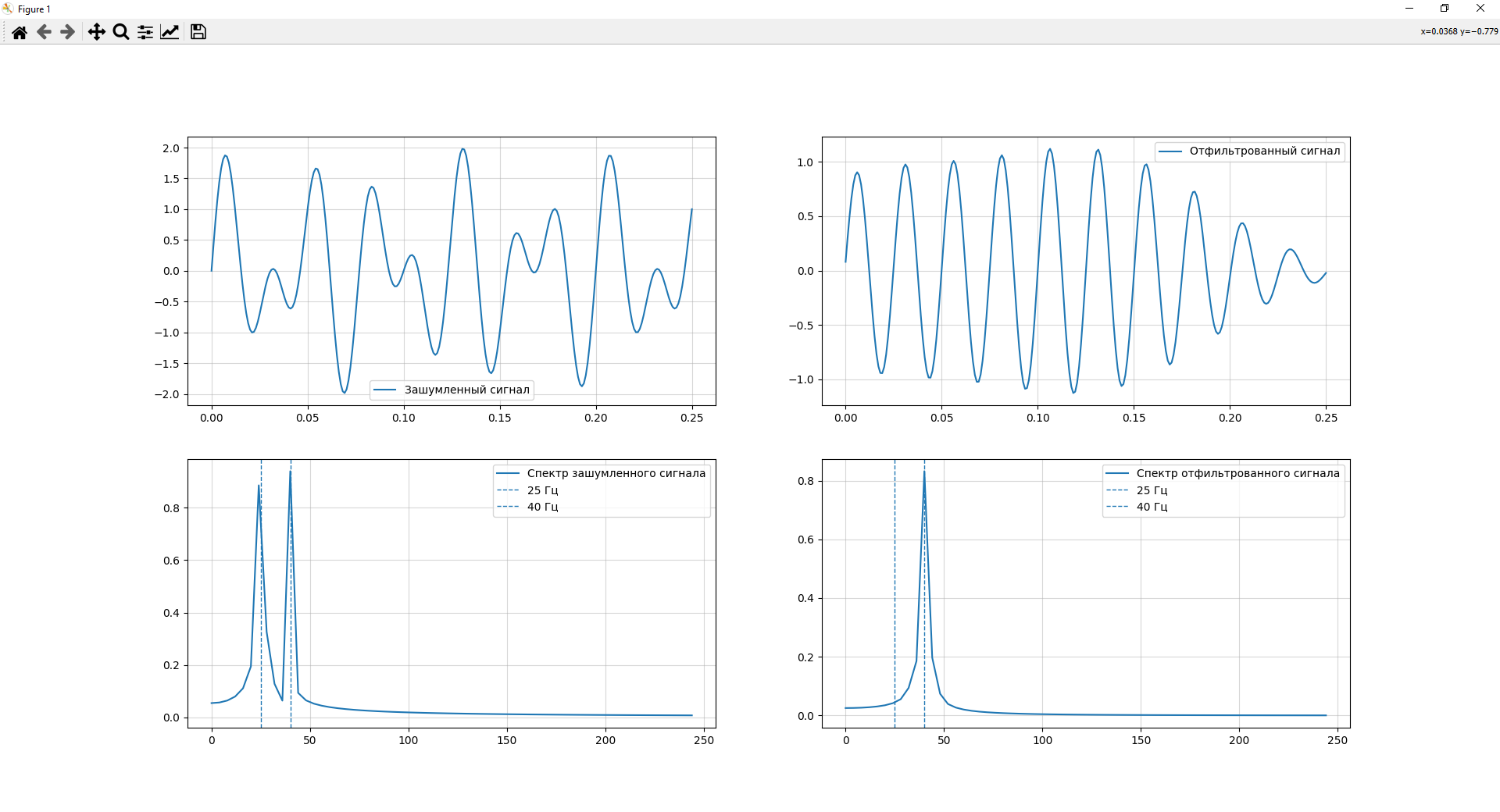


Рис. 7 Анализ фильтрации сигнала с помощью фильтра эллиптического

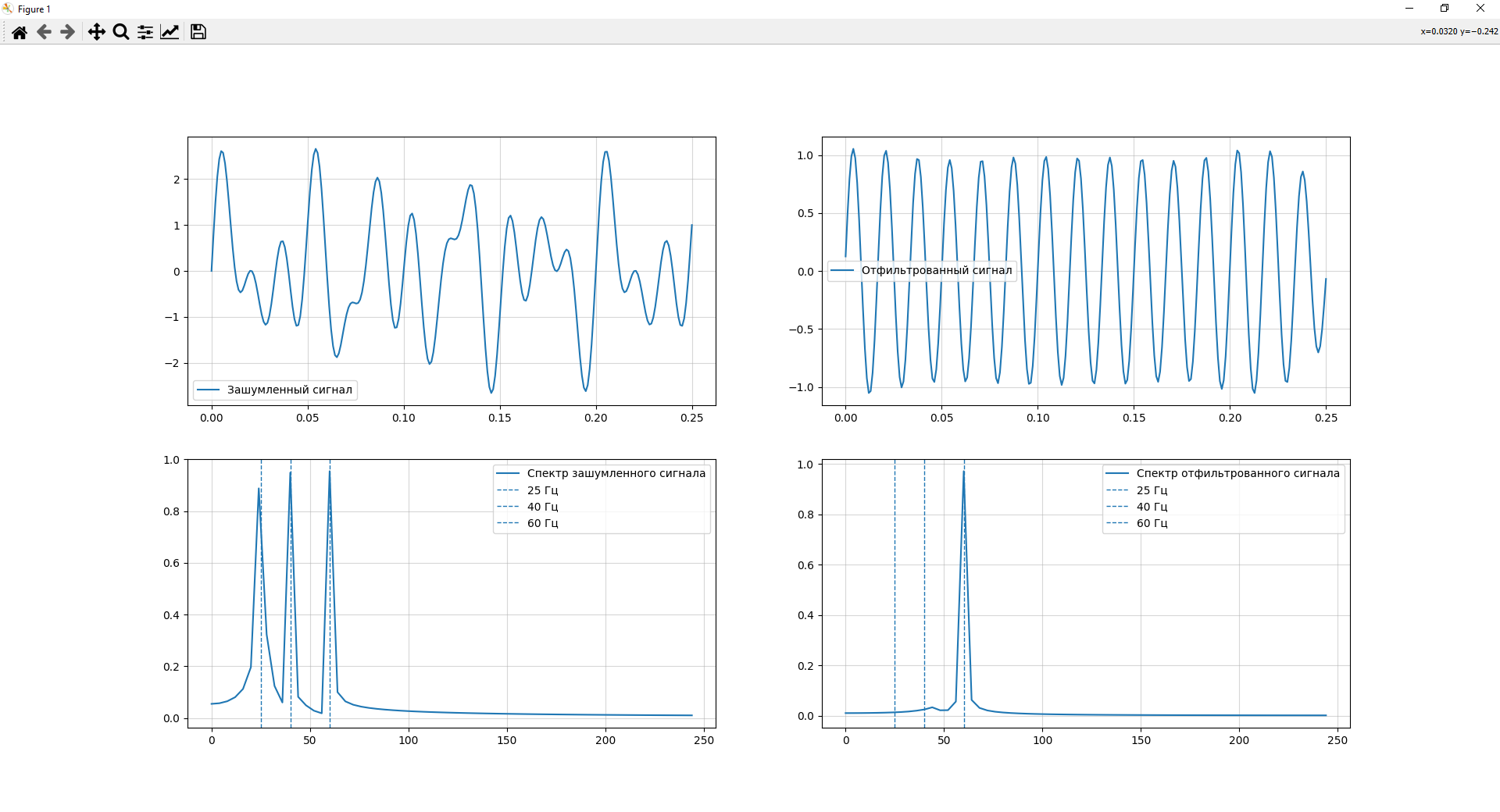


Рис. 8 Анализ фильтрации сигнала с помощью фильтра эллиптического

**Листинг кода:**

**lr5\_butterword1.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 3

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2

b, a = sgl.butter(n, s2 - delta, 'highpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//2]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//2]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.xlim(0, 100)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//2]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.xlim(0, 100)

plt.legend()

plt.show()

**lr5\_butterword2.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 7

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2 + signal3

b, a = sgl.butter(n, [s2 - delta, s2 + delta], 'bandpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_cheb11.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 10

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2

b, a = sgl.cheby1(n, Rp, [s2 - delta, s2 + delta], 'bandstop', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_cheb12.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 12

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2 + signal3

b, a = sgl.cheby1(n, Rp, s3 - delta, 'lowpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_cheb21.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 22

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2

b, a = sgl.cheby2(n, Rs, s1 + delta, 'lowpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_cheb22.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 7

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2 + signal3

b, a = sgl.cheby2(n, Rs, [s2 - delta, s2 + delta], 'bandstop', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_ellip1.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 6

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2

b, a = sgl.ellip(n, Rp, Rs, [s2 - delta, s2 + delta], 'bandpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**lr5\_ellip2.py**

import numpy as np

import scipy.signal as sgl

import matplotlib.pyplot as plt

from scipy.fft import fft, fftfreq

#butter, cheb1, cheb2, ellip

#ФВЧ - s2, РФ - s1, ФНЧ - s1, ПФ - s2 //s1+s2

#ПФ - s2, ФНЧ - s1 + s2, РФ - s1 + s3, ФВЧ - s3 // s1+s2+s3

fs = 1000

T = 0.25

N = int (fs \* T)

s1 = 25

s2 = 40

s3 = 60

delta = 8

n = 5

Rp = 0.1

Rs = 40

x = np.linspace(0, T, int(T\*fs))

signal1 = np.sin(2 \* np.pi \* s1 \* x)

signal2 = np.sin(2 \* np.pi \* s2 \* x)

signal3 = np.sin(2 \* np.pi \* s3 \* x)

noisy = signal1 + signal2 + signal3

b, a = sgl.ellip(n, Rp, Rs, s3 - delta, 'highpass', fs=fs)

filtered = sgl.filtfilt(b, a, noisy)

xf = fftfreq(N, 1.0 / fs)[:N//4]

noisyf = fft(noisy)

filteredf = fft(filtered)

plt.figure()

plt.subplot(2, 2, 1)

plt.plot(x, noisy, label=f'Зашумленный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 2)

plt.plot(x, filtered, label=f'Отфильтрованный сигнал')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 3)

plt.plot(xf, 2.0/N \* np.abs(noisyf[0:N//4]), label=f'Спектр зашумленного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.subplot(2, 2, 4)

plt.plot(xf, 2.0/N \* np.abs(filteredf[0:N//4]), label=f'Спектр отфильтрованного сигнала')

plt.axvline(x=s1, linestyle='--', linewidth=1, label=f'{s1} Гц')

plt.axvline(x=s2, linestyle='--', linewidth=1, label=f'{s2} Гц')

plt.axvline(x=s3, linestyle='--', linewidth=1, label=f'{s3} Гц')

plt.grid(True, alpha=0.5)

plt.legend()

plt.show()

**Вывод:** в ходе выполнения лабораторной работы были сформированы практические навыки анализа спектра дискретных сигналов с помощью дискретного преобразования Фурье (ДПФ).