МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Цифровая обработка сигналов»

Тема: Исследование результатов фильтрации дискретного сигнала

Студент гр. 8383 _____ Киреев К.А.

Студент гр. 8383 _____ Муковский Д.В.

Преподаватель _____ Середа А.-В.И.

Санкт-Петербург

2021

Цель работы.

Получение практических навыков выполнения фильтрации дискретных последовательностей с помощью рекурсивных и нерекурсивных фильтров, а также анализа получаемых результатов с помощью дискретного преобразования Фурье.

Основные теоретические положения.

Лабораторная работа потребует знаний:

- в области дискретизации непрерывного сигнала;
- фильтрации дискретного сигнала с помощью дискретных нерекурсивных и рекурсивных фильтров;
- дискретного преобразования Фурье (ДПФ) для дискретных последовательностей;

и умений:

- в организации вычислительных процессов;
- в проведении компьютерных расчетов с визуализацией получаемых результатов;
- проведения анализа полученных результатов и формулировка выводов.

Выполнение работы.

Формирование дискретного сигнала

Был сформирован дискретный сигнал посредством дискретизации с шагом $T = 1$ непрерывного сигнала, представляющего собой линейную комбинацию косинусоид вида $A_k \cos(\omega_k t + \varphi_k)$. Коэффициенты линейной комбинации были нормализованы посредством деления их на сумму полученных случайным образом амплитуд. Дискретная последовательность включает в себя 32 отсчета. Аналоговый сигнал представлен на рис. 1, дискретный сигнал – на рис. 2.

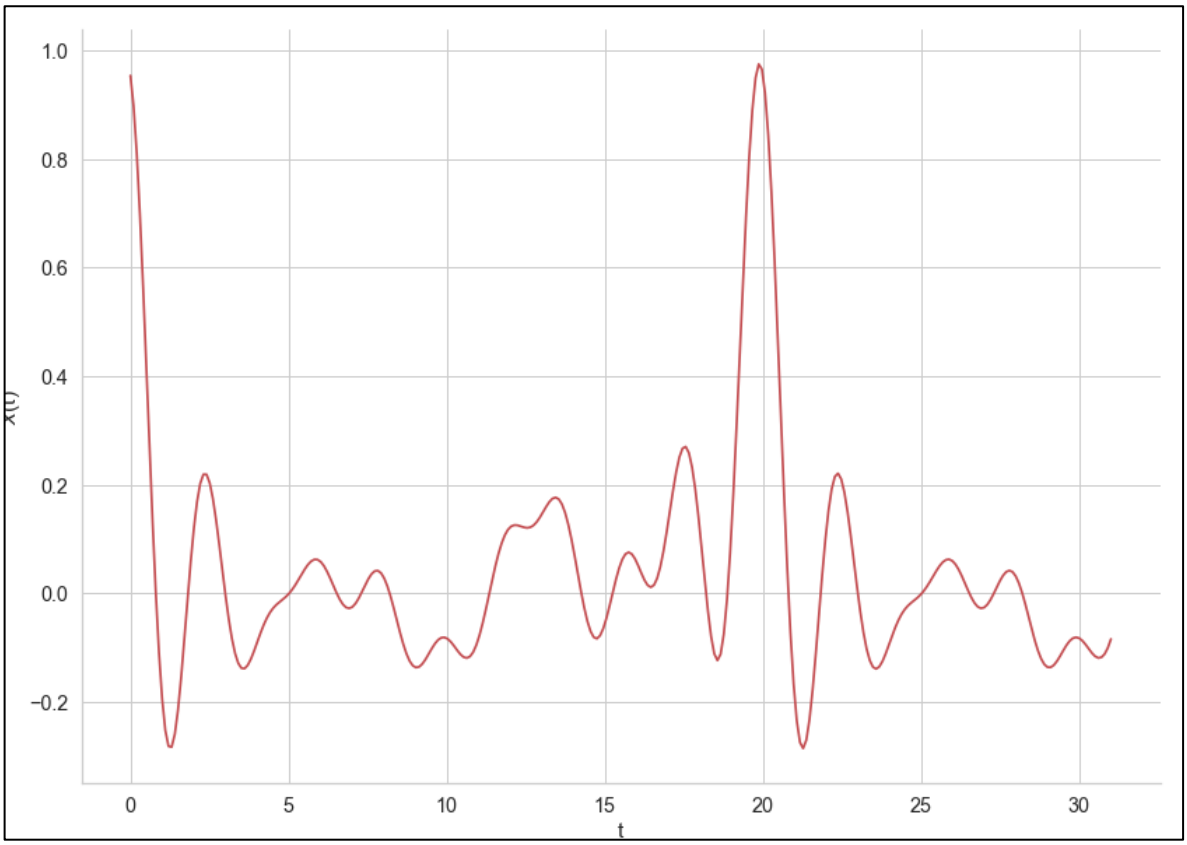

Рисунок 1 – Аналоговый сигнал

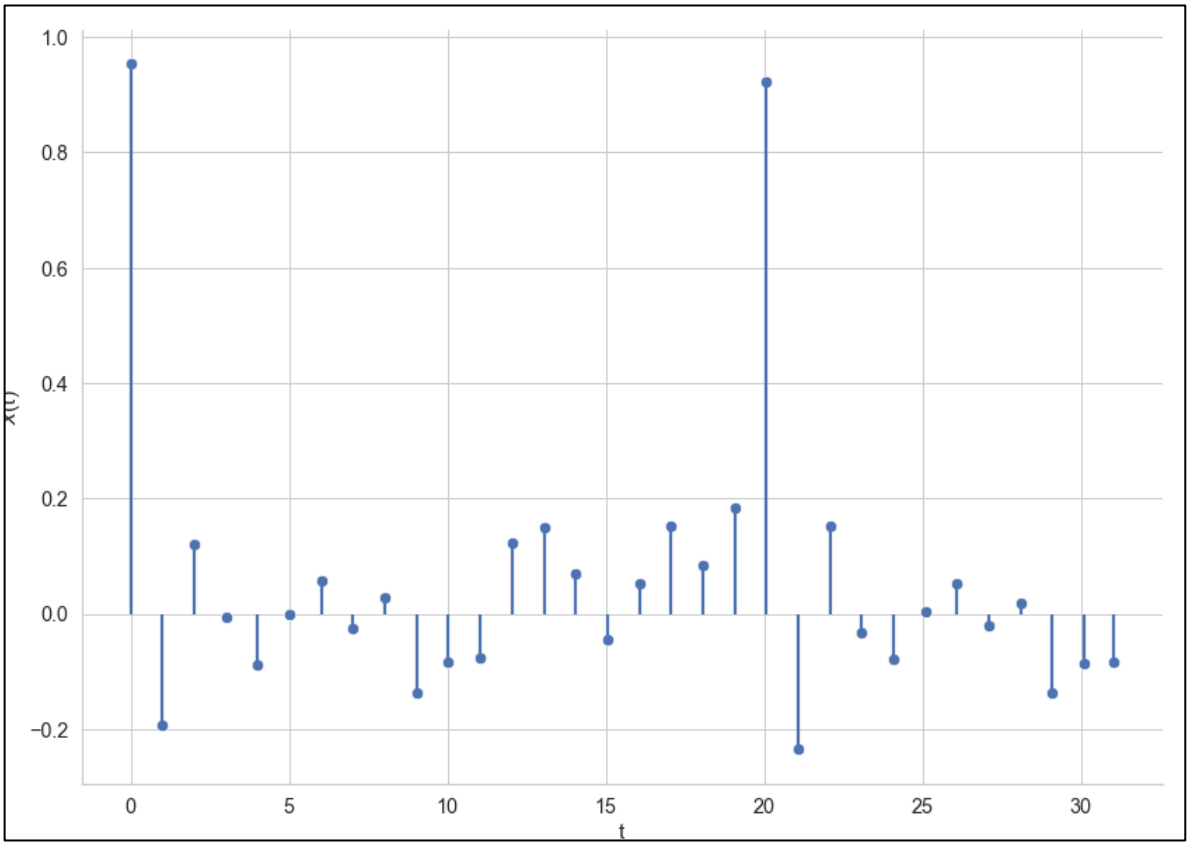

Рисунок 2 – Дискретный сигнал

Нахождение дискретных отсчетов спектра дискретного сигнала с помощью ДПФ

Представим дискретный сигнал в виде функции от времени:

$$s(t) = \sum_{k=-\infty}^{\infty} x_k \delta(t-k)$$

Тогда спектр дискретного сигнала:

$$S(\omega) = \sum_{k=-\infty}^{\infty} x_k e^{-i\omega k}$$

С другой стороны, дискретный сигнал можно представить в виде:

$$s_d(t) = s(t) \sum_{k=-\infty}^{\infty} \delta(t-kT)$$

Сумму можно представить в виде комплексного ряда Фурье, тогда

$$S_d(\omega) = \frac{1}{T} \sum_{k=-\infty}^{\infty} S\left(\omega - \frac{2\pi k}{T}\right)$$

Полученный с помощью дискретного преобразования Фурье спектр дискретного сигнала представлен на рис. 3.

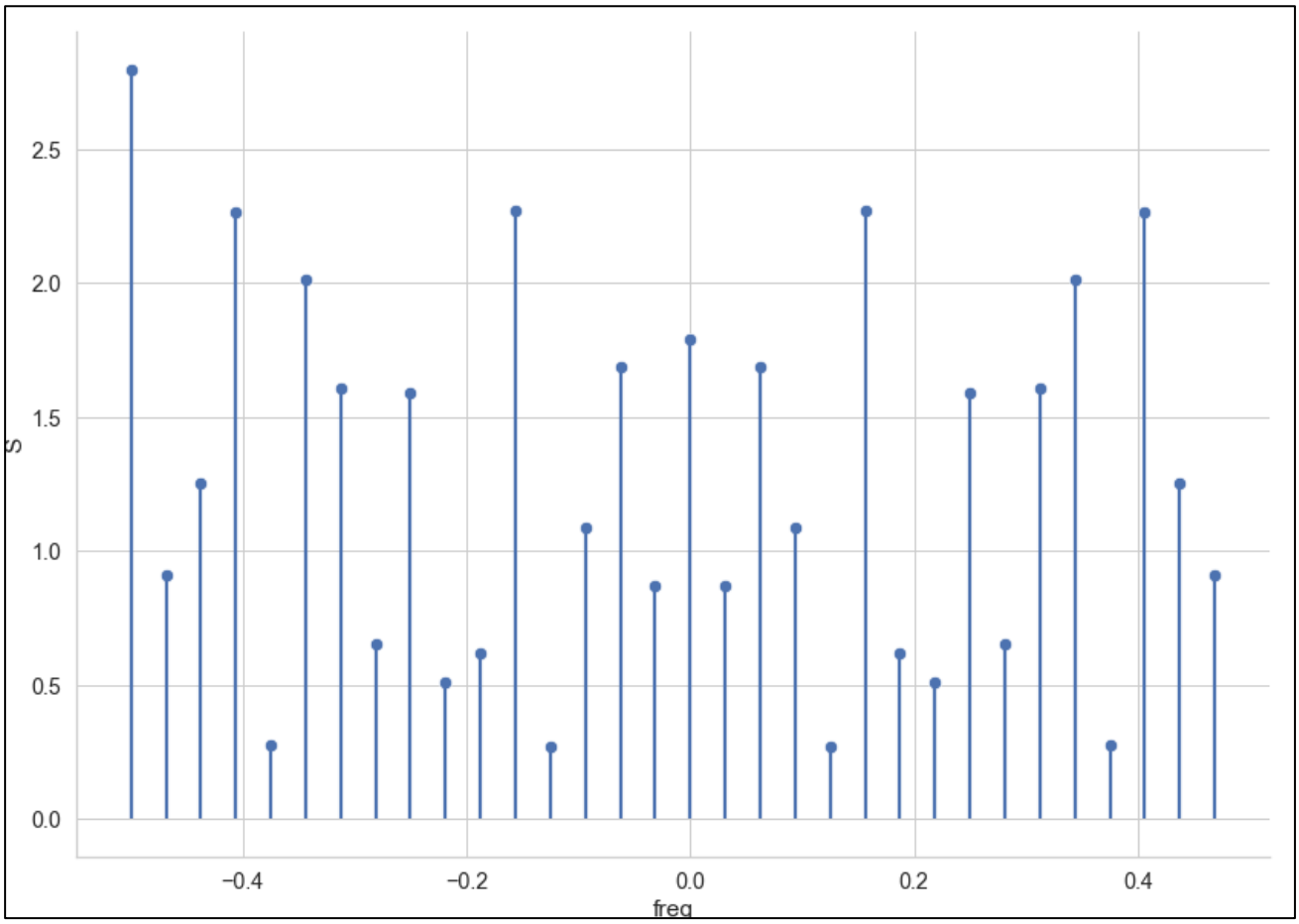

Рисунок 3 – Дискретные отсчеты спектра дискретного сигнала

Спектр симметричен относительно нуля, он представляет собой разложение исходного сигнала на линейную комбинацию простых синусоидальных функций и отражает амплитуды этих функций на разных частотах. Спектр имеет периодичность с шагом 1.

Применение линейного сглаживания по 5-ти и 9-ти точкам

Линейное сглаживание по 5-ти и 9-ти точкам осуществляется с помощью полинома первой степени, а коэффициенты в передаточной функции одинаковы. Передаточные функции 5-точечного и 9-точечного НЦФ в z-области:

$$H_5(z) = \frac{1}{5}(z^{-2} + z^{-1} + 1 + z^1 + z^2)$$

$$H_9(z) = \frac{1}{9}(z^{-4} + z^{-3} + z^{-2} + z^{-1} + 1 + z^1 + z^2 + z^3 + z^4)$$

Общая формула передаточной функции в частотной области:

$$H(\omega) = \frac{sin\left(\left(N + \frac{1}{2}\right)\omega\right)}{\left(N + \frac{1}{2}\right)\omega} = sinc\left(\left(N + \frac{1}{2}\right)\omega\right)$$

На рис. 4 представлено сравнение исходного сигнала (синий) и сигнала после применения линейного сглаживания по 5-ти точкам (оранжевый).

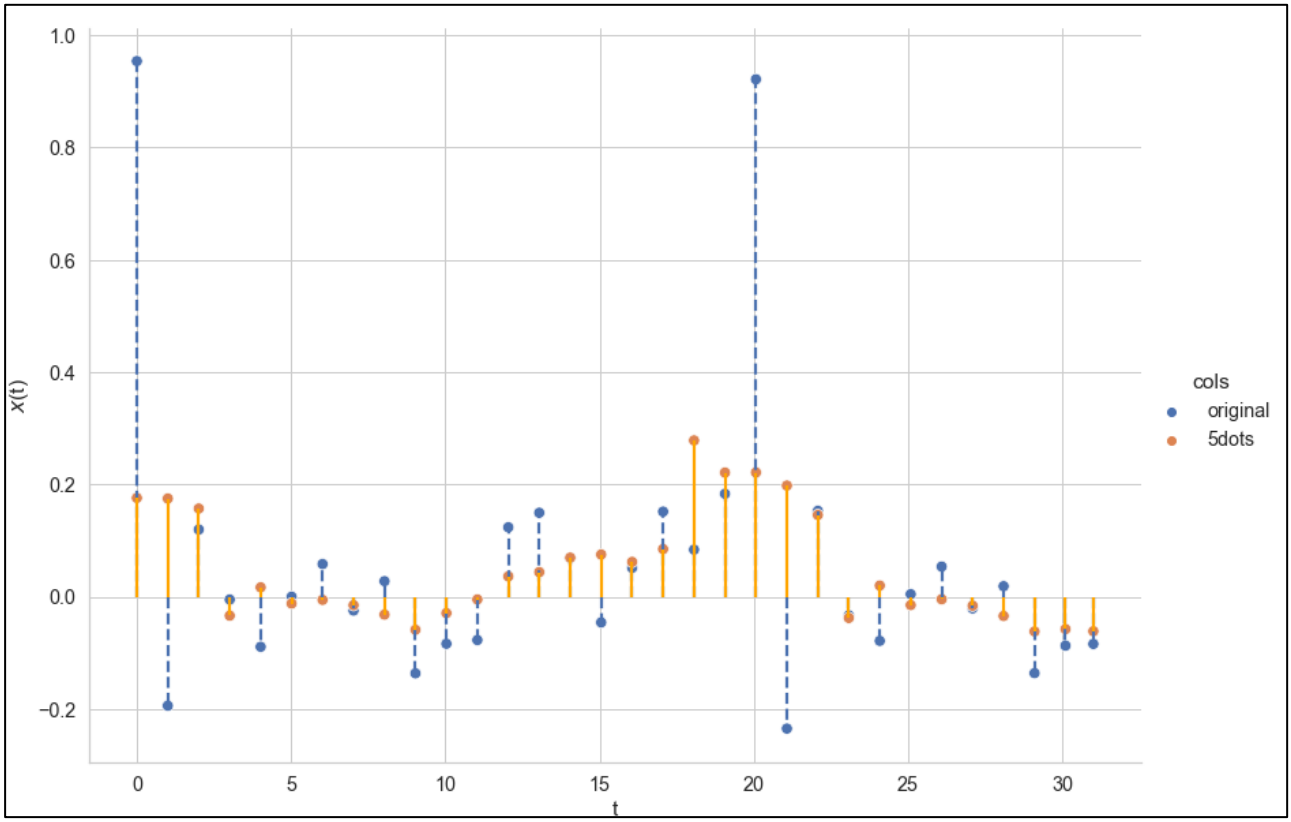

Рисунок 4 – Линейное сглаживание по 5-ти точкам

На рис. 5 представлено сравнение исходного сигнала (синий) и сигнала после применения линейного сглаживания по 9-ти точкам (оранжевый).

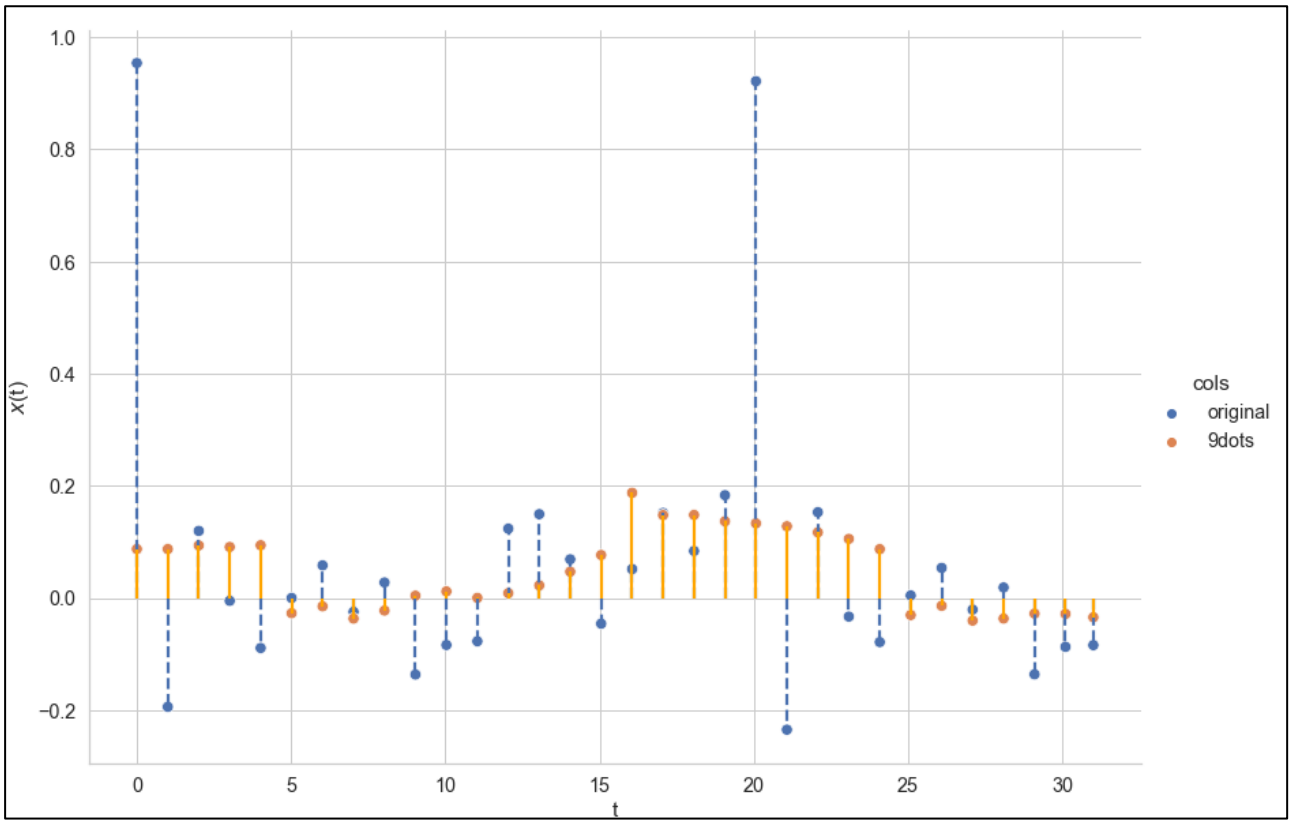

Рисунок 5 – Линейное сглаживание по 9-ти точкам

Также были получены спектры сигнала после фильтрации. На рис. 6 и 7 представлены сравнения спектров исходного сигнала и сигнала после линейного сглаживания по 5-ти и 9-ти точкам.

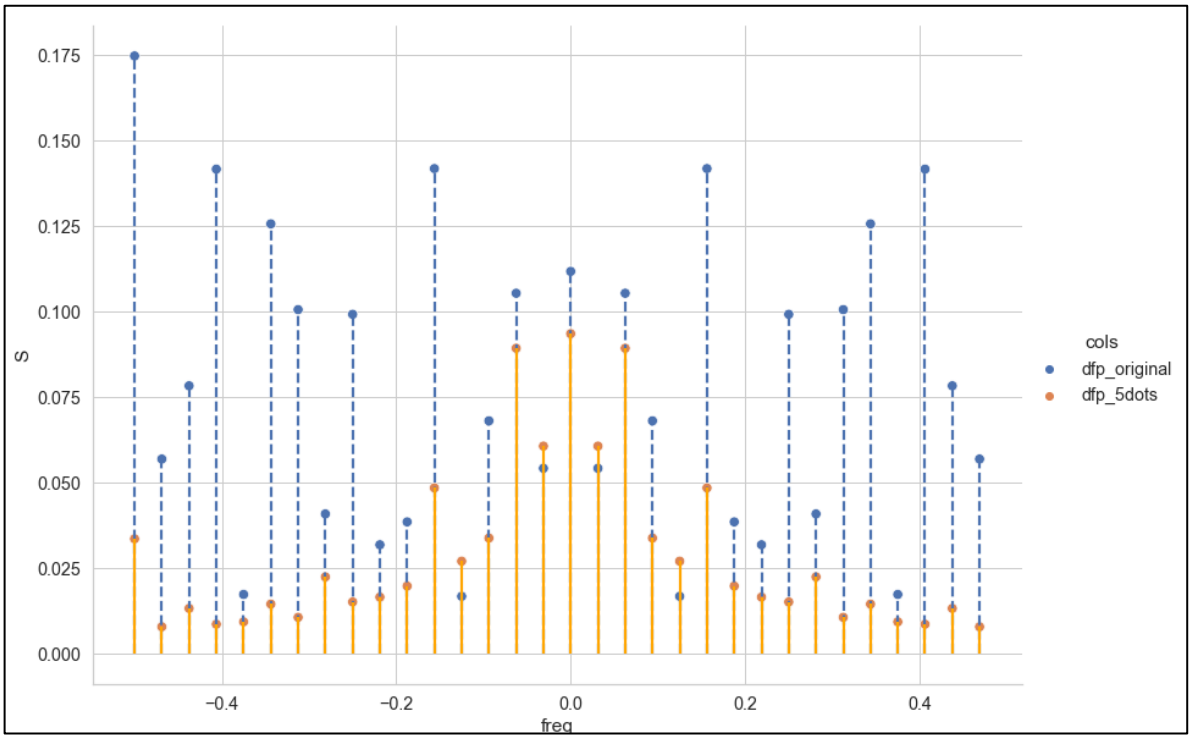

Рисунок 6 – Дискретные отсчеты спектра после линейного сглаживания по 5-ти точкам

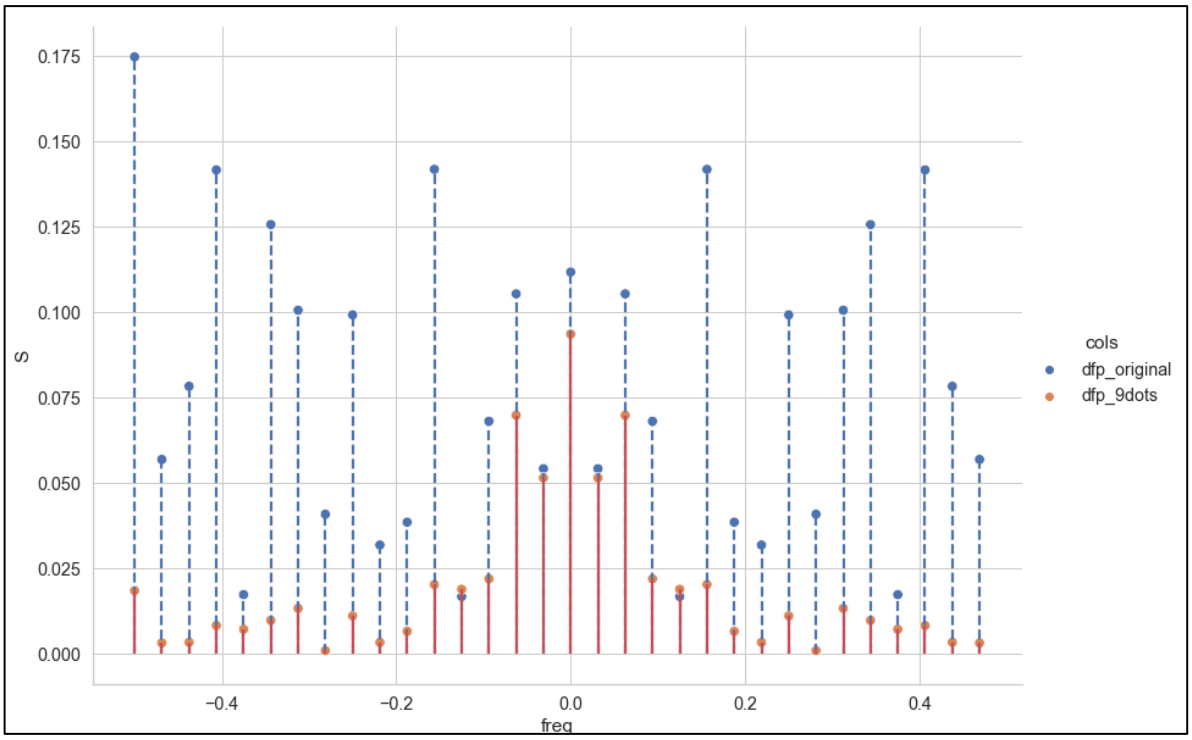

Рисунок 7 – Дискретные отсчеты спектра после линейного сглаживания по 9-ти точкам

Из спектра видно, что без ослабления пропускается только сигнал постоянного уровня (нулевой частоты). С увеличением числа точек полоса пропускания становится уже.

График передаточной функции линейного сглаживания представлен на рис. 8.

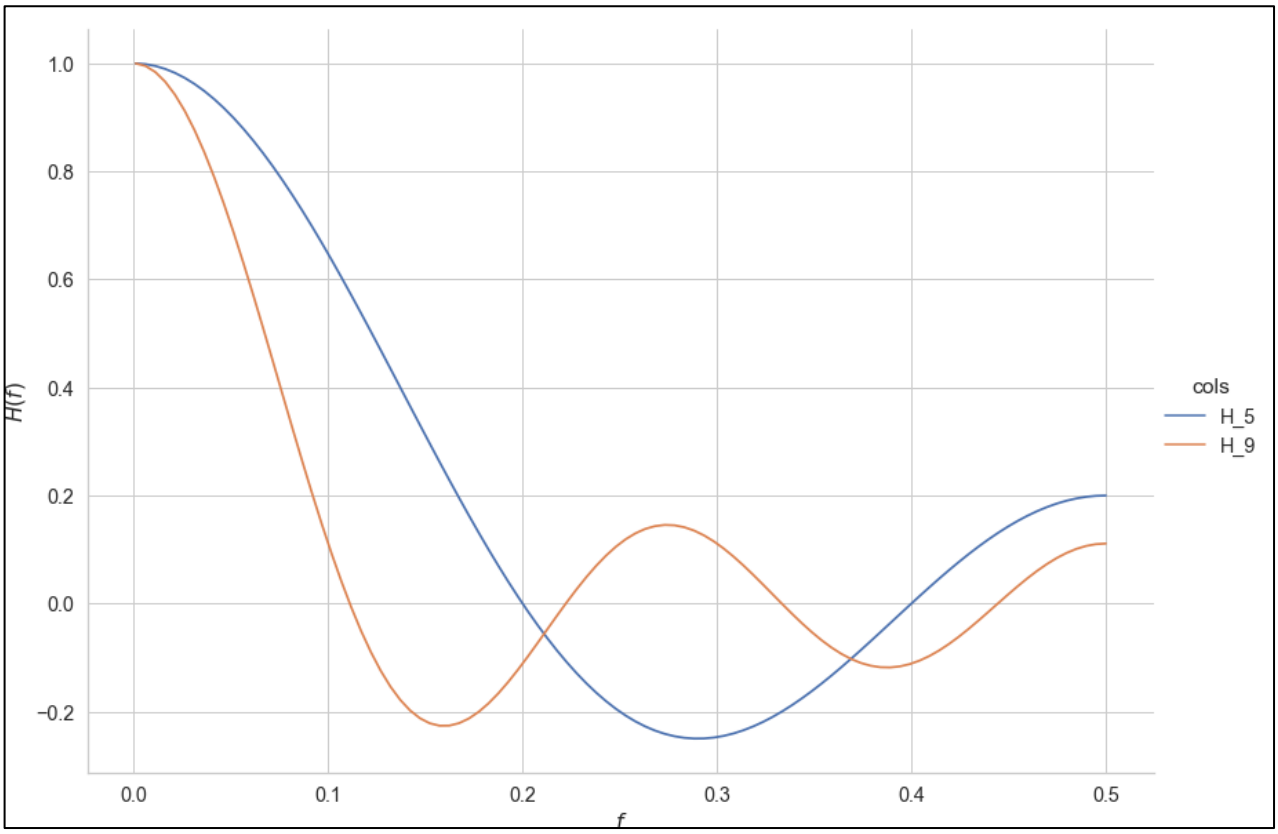

Рисунок 8 – График передаточной функции линейного сглаживания

График передаточной функции подтверждает ранее сделанные выводы относительно полосы пропускания и ослабления уровня.

Сглаживание полиномом 2-й степени по 5 и 9 узлам

Формулы передаточных функций для сглаживания полиномом второй степени:

$$H_5(\omega) = \frac{(17 + 24\cos(\omega) - 6\cos(2\omega))}{35}$$

$$H_9(\omega) = \frac{(59 + 108\cos(\omega) + 78\cos(2\omega) + 28\cos(3\omega) - 42\cos(4\omega))}{231}$$

На рис. 9 и 10 представлено сравнение исходного сигнала и сигнала после применения сглаживания полиномом 2-й степени по 5-ти и 9-ти точкам.

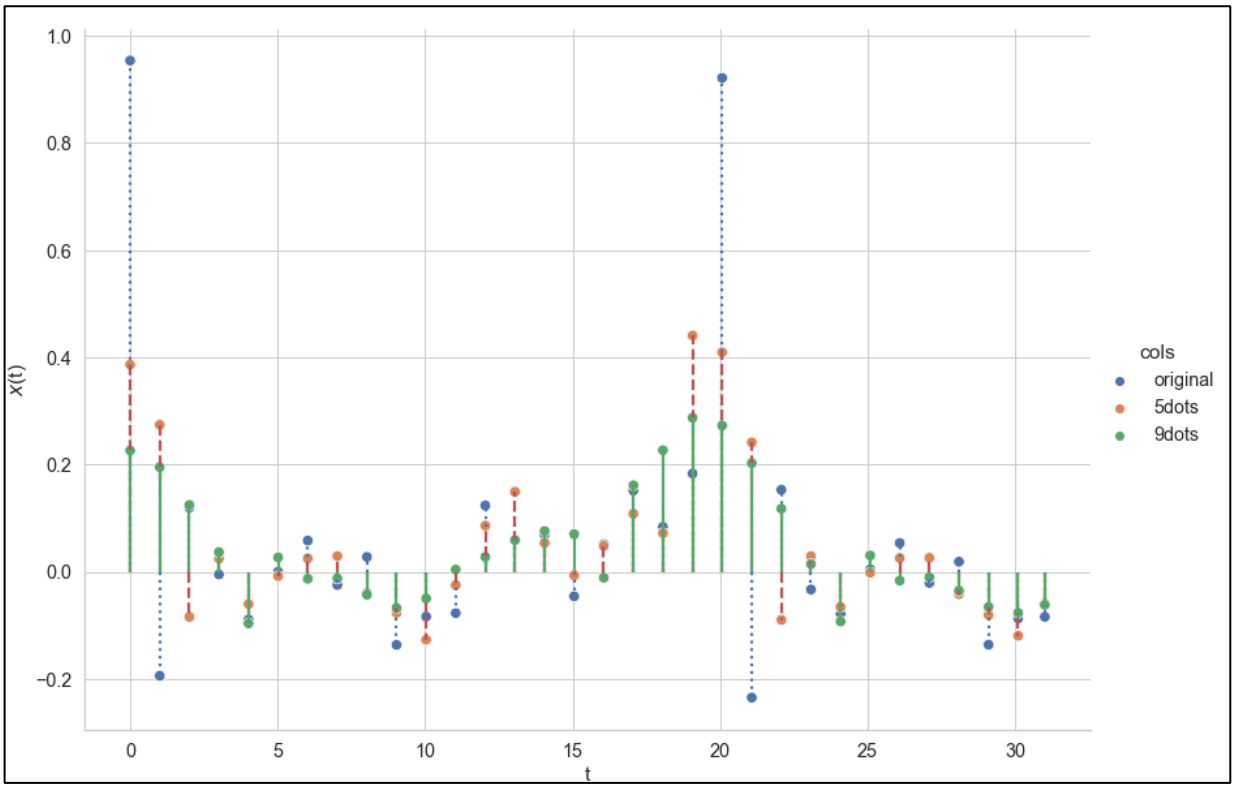

Рисунок 9 – Сглаживание полиномом 2-й степени по 5-ти и 9-ти узлам

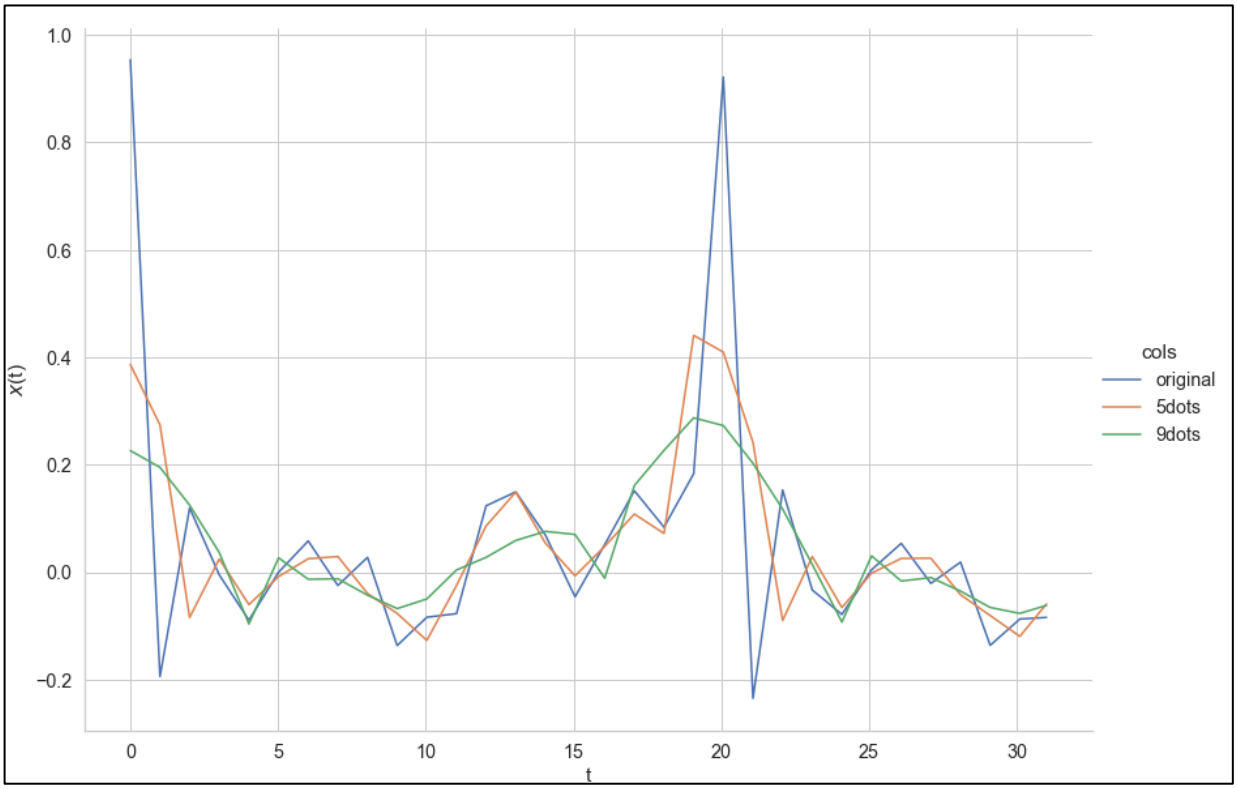

Рисунок 10 – Сглаживание полиномом 2-й степени по 5-ти и 9-ти узлам (точки соединены для лучшей визуализации)

Также были получены спектры сигнала после фильтрации. На рис. 11 и 12 представлено сравнение спектров исходного сигнала и сигнала после сглаживания полиномом 2-й степени по 5-ти и 9-ти узлам.

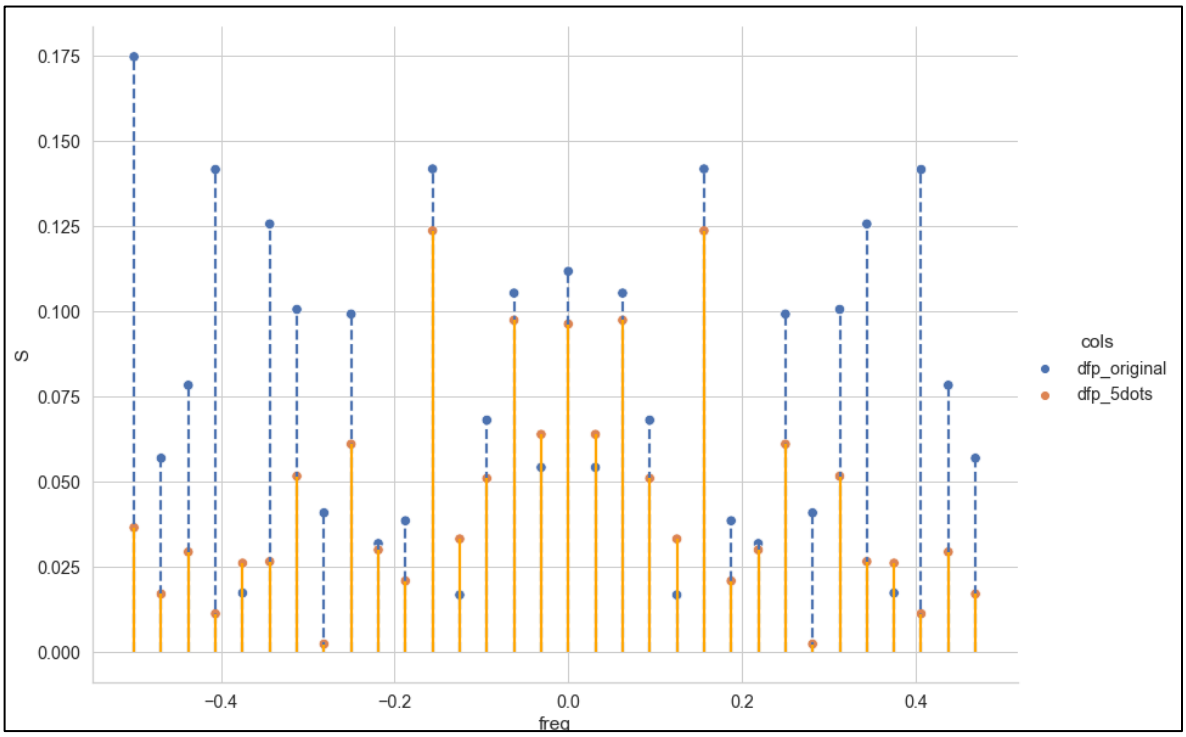

Рисунок 11 – Спектры сигналов после сглаживание фильтром 2-го порядка по 5-ти узлам

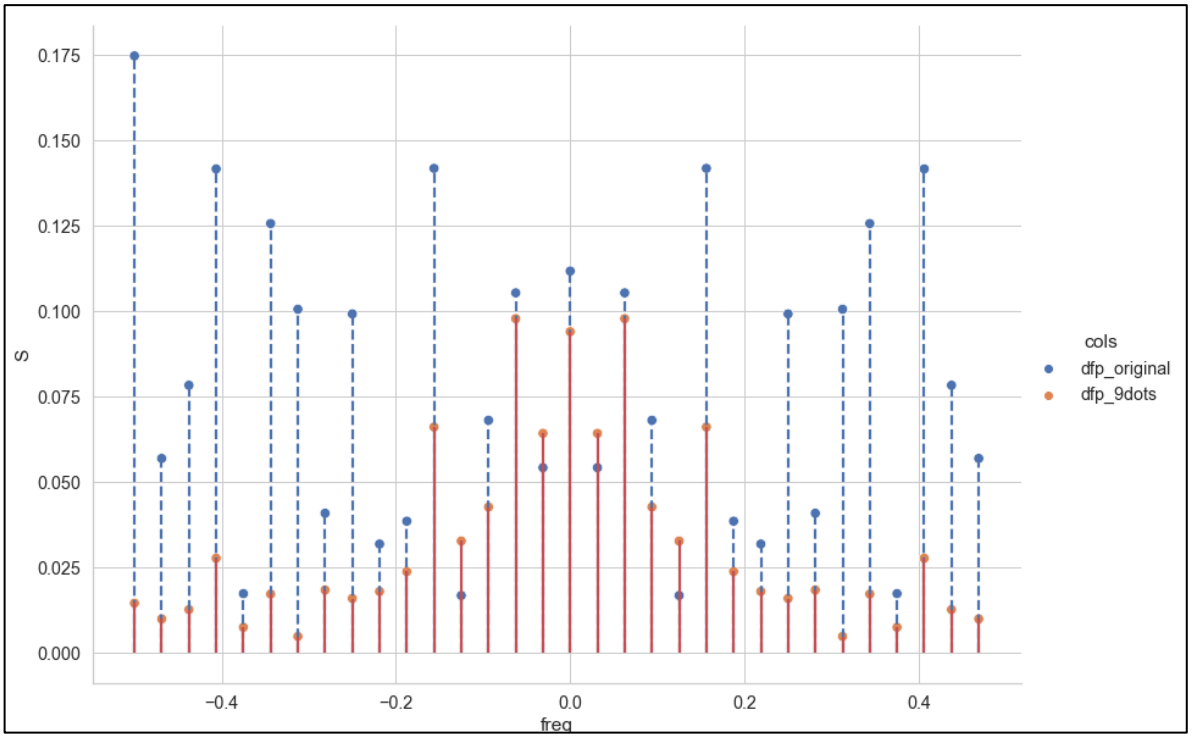

Рисунок 12 – Спектры сигналов после сглаживание фильтром 2-го порядка по 9-ти узлам

На рис. 13 представлены графики передаточных функций фильтра сглаживания полиномом второй степени.

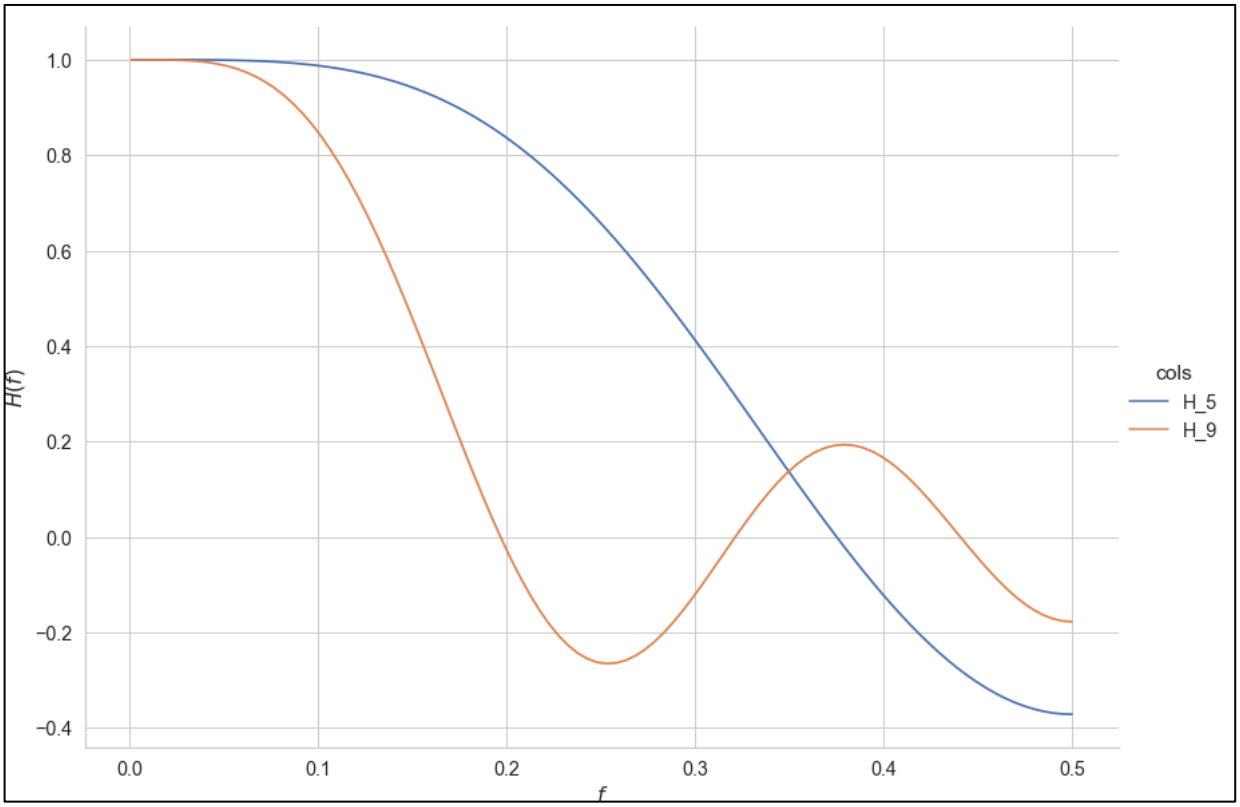

Рисунок 13 – Передаточная функция фильтра 2-го порядка

На графике видно, что при увеличении числа точек увеличивается крутизна среза и уменьшается полоса пропускания.

Также присутствует более пологая часть в полосе пропускания около 0, в отличие от линейного фильтра, поэтому пропускаются сигналы низкой частоты почти без потери амплитуды в более широком диапазоне, что видно на графиках спектра.

Сглаживание полиномом 4-й степени по 7 и 11 узлам

Формулы передаточных функций для сглаживания полиномом четвертой степени:

$$H_7(\omega) = \frac{(131 + 150\cos(\omega) - 60\cos(2\omega) + 10\cos(3\omega))}{231}$$

$$H_{11}(\omega) = \frac{(143 + 240\cos(\omega) + 120\cos(2\omega) - 20\cos(3\omega) - 90\cos(4\omega) + 36\cos(5\omega))}{429}$$

На рис. 14 и 15 представлено сравнение исходного сигнала и сигнала после применения сглаживания полиномом 4-й степени по 7-ми и 11-ти точкам.

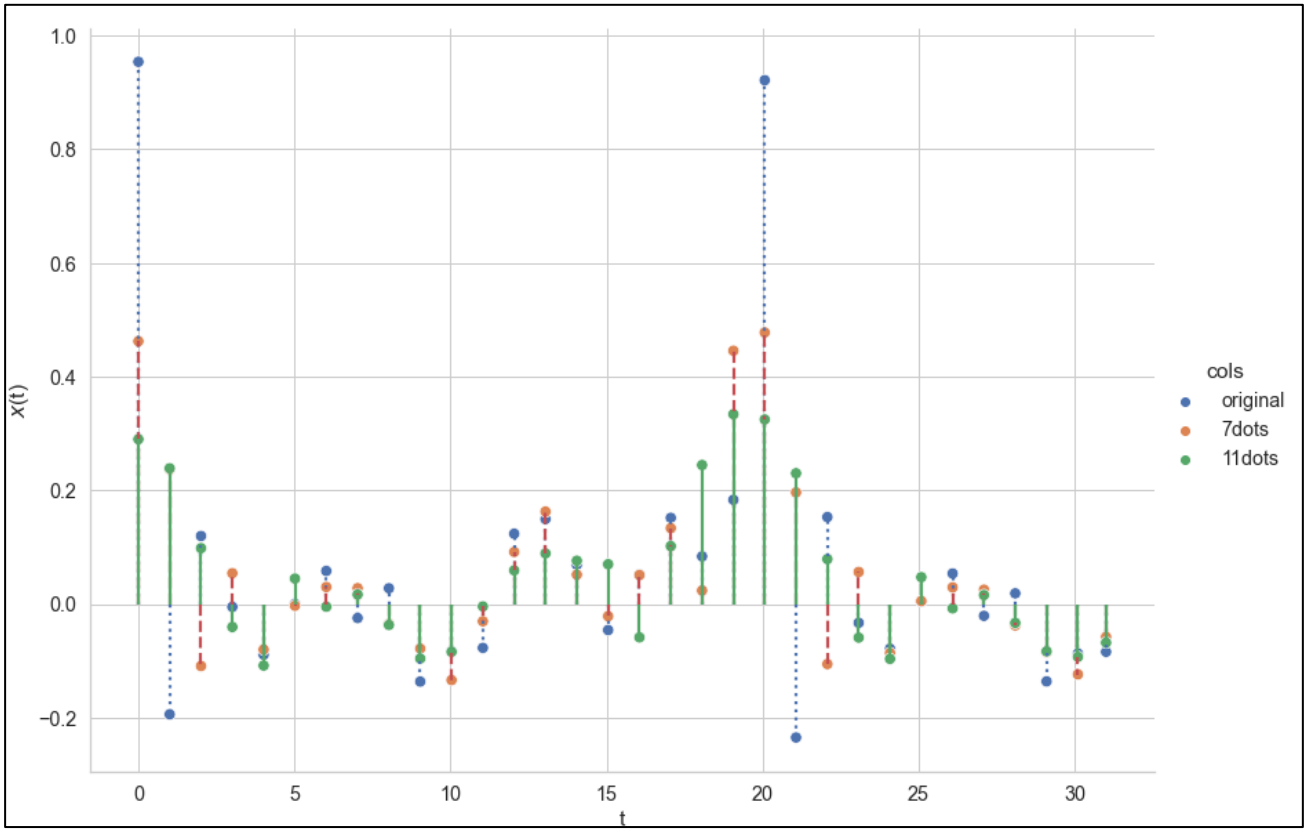

Рисунок 14 – Сглаживание полиномом 4-й степени по 7-ми и 11-ти точкам

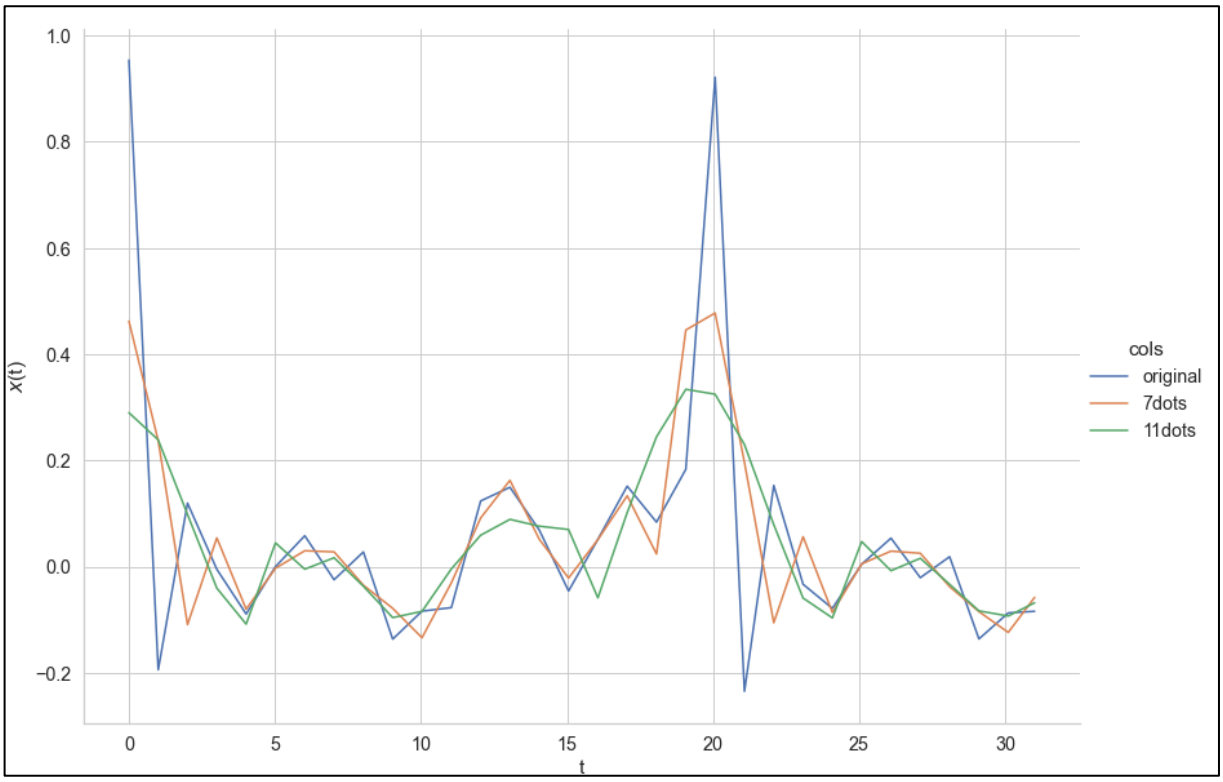

Рисунок 15 – Сглаживание полиномом 4-й степени по 7-ми и 11-ти точкам

Также были получены спектры сигнала после фильтрации. На рис. 16 и 17 представлено сравнение спектров исходного сигнала, сигнала после сглаживания полиномом 4-й степени по 7-ми и 11-ти точкам.

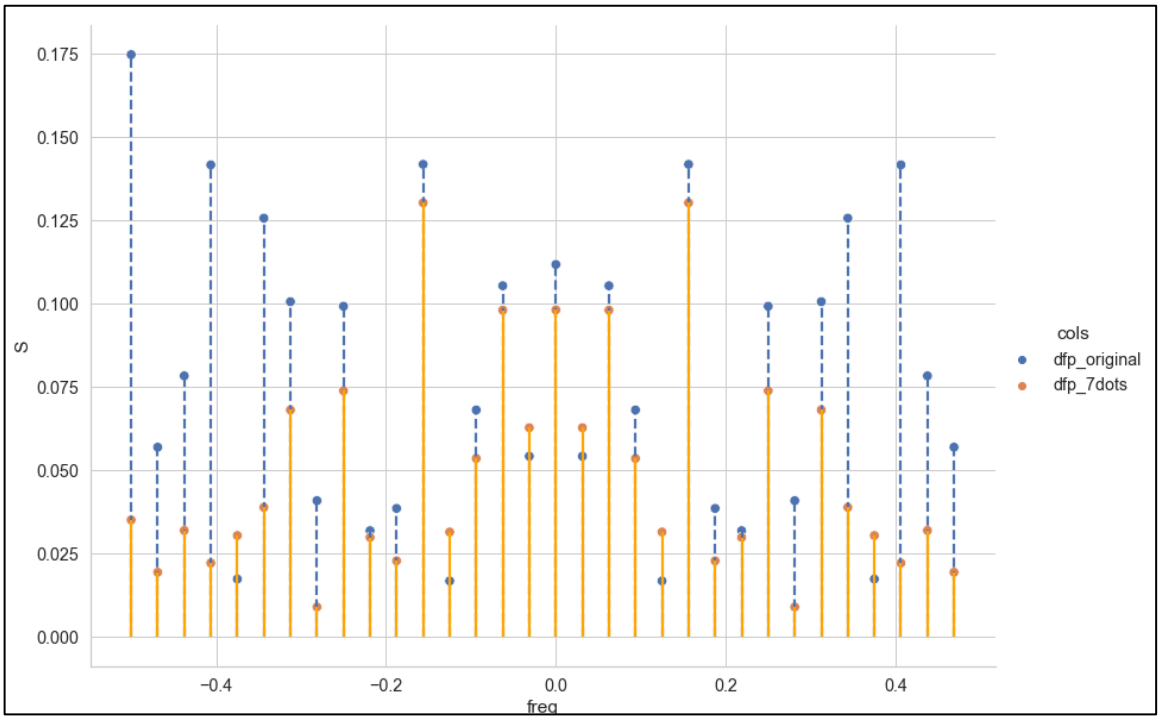

Рисунок 16 – Спектры сигналов после сглаживание фильтром 4-го порядка

по 7-ми точками

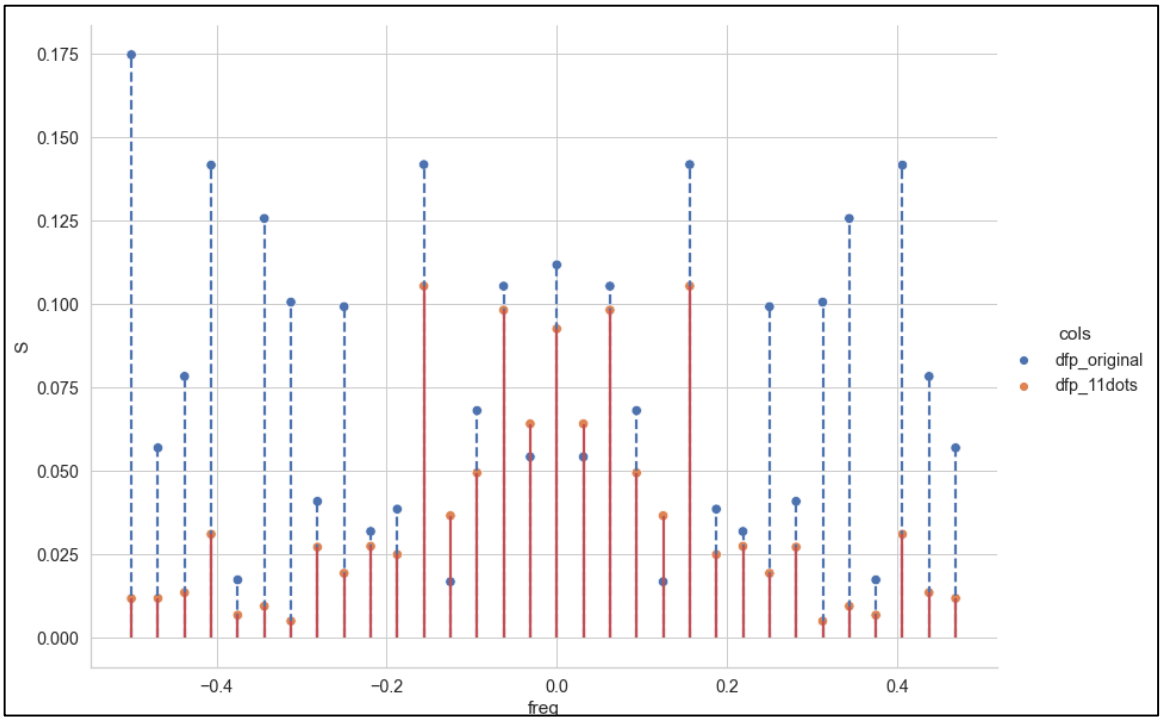

Рисунок 17 – Спектры сигналов после сглаживание фильтром 4-го порядка по 9-ти точками

На рис. 18 представлены графики передаточных функций фильтра сглаживания полиномом четвертой степени.

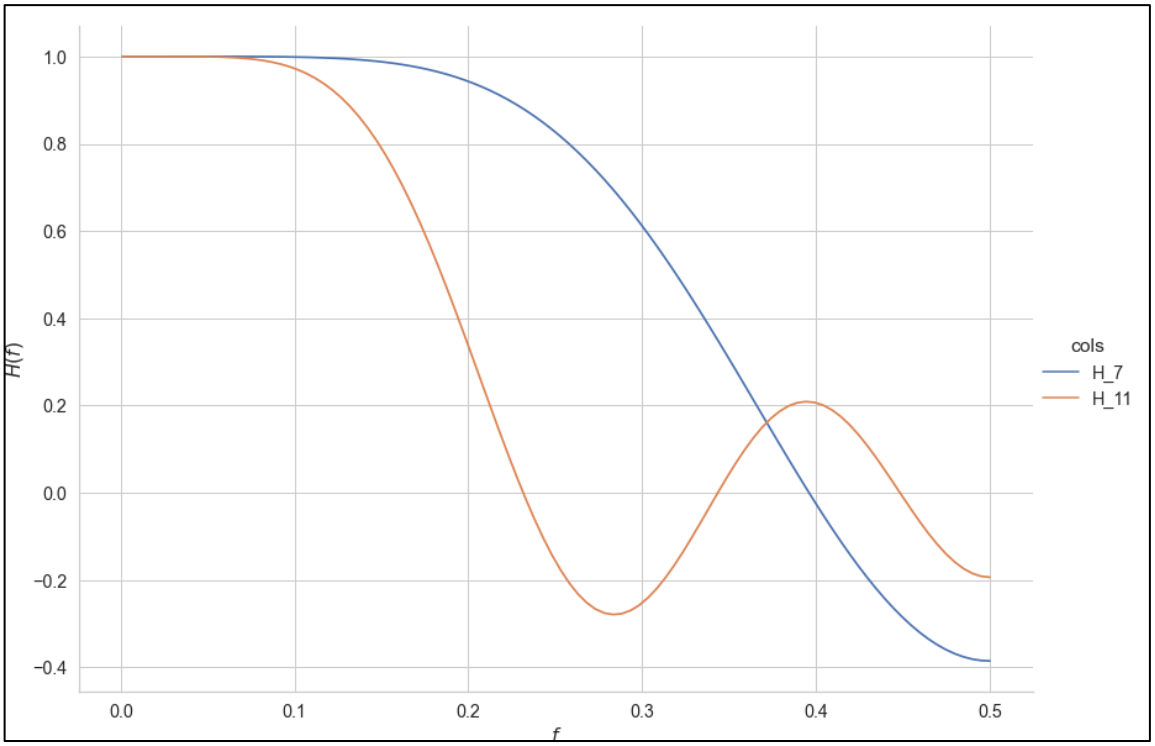

Рисунок 18 – Передаточная функция фильтра 4-го порядка

Полученные спектры после фильтрации схожи с таковыми при сглаживании полиномом 2-й степени, однако здесь наблюдается еще более широкая пологая часть пропускания низких частот почти без ослабления.

При сглаживании по 11-ти точкам можно увидеть, что полоса пропускания становится уже, что видно на спектре.

Дискретный фильтр, соответствующий численному дифференцированию 1-го порядка

Передаточная функция имеет вид:

$$H(\omega) = \frac{e^{i\omega} - e^{-i\omega}}{2} = i\sin(\omega)$$

На рис. 19 и 20 представлено сравнение исходного сигнала и сигнала после применения сглаживания фильтром дифференцирования первого порядка:

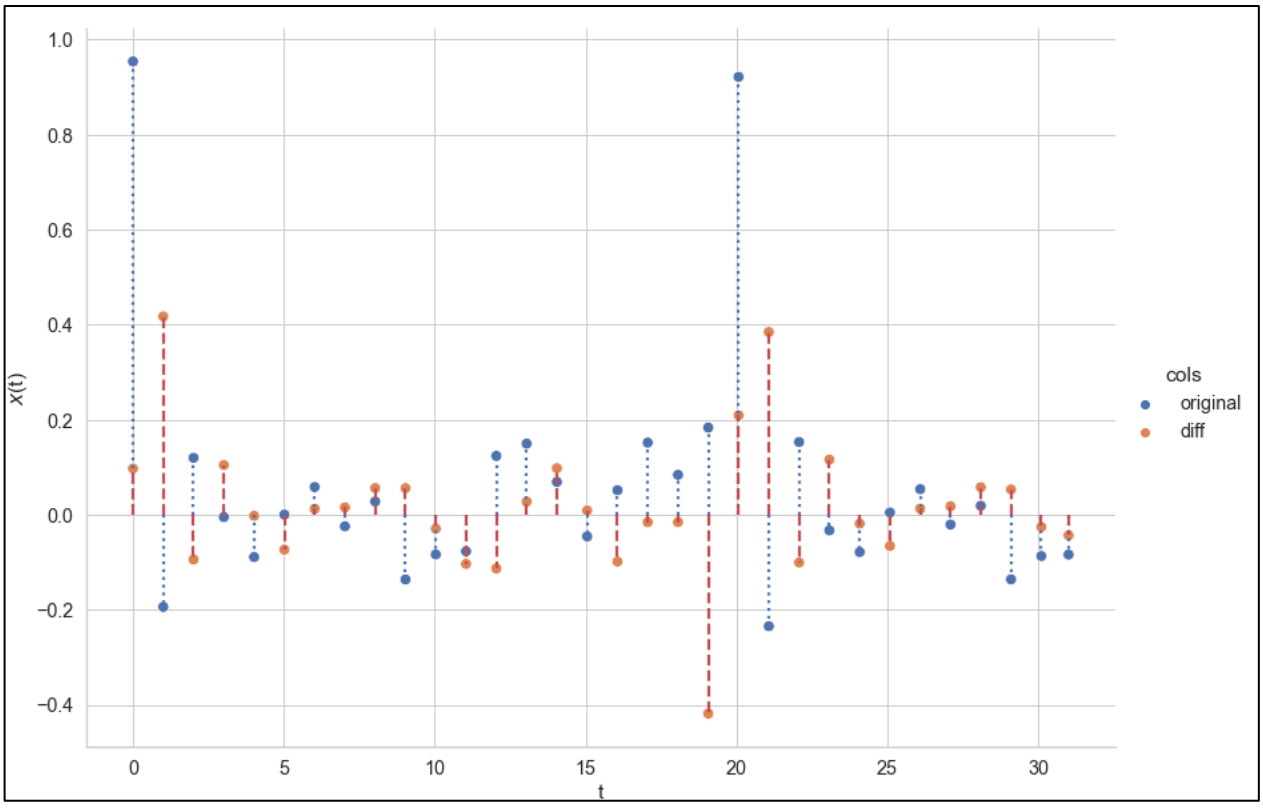

Рисунок 19 – Сглаживание фильтром дифференцирования первого порядка

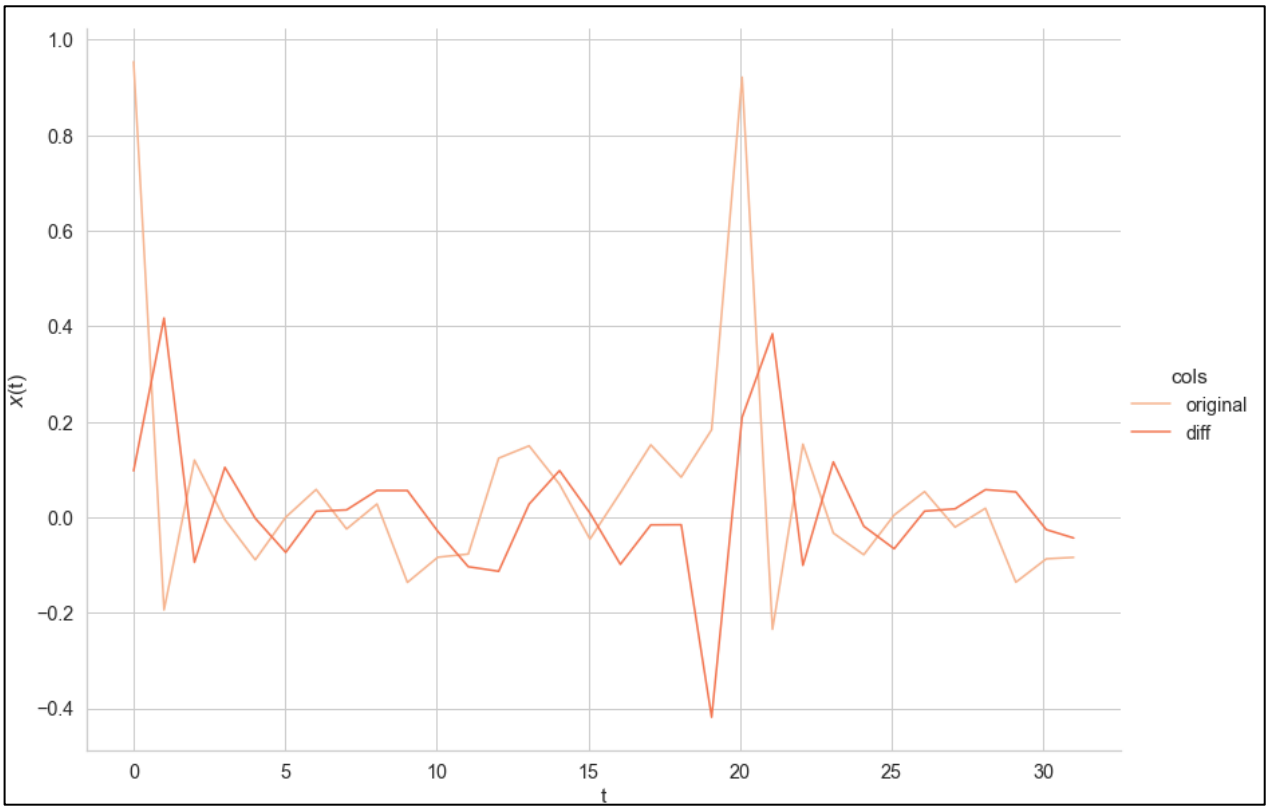

Рисунок 20 – Сглаживание фильтром дифференцирования первого порядка

Также был получен спектр сигнала после фильтрации. Сравнение спектров исходного и фильтрованного сигнала представлено на рис. 21.

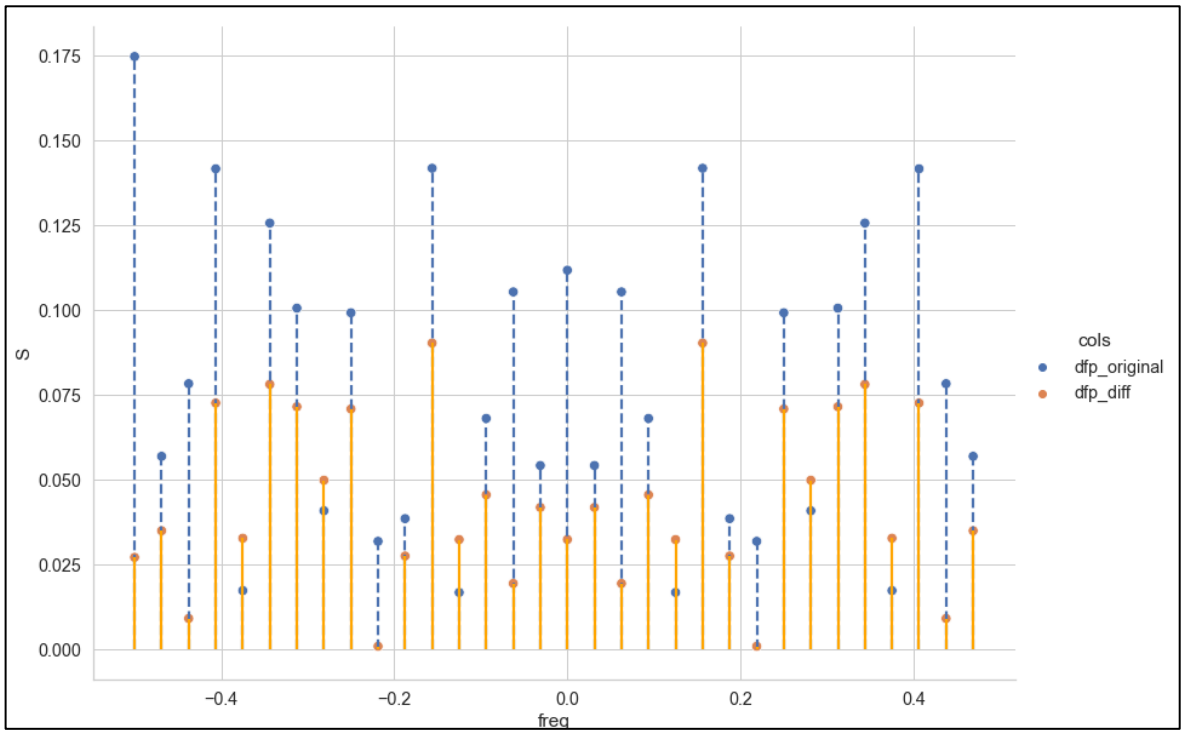

Рисунок 21 – Сравнение спектров исходного и фильтрованного сигнала

График передаточной функции представлен на рис. 22.

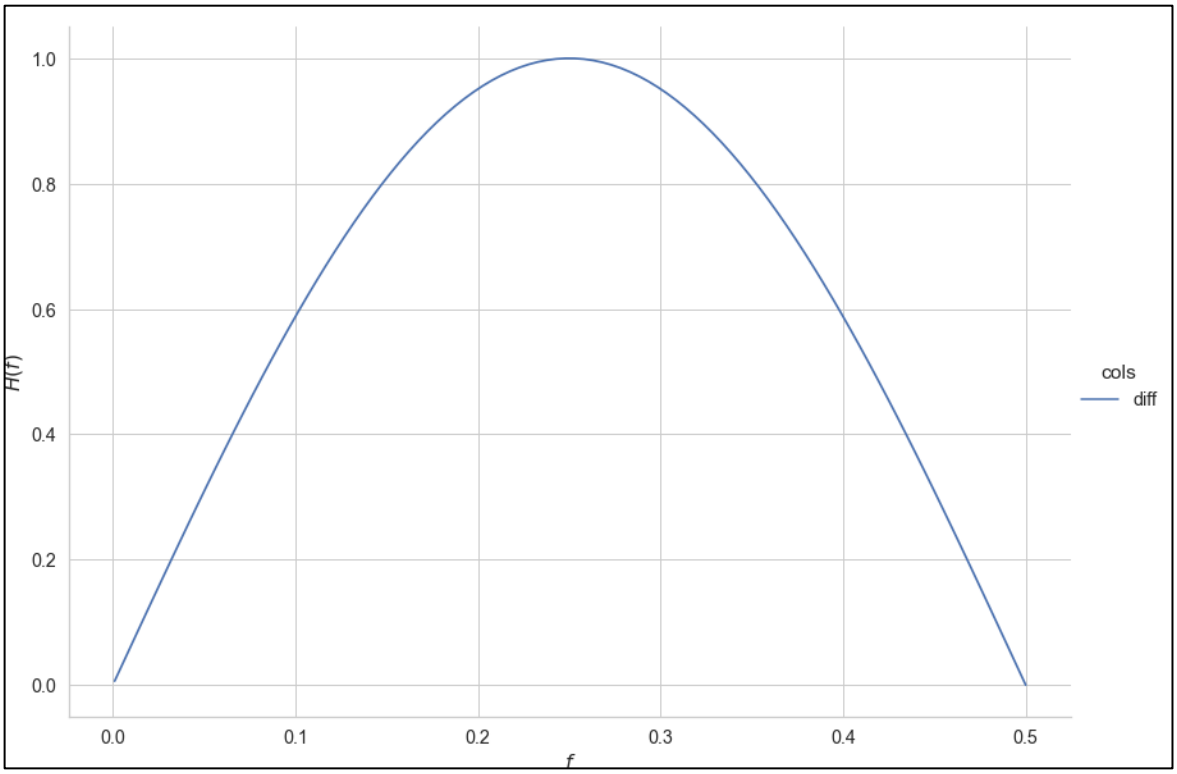

Рисунок 22 – Передаточная функция

По графику спектра видно, что средние частоты действительно усиливаются, при этом низкие и высокие частоты сглаживаются, что подтверждается графиком передаточной функции.

Дискретный фильтр, соответствующий численному интегрированию (прямоугольников, трапеций, Симпсона)

Передаточная функция для рекурсивного фильтра, соответствующего численному интегрированию прямоугольников:

$$H(\omega) = \frac{e^{\frac{i\omega}{2}}}{e^{i\omega} - 1} = \frac{1}{2isin\frac{\omega}{2}}$$

Передаточная функция для рекурсивного фильтра, соответствующего численному интегрированию трапеций:

$$H(\omega) = \frac{e^{i\omega} + 1}{2(e^{i\omega} - 1)} = \frac{cos\frac{\omega}{2}}{2isin\frac{\omega}{2}}$$

Передаточная функция для рекурсивного фильтра, соответствующего численному интегрированию по формуле Симпсона:

$$H(\omega) = \frac{e^{-i\omega} + 4 + e^{i\omega}}{3(e^{i\omega} - e^{-i\omega})} = \frac{cos\,\omega + 2}{3isin\omega}$$

На рис. 23 представлено сравнение исходного сигнала и сигнала после применения сглаживания фильтрами интегрирования.

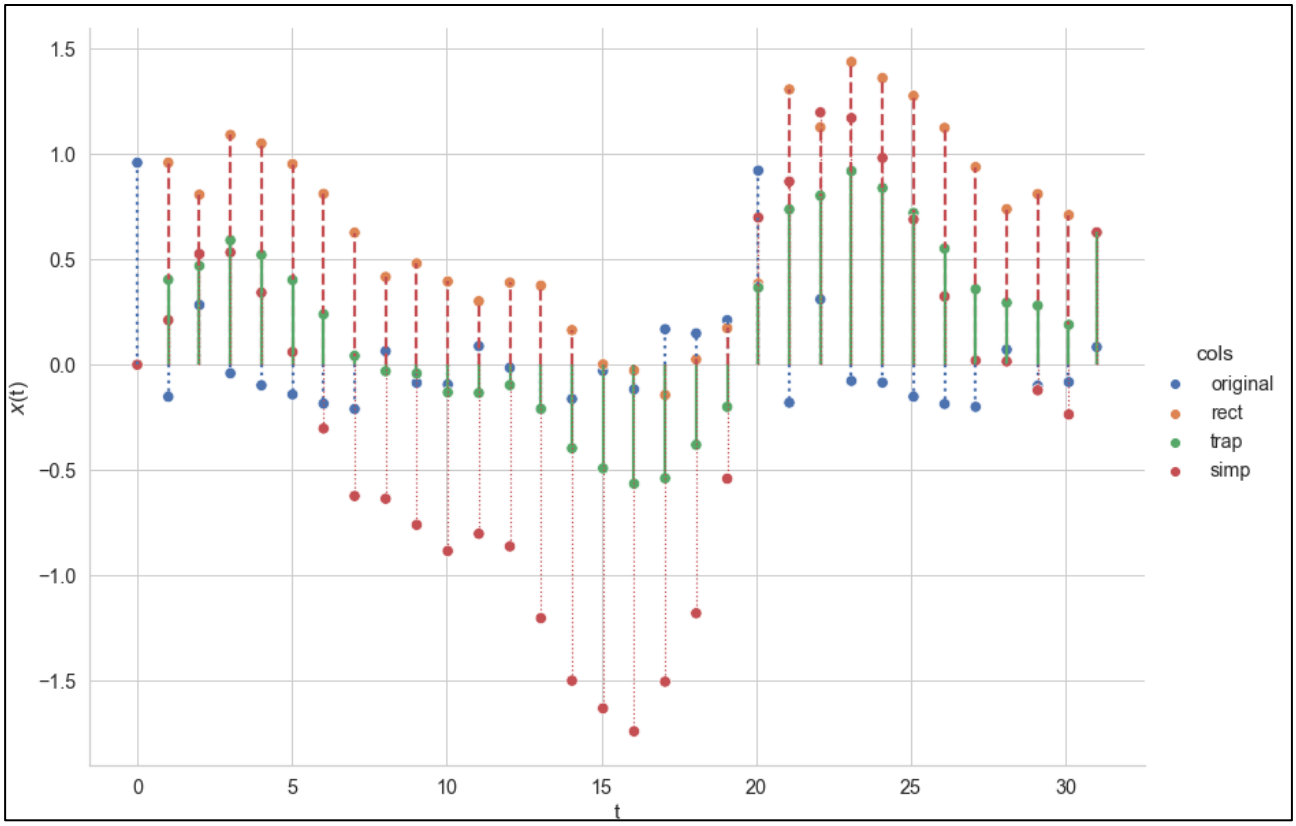

Рисунок 23 – Сравнение исходного и отфильтрованного сигналов

Также был получен спектр сигнала после фильтрации. Сравнение спектров исходного и фильтрованного сигнала представлено на рис. 24, 25 и 26.

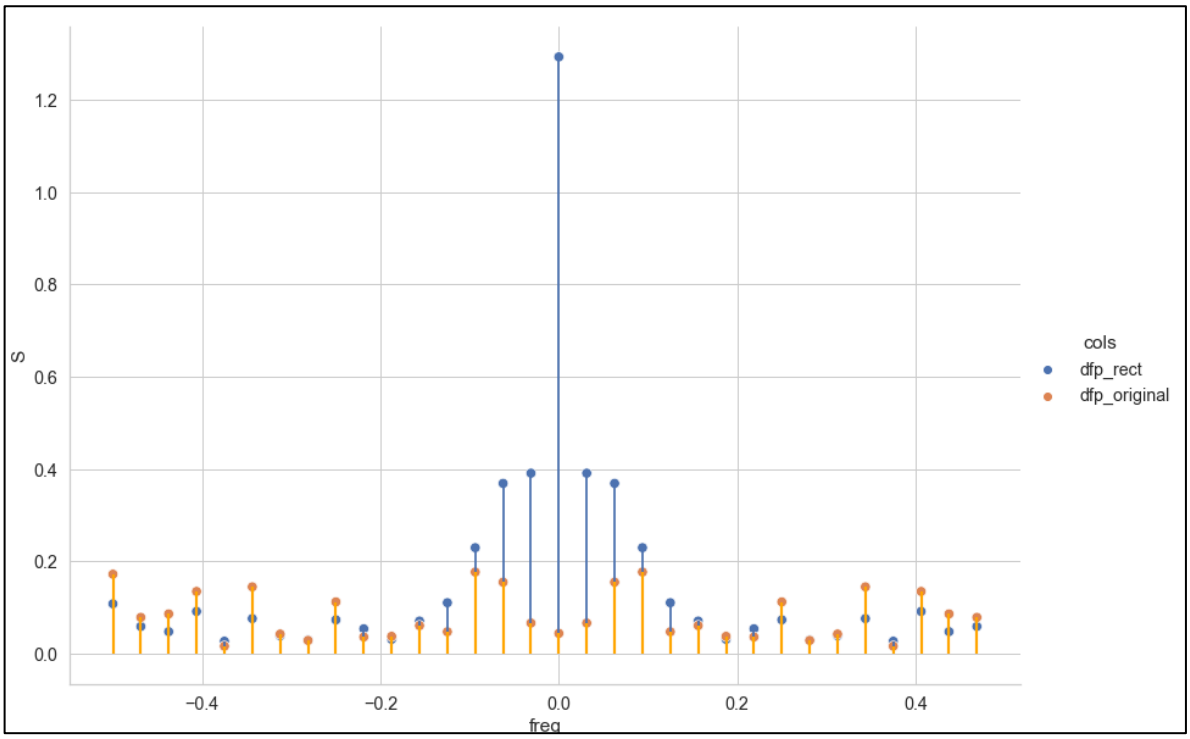

Рисунок 24 – Спектр исходного и отфильтрованных сигналов

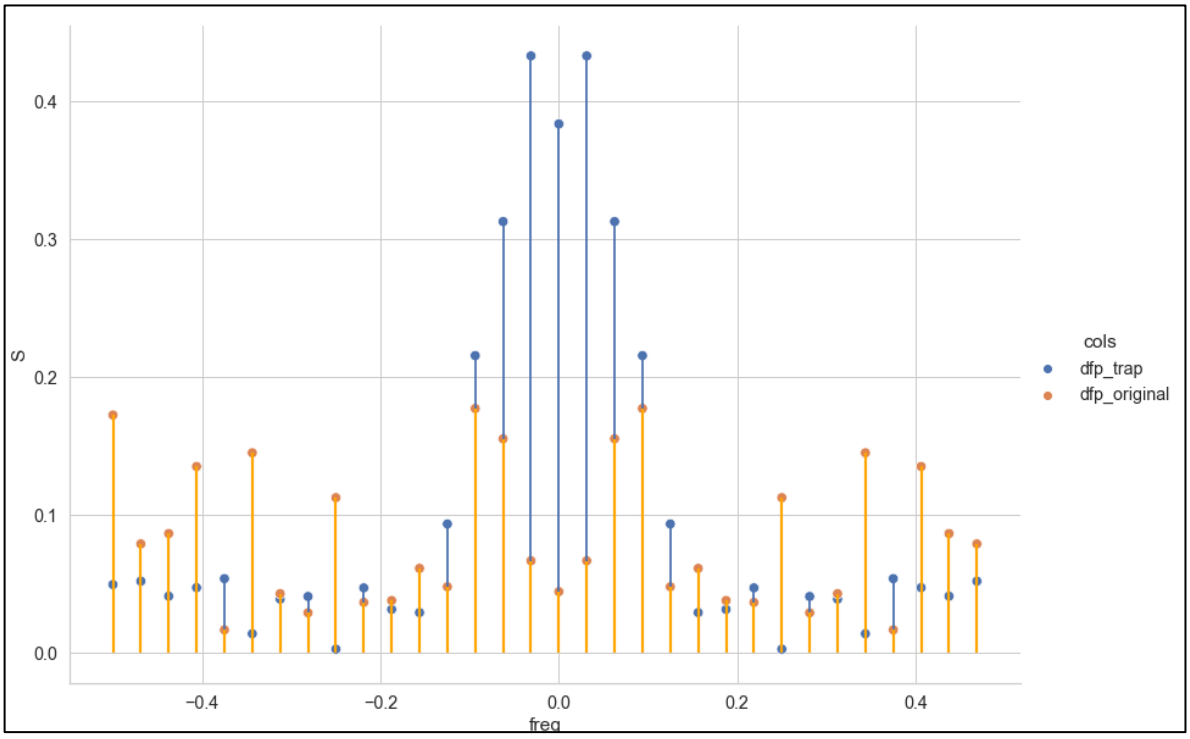

Рисунок 25 – Спектр исходного и отфильтрованных сигналов

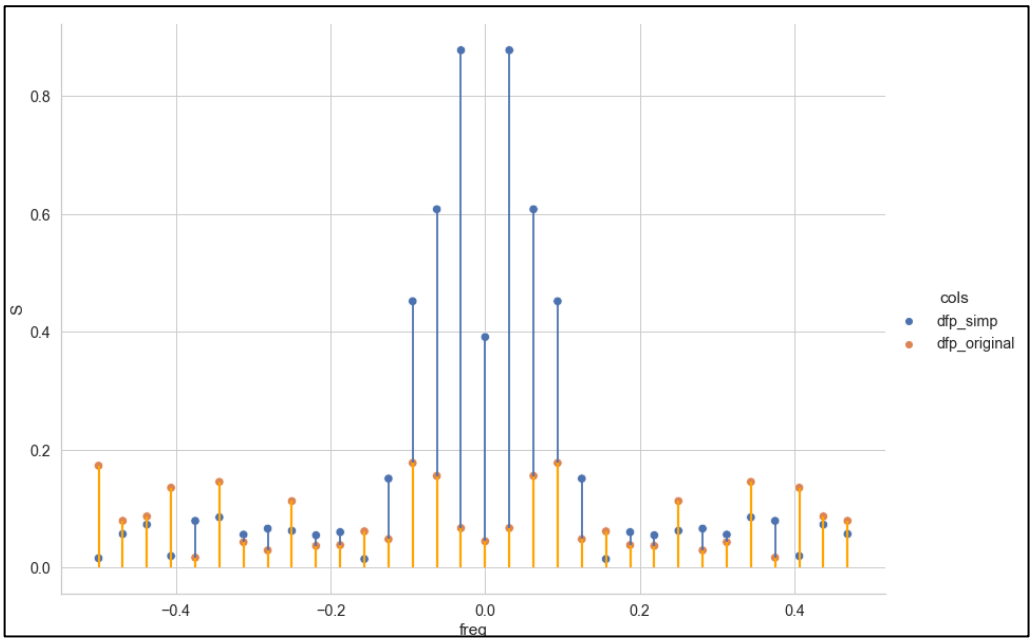

Рисунок 26 – Спектр исходного и отфильтрованных сигналов

Из спектров можно увидеть отличие сглаживаний интегрированием с помощью различных формул: сглаживание по формулам прямоугольников значительно усиливает низкие частоты в небольшом диапазоне; сглаживание по формулам трапеции усиливает низкие частоты, но в диапазоне шире и с меньшим усилением; сглаживание по формулам Симпсона усиливает весь спектр частот, и особенно нижние. Графики передаточных функций представлены на рис. 27.

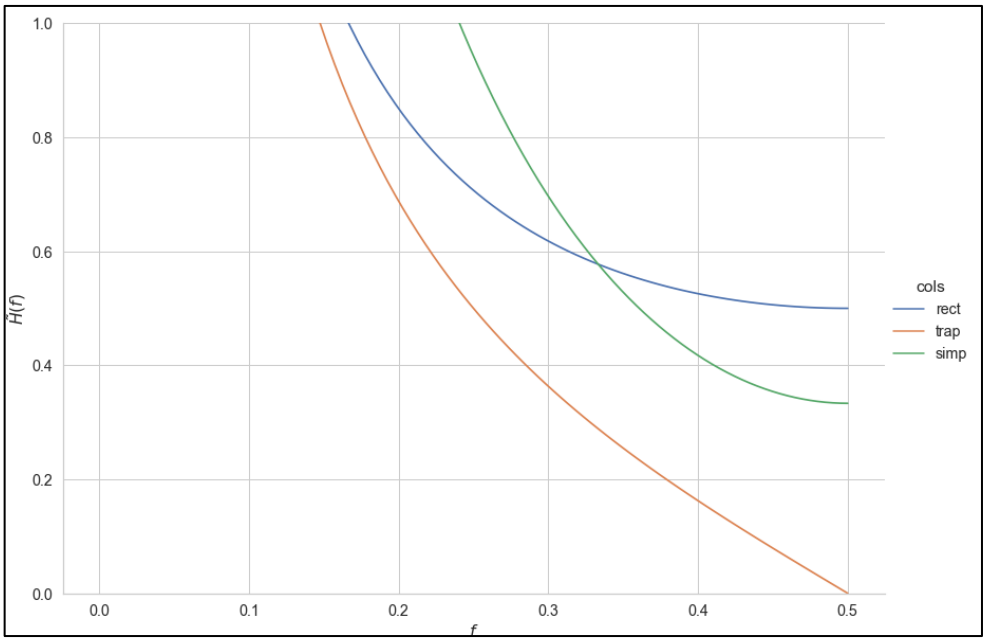

Рисунок 27 – Передаточная функция

Выводы

В результате выполнения лабораторной работы был сгенерирован аналоговый сигнал, дискретизирован, построен спектр дискретного сигнала. Было выяснено, что спектр дискретного сигнала симметричен относительно 0, в спектре представлено множество частот. Были применены фильтры: линейного сглаживания по 5, 9 узлам, сглаживания полиномом 2-й по 5, 9 узлам и 4-й степени по 7, 11 узлам, построены графики сигнала и спектра, передаточной функции. В результате по спектру было определено, что обеспечивается фильтрация высоких частот и ширина полосы пропускания уменьшается с увеличением числа точек. Был применен дискретный фильтр, соответствующий численному дифференцированию, построен график сигнала, спектр, передаточная функция. В результате графики спектра и передаточной функции показали, что фильтр имеет полосу пропускания в области средних частот, уменьшает амплитуду низких и высоких частот. Были применены фильтры, соответствующие численному интегрированию по формулам прямоугольников, трапеций, Симпсона. Фильтры отличаются усилением сигнала в области низких частот.

ПРИЛОЖЕНИЕ А.
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
import pandas as pd
import numpy as np
from scipy.fftpack import fft, ifft, fftshift, rfft
import seaborn as sns
import matplotlib.pyplot as plt
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"


# ## Пункт 1

def signal(n):
    t, Y = np.linspace(0, 31, n), 0
    A = np.random.randint(1, 11, 11)
    W = np.arange(0, 1.01*np.pi, 0.1*np.pi)
    Phi = np.random.uniform(0, 0.5, [11,])
    for a, w, phi in zip(A, W, Phi):
        Y += a*np.cos(w*t+phi)
    Y = Y/np.sum(A)
    return t, Y


# ## Пункт 2

t, Y = signal(310)
df_a = pd.DataFrame({'t': t, 'Y': Y})

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_a, x='t', y='Y', linewidth=1.7, color='r',
                 kind='line', height=8.27, aspect=11.7/8.27)
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/2_1.png')
plt.show()

df = df_a.iloc[np.concatenate((np.linspace(0, 300, 31), [309]), axis=0)]

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df, x='t', y='Y',  color='b', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df['t'], ymin=0, ymax=df['Y'], linewidth=2.2)
```

```
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/2_2.png')
plt.show()


# ## Пункт 3

def DFT(x):
    x = np.asarray(x, dtype=float)
    N = x.shape[0]
    n = np.arange(N)
    k = n.reshape((N, 1))
    M = np.exp(-2j * np.pi * k * n / N)
    return np.dot(M, x)

df_3 = df.copy()
df_3['Y'] = 2 * np.abs(fft(df_3['Y'].values)) / 32

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_3, x='t', y='Y',  color='b', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_3['t'], ymin=0, ymax=df_3['Y'], linewidth=2.2)
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/3.png')
plt.show()


# ## Пункт 5
#
# ### Линейное сглаживание по 5 и 9 точкам

df_47 = df.copy()
df_47['original'] = df_47['Y']
df_47['5dots'] = np.convolve(df['Y'], np.ones(5), 'same') / 5
df_47['9dots'] = np.convolve(df['Y'], np.ones(9), 'same') / 9
df_47m5    =    df_47.melt(id_vars='t',    value_vars=['original',    '5dots'],
var_name='cols', value_name='vals')
df_47m9    =    df_47.melt(id_vars='t',    value_vars=['original',    '9dots'],
var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_47m5, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_47['t'],    ymin=0,    ymax=df_47['original'],    linewidth=2.2,
linestyles='--')
```

```
plt.vlines(x=df_47['t'],     ymin=0,     ymax=df_47['5dots'],     linewidth=2.2,
color='orange')
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/5_1.png')
plt.show()

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_47m9, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_47['t'],    ymin=0,    ymax=df_47['original'],    linewidth=2.2,
linestyles='--')
plt.vlines(x=df_47['t'],     ymin=0,     ymax=df_47['9dots'],     linewidth=2.2,
color='orange')
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/5_2.png')
plt.show()


# ## Пункт 6

df_47['dfp_original'] = 2 * np.abs(fft(df_47['Y'].values)) / 32
df_47['dfp_5dots'] = 2 * np.abs(fft(df_47['5dots'].values)) / 32
df_47['dfp_9dots'] = 2 * np.abs(fft(df_47['9dots'].values)) / 32
df_47m5dfp = df_47.melt(id_vars='t', value_vars=['dfp_original', 'dfp_5dots'],
var_name='cols', value_name='vals')
df_47m9dfp = df_47.melt(id_vars='t', value_vars=['dfp_original', 'dfp_9dots'],
var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_47m5dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_47['t'],   ymin=0,   ymax=df_47['dfp_original'],   linewidth=2.2,
linestyles='--')
plt.vlines(x=df_47['t'],    ymin=0,    ymax=df_47['dfp_5dots'],    linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/6_1.png')
plt.show()

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_47m9dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_47['t'],   ymin=0,   ymax=df_47['dfp_original'],   linewidth=2.2,
linestyles='--')
```

```
plt.vlines(x=df_47['t'],   ymin=0,   ymax=df_47['dfp_9dots'],   linewidth=2.2,
color='r')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/6_2.png')
plt.show()


# ## Пункт 8
#
# ### А. Сглаживание полиномом 2-ой степени по 5 и 9 узлам

df_8a = df.copy()
df_8a['original'] = df_8a['Y']
df_8a = df_8a.drop(['Y'], axis=1)
df_8a['5dots'] = np.convolve(df['Y'], np.array([-3, 12, 17, 12, -3]), 'same') /
35
df_8a['9dots'] = np.convolve(df['Y'], np.array([-21, 14, 39, 54, 59, 54, 39, 14,
-21]), 'same') / 231
df_8a_m = df_8a.melt(id_vars='t', var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8a_m, x='t', y='vals', hue='cols',
                 kind='line', height=8.27, aspect=11.7/8.27)
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/8a_1.png')
plt.show()

df_8a['dfp_original'] = 2 * np.abs(fft(df_8a['original'].values)) / 32
df_8a['dfp_5dots'] = 2 * np.abs(fft(df_8a['5dots'].values)) / 32
df_8a['dfp_9dots'] = 2 * np.abs(fft(df_8a['9dots'].values)) / 32
df_8a_m5dfp = df_8a.melt(id_vars='t', value_vars=['dfp_original', 'dfp_5dots'],
var_name='cols', value_name='vals')
df_8a_m9dfp = df_8a.melt(id_vars='t', value_vars=['dfp_original', 'dfp_9dots'],
var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8a_m5dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8a['t'],   ymin=0,   ymax=df_8a['dfp_original'],   linewidth=2.2,
linestyles='--')
plt.vlines(x=df_8a['t'],   ymin=0,   ymax=df_8a['dfp_5dots'],   linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8a_2.png')
plt.show()
```

```
sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8a_m9dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8a['t'],   ymin=0,   ymax=df_8a['dfp_original'],   linewidth=2.2,
linestyles='--')
plt.vlines(x=df_8a['t'],    ymin=0,    ymax=df_8a['dfp_9dots'],    linewidth=2.2,
color='r')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8a_3.png')
plt.show()


# ### B. Сглаживание полиномом 4-ой степени по 7 и 11 узлам

df_8b = df.copy()
df_8b['original'] = df_8b['Y']
df_8b = df_8b.drop(['Y'], axis=1)
df_8b['7dots'] = np.convolve(df['Y'], np.array([5, -30, 75, 131, 75, -30, 5]),
'same') / 231
df_8b['11dots'] = np.convolve(df['Y'], np.array([13, -45, -10, 60, 120, 143, 120,
60, -10, -45, 13]), 'same') / 429
df_8b_m = df_8b.melt(id_vars='t', var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8b_m, x='t', y='vals', hue='cols',
                 kind='line', height=8.27, aspect=11.7/8.27)
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/8b_1.png')
plt.show()

df_8b['dfp_original'] = 2 * np.abs(fft(df_8b['original'].values)) / 32
df_8b['dfp_7dots'] = 2 * np.abs(fft(df_8b['7dots'].values)) / 32
df_8b['dfp_11dots'] = 2 * np.abs(fft(df_8b['11dots'].values)) / 32
df_8b_m7dfp = df_8b.melt(id_vars='t', value_vars=['dfp_original', 'dfp_7dots'],
var_name='cols', value_name='vals')
df_8b_m11dfp      =      df_8b.melt(id_vars='t',      value_vars=['dfp_original',
'dfp_11dots'], var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8b_m7dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8b['t'],   ymin=0,   ymax=df_8b['dfp_original'],   linewidth=2.2,
linestyles='--')
```

```
plt.vlines(x=df_8b['t'],    ymin=0,    ymax=df_8b['dfp_7dots'],    linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8b_2.png')
plt.show()

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8b_m11dfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8b['t'],   ymin=0,   ymax=df_8b['dfp_original'],   linewidth=2.2,
linestyles='--')
plt.vlines(x=df_8b['t'],    ymin=0,    ymax=df_8b['dfp_11dots'],    linewidth=2.2,
color='r')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8b_3.png')
plt.show()


# ### C. Дискретный фильтр, соответствующий численному дифференцированию 1-го
порядка

df_8c = df.copy()
df_8c['original'] = df_8c['Y']
df_8c = df_8c.drop(['Y'], axis=1)
df_8c['diff'] = np.convolve(df['Y'], np.array([-1, 0, 1]), 'same') / 2
df_8c_m = df_8c.melt(id_vars='t', var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",     palette='rocket_r',     context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8c_m, x='t', y='vals', hue='cols',
                 kind='line', height=8.27, aspect=11.7/8.27)
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/8c_1.png')
plt.show()

df_8c['dfp_original'] = 2 * np.abs(fft(df_8c['original'].values)) / 32
df_8c['dfp_diff'] = 2 * np.abs(fft(df_8c['diff'].values)) / 32
df_8c_m_diffdfp     =     df_8c.melt(id_vars='t',     value_vars=['dfp_original',
'dfp_diff'], var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8c_m_diffdfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8c['t'],   ymin=0,   ymax=df_8c['dfp_original'],   linewidth=2.2,
linestyles='--')
```

```
plt.vlines(x=df_8c['t'],    ymin=0,    ymax=df_8c['dfp_diff'],    linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8c_2.png')
plt.show()


# ### D. Дискретный фильтр, соответствующий численному интегрированию
(прямоугольников, трапеций, Симпсона)

def rect(orig):
    integr = np.empty(len(orig))
    integr[0] = orig[0]
    for i in range(1, len(orig)):
        integr[i] = integr[i-1] + orig[i]
    return integr

def simpson(orig):
    integr = np.empty(len(orig))
    integr[0] = (0 + 4*orig[0] + orig[1]) / 3
    for i in range(1, len(orig)-1):
        integr[i] = integr[i-1] + (orig[i-1] + orig[i] + 4*orig[i+1]) / 3
    return integr

def trap(orig):
    integr = np.empty(len(orig))
    integr[0] = (orig[0] + orig[1]) / 2
    for i in range(1, len(orig)-1):
        integr[i] = integr[i - 1] + (orig[i] + orig[i+1]) / 2
    return integr

df_8d = df.copy()
df_8d['original'] = df_8d['Y']
df_8d = df_8d.drop(['Y'], axis=1)
df_8d['rect'] = rect(df['Y'].values)
df_8d['trap'] = trap(df['Y'].values)
df_8d['simp'] = simpson(df['Y'].values)
df_8d_m = df_8d.melt(id_vars='t', var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8d_m, x='t', y='vals', hue='cols',
                 kind='line', height=8.27, aspect=11.7/8.27)
ax.set_axis_labels('t', r'$x$(t)')
plt.savefig('pics/8d_1.png')
plt.show()
```

```
df_8d['dfp_original'] = 2 * np.abs(fft(df_8d['original'].values)) / 32
df_8d['dfp_rect'] = 2 * np.abs(fft(df_8d['rect'].values)) / 32
df_8d['dfp_trap'] = 2 * np.abs(fft(df_8d['trap'].values)) / 32
df_8d['dfp_simp'] = 2 * np.abs(fft(df_8d['simp'].values)) / 32
df_8d_mRdfp = df_8d.melt(id_vars='t', value_vars=['dfp_rect', 'dfp_original'],
var_name='cols', value_name='vals')
df_8d_mTdfp = df_8d.melt(id_vars='t', value_vars=['dfp_trap', 'dfp_original'],
var_name='cols', value_name='vals')
df_8d_mSdfp = df_8d.melt(id_vars='t', value_vars=['dfp_simp', 'dfp_original'],
var_name='cols', value_name='vals')

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8d_mRdfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8d['t'],    ymin=0,    ymax=df_8d['dfp_rect'],    linewidth=1.8,
linestyles='solid')
plt.vlines(x=df_8d['t'],   ymin=0,   ymax=df_8d['dfp_original'],   linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8d_2.png')
plt.show()

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8d_mTdfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8d['t'],    ymin=0,    ymax=df_8d['dfp_trap'],    linewidth=1.8,
linestyles='solid')
plt.vlines(x=df_8d['t'],   ymin=0,   ymax=df_8d['dfp_original'],   linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8d_3.png')
plt.show()

sns.set_theme(style="whitegrid",       palette='deep',       context='notebook',
font_scale=1.3)
ax = sns.relplot(data=df_8d_mSdfp, x='t', y='vals', hue='cols', s=70,
                 kind='scatter', height=8.27, aspect=11.7/8.27)
plt.vlines(x=df_8d['t'],    ymin=0,    ymax=df_8d['dfp_simp'],    linewidth=1.8,
linestyles='solid')
plt.vlines(x=df_8d['t'],   ymin=0,   ymax=df_8d['dfp_original'],   linewidth=2.2,
color='orange')
ax.set_axis_labels('freq', r'level')
plt.savefig('pics/8d_4.png')
plt.show()
```