



Федеральное государственное автономное образовательное учреждение высшего образования «Национальный Исследовательский Университет ИТМО»

ЛАБОРАТОРНАЯ РАБОТА №3
ПРЕДМЕТ «ЧАСТОТНЫЕ МЕТОДЫ»
ТЕМА «ЖЕСТКАЯ ФИЛЬТРАЦИЯ»

Лектор: Перегудин А. А.
Практик: Пашенко А. В.
Студент: Румянцев А. А.
Поток: ЧАСТ.МЕТ. 1.3

Факультет: СУиР
Группа: R3241

Санкт-Петербург
2024

Содержание

1 Задание 1. Жесткие фильтры.	2
1.1 Убираем высокие частоты. Фильтр нижних частот.	2
1.2 Убираем низкие частоты. Фильтр верхних частот.	12
1.3 Убираем специфические частоты. Фильтр специфических частот.	28
2 Задание 2. Фильтрация звука.	29
3 Листинги программных реализаций	44

1 Задание 1. Жесткие фильтры.

Зададим такие числа a, t_1, t_2 , что $t_1 < t_2$, и рассмотрим функцию g такую, что $g(t) = a$ при $t \in [t_1, t_2]$ и $g(t) = 0$ при других t .

$$\square a = 2, \quad t_1 = -1.5, \quad t_2 = 2.5, \quad g(t) = \begin{cases} 2, & t \in [t_1, t_2] \\ 0, & \text{otherwise} \end{cases}$$

Выберем интервал времени $T = 10$ и шаг дискретизации $dt = 0.1$. Зададим в python массив времени t от $-0.5 \cdot T$ до $0.5 \cdot T + dt$ с шагом dt и включим последнюю точку. Найдем список значений g и зададим зашумленную версию сигнала как

$$u = g + b \cdot (\text{random}(\text{len}(t)) - 0.5) + c \cdot \sin(d \cdot t);$$

В данном задании мы выполняем жесткую фильтрацию сигнала u . Алгоритм следующий: находится Фурье-образ от сигнала, обнуляются его значения на некоторых диапазонах частот, затем сигнал восстанавливается обратным преобразованием Фурье. Далее строятся графики с помощью программы на языке python. Используемый код с пояснениями находится в отдельной секции.

В задаваемом сигнале параметр a отвечает за высоту, на которую поднимется часть сигнала от нуля, а t_1 и t_2 – начало и конец промежутка с подъемом соответственно. Таким образом, на интервале длины $t_2 - t_1 = 2.5 + 1.5 = 4$, начиная с $t_1 = -1.5$ и заканчивая $t_2 = 2.5$, на высоте $a = 2$ будет находиться часть от всего сигнала, который, в свою очередь, располагается на промежутке $[-0.5 \cdot T, 0.5 \cdot T] = [-5, 5]$ длины $2 \cdot T \cdot 0.5 = 10$. Параметры b, c, d отвечают за шум, присутствующий в сигнале. Далее будут рассмотрены графики и сделаны выводы о влиянии каждого параметра на сам сигнал и на его результат фильтрации.

1.1 Убираем высокие частоты. Фильтр нижних частот.

Возьмем параметр $c = 0$. Далее действуем в соответствии с алгоритмом. Возьмем некоторый диапазон частот $[-\nu_0, \nu_0]$, на котором оставим Фурье-образ сигнала u неизменным, а на остальных частотах обнулим его значения. Такое поведение соответствует фильтру **нижних** частот, так как он пропускает все частоты ниже частоты среза. Построим сравнительные графики исходного и фильтрованного сигналов на некотором интервале $[t_1, t_2]$, а также модуля Фурье-образа исходного и фильтрованного сигналов. Исследуем влияние частоты среза ν_0 и значения параметра b на эффективность фильтрации.

Далее будут приведены рисунки полученных графиков. На каждом графике подписаны выбранные значения b, c, d, ν_0 (хотя, при условии, что $c = 0$, менять или рассматривать параметр d не требуется). Также отмечена легенда – синим цветом обозначается оригинальный сигнал, красным фильтрованный.

Исходя из графиков можно сделать вывод, что значение параметра b отвечает за амплитуду каждой волны. Чем больше значение b , тем зашумленнее, «грязнее» выглядит сигнал, так как амплитуды волн возрастают. Фильтрованный сигнал при больших значениях b также имеет более глубокие ямы и высокие подъемы, то есть испытывает увеличение амплитуды. Наглядно такое возрастание можно наблюдать на графиках модуля Фурье-образа исходного и фильтрованного сигналов – на рисунке 2 амплитуды частот больше прижимаются к линии $A = 0$, а на рисунке 8 они больше стремятся к линии $A = 50$, где A – амплитуда. Само значение параметра b не влияет на эффективность фильтрации в

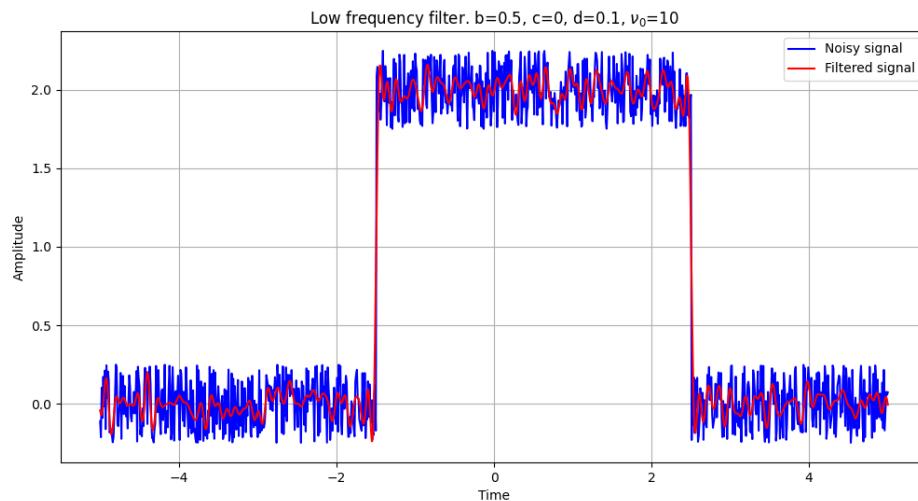


Рис. 1: График исходного и фильтрованного сигналов.

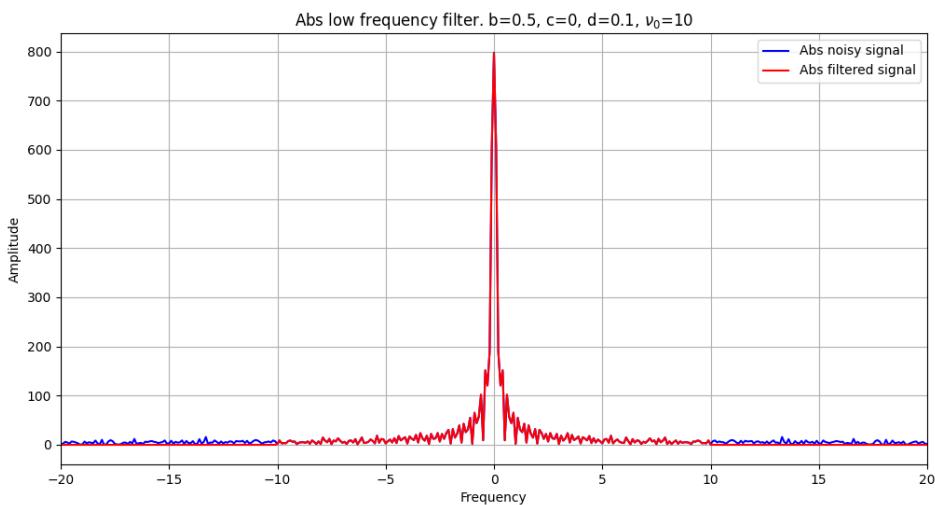


Рис. 2: График модуля Фурье-образа исходного и фильтрованного сигналов.

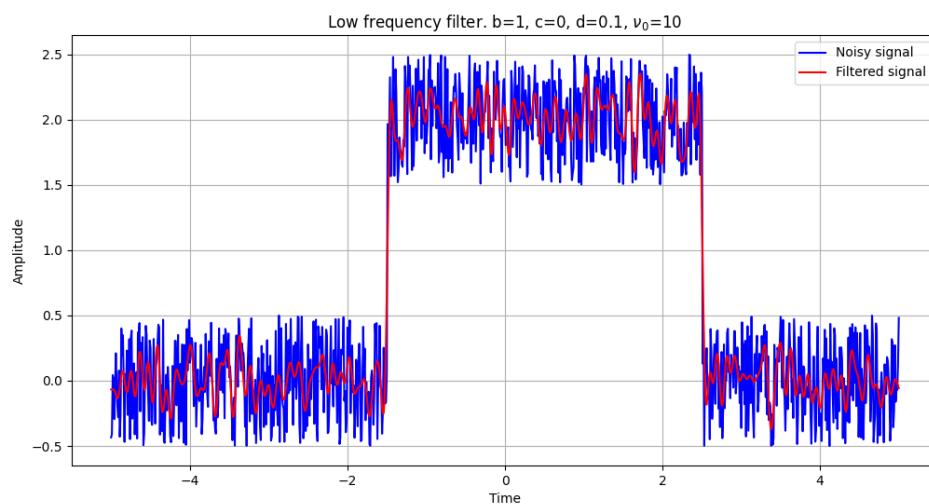


Рис. 3: График исходного и фильтрованного сигналов.

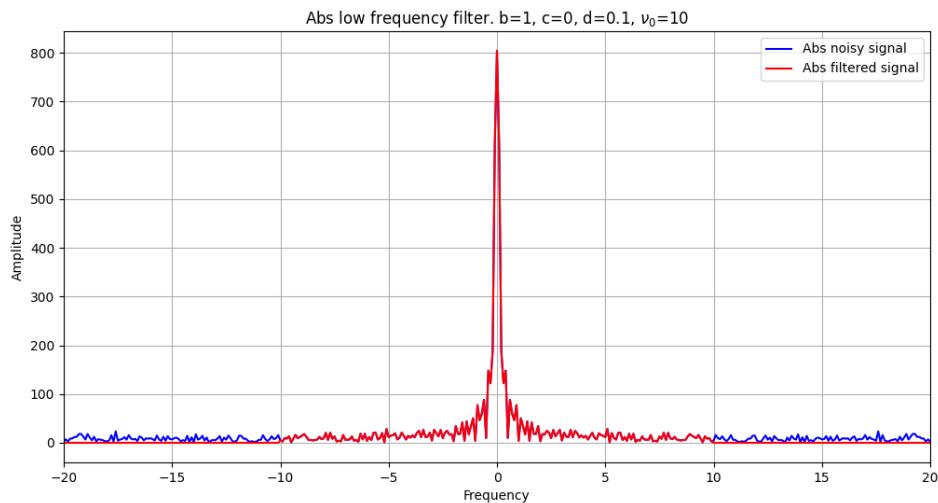


Рис. 4: График модуля Фурье-образа исходного и фильтрованного сигналов.

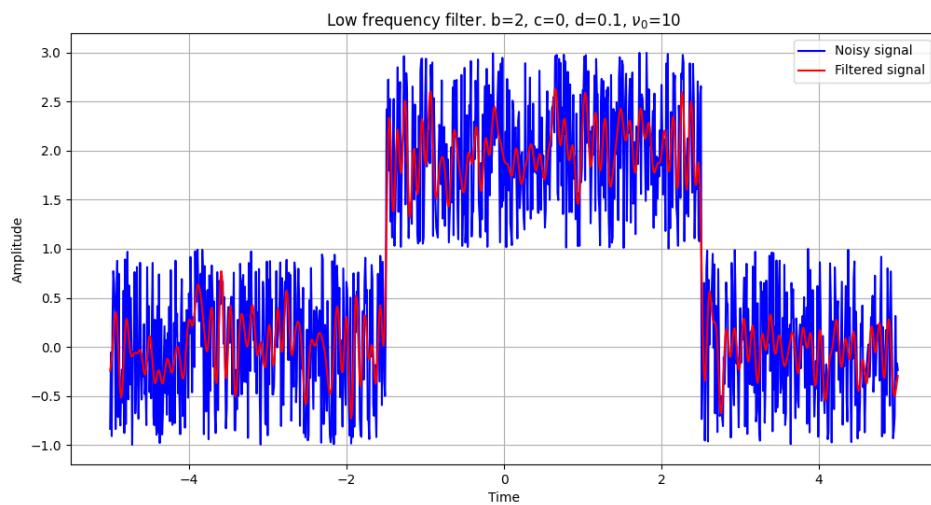


Рис. 5: График исходного и фильтрованного сигналов.

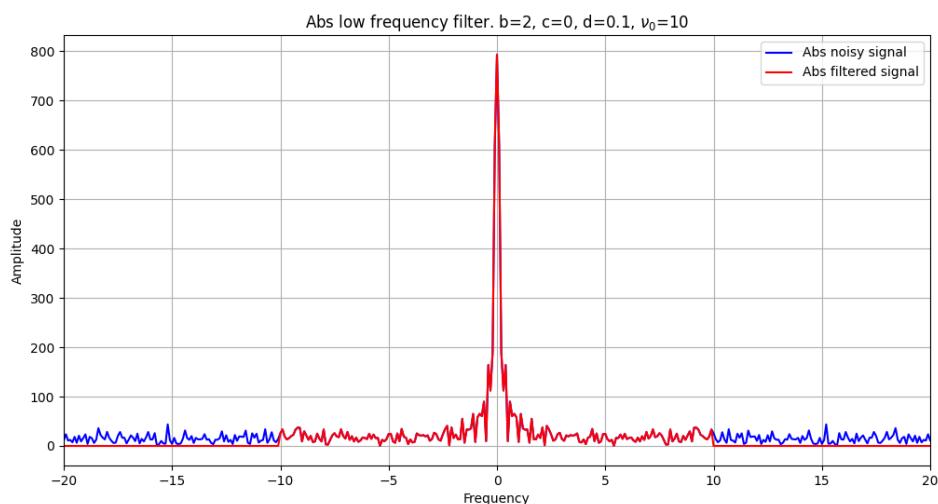


Рис. 6: График модуля Фурье-образа исходного и фильтрованного сигналов.

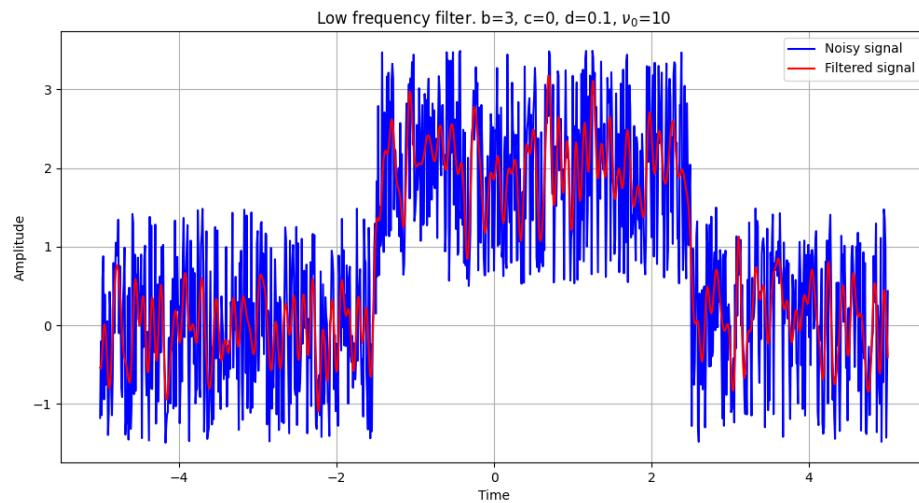


Рис. 7: График исходного и фильтрованного сигналов.

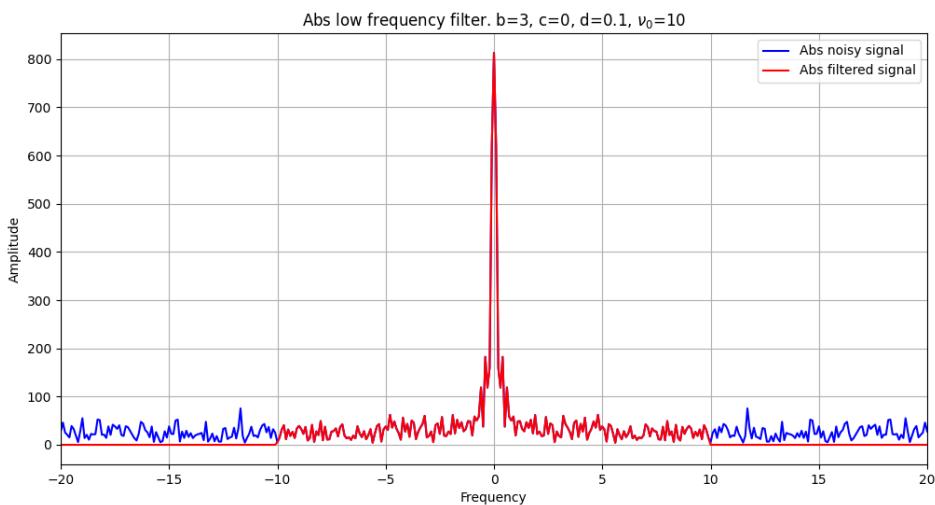


Рис. 8: График модуля Фурье-образа исходного и фильтрованного сигналов.

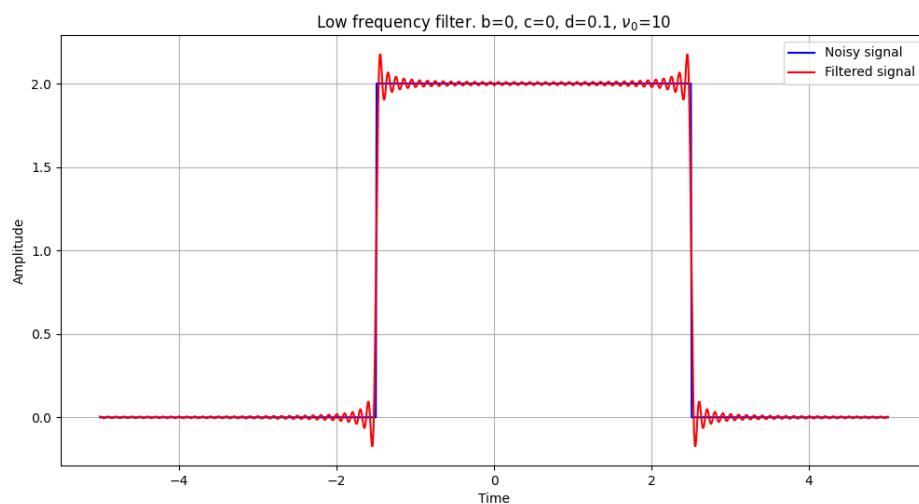


Рис. 9: График исходного и фильтрованного сигналов.

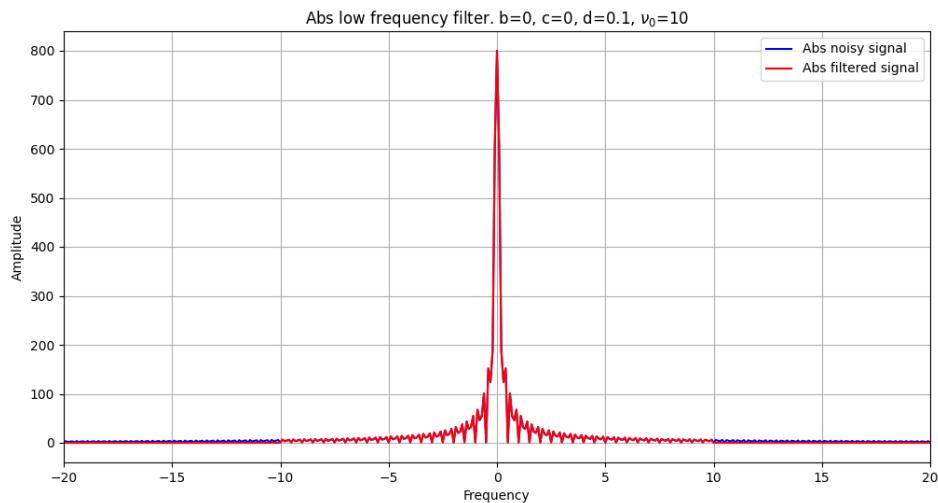


Рис. 10: График модуля Фурье-образа исходного и фильтрованного сигналов.

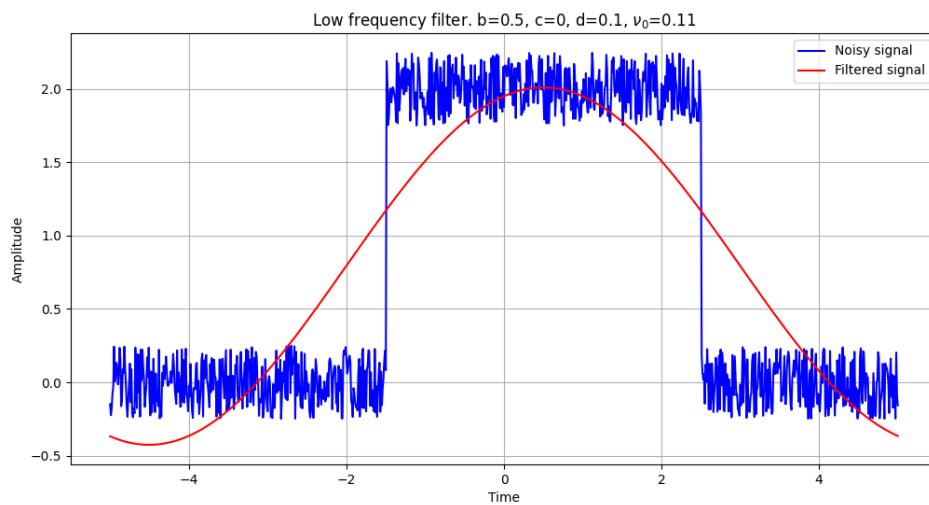


Рис. 11: График исходного и фильтрованного сигналов.

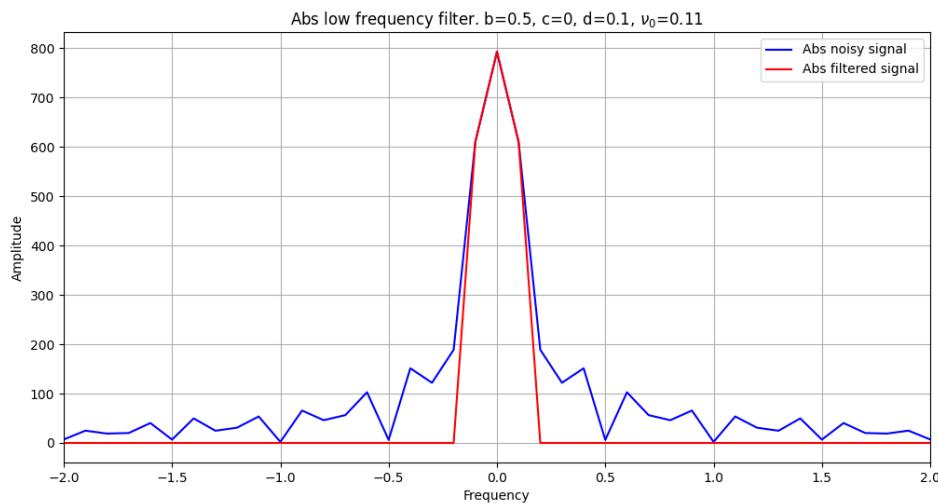


Рис. 12: График модуля Фурье-образа исходного и фильтрованного сигналов.

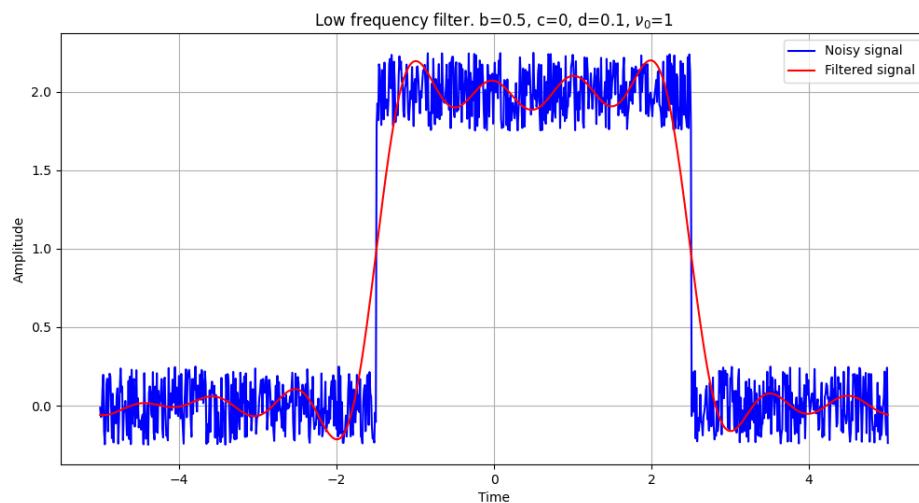


Рис. 13: График исходного и фильтрованного сигналов.

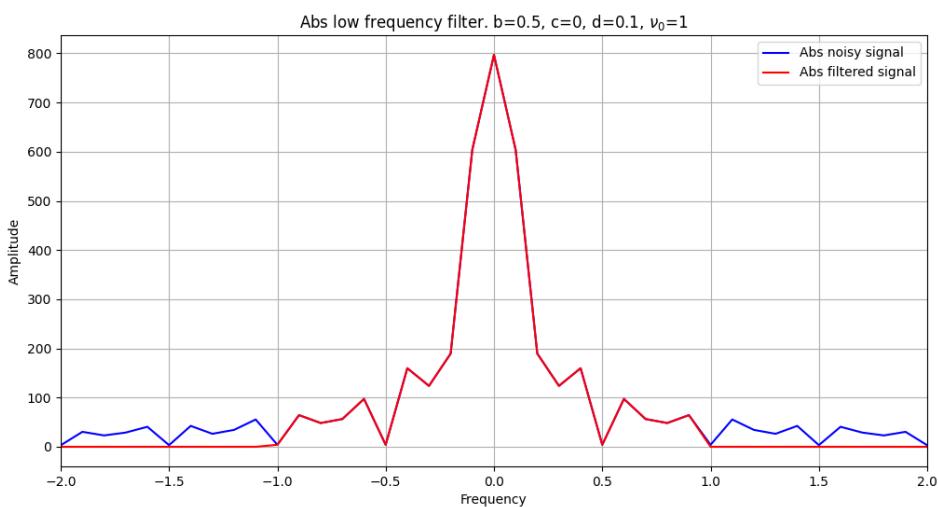


Рис. 14: График модуля Фурье-образа исходного и фильтрованного сигналов.

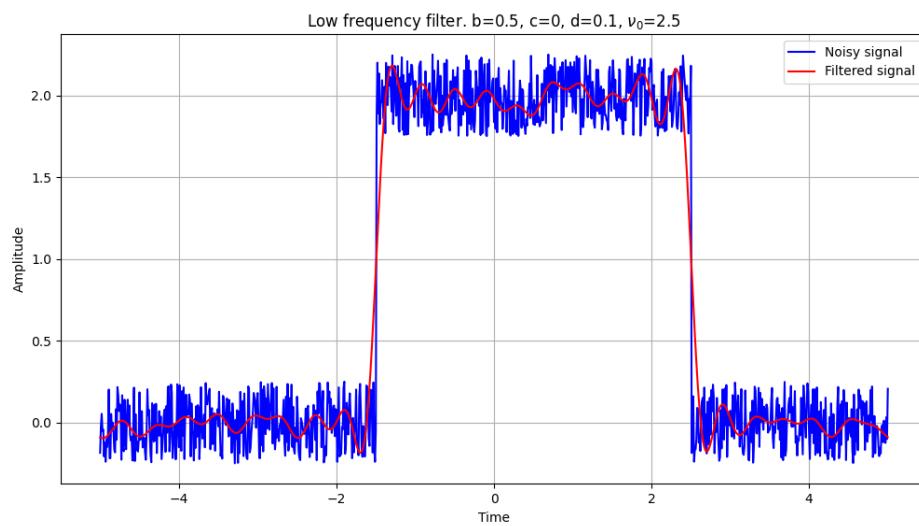


Рис. 15: График исходного и фильтрованного сигналов.

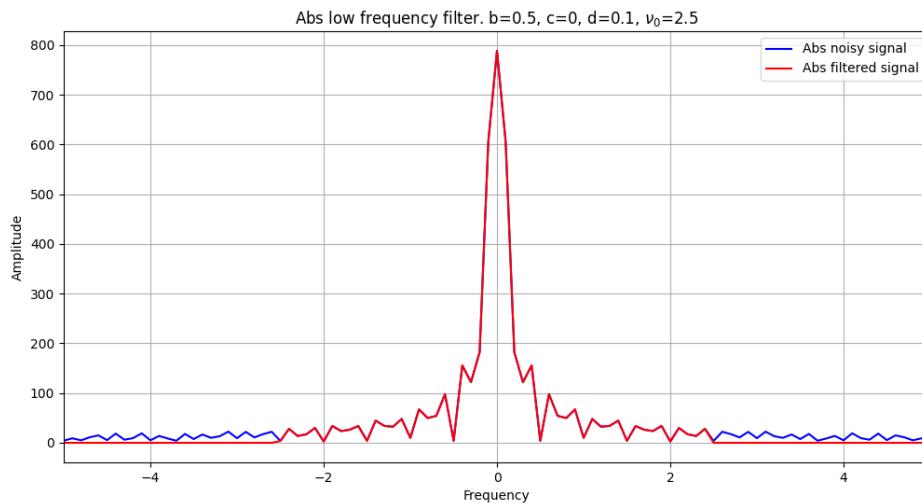


Рис. 16: График модуля Фурье-образа исходного и фильтрованного сигналов.

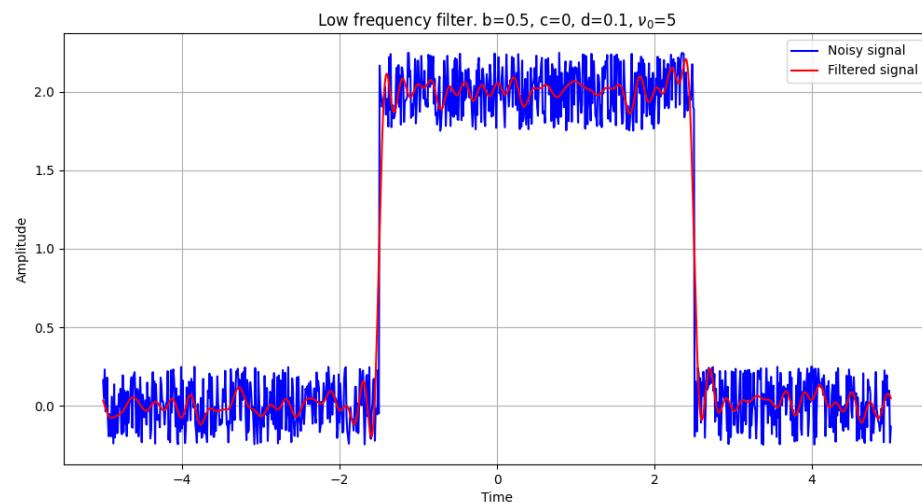


Рис. 17: График исходного и фильтрованного сигналов.

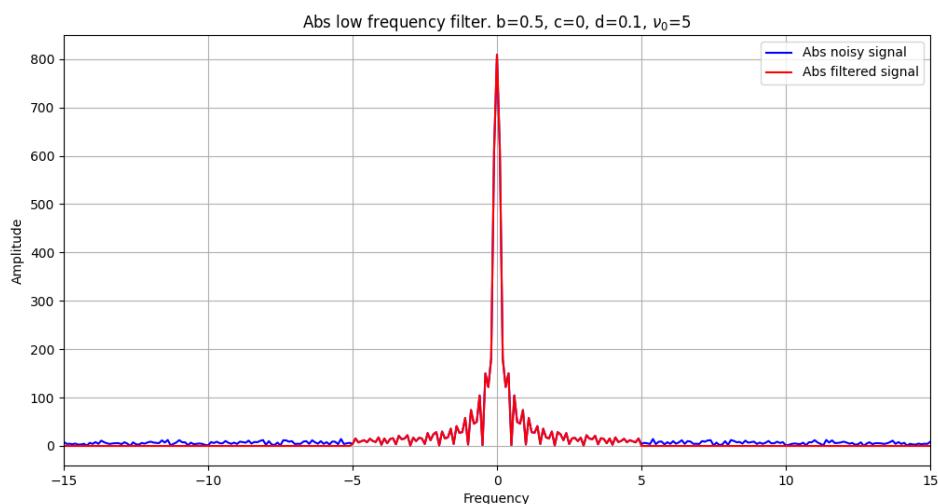


Рис. 18: График модуля Фурье-образа исходного и фильтрованного сигналов.

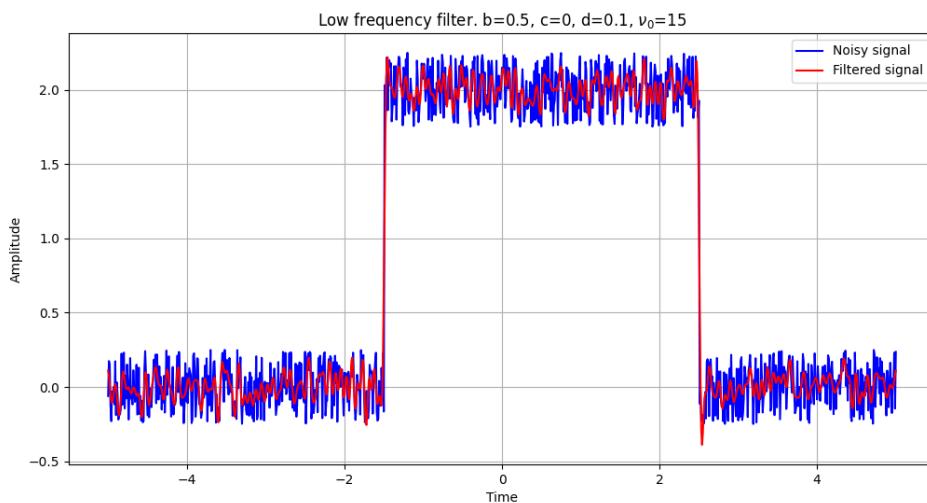


Рис. 19: График исходного и фильтрованного сигналов.

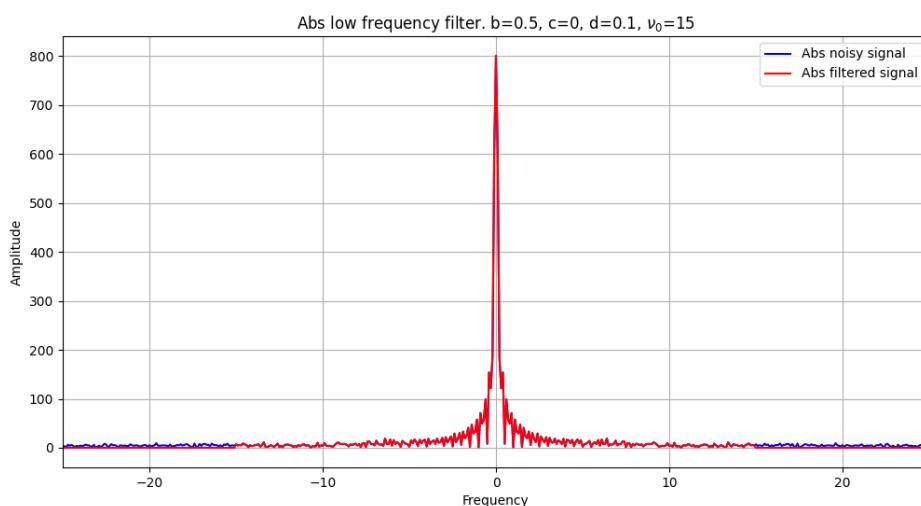


Рис. 20: График модуля Фурье-образа исходного и фильтрованного сигналов.

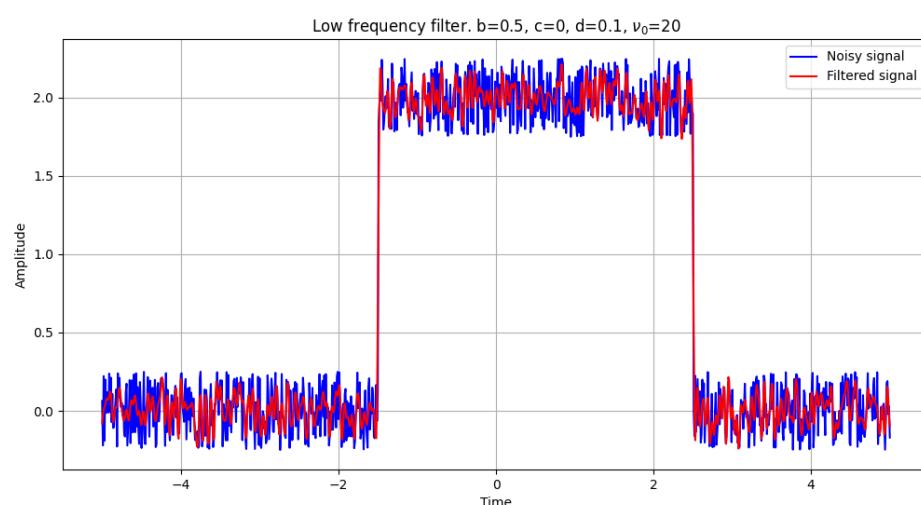


Рис. 21: График исходного и фильтрованного сигналов.

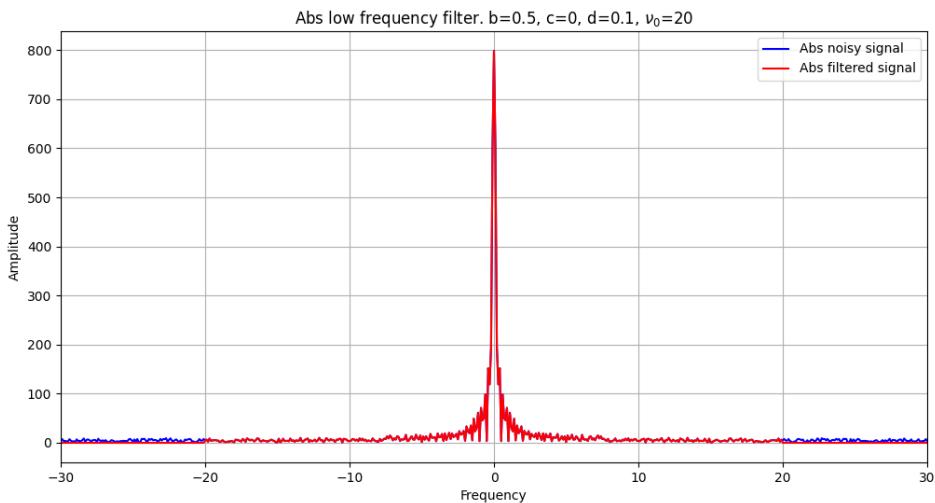


Рис. 22: График модуля Фурье-образа исходного и фильтрованного сигналов.

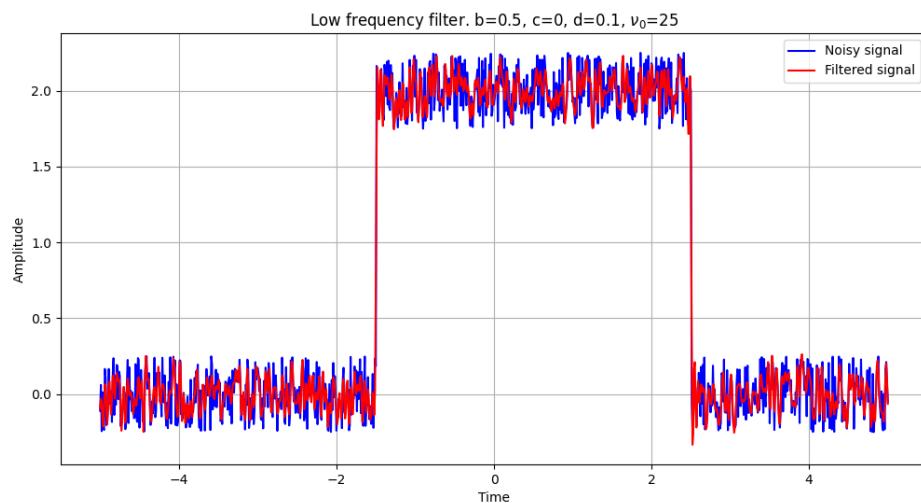


Рис. 23: График исходного и фильтрованного сигналов.

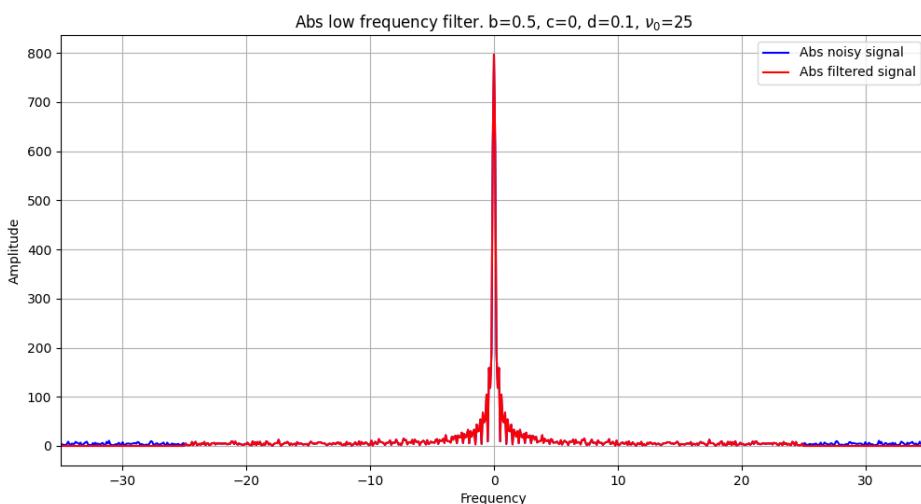


Рис. 24: График модуля Фурье-образа исходного и фильтрованного сигналов.

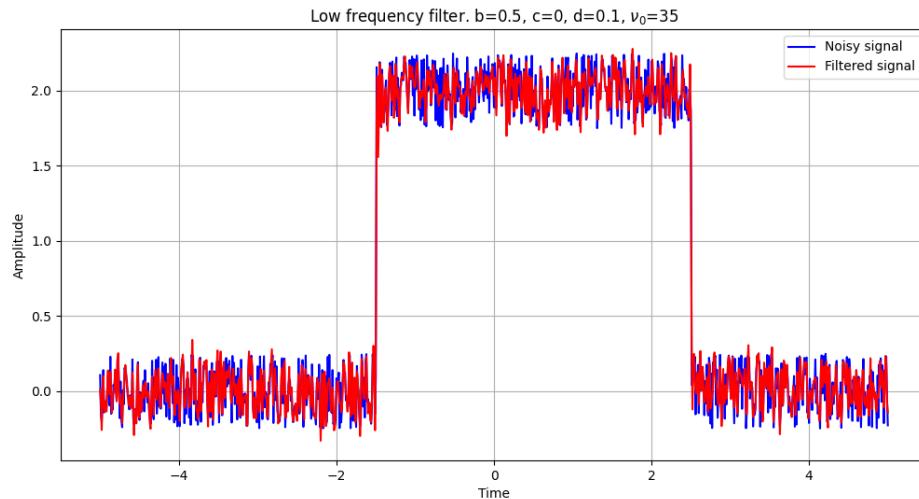


Рис. 25: График исходного и фильтрованного сигналов.

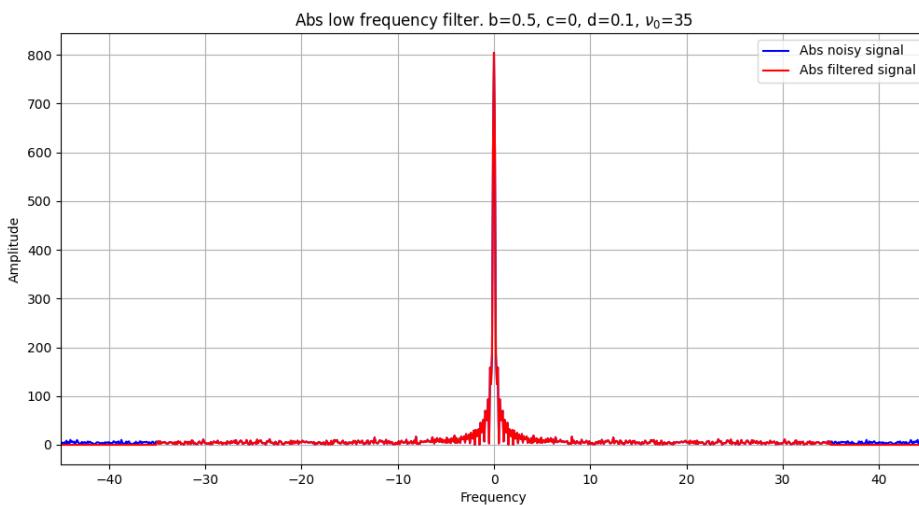


Рис. 26: График модуля Фурье-образа исходного и фильтрованного сигналов.

локальном смысле, то есть из плохого зашумленного сигнала получается фильтрованный плохой сигнал, но в глобальном смысле чем меньше значение b , тем чище сигнал и тем лучше сама фильтрация. При $b = 0$ получается особый случай — сигнал превращается в прямоугольную функцию, а фильтрованный сигнал выглядит как ее аппроксимация.

Больше всего влияния на эффективность фильтрации оказывает частота среза ν_0 . Этот параметр необходимо подобрать так, чтобы оставались только те частоты, которые имеют заметно более высокую амплитуду по сравнению с остальными. Чем меньше частота среза ν_0 , тем заметно лучше фильтрация. Однако если взять слишком маленькое значение ν_0 , то фильтрация будет слишком сильной, и мы потеряем большую часть значащих частот в сигнале, что можно увидеть на рисунках 11, 12. Это происходит потому, что мы задели и нижние частоты тоже, оставив лишь малую их часть. Можно сказать, что ν_0 — это порог, которым мы определяем, является ли частота верхней или нижней. Такой параметр нужно подбирать с умом, чтобы получить от фильтрации ожидаемый результат. Если взять слишком большое значение ν_0 , то фильтрация будет не очень эффективной (см. рис. 25, 26), так как мы оставили некоторые верхние частоты, которые можно было обнулить.

1.2 Убираем низкие частоты. Фильтр верхних частот.

Рассмотрим графики, где в некоторой окрестности точки $\nu = 0$ обнулим все значения частот Фурье-образа. Такое поведение соответствует фильтру *верхних* частот, так как он пропускает все частоты выше частоты среза. Окрестность будет настраиваться выбором диапазона частот $[-\nu_0, \nu_0]$. Для лучшей показательности выбраны как маленькие окрестности около нуля, так и большие. Также рассмотрим поведение фильтрации при различных параметрах b, c, d .

Далее будут приведены рисунки полученных графиков. На каждом графике подписаны выбранные значения b, c, d, ν_0 . Синим цветом обозначается оригинальный сигнал, красным фильтрованный.

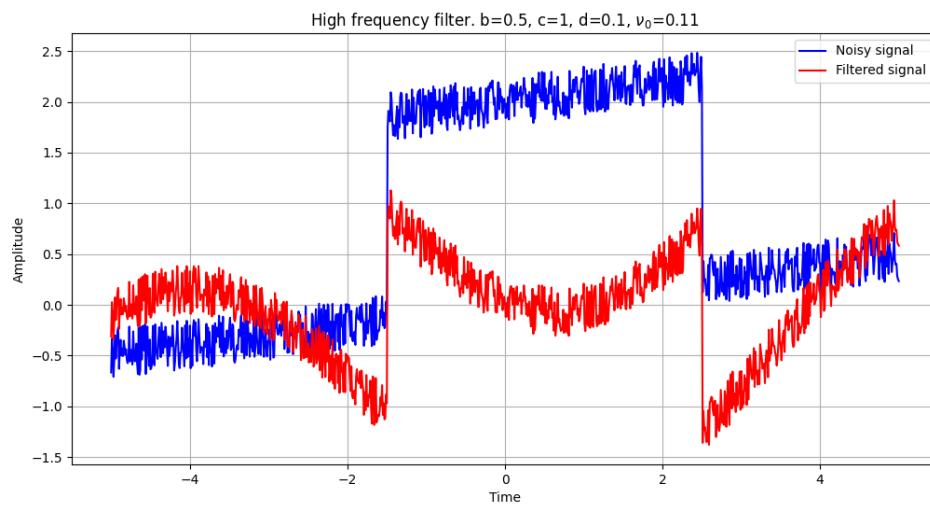


Рис. 27: График исходного и фильтрованного сигналов.

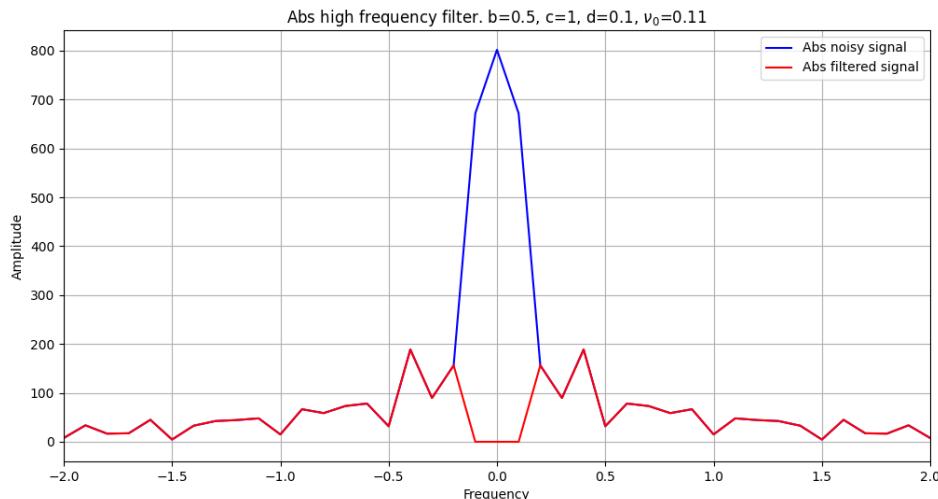


Рис. 28: График модуля Фурье-образа исходного и фильтрованного сигналов.

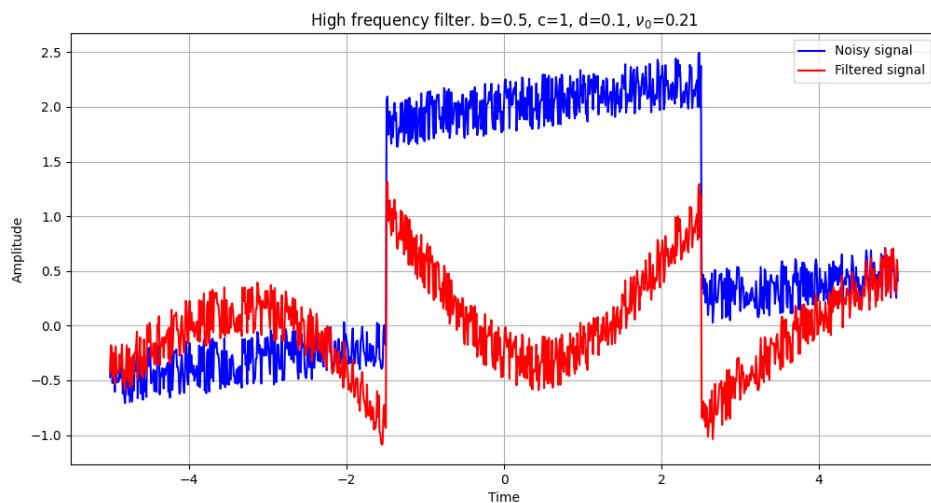


Рис. 29: График исходного и фильтрованного сигналов.

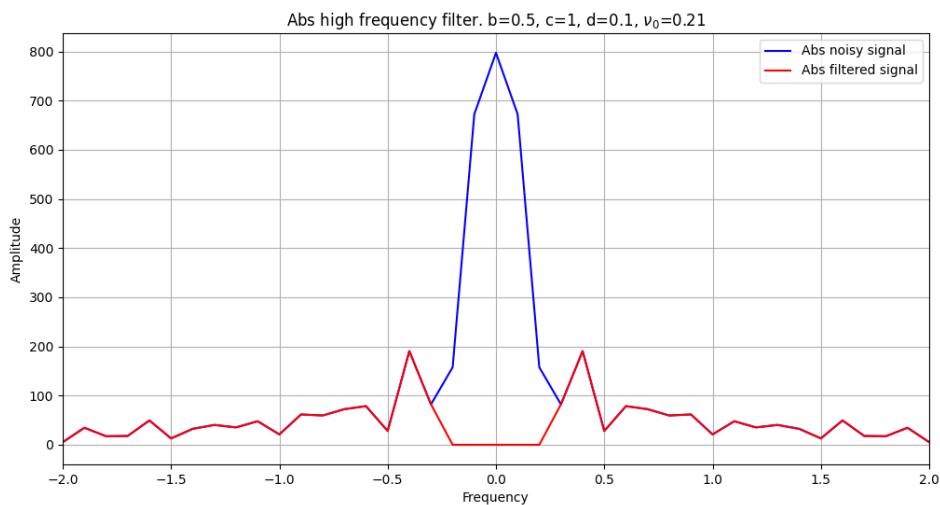


Рис. 30: График модуля Фурье-образа исходного и фильтрованного сигналов.

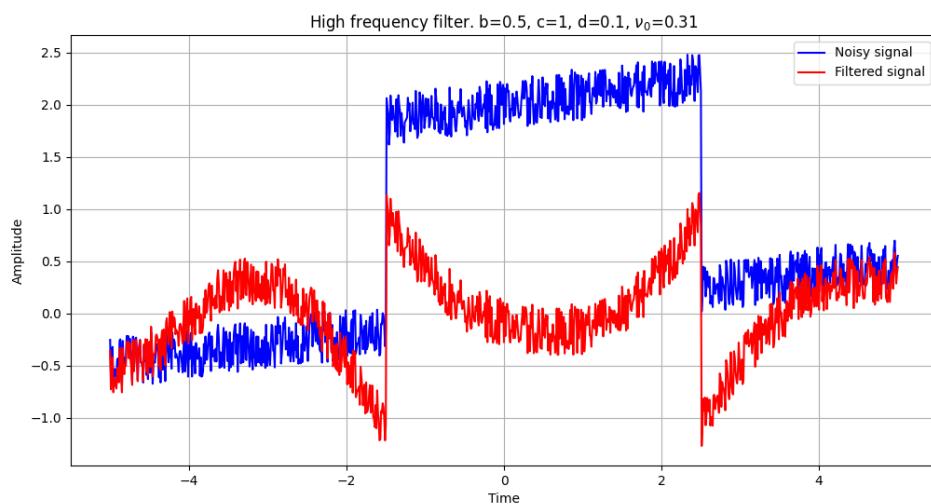


Рис. 31: График исходного и фильтрованного сигналов.

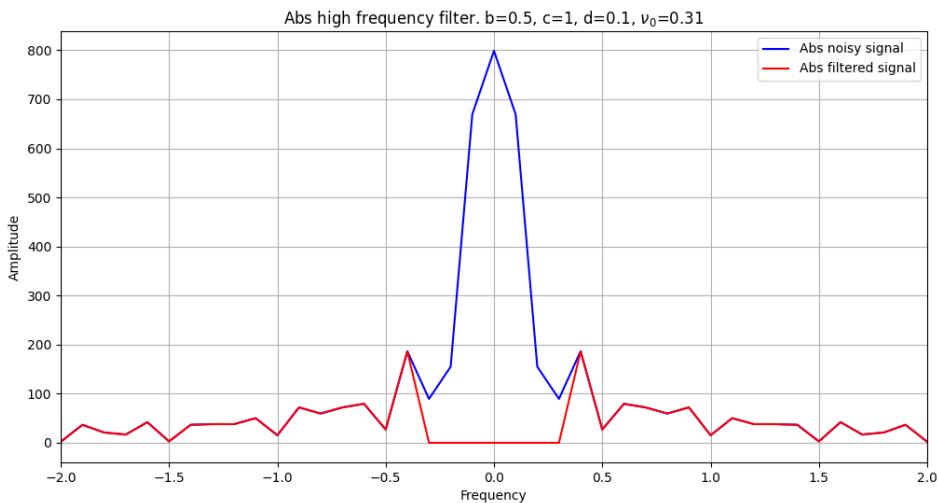


Рис. 32: График модуля Фурье-образа исходного и фильтрованного сигналов.

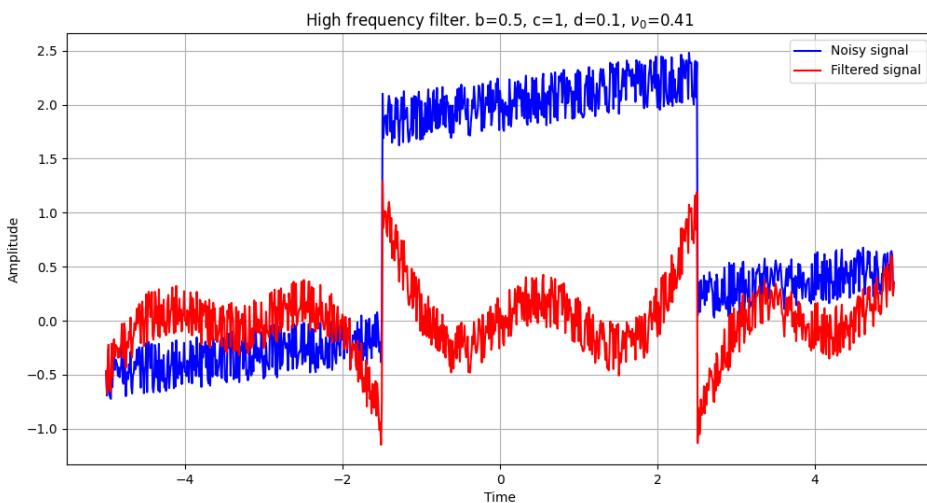


Рис. 33: График исходного и фильтрованного сигналов.

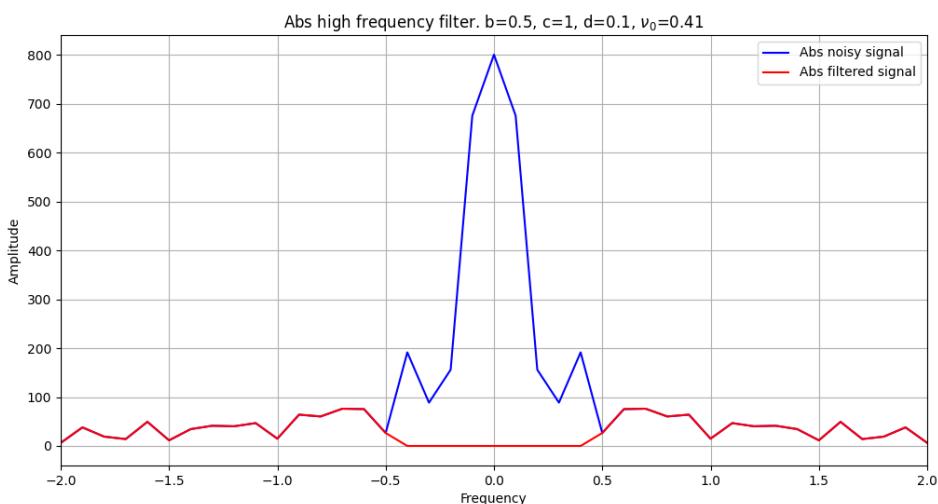


Рис. 34: График модуля Фурье-образа исходного и фильтрованного сигналов.

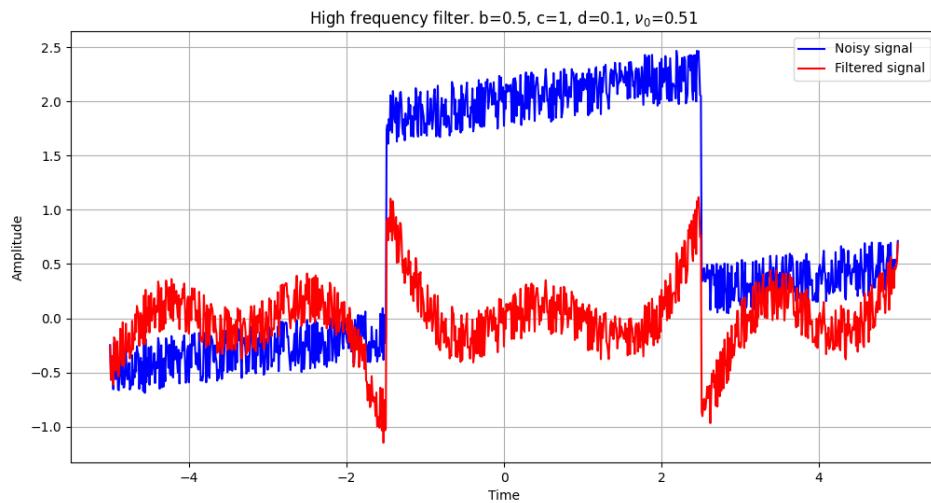


Рис. 35: График исходного и фильтрованного сигналов.

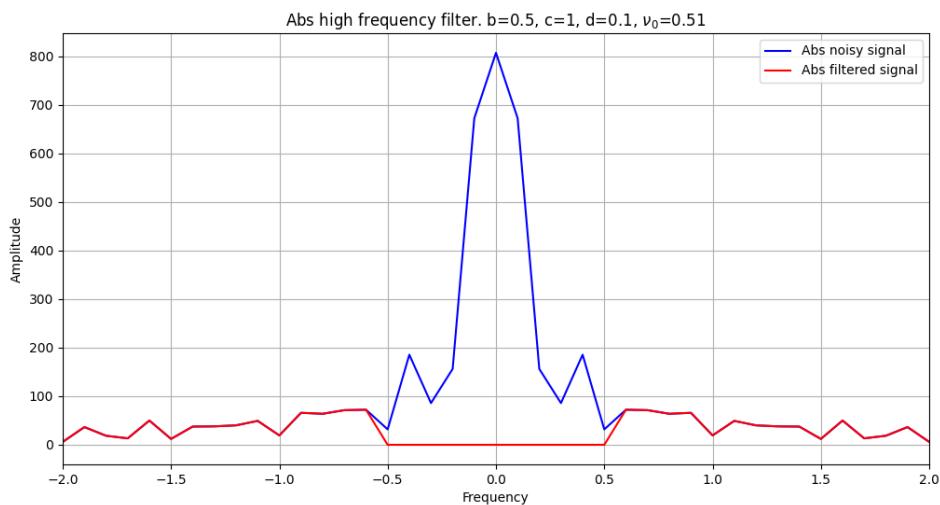


Рис. 36: График модуля Фурье-образа исходного и фильтрованного сигналов.

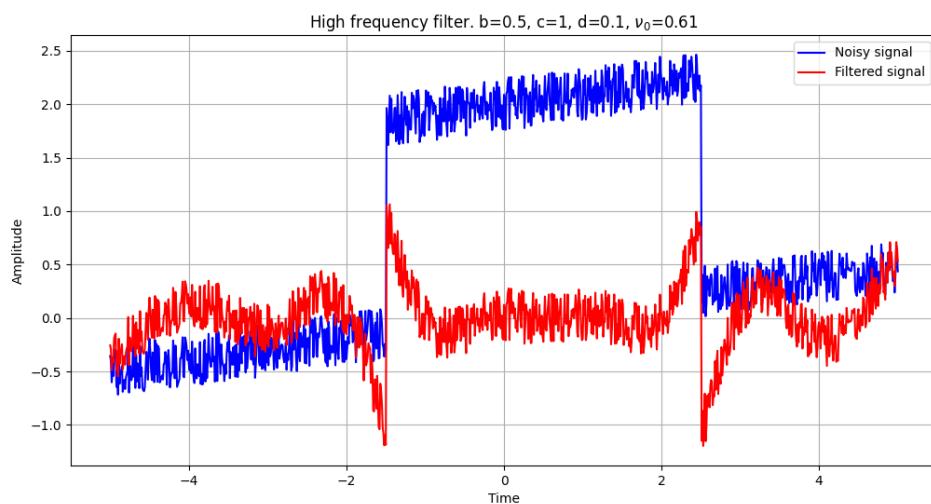


Рис. 37: График исходного и фильтрованного сигналов.

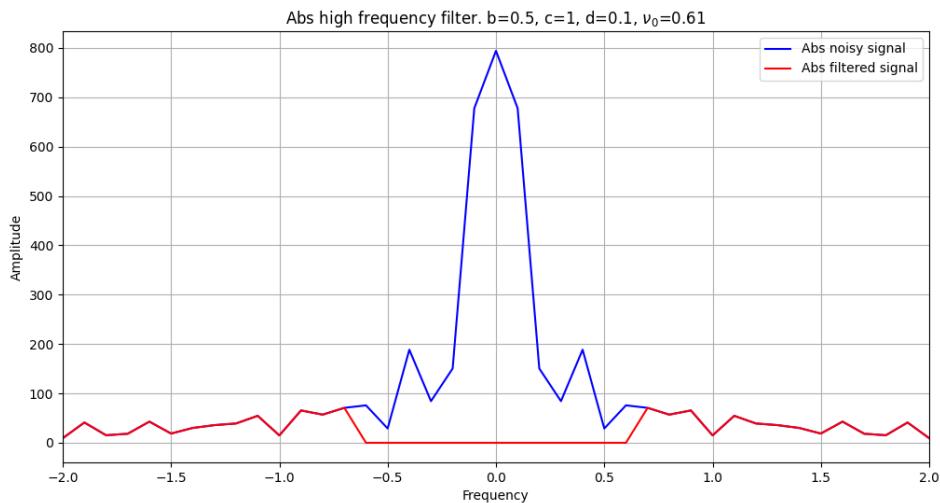


Рис. 38: График модуля Фурье-образа исходного и фильтрованного сигналов.

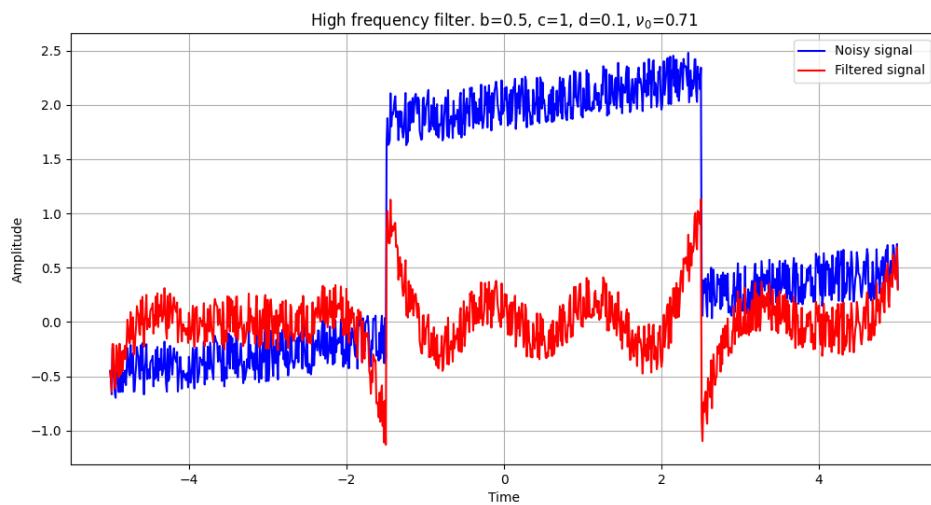


Рис. 39: График исходного и фильтрованного сигналов.

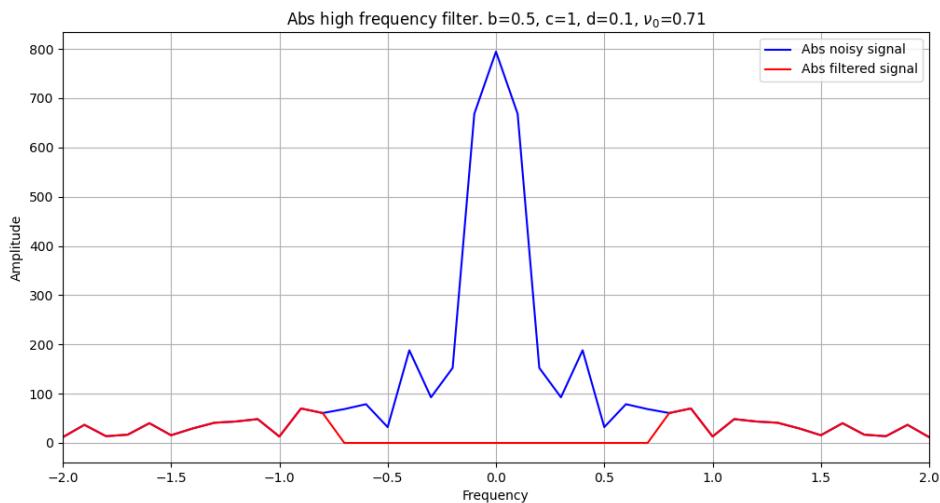


Рис. 40: График модуля Фурье-образа исходного и фильтрованного сигналов.

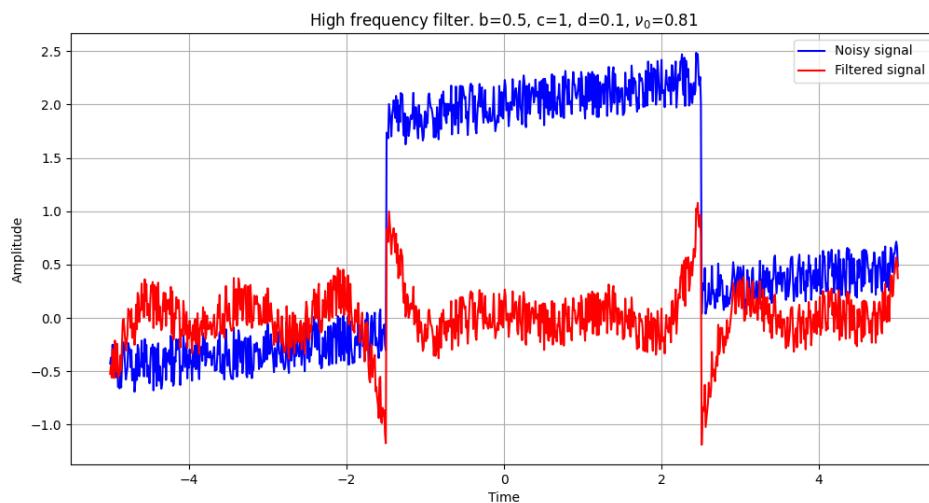


Рис. 41: График исходного и фильтрованного сигналов.

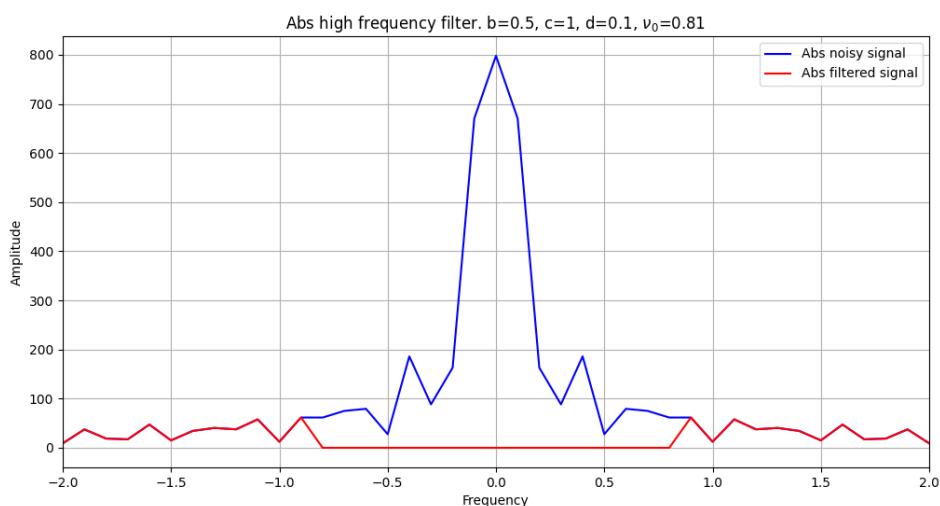


Рис. 42: График модуля Фурье-образа исходного и фильтрованного сигналов.

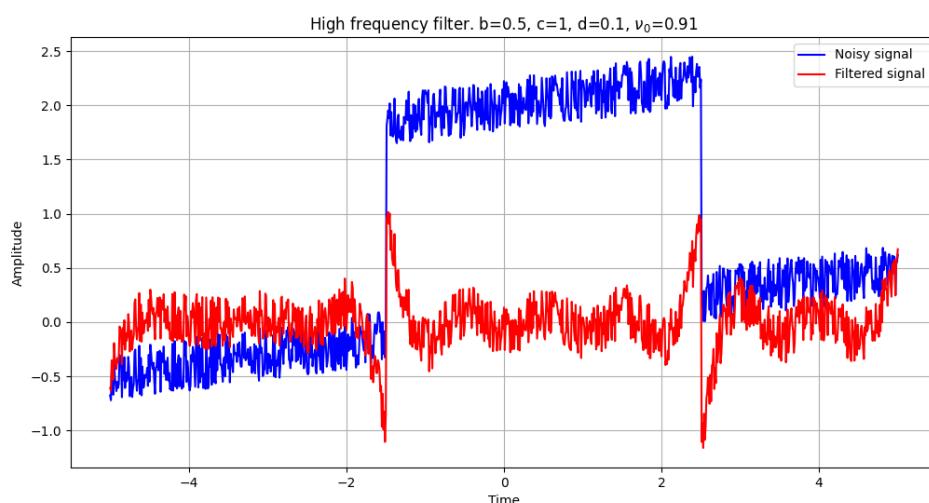


Рис. 43: График исходного и фильтрованного сигналов.

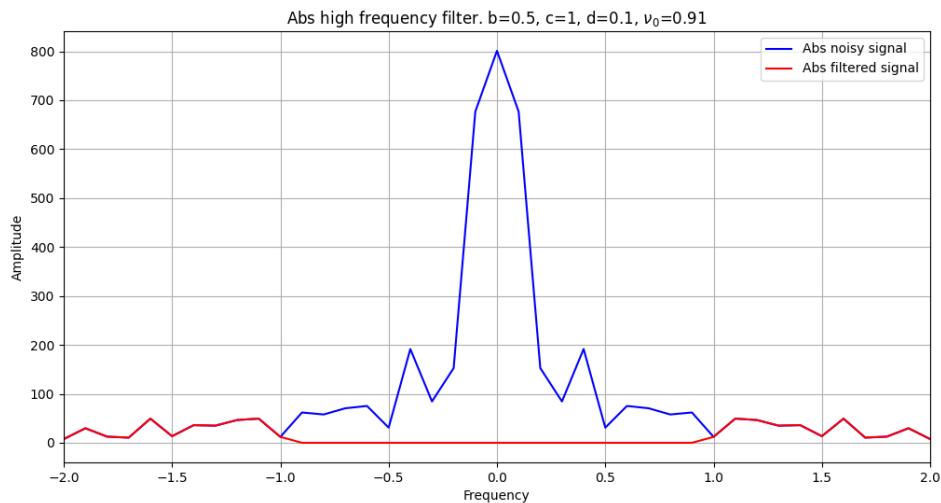


Рис. 44: График модуля Фурье-образа исходного и фильтрованного сигналов.

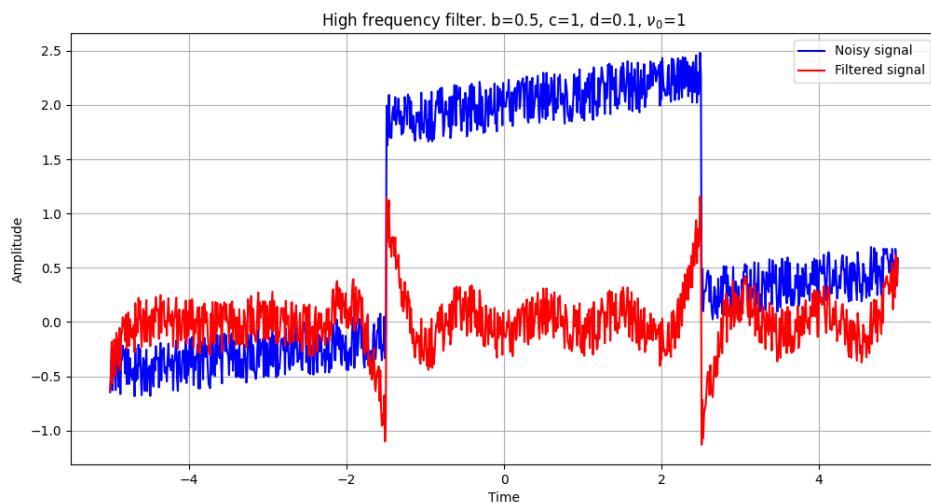


Рис. 45: График исходного и фильтрованного сигналов.

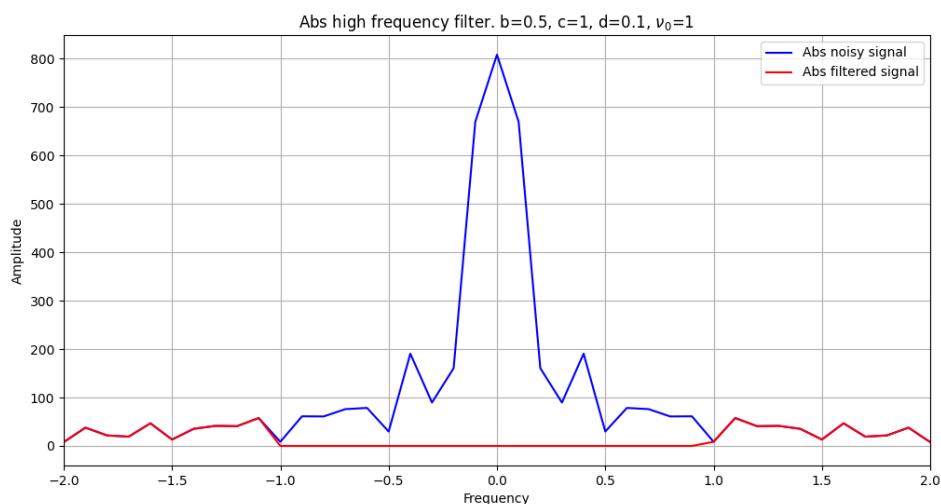


Рис. 46: График модуля Фурье-образа исходного и фильтрованного сигналов.

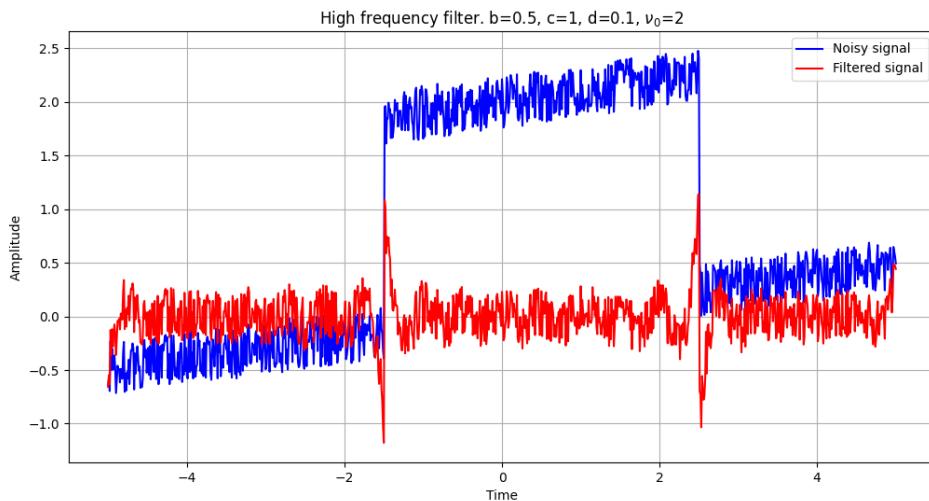


Рис. 47: График исходного и фильтрованного сигналов.

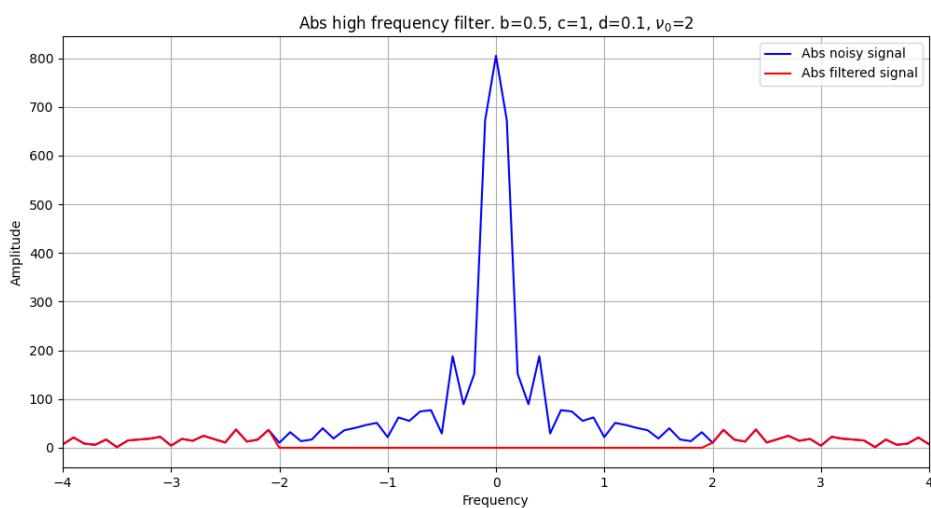


Рис. 48: График модуля Фурье-образа исходного и фильтрованного сигналов.

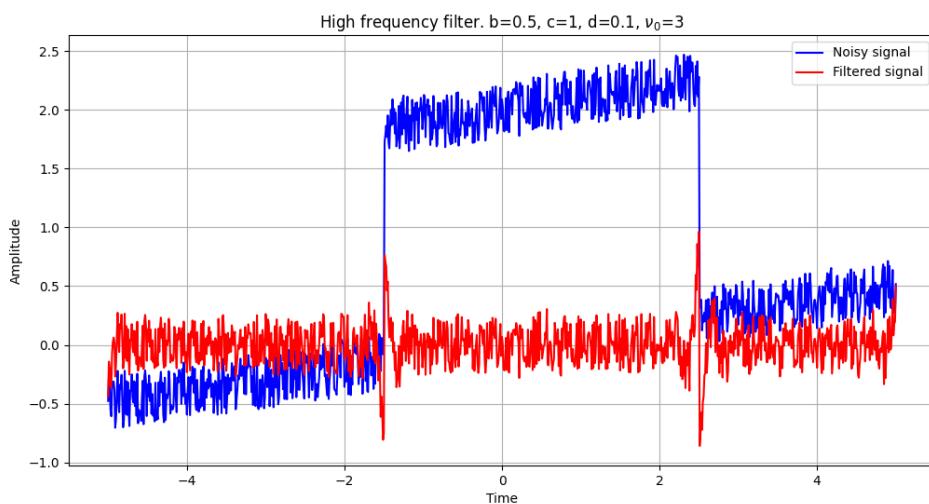


Рис. 49: График исходного и фильтрованного сигналов.

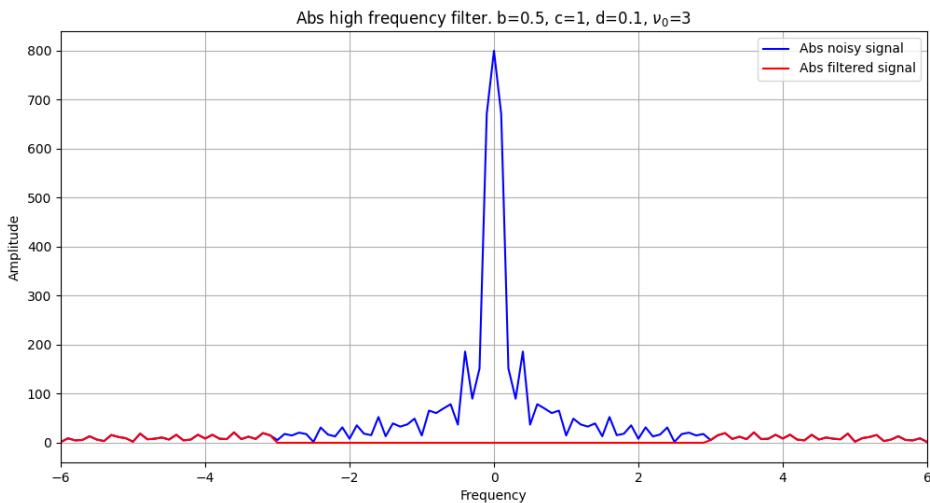


Рис. 50: График модуля Фурье-образа исходного и фильтрованного сигналов.

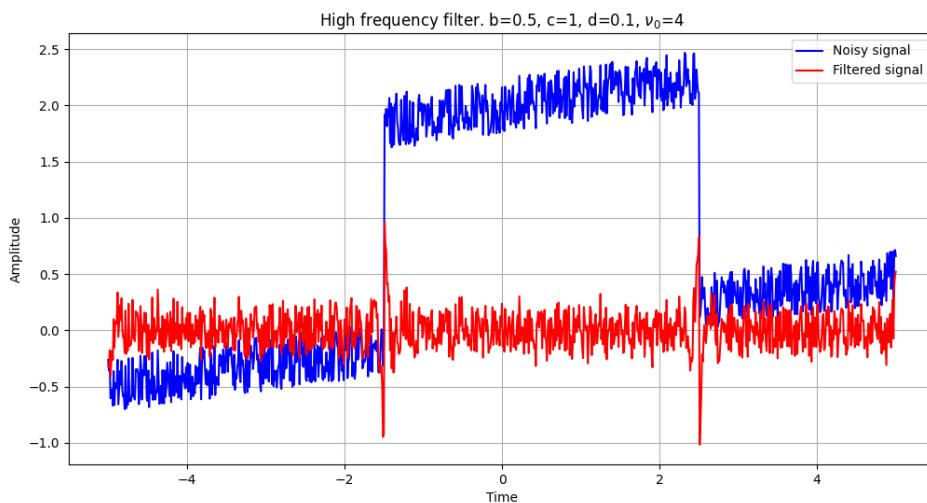


Рис. 51: График исходного и фильтрованного сигналов.

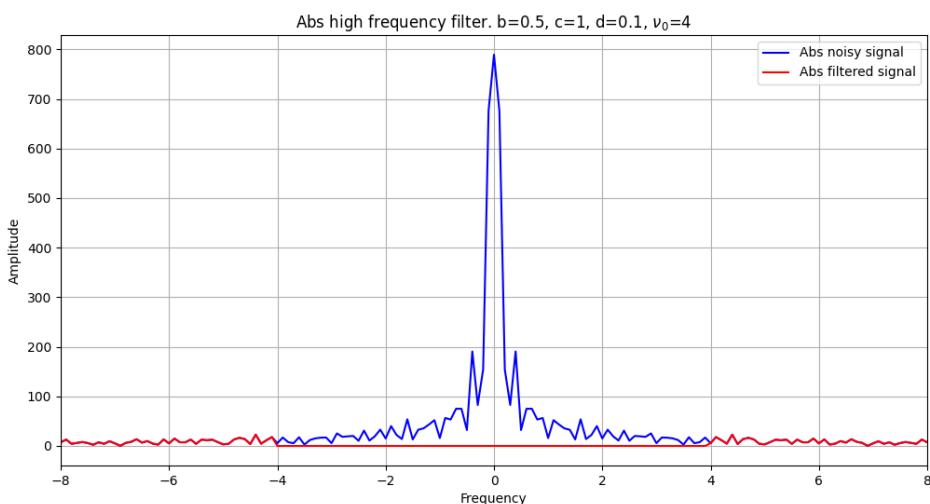


Рис. 52: График модуля Фурье-образа исходного и фильтрованного сигналов.

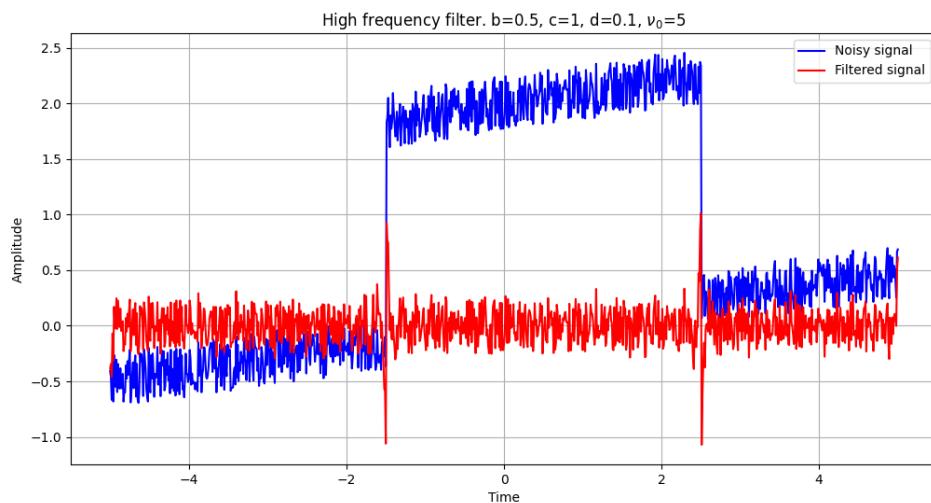


Рис. 53: График исходного и фильтрованного сигналов.

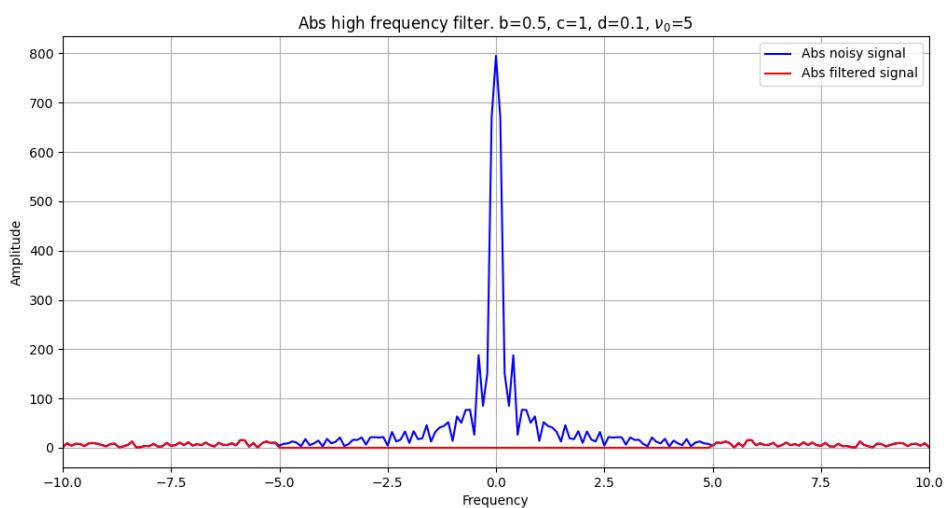


Рис. 54: График модуля Фурье-образа исходного и фильтрованного сигналов.

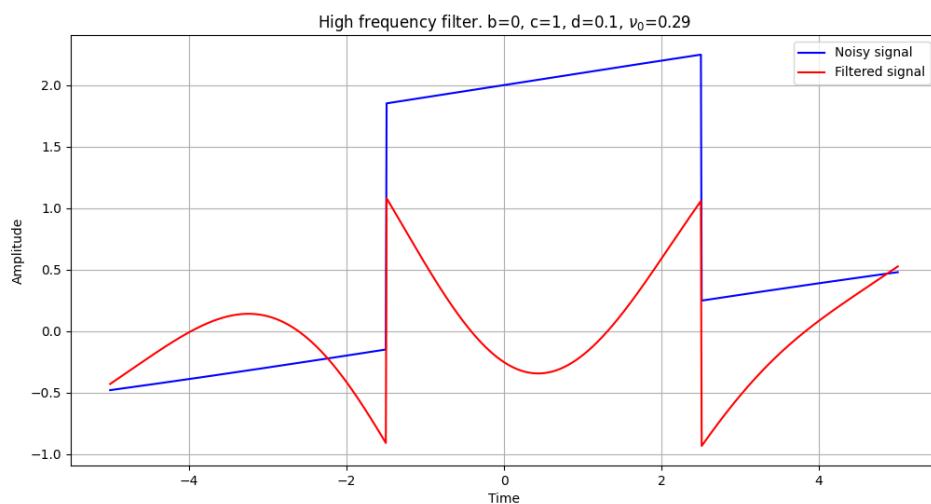


Рис. 55: График исходного и фильтрованного сигналов.

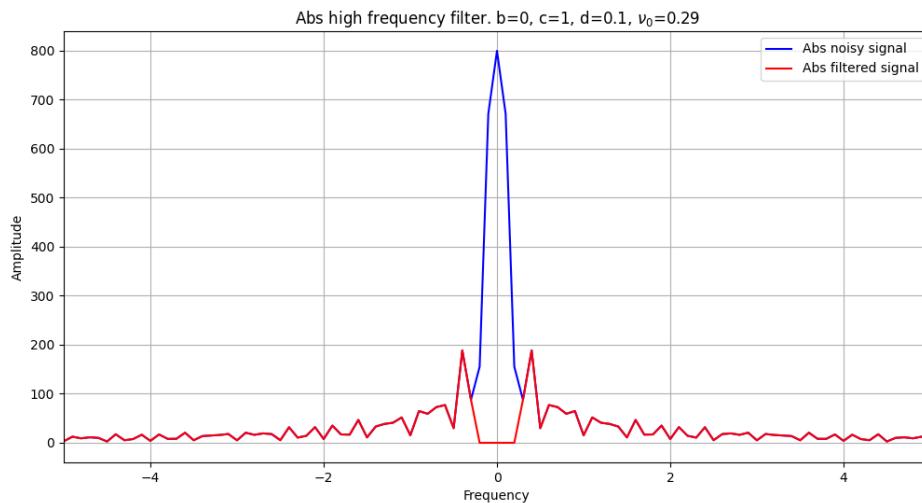


Рис. 56: График модуля Фурье-образа исходного и фильтрованного сигналов.

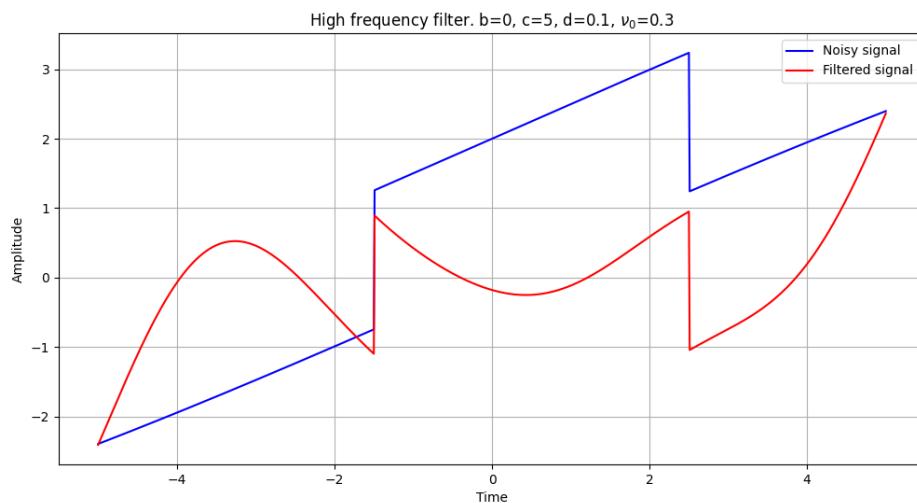


Рис. 57: График исходного и фильтрованного сигналов.

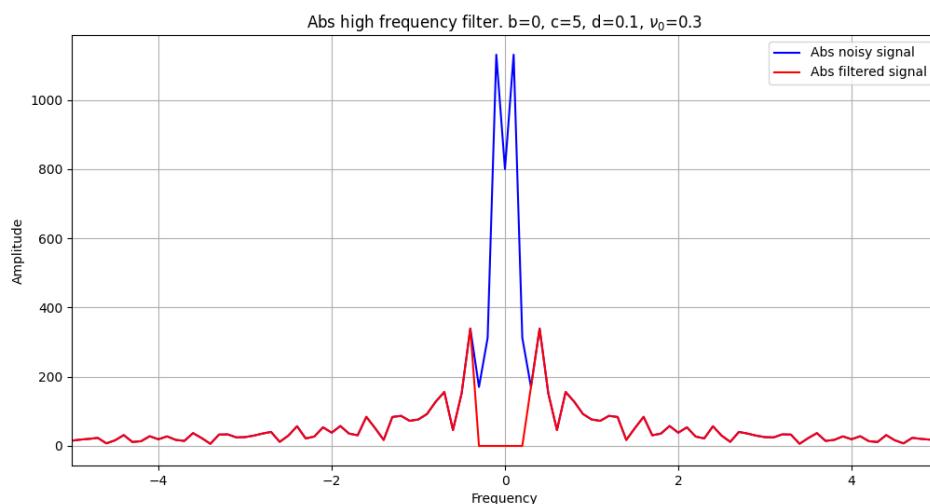


Рис. 58: График модуля Фурье-образа исходного и фильтрованного сигналов.

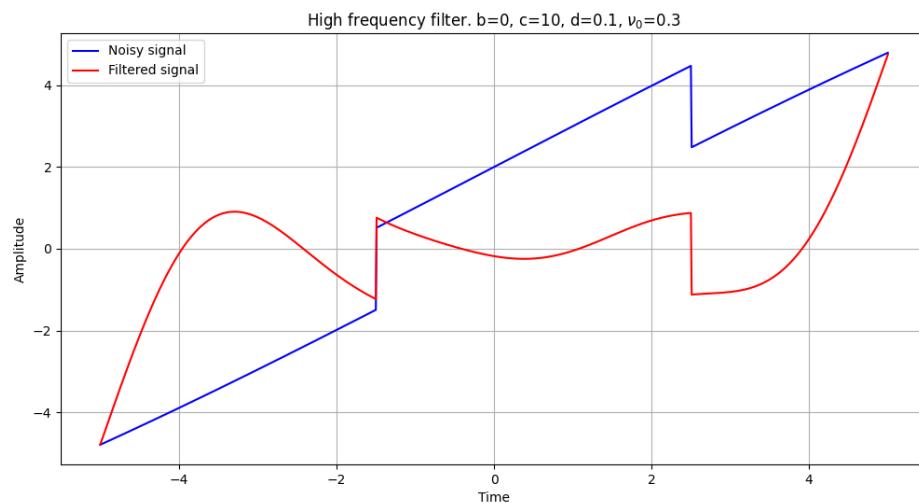


Рис. 59: График исходного и фильтрованного сигналов.

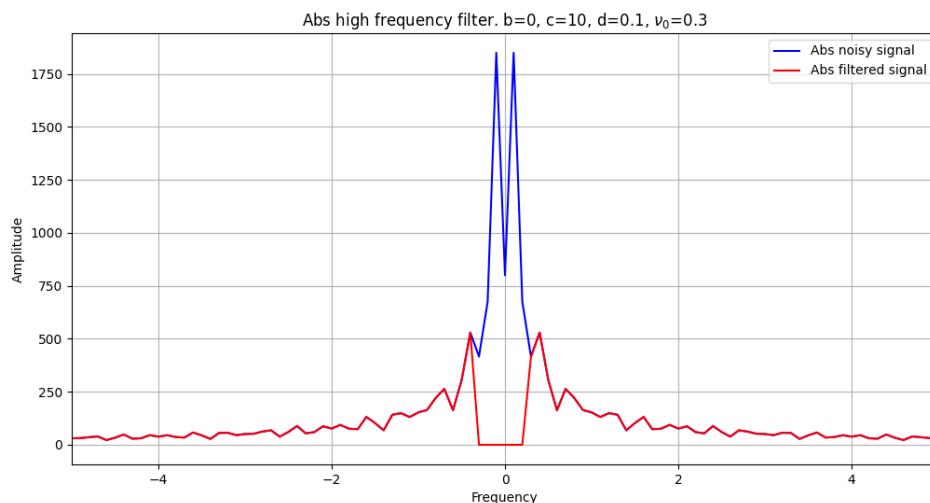


Рис. 60: График модуля Фурье-образа исходного и фильтрованного сигналов.

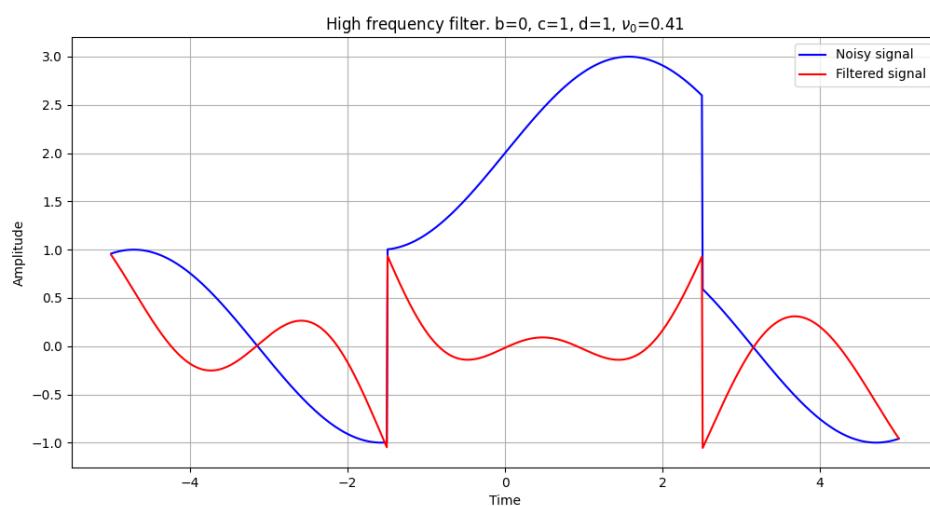


Рис. 61: График исходного и фильтрованного сигналов.

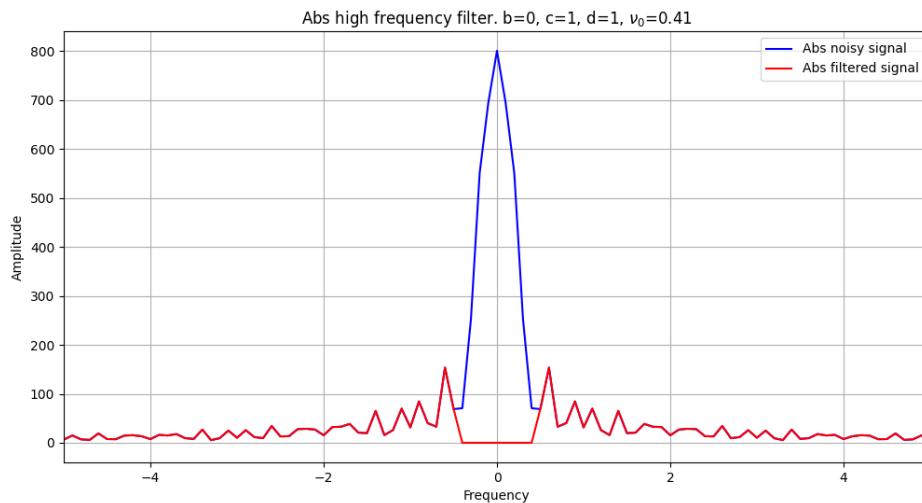


Рис. 62: График модуля Фурье-образа исходного и фильтрованного сигналов.

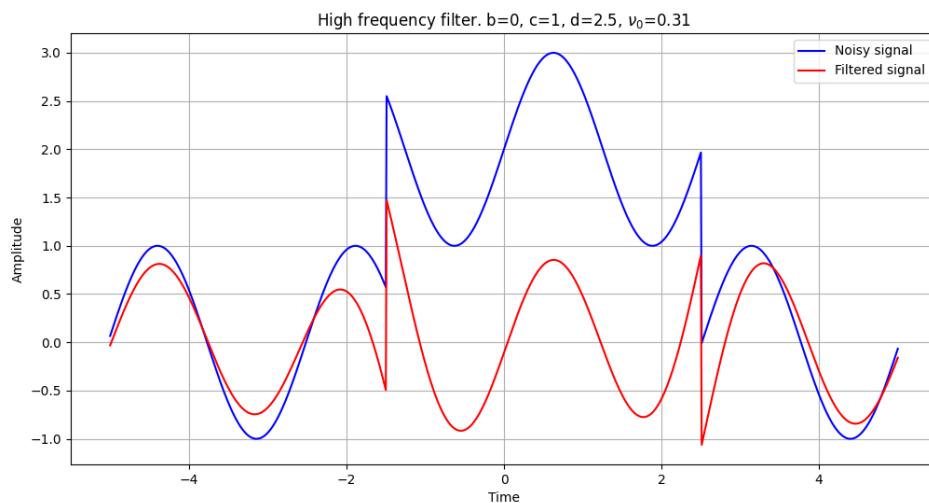


Рис. 63: График исходного и фильтрованного сигналов.

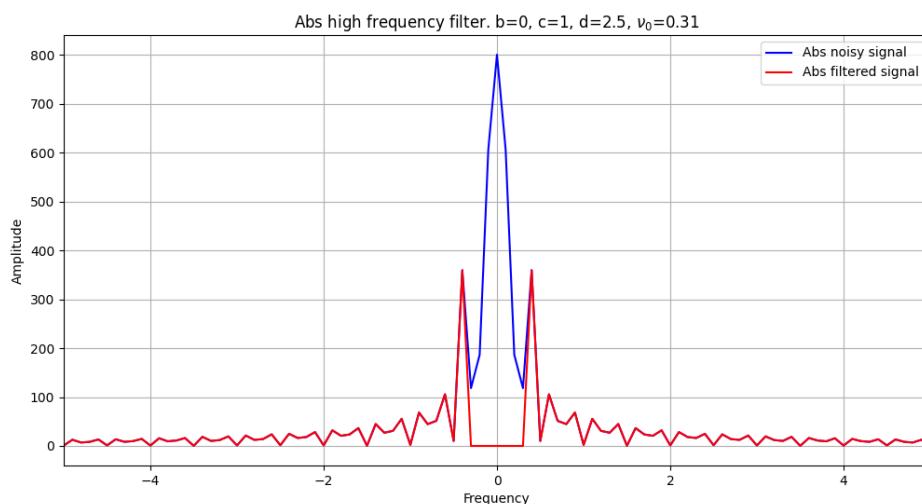


Рис. 64: График модуля Фурье-образа исходного и фильтрованного сигналов.

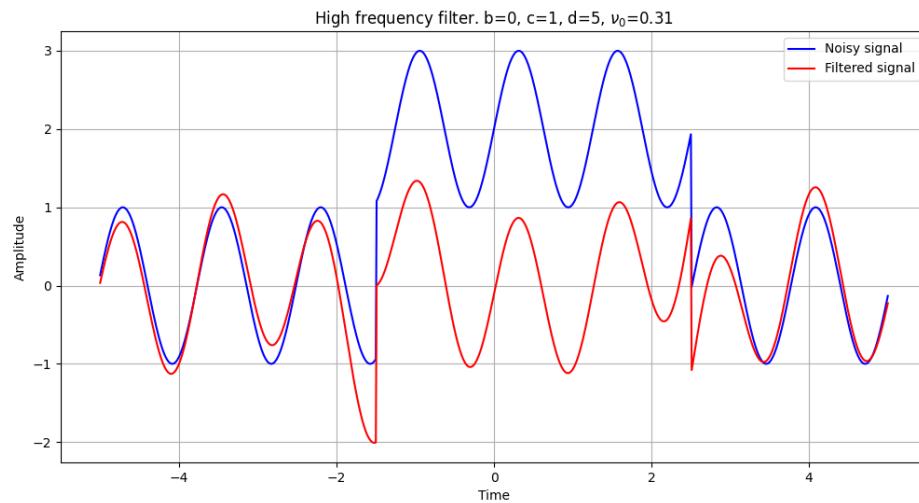


Рис. 65: График исходного и фильтрованного сигналов.

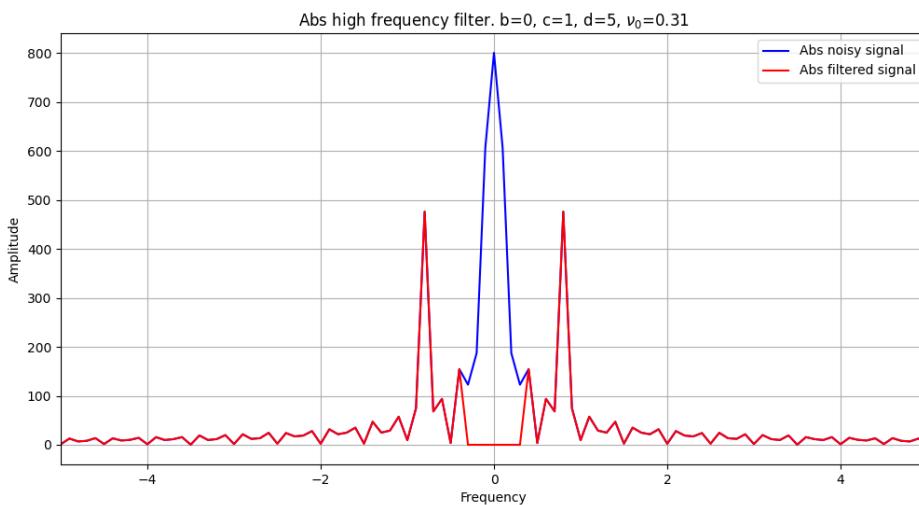


Рис. 66: График модуля Фурье-образа исходного и фильтрованного сигналов.

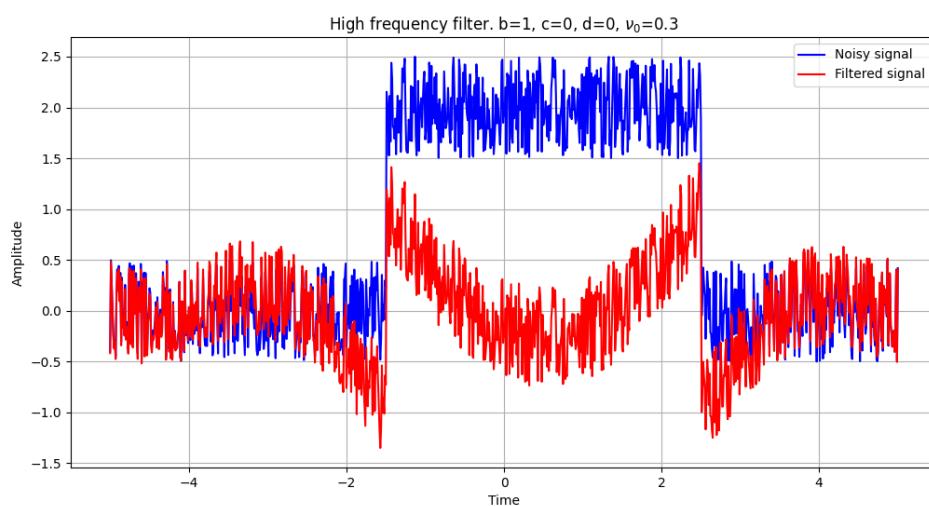


Рис. 67: График исходного и фильтрованного сигналов.

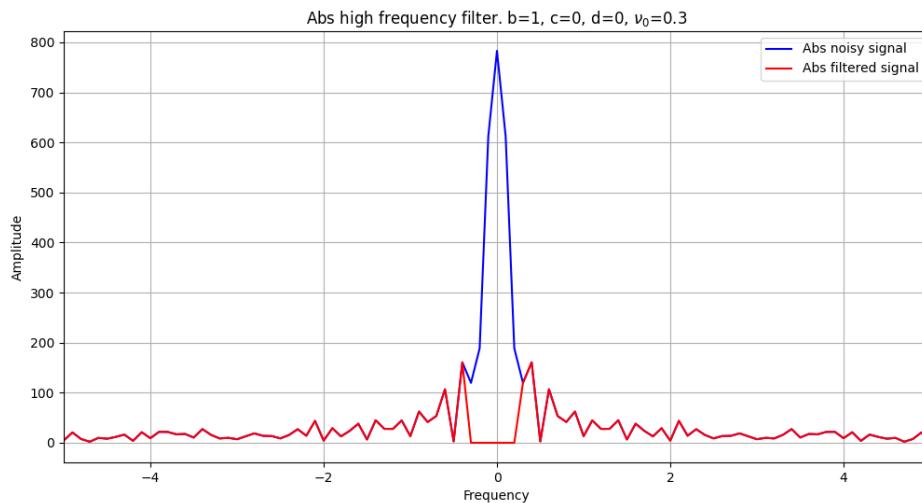


Рис. 68: График модуля Фурье-образа исходного и фильтрованного сигналов.

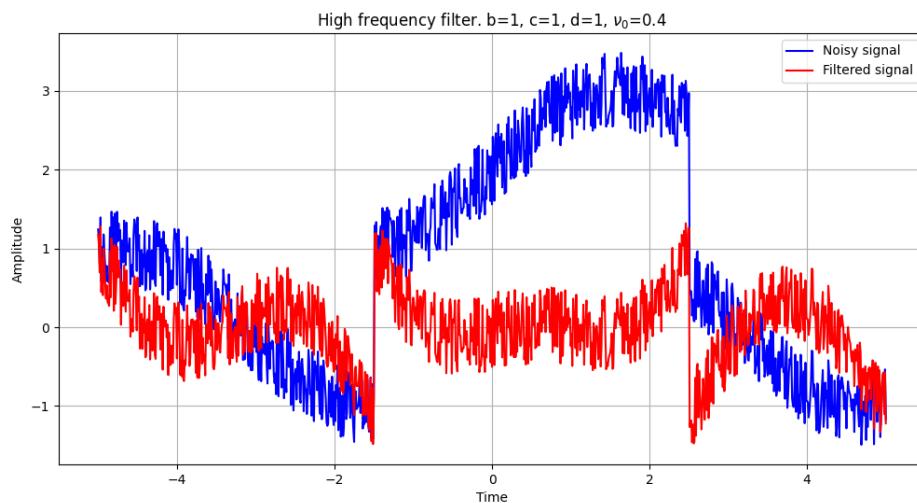


Рис. 69: График исходного и фильтрованного сигналов.

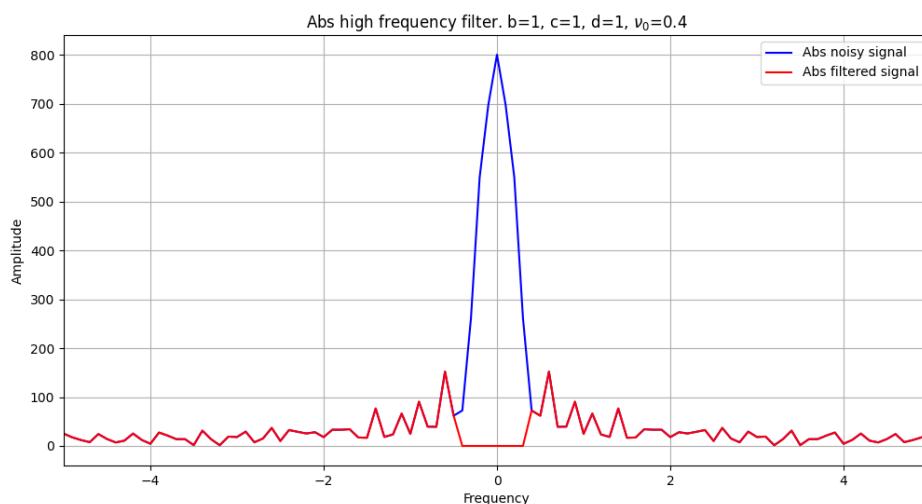


Рис. 70: График модуля Фурье-образа исходного и фильтрованного сигналов.

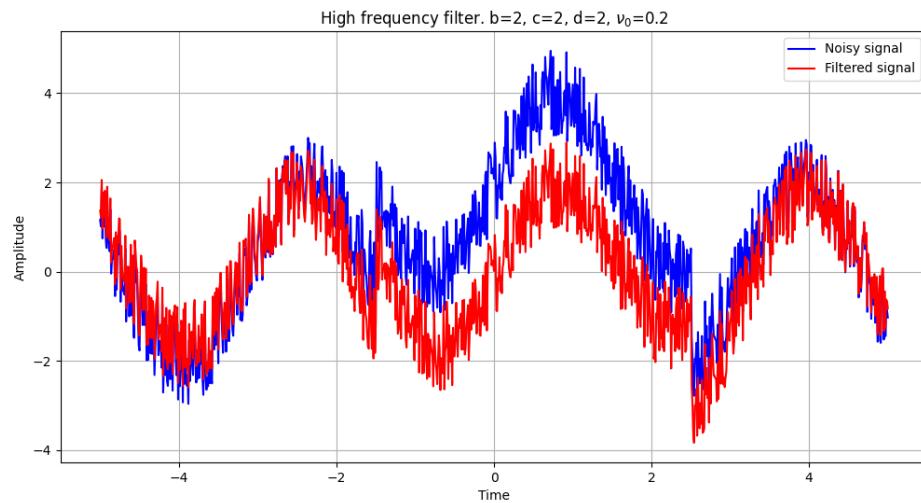


Рис. 71: График исходного и фильтрованного сигналов.

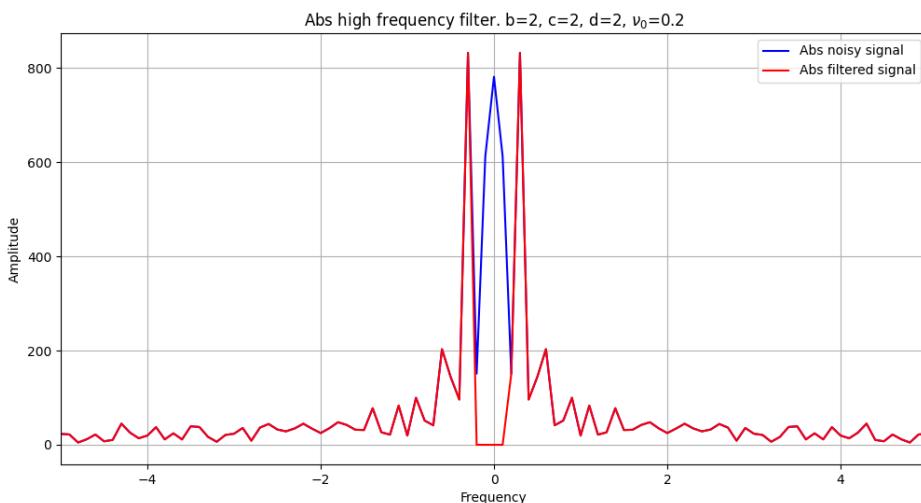


Рис. 72: График модуля Фурье-образа исходного и фильтрованного сигналов.

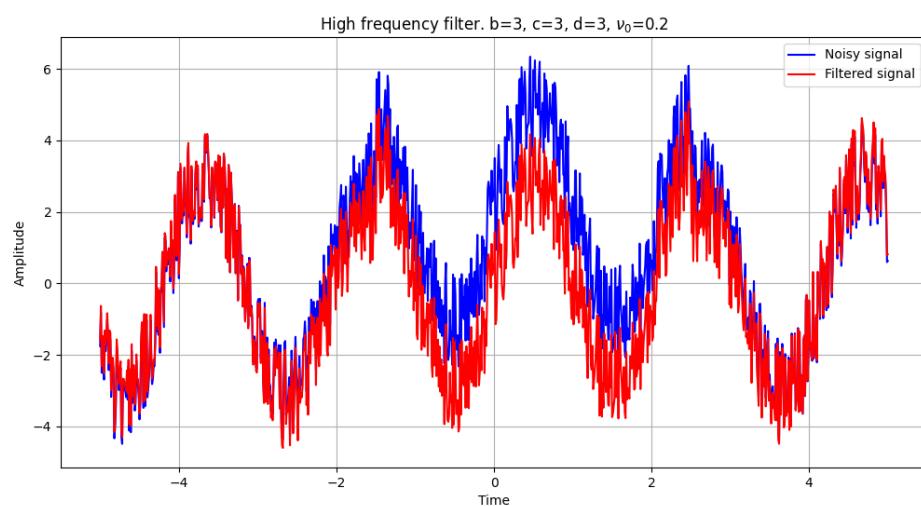


Рис. 73: График исходного и фильтрованного сигналов.

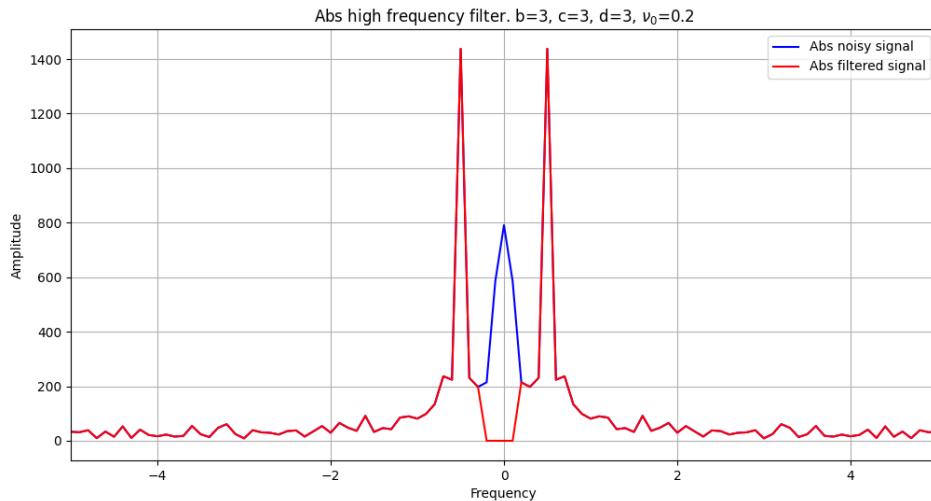


Рис. 74: График модуля Фурье-образа исходного и фильтрованного сигналов.

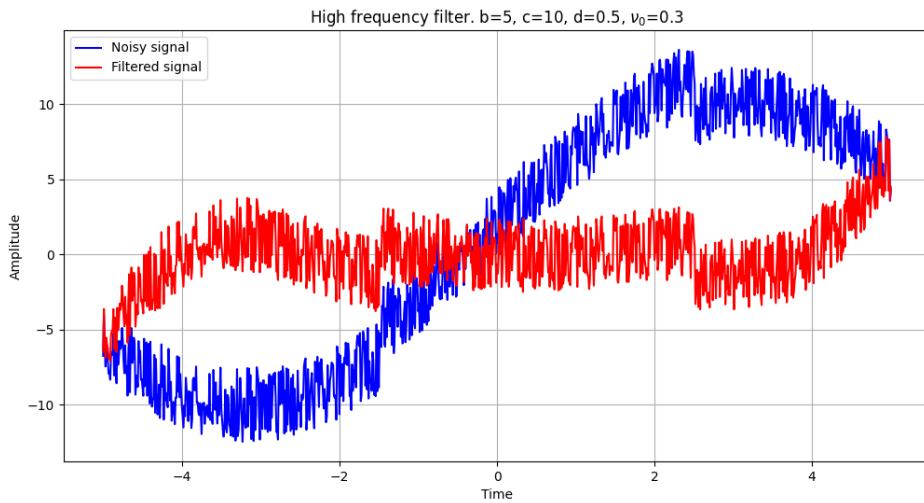


Рис. 75: График исходного и фильтрованного сигналов.

1.3 Убираем специфические частоты. Фильтр специфических частот.

Возьмем ненулевые параметры b , c , d и проделаем то же самое, что и раньше. Сначала попробуем обнулять некоторые диапазоны частот, потом обнулим верхние или нижние частоты, а также совместим различные варианты фильтрации, чтобы по возможности убрать влияние обеих компонент помехи. Исследуем влияние частот среза и значений параметров b , c , d на вид помехи и эффективность фильтрации. Кроме того, отдельно рассмотрим случай для $b = 0$.

Далее будут приведены рисунки полученных графиков. На каждом графике подписаны выбранные значения b , c , d , ν_0 . Синим цветом обозначается оригинальный сигнал, красным фильтрованный. Для первого случая будут добавлены два дополнительных рисунка – графики сигнала и его Фурье-образа. Второй из двух нужен для того, чтобы вручную определить диапазон частот, который мы будем обнулять. В ходе эксперимента с различными диапазонами был сделан вывод, что нужно попробовать обнулять те частоты,

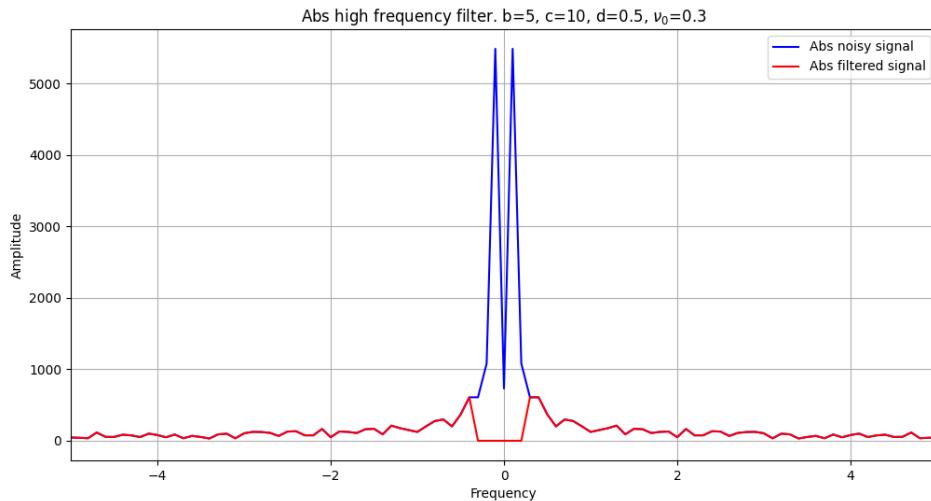


Рис. 76: График модуля Фурье-образа исходного и фильтрованного сигналов.

ты, которые имеют пиковые амплитуды рядом с наивысшей амплитудой в точке 0. Если наивысшая амплитуда не одна, а, например, их две, и они находятся где-то в окрестности точки 0, то обнулять будем пики частот слева от левой и справа от правой наивысших амплитуд. Частоты в этих диапазонах будем называть специфическими.

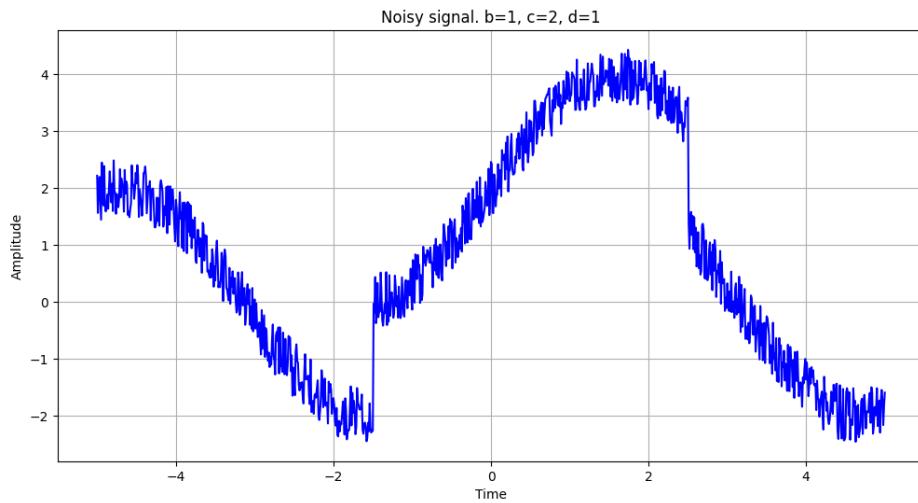


Рис. 77: График исходного сигнала.

2 Задание 2. Фильтрация звука.

В данном задании необходимо убрать шумы из записи голоса в файле МИУНА.wav так, чтобы остался только голос. При прослушивании записи голос слышно не очень хорошо, так как присутствует громкий гул. Построим графики исходного сигнала аудиозаписи и его Фурье-образа. Определим по второму графику, какие частоты могут создавать шумы. Так как гул громкий, нам необходимо вырезать частоты из аудиозаписи, имеющие наибольшую амплитуду. Такие частоты мы можем наблюдать примерно в диапазоне $[-300, 300]$ Гц. Чтобы вырезать эти частоты, потребуется фильтр верхних частот, который мы рассматривали в задании 1. После применения фильтра построим сравнительный

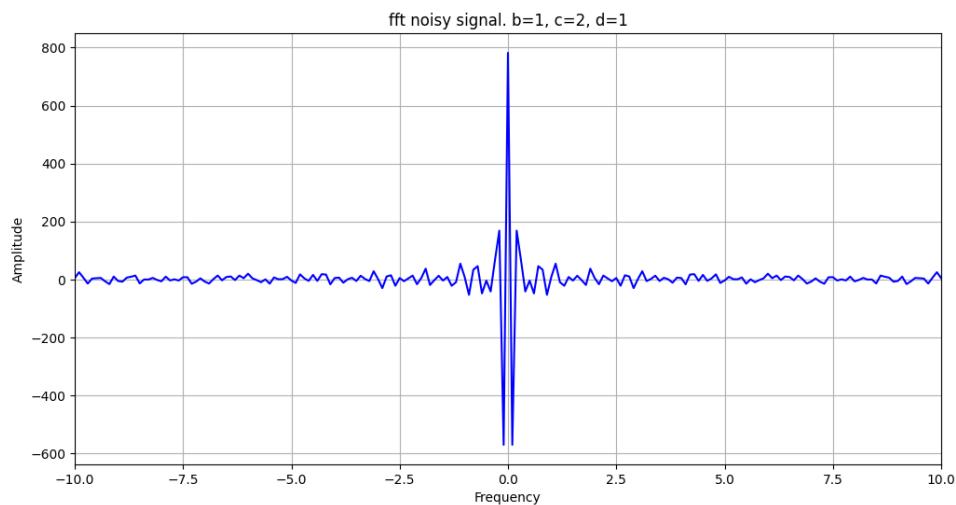


Рис. 78: График Фурье-образа исходного сигнала.

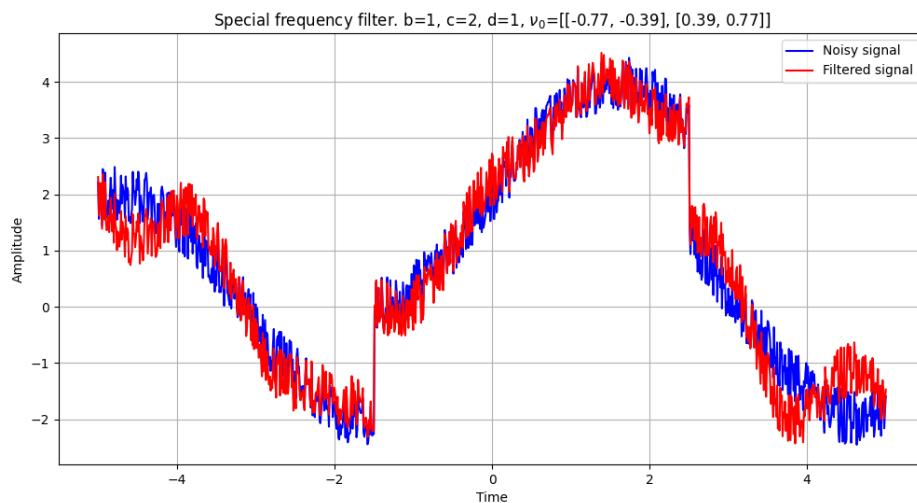


Рис. 79: График исходного и фильтрованного сигналов.

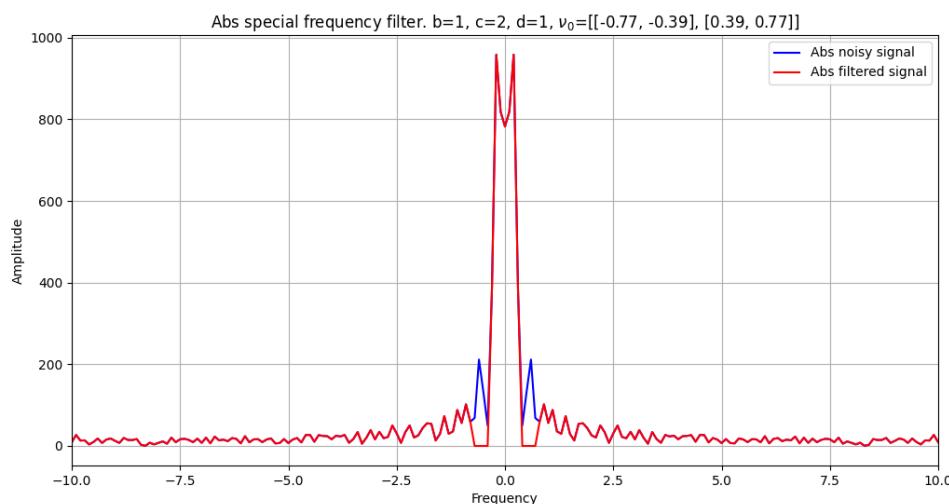


Рис. 80: График модуля Фурье-образа исходного и фильтрованного сигналов.

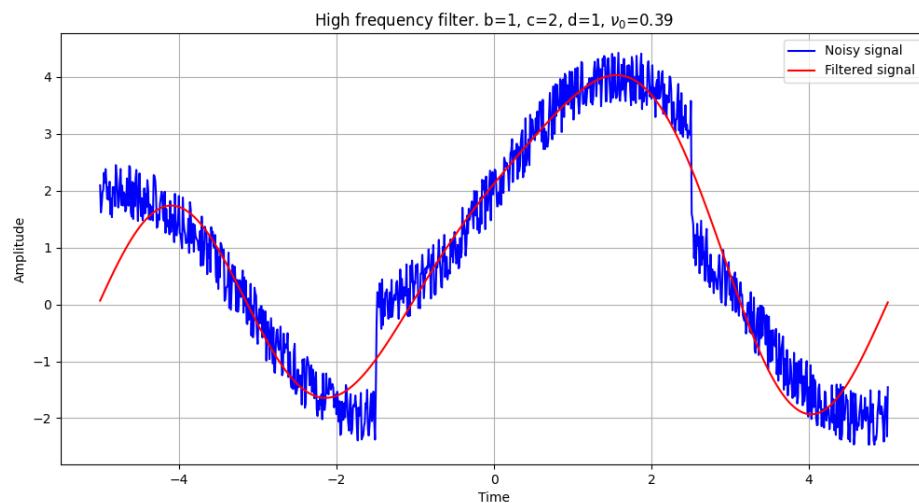


Рис. 81: График исходного и фильтрованного сигналов.

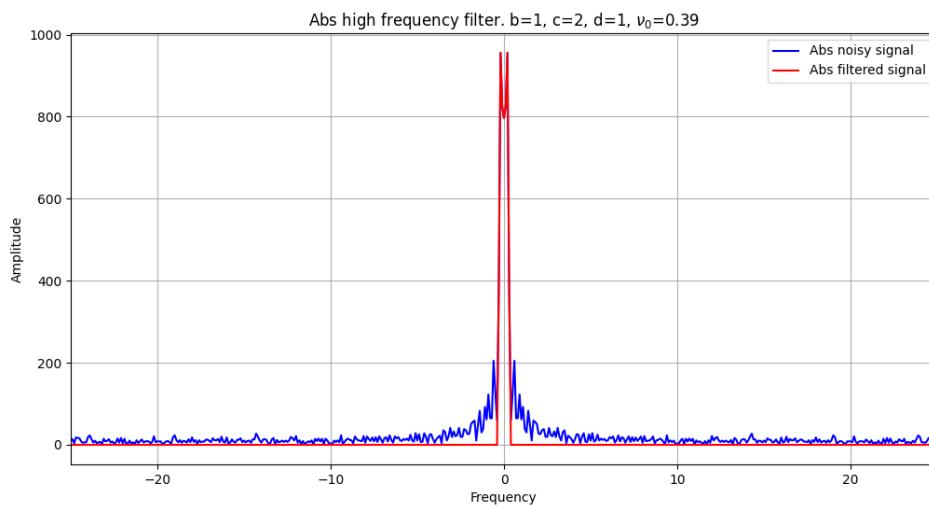


Рис. 82: График модуля Фурье-образа исходного и фильтрованного сигналов.

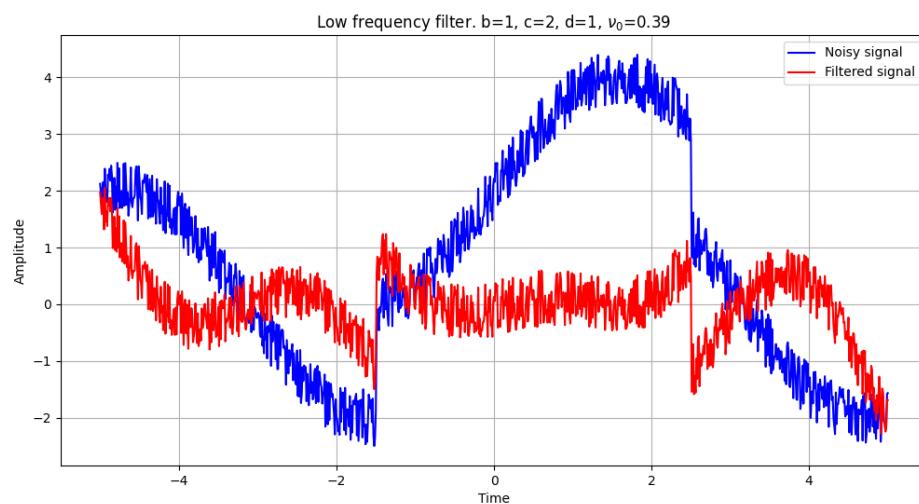


Рис. 83: График исходного и фильтрованного сигналов.

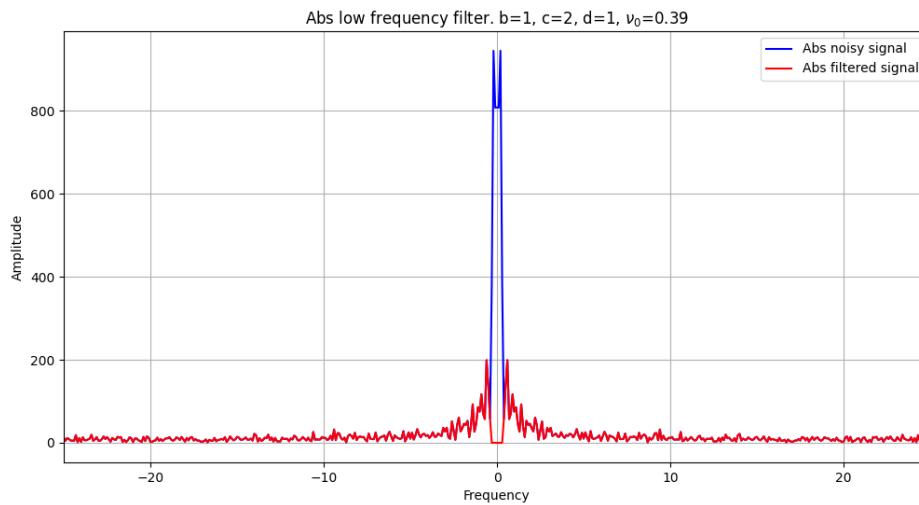


Рис. 84: График модуля Фурье-образа исходного и фильтрованного сигналов.

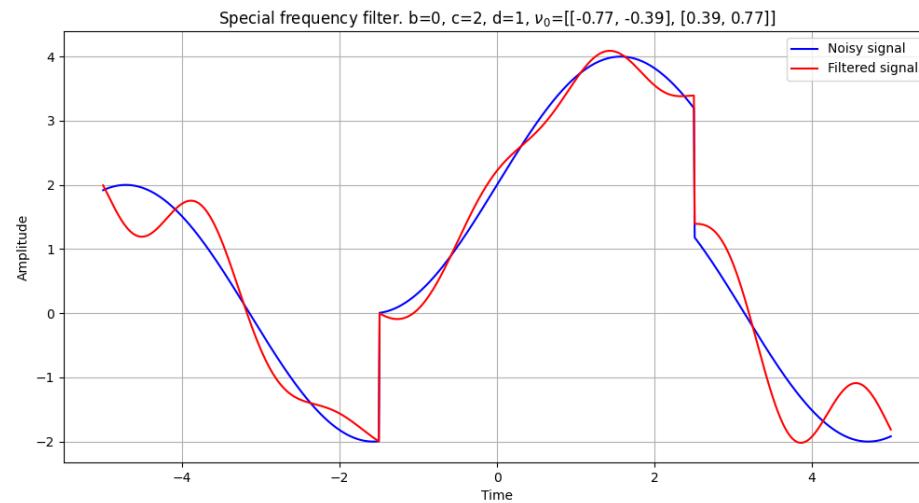


Рис. 85: График исходного и фильтрованного сигналов.

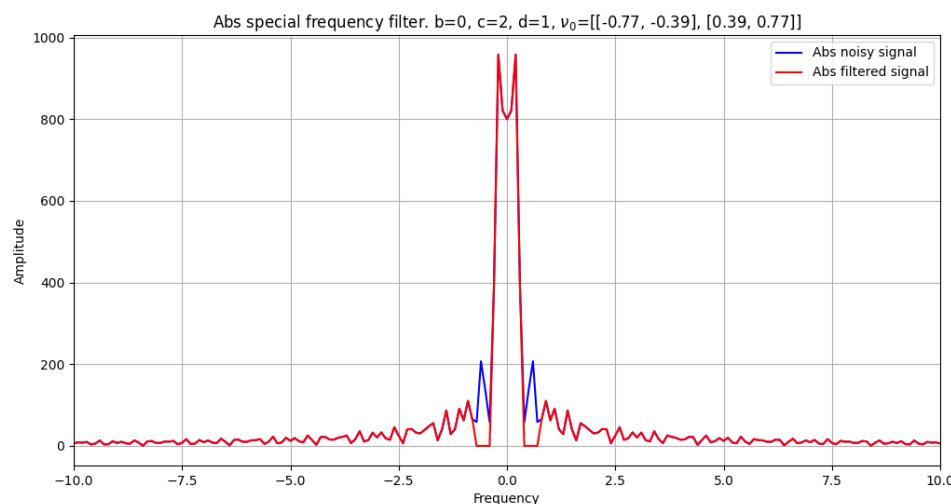


Рис. 86: График модуля Фурье-образа исходного и фильтрованного сигналов.

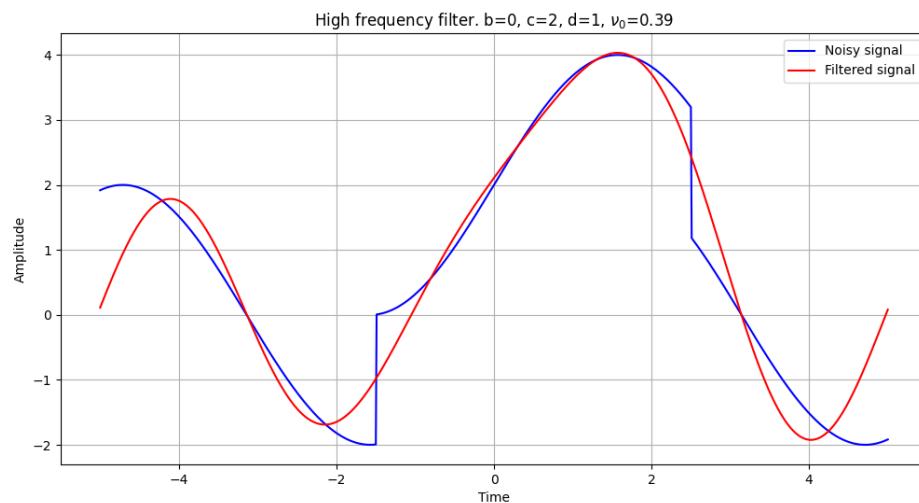


Рис. 87: График исходного и фильтрованного сигналов.

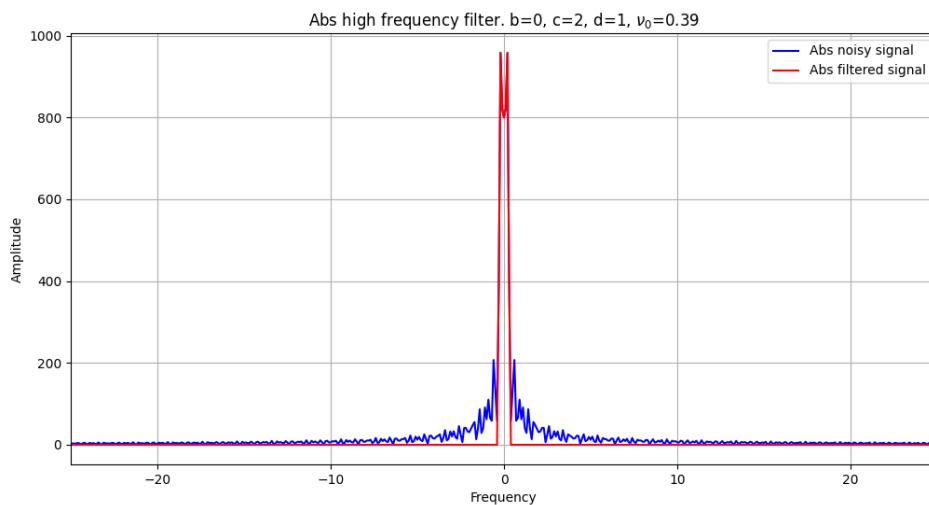


Рис. 88: График модуля Фурье-образа исходного и фильтрованного сигналов.

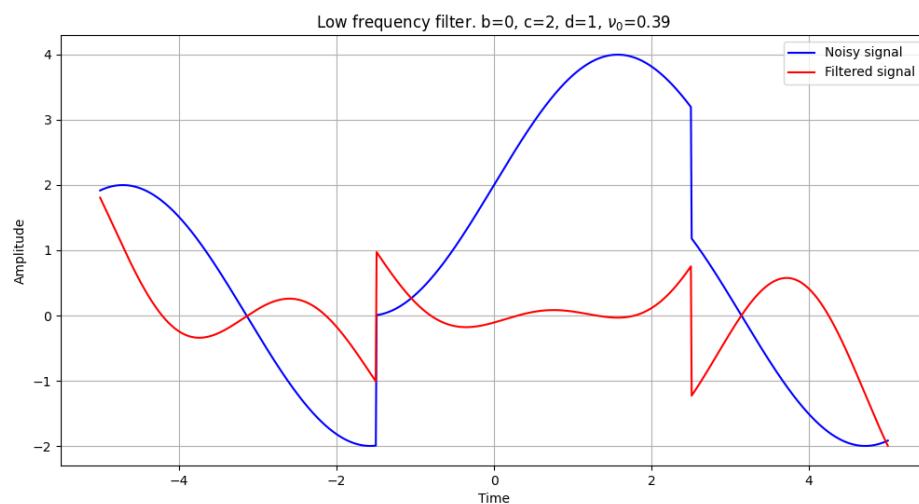


Рис. 89: График исходного и фильтрованного сигналов.

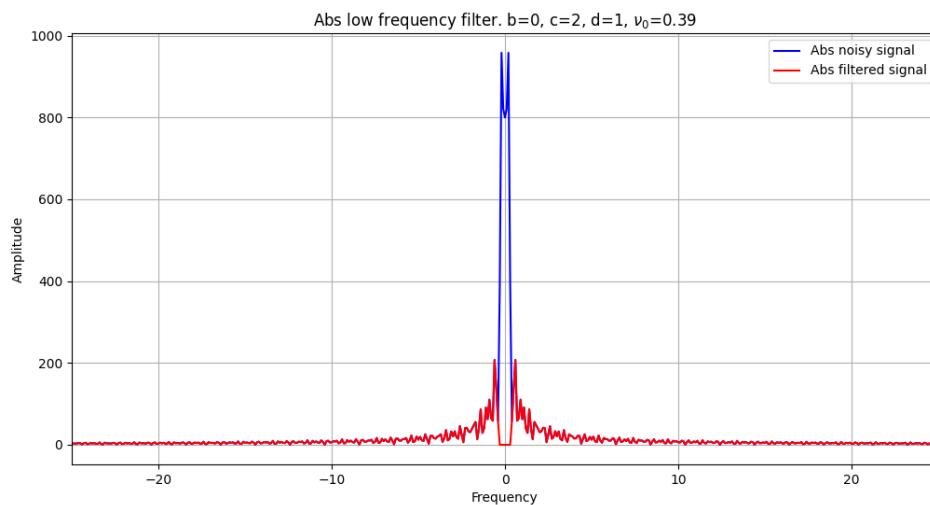


Рис. 90: График модуля Фурье-образа исходного и фильтрованного сигналов.

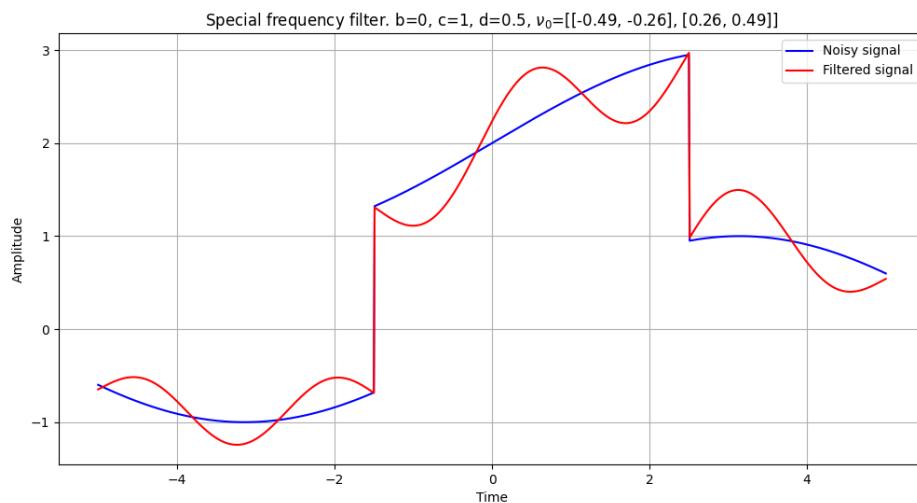


Рис. 91: График исходного и фильтрованного сигналов.

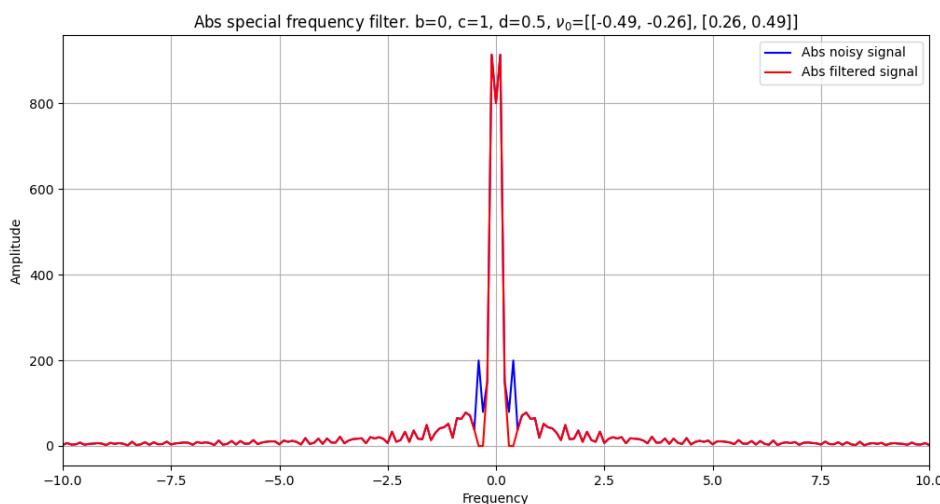


Рис. 92: График модуля Фурье-образа исходного и фильтрованного сигналов.

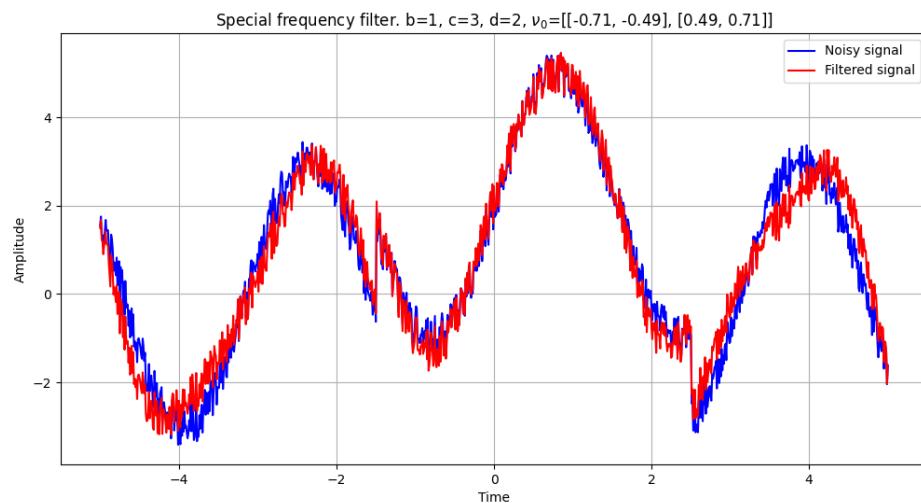


Рис. 93: График исходного и фильтрованного сигналов.

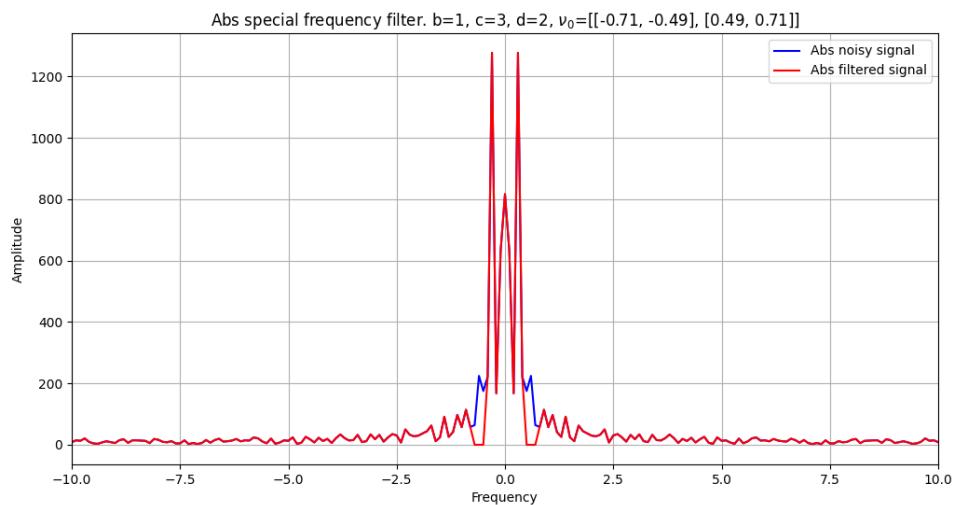


Рис. 94: График модуля Фурье-образа исходного и фильтрованного сигналов.

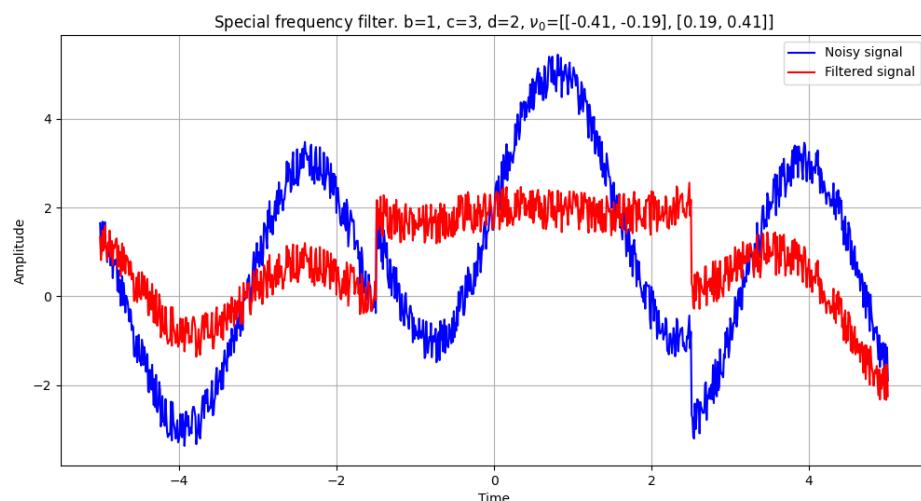


Рис. 95: График исходного и фильтрованного сигналов.

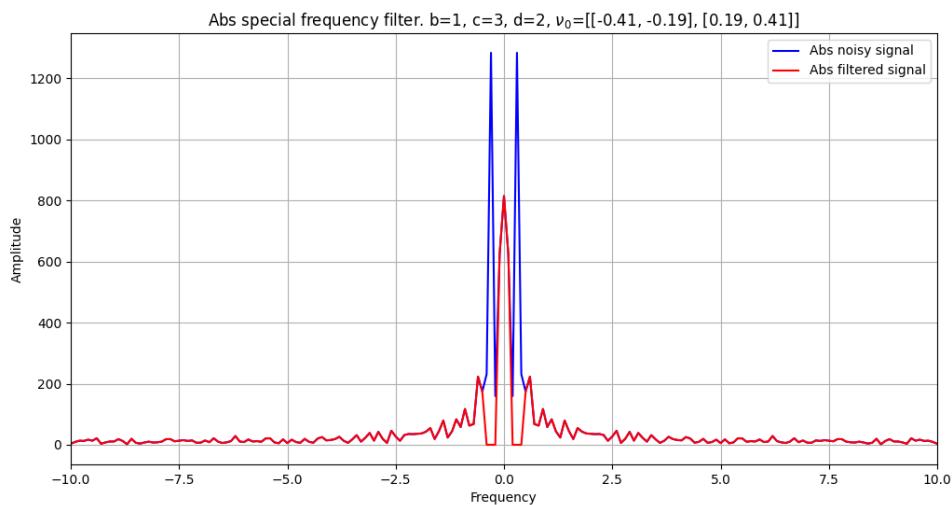


Рис. 96: График модуля Фурье-образа исходного и фильтрованного сигналов.

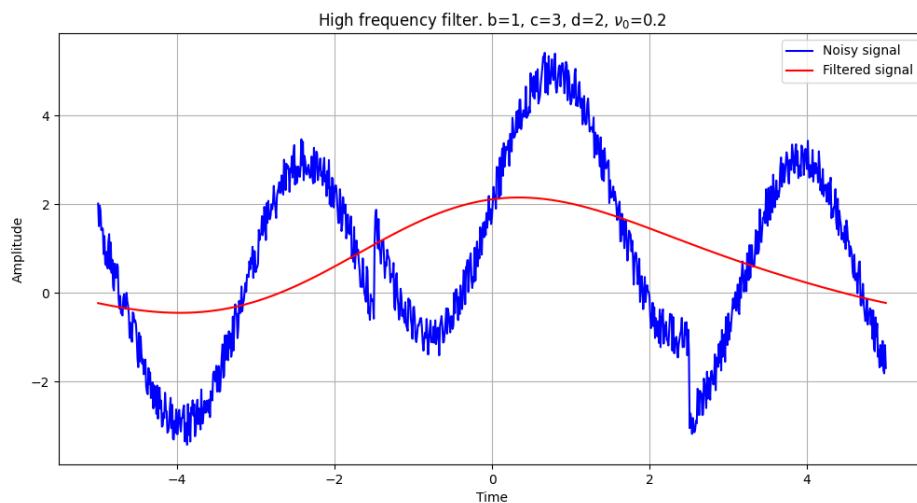


Рис. 97: График исходного и фильтрованного сигналов.

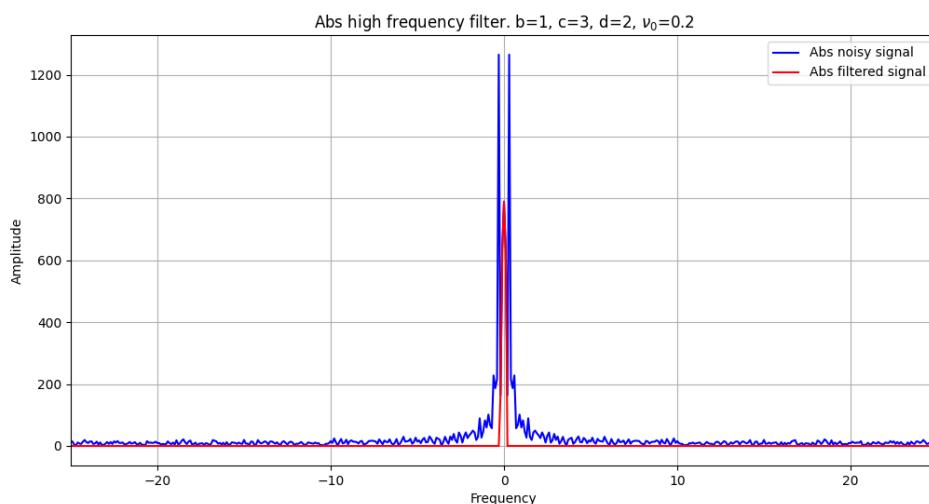


Рис. 98: График модуля Фурье-образа исходного и фильтрованного сигналов.

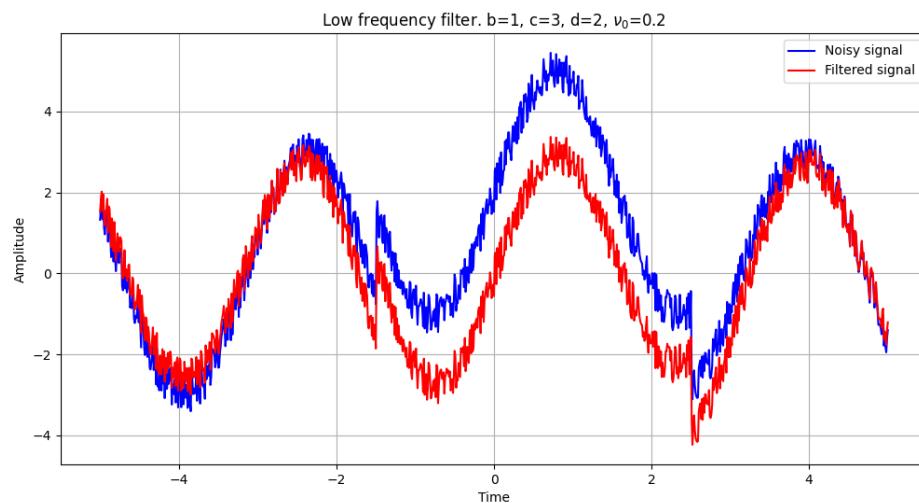


Рис. 99: График исходного и фильтрованного сигналов.

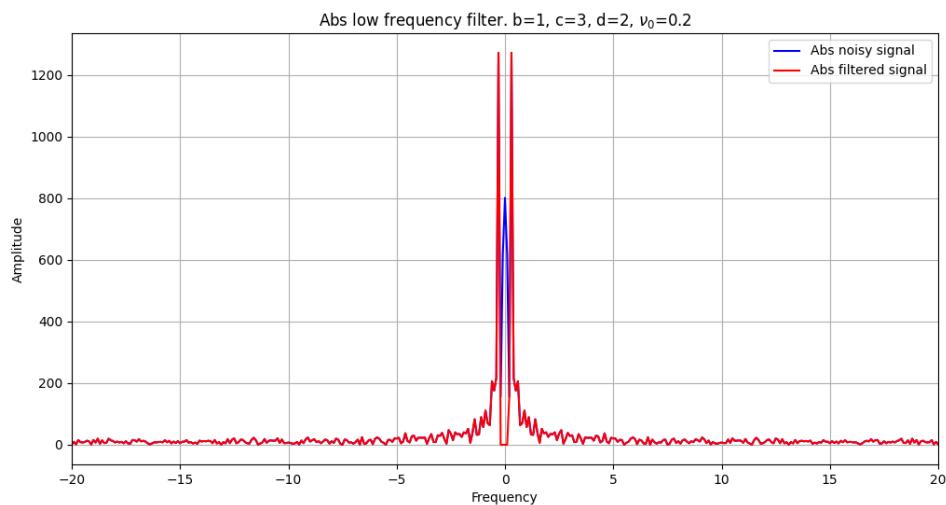


Рис. 100: График модуля Фурье-образа исходного и фильтрованного сигналов.

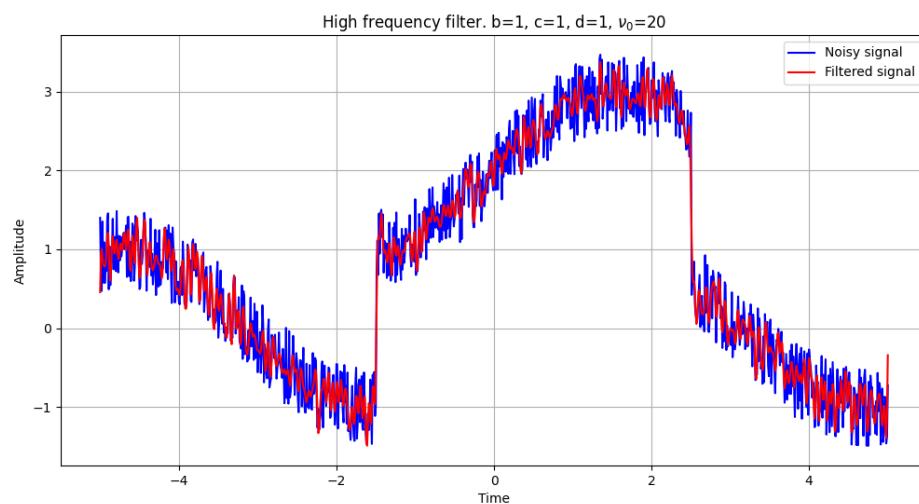


Рис. 101: График исходного и фильтрованного сигналов.

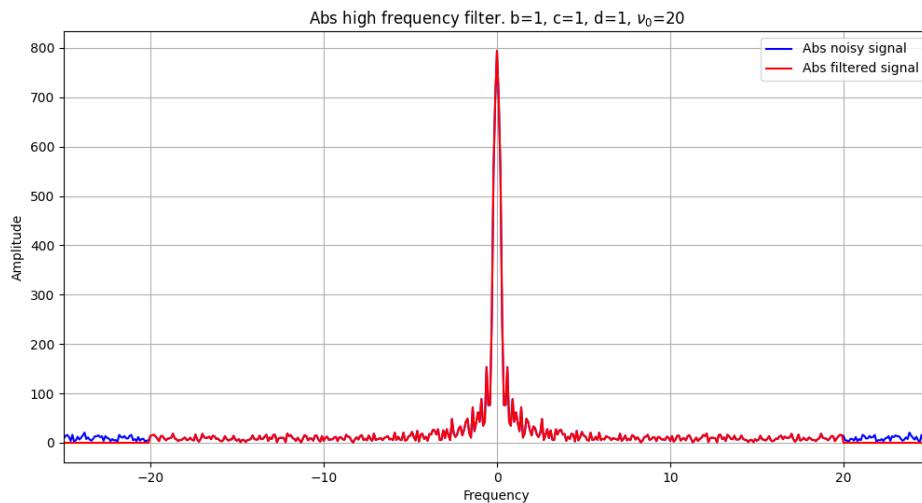


Рис. 102: График модуля Фурье-образа исходного и фильтрованного сигналов.

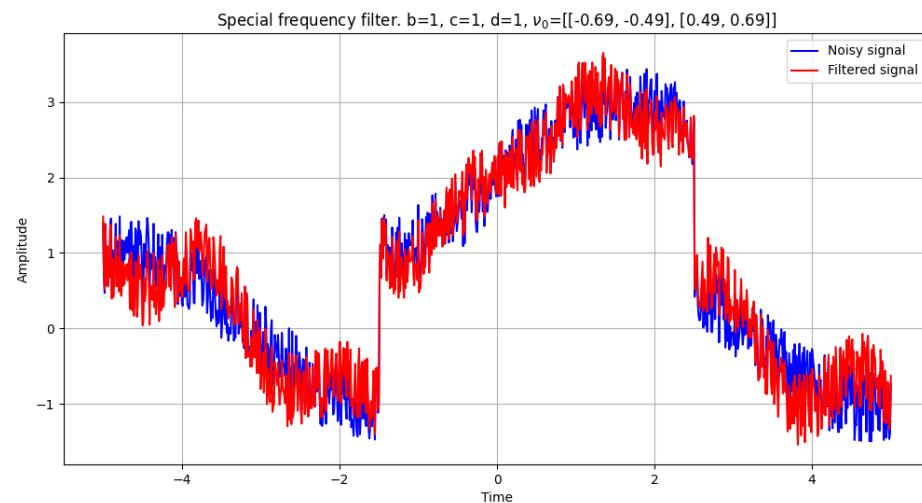


Рис. 103: График исходного и фильтрованного сигналов.

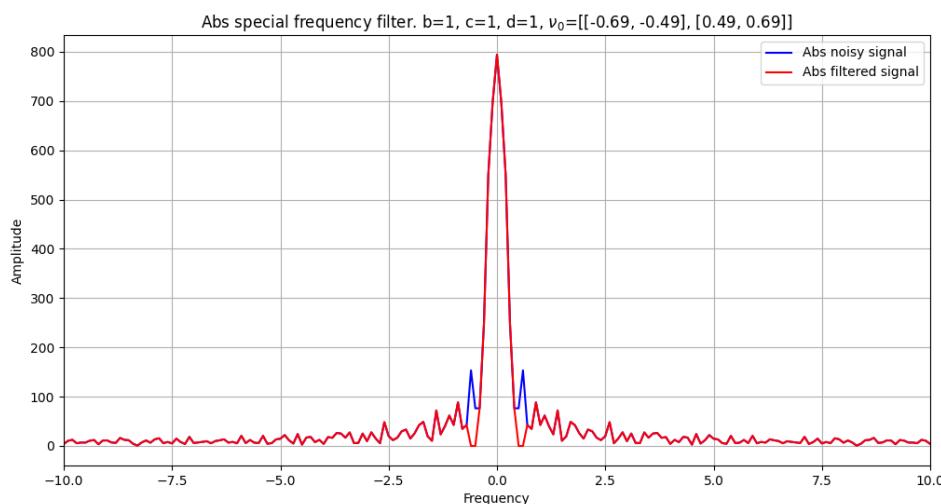


Рис. 104: График модуля Фурье-образа исходного и фильтрованного сигналов.

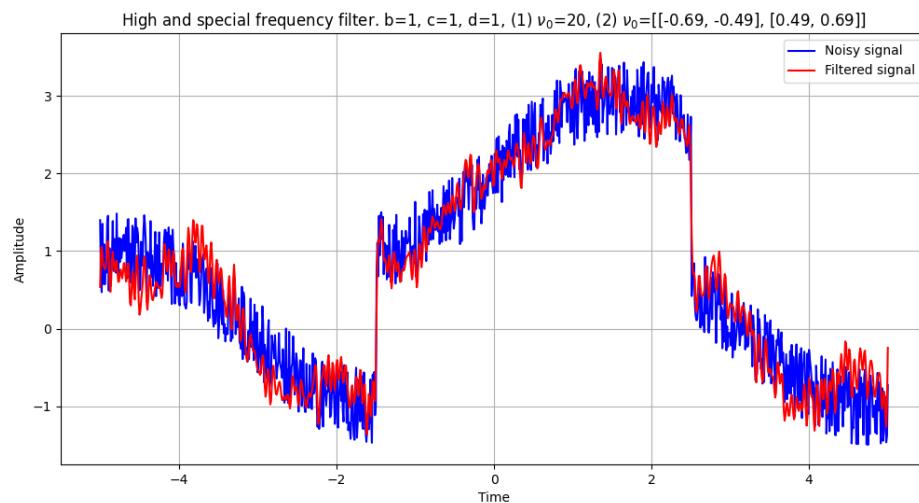


Рис. 105: График исходного и фильтрованного сигналов.

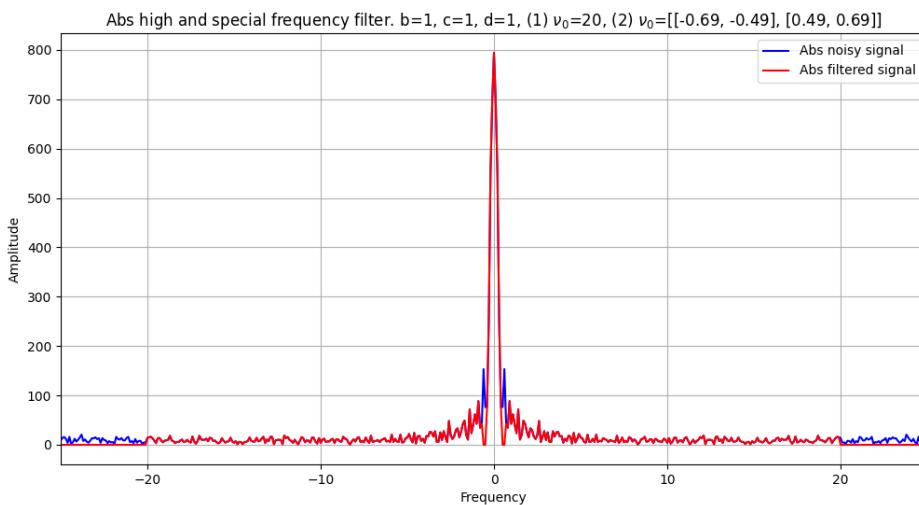


Рис. 106: График модуля Фурье-образа исходного и фильтрованного сигналов.

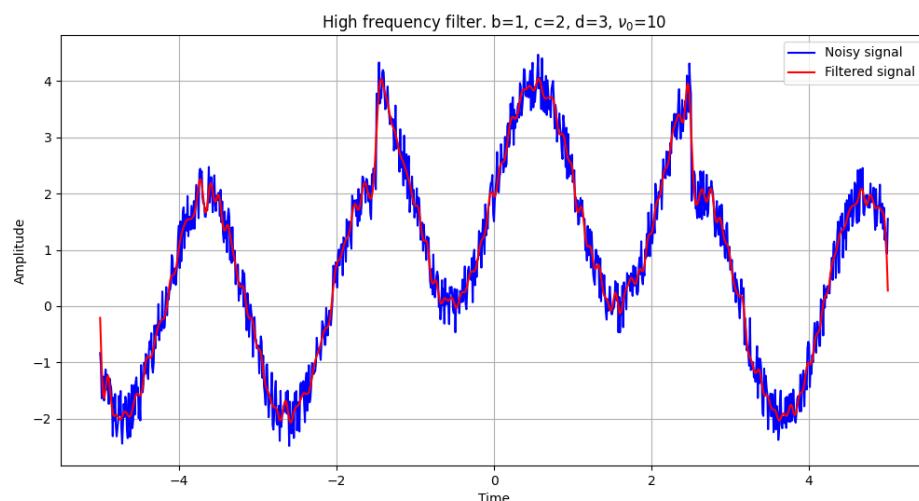


Рис. 107: График исходного и фильтрованного сигналов.

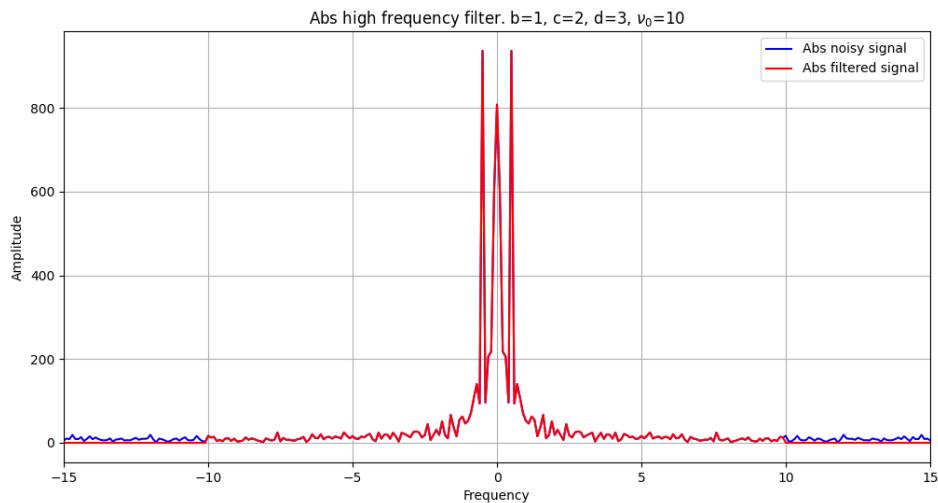


Рис. 108: График модуля Фурье-образа исходного и фильтрованного сигналов.

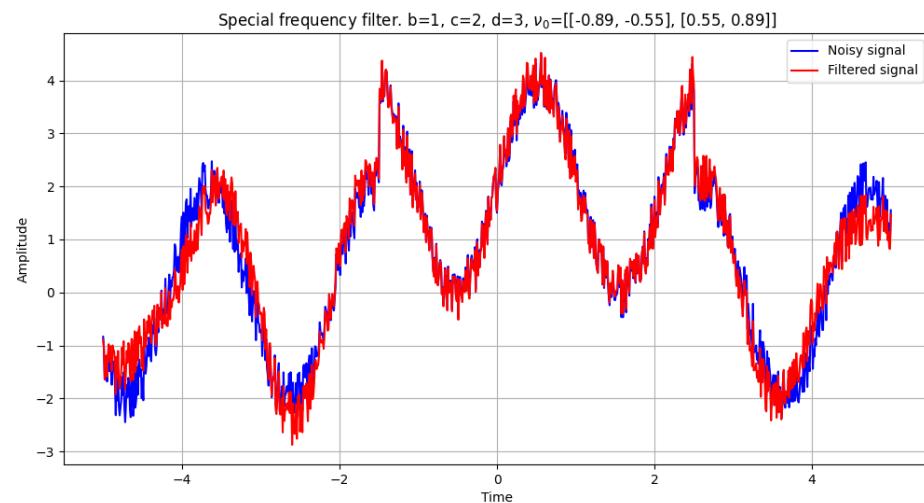


Рис. 109: График исходного и фильтрованного сигналов.

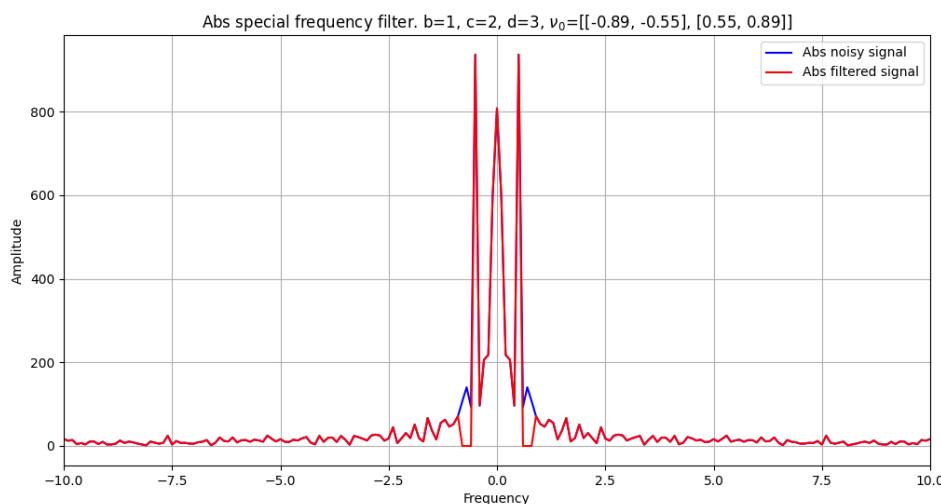


Рис. 110: График модуля Фурье-образа исходного и фильтрованного сигналов.

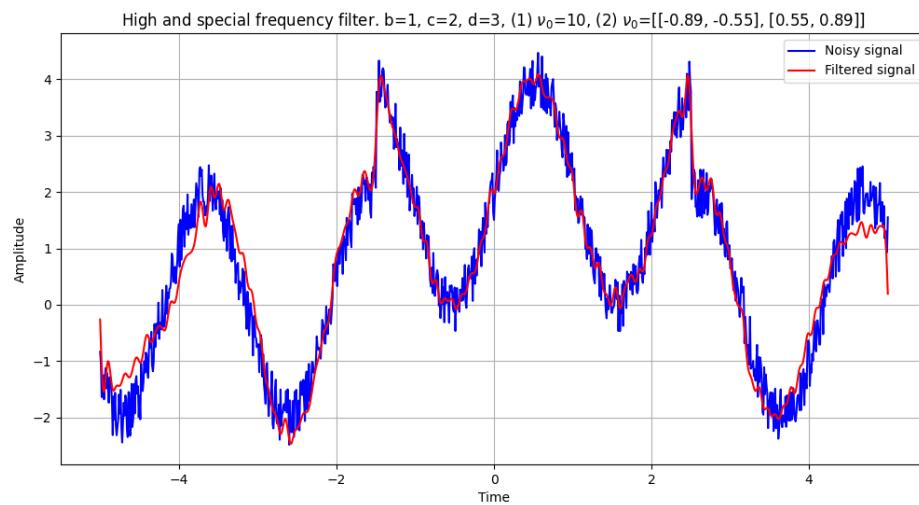


Рис. 111: График исходного и фильтрованного сигналов.

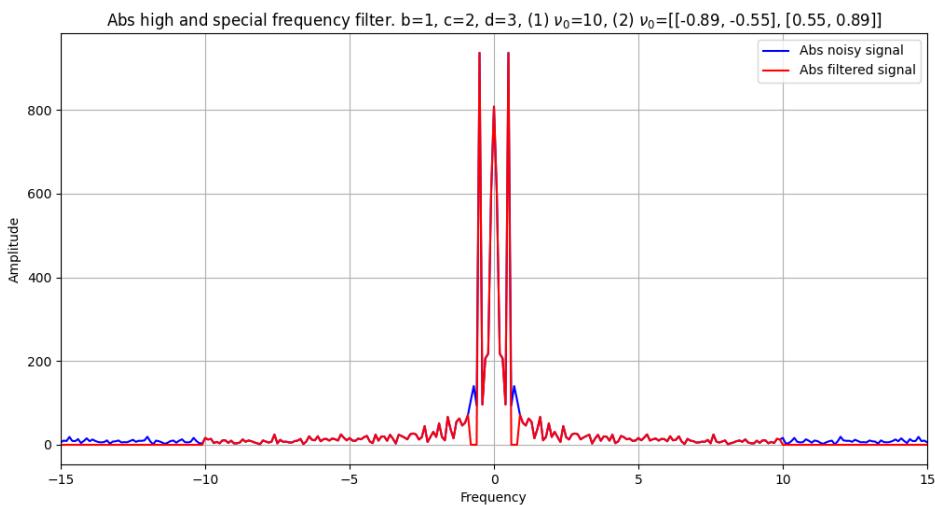


Рис. 112: График модуля Фурье-образа исходного и фильтрованного сигналов.

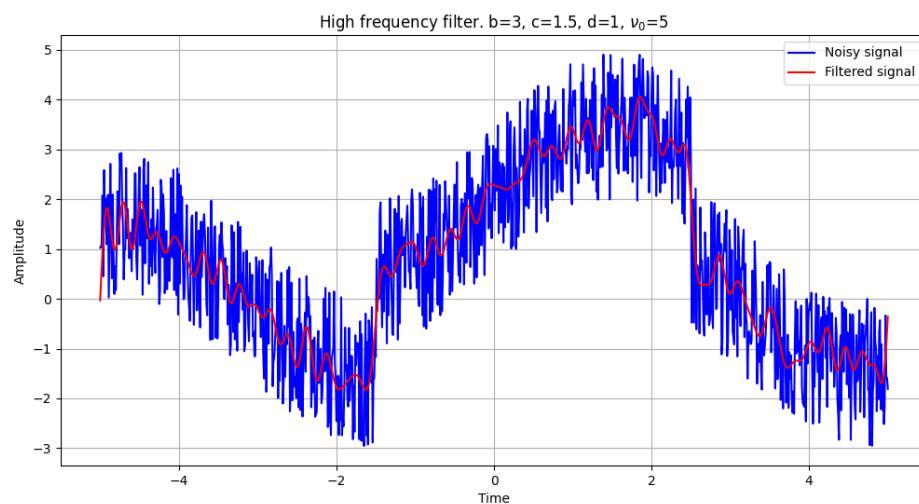


Рис. 113: График исходного и фильтрованного сигналов.

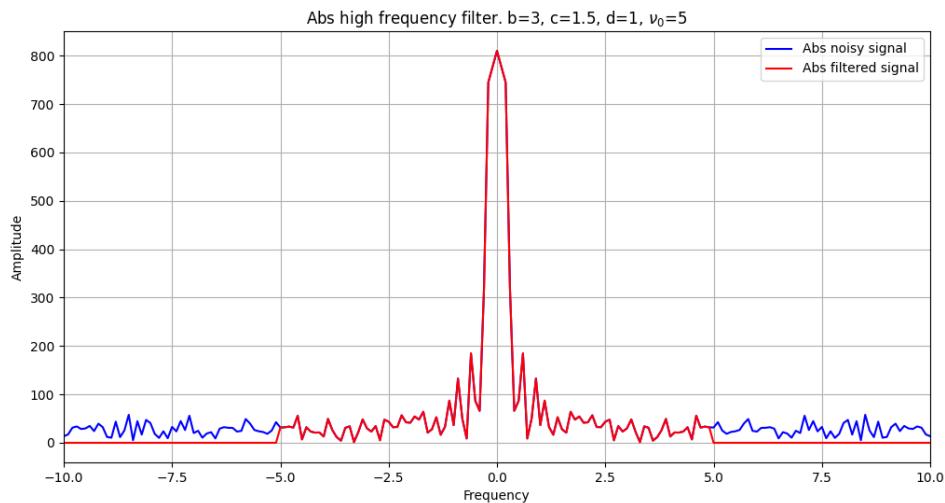


Рис. 114: График модуля Фурье-образа исходного и фильтрованного сигналов.

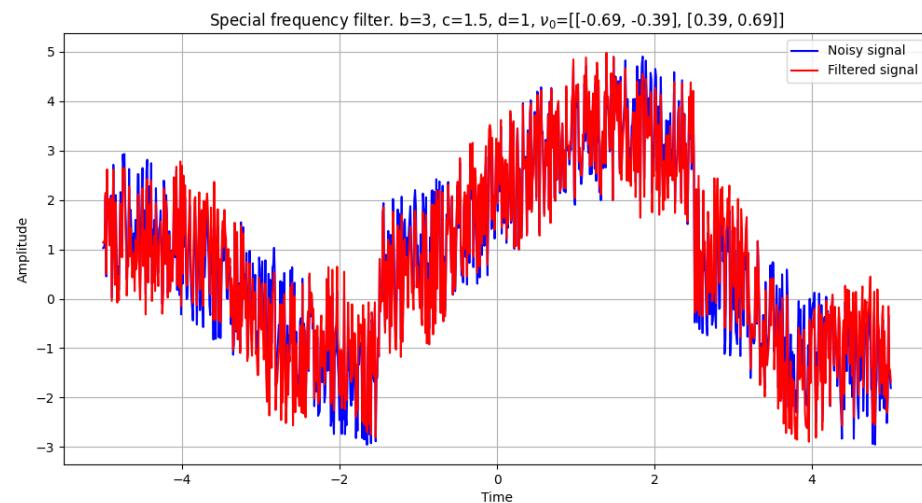


Рис. 115: График исходного и фильтрованного сигналов.

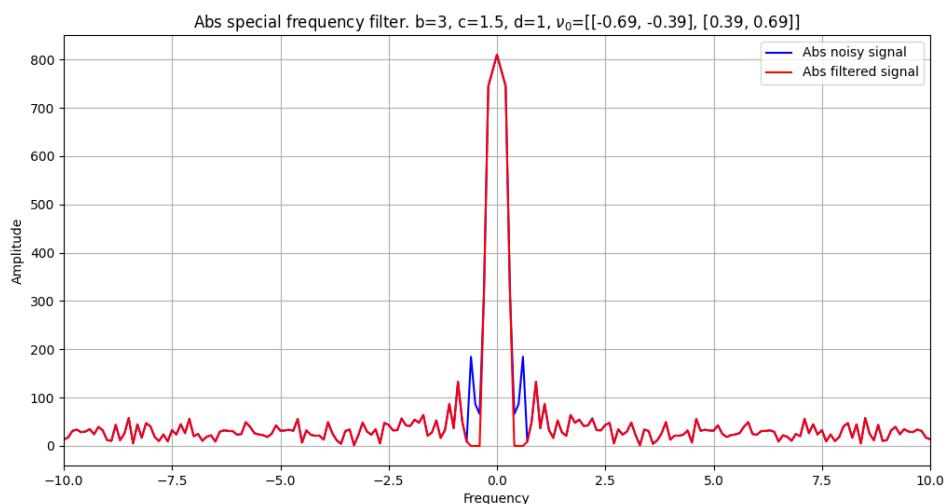


Рис. 116: График модуля Фурье-образа исходного и фильтрованного сигналов.

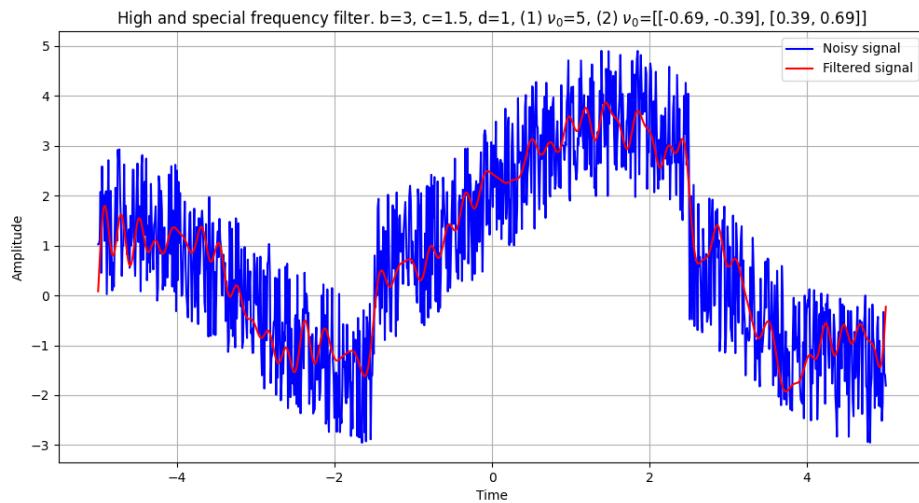


Рис. 117: График исходного и фильтрованного сигналов.

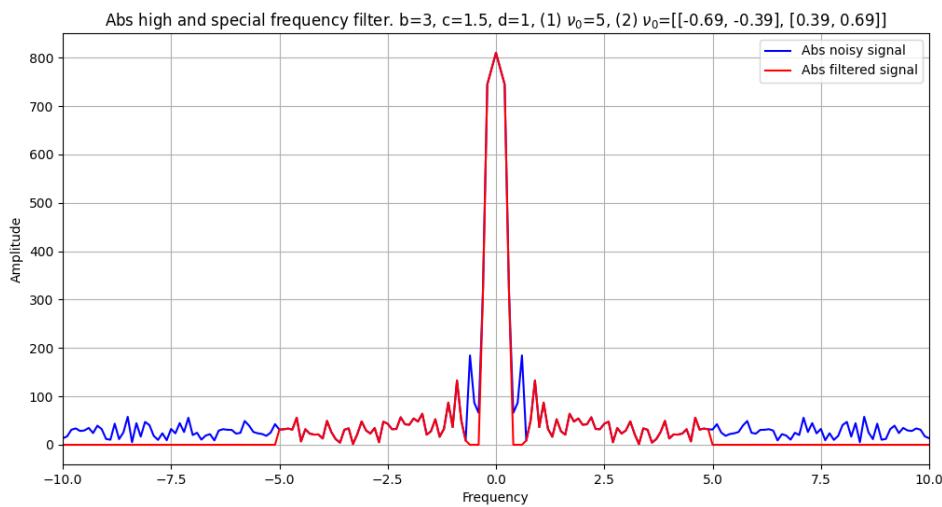


Рис. 118: График модуля Фурье-образа исходного и фильтрованного сигналов.

график исходного и фильтрованного сигналов аудиозаписи. Можем наблюдать, насколько чище стал сигнал — оставшиеся амплитуды являются голосом, который нам нужен. Это легко понять исходя из того, что между каждым возрастанием амплитуд есть интервал с падением амплитуд, где они находятся в некоторой окрестности нуля — это паузы в предложении, которое говорит человек на записи. Также построим сравнительный график модуля исходного и фильтрованного сигналов аудиозаписи. На нем мы видим, что мы успешно вырезали низкие частоты с наибольшей амплитудой, которые соответствовали громкому гулу. Теперь послушаем аудиозапись `filtered_MUHA.wav`, которая оставлена на этом [гугл-диске](#), и убедимся в том, что все посторонние шумы пропали. На записи голос слышно хорошо, однако остался некоторый звуковой эффект фейзер. Избавиться от него удалось обрезав частоты в диапазоне $[-22050, -1000]$ и $[1000, 22050]$ Гц, то есть применив фильтр низких частот, однако голос сильно потерял в качестве. Прослушать этот вариант можно на том же [гугл-диске](#), файл выложен под названием `filtered_MUHA_2.wav`.

Далее представлены рисунки с графиками, о которых говорилось в предыдущем абзаце. Синим цветом обозначен исходный сигнал, красным — фильтрованный. Также представлены графики для фильтрованной аудиозаписи `filtered_MUHA_2.wav`. Стоит заметить, что на рисунке 124 видно, как окрестность нуля стала меньше по сравнению с ри-

сунком 121, то есть в подобных диапазонах амплитуды частот уменьшились (убрали высокочастотный эффект фейзер, оставив диапазон с низкими частотами, но без тех, что создавали громкий гул).

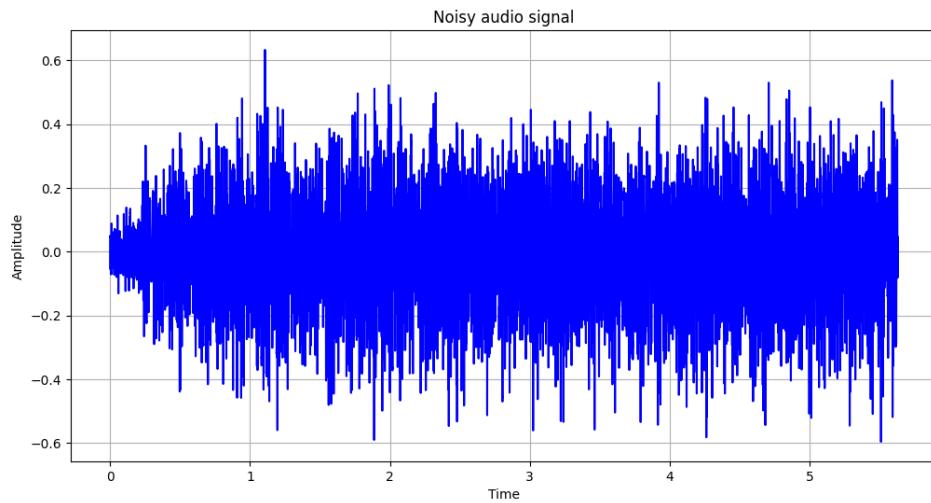


Рис. 119: График исходного сигнала аудиозаписи.

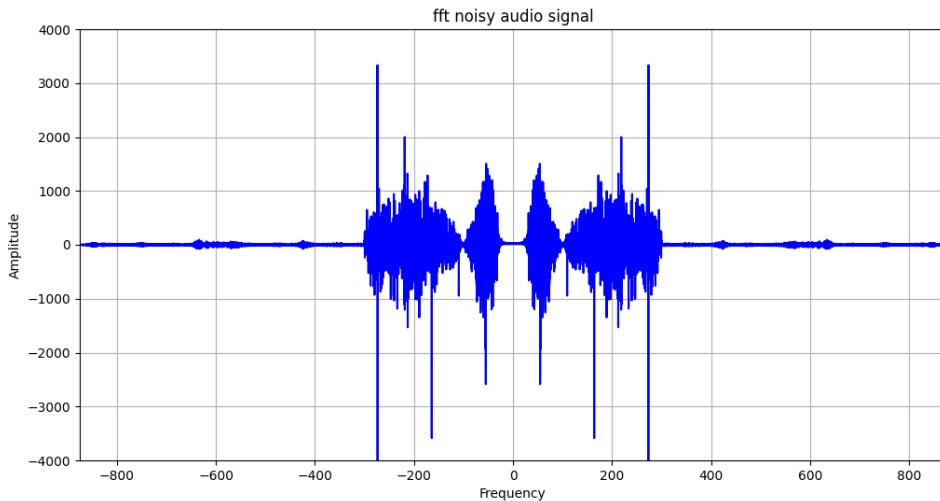


Рис. 120: График Фурье-образа исходного сигнала аудиозаписи.

3 Листинги программных реализаций

В этой секции будут рассмотрены программные реализации, которые использовались по ходу выполнения лабораторной работы. Программы написаны на языке python с подключенными библиотеками numpy и matplotlib.

Два файла, которые необходимы для задания массива времени и частот, вычисления массива функций g , задания сигнала u , а также подсчета Фурье-образа сигнала u . Достаточно передать в функции необходимые параметры и далее их результат можно использовать для выполнения фильтрации и построения графиков.

```
1 def get_t(T, dt):
2     return np.arange(-T/2, T/2 + dt, dt)
```

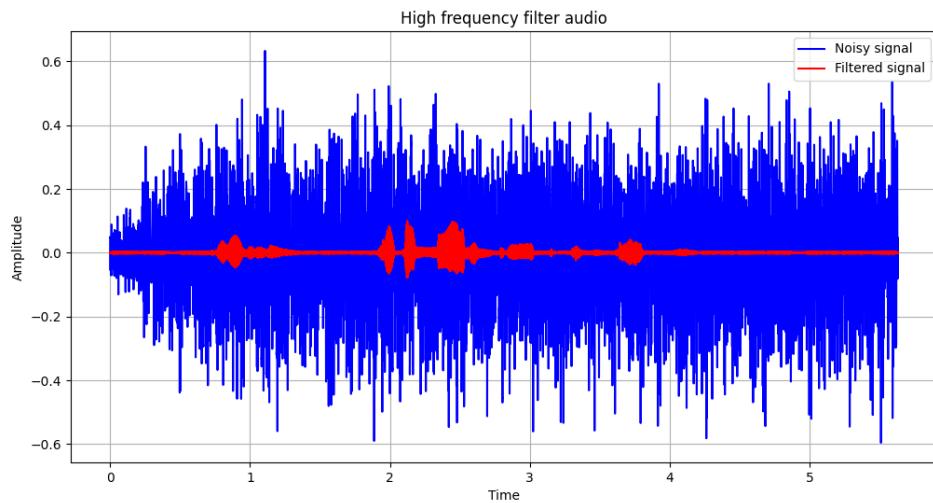


Рис. 121: График исходного и фильтрованного сигналов аудиозаписи.

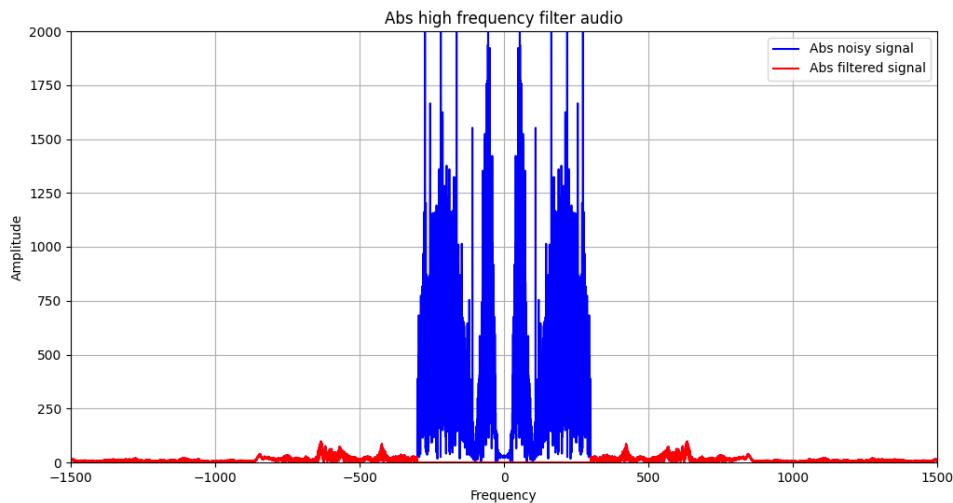


Рис. 122: График модуля Фурье-образа исходного и фильтрованного сигналов аудиозаписи.

```

3
4     def get_v(v, dv):
5         return np.arange(-v/2, v/2 + dv, dv)
6
7     def get_U(u):
8         return np.fft.fftshift(np.fft.fft(u))
9
10    def g_f(t, t_1, t_2, a):
11        if (t_1 <= t <= t_2):
12            return a
13        return 0
14
15    def u_f(g_fs: list, time: list, b, c, d):
16        return np.array(g_fs) + \
17            b*(np.random.rand(len(time))-0.5) + \
18            c*np.sin(d*time)
19
20    def get_g_fs(time: list, t_1, t_2, a):
21        gs = []

```

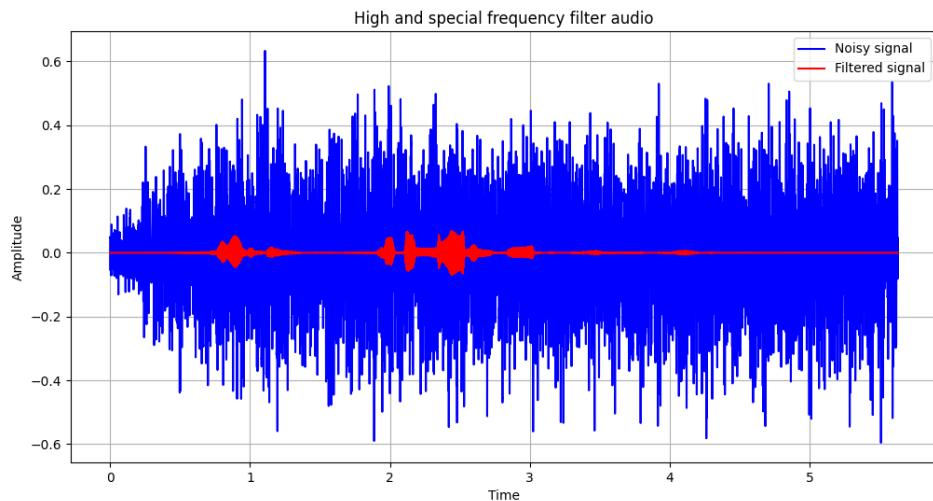


Рис. 123: График исходного и фильтрованного сигналов аудиозаписи.

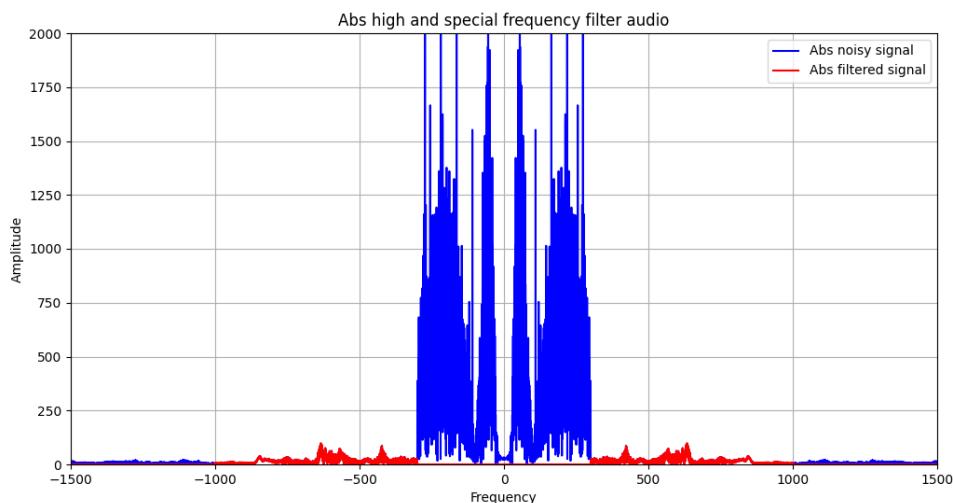


Рис. 124: График модуля Фурье-образа исходного и фильтрованного сигналов аудиозаписи.

```

22     for t in range(len(time)):
23         gs.append(g_f(time[t], t_1, t_2, a))
24
25     return gs

```

Листинг 1: Файл help.py. Вспомогательные функции.

```

1   a = 2
2   t_1 = -1.5
3   t_2 = 2.5
4
5   T = 10
6   dt = 0.01
7
8   V = 1/dt
9   dv = 1/T
10
11  time = hp.get_t(T, dt)
12  freq = hp.get_v(V, dv)

```

```
13     g_fs = hp.get_g_fs(time, t_1, t_2, a)
```

Листинг 2: Файл static.py. Вспомогательные переменные.

Программа для построения сравнительных графиков исходного и фильтрованного сигналов (функция build_u__flt_u), модуля Фурье-образа исходного и фильтрованного сигналов (функции build_abs_U__flt_U и build_abs_u_to_U__flt_U, где вторая из приведенных в данном абзаце – вспомогательная). В нее можно передать исходный сигнал u , который преобразуется в Фурье-образ. Написана лишь для удобства пользователя) и графиков исходного сигнала или его Фурье-образа (функции build_u_or_U и build_u_to_U, где вторая из приведенных в данном абзаце нужна для удобства). Все функции принимают максимально много необязательных параметров для гибкой настройки графика.

```
1  def build_u_to_U(freq: list, u: list, clr='b',
2                     x11=None, x12=None, y11=None,
3                     y12=None, xlab='Frequency', ylab='Amplitude',
4                     label=None, title=None, fz1=6.4,
5                     fz2=4.8, legend: bool = True, grid: bool = True):
6     U = get_U(u)
7     build_u_or_U(freq, U, clr,
8                   x11, x12, y11,
9                   y12, xlab, ylab,
10                  label, title, fz1,
11                  fz2, legend, grid)
12
13    def build_u_or_U(torv: list, uorU: list, clr='b',
14                      x11=None, x12=None, y11=None,
15                      y12=None, xlab=None, ylab='Amplitude',
16                      label=None, title=None, fz1=6.4,
17                      fz2=4.8, legend: bool = True, grid: bool = True):
18        plt.plot(torv, uorU, color=clr, label=label)
19        plt.xlabel(xlab)
20        plt.ylabel(ylab)
21        plt.xlim(x11, x12)
22        plt.ylim(y11, y12)
23        plt.title(title)
24        if legend:
25            plt.legend()
26        plt.grid(grid)
27        plt.gcf().set_size_inches(fz1, fz2)
28        plt.show()
29
30    def build_u__flt_u(time: list, u: list, flt_u: list,
31                       clr1='b', clr2='r', lab1='Noisy signal',
32                       lab2='Filtered signal', xlab='Time', ylab='Amplitude',
33                       title=None, fz1=6.4, fz2=4.8,
34                       legend: bool = True, grid: bool = True, x11=None,
35                       x12=None, y11=None, y12=None):
36        plt.plot(time, u, color=clr1, label=lab1)
37        plt.plot(time, flt_u, color=clr2, label=lab2)
38        plt.xlabel(xlab)
39        plt.ylabel(ylab)
40        plt.xlim(x11, x12)
41        plt.ylim(y11, y12)
42        plt.title(title)
43        if legend:
44            plt.legend()
45        plt.grid(grid)
46        plt.gcf().set_size_inches(fz1, fz2)
47        plt.show()
```

```

48
49     def build_abs_u_to_U_flt_U(freq: list, u: list, flt_U: list,
50         clr1='b', clr2='r', lab1='Abs noisy signal',
51         lab2='Abs filtered signal', xlab='Frequency', ylab='
52             Amplitude',
53             xl1=None, xl2=None, yl1=None,
54             yl2=None, title=None, fz1=6.4,
55             fz2=4.8, legend: bool = True, grid: bool = True):
56     U = get_U(u)
57     build_abs_U_flt_U(freq, U, flt_U,
58         clr1, clr2, lab1,
59         lab2, xlab, ylab,
60         xl1, xl2, yl1,
61         yl2, title, fz1,
62         fz2, legend, grid)
63
64     def build_abs_U_flt_U(freq: list, U: list, flt_U: list,
65         clr1='b', clr2='r', lab1='Abs noisy signal',
66         lab2='Abs filtered signal', xlab='Frequency', ylab='
67             Amplitude',
68             xl1=None, xl2=None, yl1=None,
69             yl2=None, title=None, fz1=6.4,
70             fz2=4.8, legend: bool = True, grid: bool = True):
71         plt.plot(freq, np.abs(U), color=clr1, label=lab1)
72         plt.plot(freq, np.abs(flt_U), color=clr2, label=lab2)
73         plt.xlabel(xlab)
74         plt.ylabel(ylab)
75         plt.xlim(xl1, xl2)
76         plt.ylim(yl1, yl2)
77         plt.title(title)
78         if legend:
79             plt.legend()
80         plt.grid(grid)
81         plt.gcf().set_size_inches(fz1, fz2)
82         plt.show()

```

Листинг 3: Файл builder.py. Реализация построения графиков.

Программа, реализующая фильтрацию. Пользователь может вызвать такие функции, как filter_high, filter_low, filter_special_out/in для фильтрации верхних, нижних и специфических частот соответственно. Первые две функции принимают некоторый параметр ν_0 , а последняя принимает список диапазонов с некоторыми ν_0^i и ν_0^j (где i, j – индексы, не степени). Здесь реализован алгоритм, описанный в первом задании – берется Фурье-образ, фильтруется и преобразуется обратно из частотного пространства во временное. Функция filter_U обнуляет частоты, соответствующие конкретному шагу в зависимости от решения фильтров проверки high_filter, low_filter и special_filter_out/in.

```

1     def filter_U(u: list, freq: list, v_0, filter):
2         flt_U = get_U(u)
3         for i in range(len(freq)):
4             freq_i = freq[i]
5             if filter(freq_i, v_0):
6                 flt_U[i] = 0
7
8         return flt_U
9
10    def low_filter(freq, v_0):
11        if -v_0 <= freq <= v_0:
12            return False
13        return True

```

```

14
15     def high_filter(freq, v_0):
16         return not low_filter(freq, v_0)
17
18     def special_filter_in(freq, v_0: list):
19         for i in range(len(v_0)):
20             if v_0[i][0] <= freq <= v_0[i][1]:
21                 return False
22         return True
23
24     def special_filter_out(freq, v_0: list):
25         return not special_filter_in(freq, v_0)
26
27     def filter_low(freq: list, u: list, v_0):
28         if isinstance(v_0, list) or \
29             len(freq) <= 0 or \
30             len(u) <= 0:
31             return None
32
33         flt_U = filter_U(u, freq, v_0, low_filter)
34         flt_u = np.fft.ifft(np.fft.ifftshift(flt_U))
35         return flt_u, flt_U
36
37     def filter_high(freq: list, u: list, v_0):
38         if isinstance(v_0, list) or \
39             len(freq) <= 0 or \
40             len(u) <= 0:
41             return None
42
43         flt_U = filter_U(u, freq, v_0, high_filter)
44         flt_u = np.fft.ifft(np.fft.ifftshift(flt_U))
45         return flt_u, flt_U
46
47     def filter_special_in(freq: list, u: list, v_0: list):
48         if not isinstance(v_0, list) or \
49             len(v_0) <= 0 or \
50             len(freq) <= 0 or \
51             len(u) <= 0:
52             return None
53
54         flt_U = filter_U(u, freq, v_0, special_filter_in)
55         flt_u = np.fft.ifft(np.fft.ifftshift(flt_U))
56         return flt_u, flt_U
57
58     def filter_special_out(freq: list, u: list, v_0: list):
59         if not isinstance(v_0, list) or \
60             len(v_0) <= 0 or \
61             len(freq) <= 0 or \
62             len(u) <= 0:
63             return None
64
65         flt_U = filter_U(u, freq, v_0, special_filter_out)
66         flt_u = np.fft.ifft(np.fft.ifftshift(flt_U))
67         return flt_u, flt_U

```

Листинг 4: Файл filters.py. Реализация фильтров.

Далее представлены программы, в которых используются все предыдущие наработки. Типовой алгоритм – задать параметры b, c, d и некоторый ν_0 , далее воспользоваться функциями создания зашумленного сигнала, фильтрации нужных частот и построения

необходимых графиков. Любые остальные добавления в код нужны лишь для удобства, например, когда нужно построить много графиков и сравнивать их друг с другом или анализировать различные результаты по типу совмещения нескольких фильтров на одном сигнале. В случае очистки специфических частот задается список ν_0^i, ν_0^j . В каджый файл импортируются numpy как np, static как st, help как hp и filters как ft. Для работы с аудиозаписью подключены библиотеки librosa, scipy и playsound.

```

1   time = st.time
2   freq = st.freq
3   g_fs = st.g_fs
4
5   b = 0.5
6   c = 0
7   d = 0.1
8   v_0 = st.V / 10
9
10  u = hp.u_f(g_fs, time, b, c, d)
11  flt_u, flt_U = ft.filter_low(freq, u, v_0)
12
13  bd.build_u__flt_u(time, u, flt_u,
14      title=rf'Low frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
15      fz1=12, fz2=6)
16  bd.build_abs_u_to_U__flt_U(freq, u, flt_U,
17      title=rf'Abs low frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
18      x11=-20, x12=20, fz1=12, fz2=6)
```

Листинг 5: Файл nohigh.py. Фильтрация нижних частот.

```

1   time = st.time
2   freq = st.freq
3   g_fs = st.g_fs
4
5   b = 0.5
6   c = 1
7   d = 0.1
8   v_0 = st.V / 10
9
10  u = hp.u_f(g_fs, time, b, c, d)
11  flt_u, flt_U = ft.filter_high(freq, u, v_0)
12
13  bd.build_u__flt_u(time, u, flt_u,
14      title=rf'High frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
15      fz1=12, fz2=6)
16  bd.build_abs_u_to_U__flt_U(freq, u, flt_U,
17      title=rf'Abs high frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
18      fz1=12, fz2=6, x11=-20, x12=20)
```

Листинг 6: Файл nolow.py. Фильтрация верхних частот.

```

1   time = st.time
2   freq = st.freq
3   g_fs = st.g_fs
4
5   b = 1
6   c = 2
7   d = 1
8   v_0 = 0.39
9   v_0_1 = 0.39
10  v_0_2 = [[-0.77, -0.39], [0.39, 0.77]]
11  u = hp.u_f(g_fs, time, b, c, d)
12
```

```

13     build_u_U = True
14     filter_low = False
15     filter_high = False
16     filter_special_out = True
17
18     flt_u, flt_U = None, None
19     flt_u_0, flt_U_0 = None, None
20     flt_u_1, flt_U_1 = None, None
21     flt_u_2, flt_U_2 = None, None
22
23 if build_u_U:
24     bd.build_u_or_U(time, u, xlab='Time',
25                     title=f'Noisy signal. b={b}, c={c}, d={d}',
26                     legend=False, fz1=12, fz2=6)
27     bd.build_u_to_U(freq, u,
28                     title=f'fft noisy signal. b={b}, c={c}, d={d}',
29                     legend=False, fz1=12, fz2=6, xl1=-10, xl2=10)
30
31 if filter_low:
32     flt_u, flt_U = ft.filter_low(freq, u, v_0)
33     bd.build_u__flt_u(time, u, flt_u,
34     title=rf'Low frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
35     fz1=12, fz2=6)
36     bd.build_abs_u_to_U__flt_U(freq, u, flt_U,
37     title=rf'Abs low frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0},
38     fz1=12, fz2=6, xl1=-25, xl2=25)
39
40 if filter_high:
41     flt_u_1, flt_U_1 = ft.filter_high(freq, u, v_0_1)
42     bd.build_u__flt_u(time, u, flt_u_1,
43     title=rf'High frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0_1},
44     fz1=12, fz2=6)
45     bd.build_abs_u_to_U__flt_U(freq, u, flt_U_1,
46     title=rf'Abs high frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0_1},
47     fz1=12, fz2=6, xl1=-25, xl2=25)
48
49 if filter_special_out:
50     flt_u_0, flt_U_0 = ft.filter_special_out(freq, u, v_0_2)
51     bd.build_u__flt_u(time, u, flt_u_0,
52     title=rf'Special frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0_2},
53     fz1=12, fz2=6)
54     bd.build_abs_u_to_U__flt_U(freq, u, flt_U_0,
55     title=rf'Abs special frequency filter. b={b}, c={c}, d={d}, $\nu_0$={v_0_2},
56     fz1=12, fz2=6, xl1=-10, xl2=10)
57
58 if filter_low and filter_special_out:
59     flt_u_2, flt_U_2 = ft.filter_special_out(freq, flt_u, v_0_2)
60     bd.build_u__flt_u(time, u, flt_u_2,
61     title=rf'Low and special frequency filter. b={b}, c={c}, d={d}, (1)
62     $\nu_0$={v_0}, (2) $\nu_0$={v_0_2}, fz1=12, fz2=6)
63     bd.build_abs_u_to_U__flt_U(freq, u, flt_U_2,
64     title=rf'Abs low and special frequency filter. b={b}, c={c}, d={d},
65     (1) $\nu_0$={v_0}, (2) $\nu_0$={v_0_2}, fz1=12, fz2=6,
66     xl1=-25, xl2=25)
67
68 if filter_high and filter_special_out:
69     flt_u_2, flt_U_2 = ft.filter_special_out(freq, flt_u_1, v_0_2)
70     bd.build_u__flt_u(time, u, flt_u_2,
71     title=rf'High and special frequency filter. b={b}, c={c}, d={d}, (1)
72     $\nu_0$={v_0_1}, (2) $\nu_0$={v_0_2}, fz1=12, fz2=6)

```

```

67     bd.build_abs_u_to_U__flt_U(freq, u, flt_U_2,
68         title=rf'Abs high and special frequency filter. b={b}, c={c}, d={d},
69             (1) $\nu_0$={v_0_1}, (2) $\nu_0$={v_0_2}', fz1=12, fz2=6,
69             xl1=-25, xl2=25)

```

Листинг 7: Файл nospec.py. Фильтрация специфических частот.

```

1      filter_all = True
2      build_U = True
3
4      title = 'High frequency filter audio'
5      title2 = 'Abs high frequency filter audio'
6      if filter_all:
7          title = 'High and special frequency filter audio'
8          title2 = 'Abs high and special frequency filter audio'
9
10     src = 'sound/MUHA.wav'
11     filename = 'sound/filtered_MUHA_2.wav'
12     try:
13         audio, rate = librosa.load(src, sr=None)
14     except:
15         lab = 'fm_lab3/'
16         src = lab + src
17         filename = lab + filename
18         audio, rate = librosa.load(src, sr=None)
19
20     dt = 1 / rate
21     T = len(audio) * dt
22
23     time = np.linspace(0, T, len(audio), endpoint=False)
24     freq = np.linspace(-rate / 2, rate / 2, len(audio), endpoint=False)
25
26     flt_u, flt_U = ft.filter_high(freq, audio, 300)
27     if filter_all:
28         flt_u, flt_U = ft.filter_special_out(freq, flt_u,
29                                             [[freq[0], -1000], [1000, freq[-1]]])
30         flt_u_float = flt_u.real.astype(np.float32)
31
32
33     if build_U:
34         bd.build_u_or_U(time, audio, xlab='Time',
35                         title='Noisy audio signal', fz1=12, fz2=6,
36                         legend=False)
37         bd.build_u_to_U(freq, audio, title='fft noisy audio signal',
38                         legend=False, fz1=12, fz2=6,
39                         xl1=-875, xl2=875, yl1=-4000,
40                         yl2=4000)
41
42         bd.build_u__flt_u(time, audio, flt_u,
43                             title=title, fz1=12, fz2=6)
44         bd.build_abs_u_to_U__flt_U(freq, audio, flt_U,
45                                     title=title2, xl1=-1500, xl2=1500,
46                                     yl1=0, yl2=2000, fz1=12, fz2=6)
47
48     write(filename, rate, flt_u_float)
49     playsound(filename)

```

Листинг 8: Файл audio.py. Фильтрация шумов в аудиозаписи.