МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Университет ИТМО

Факультет систем управления и робототехники

ОТЧЁТ
по дисциплине
”Частотные методы”

по теме:
ЖЁСТКАЯ ФИЛЬТРАЦИЯ

Студент:
Группа R3236 *Поляков А.А.*

Предподаватель:
к.т.н., доцент *Перегудин А.А.*

Санкт-Петербург
2024

СОДЕРЖАНИЕ

1 ВСПОМОГАТЕЛЬНЫЕ ФУНКЦИИ И БИБЛИОТЕКИ

Онлайн версию кода здесь нет, потому что делал основные вычисления в live-script матлабовских, в репозитории можно найти исходники. Так как в большинстве своём выполнение лабораторной работы сводилось к применению встроенных функций и построению красивых графиков, то не вижу смысла приводить отдельные фрагменты кода, потому что проще увидеть всю картину целиком - заглянуть в исходники, над их оформлением я тоже немного постарался.

2 ЗАДАНИЕ 1. ЖЁСТКИЕ ФИЛЬТРЫ

Для начала задаём константы a, b, c, d, t_1, t_2, что $t_1 < t_2$, После составляем функцию:

$$g(t) = \begin{cases} a, t \in [t_1; t_2] \\ 0, t \in [else] \end{cases}$$

Также задаём большой интервал времени T и маленький шаг дискретизации dt. На основе всего зашумлённая версия сигнала будет выглядеть так:

```
u = g + b*(rand(size(t))-0.5) + c*sin(d*t);
```

2.1 Убираем высокие частоты

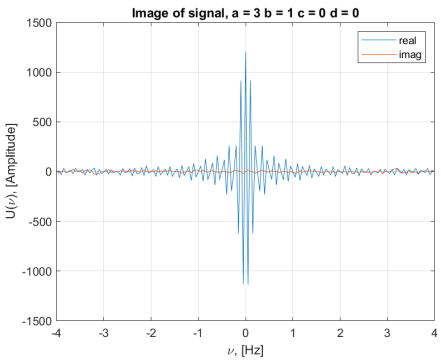
Берём $d = c = 0$. Тогда в этом пункте мы будем работать со следующей версией шумного сигнала:

```
u = g + b*(rand(size(t))-0.5)
```

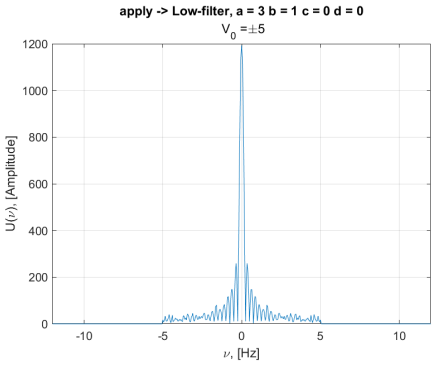
…из чего сразу следует, что у нас добавляется только ”случайный”шум.

В дальнейший прогонах фильтра мы будем каждое испытание показывать вместе, в таком порядке: сначала ищем образ Фурье зашумленного сигнала, после фильтруем лишнее, и делаем обратное преобразование Фурье и сравниваем с оригиналом.

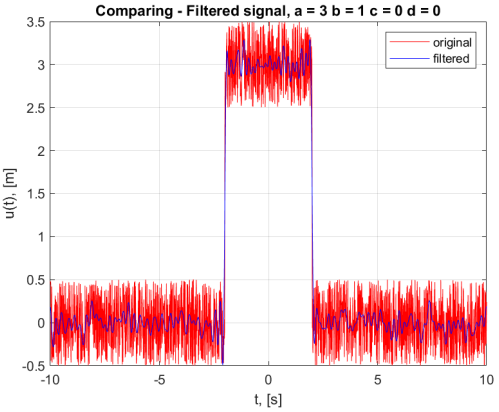
2.1.1 Испытание 1



(а)



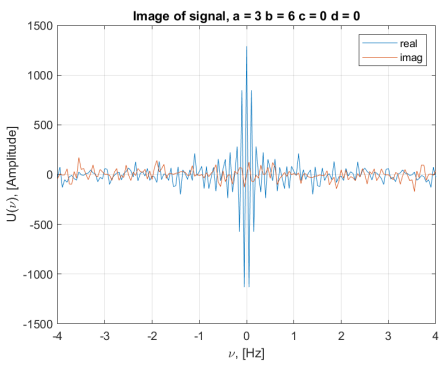
(б)



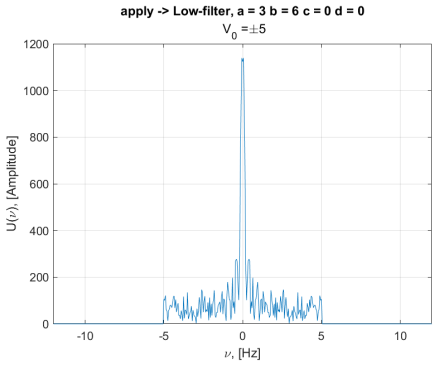
(в)

Рисунок 1 — Испытание 1

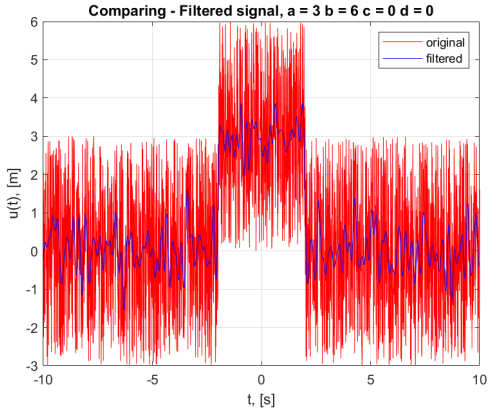
2.1.2 Испытание 2



(а)



(б)



(в)

Рисунок 2 — Испытание 2

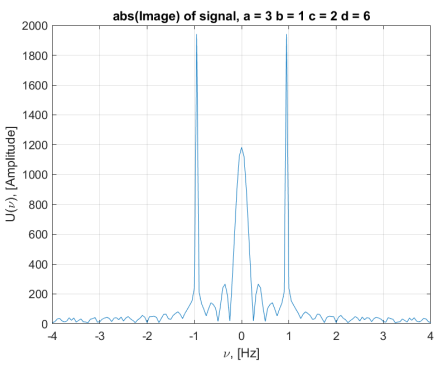
2.1.3 Выводы

При увеличении b мы добавляем больше сбвига амплитуд в исходную синусоиду, что усложняет дальнешую фильтрацию и возвращение к исходной функции, поэтому при фильтрации с бОльшой b очищенная функция была ”неуверена”в себе. Также очевидно, при большем диапазоне фильтра мы оставляем больше частот, а значит и отфильтрованная функция будет больше колебаться, т.е. опять ”неуверена”в себе. Поэтому оптимальный случай для фильтрации таким фильтром - небольшой b и небольшой срез.

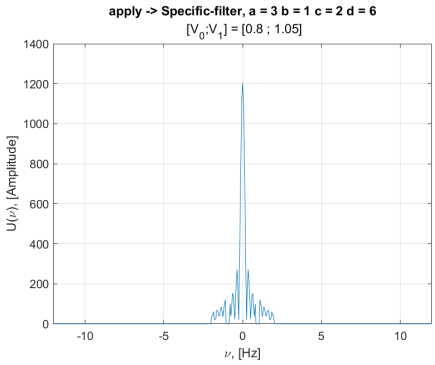
2.2 Убираем специфические частоты

Выберем все параметры b, c, d ненулевыми. Теперь мы уже имеем дело с двумя компонентами шума - случайным и гармоническим. Соответственно, чтобы убрать обе компоненты, надо применить два фильтра, один из них мы уже нашли в пункте до(специфический+низкий).

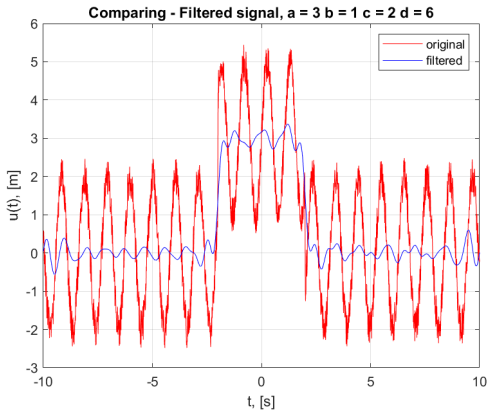
2.2.1 Испытание 1



(а)



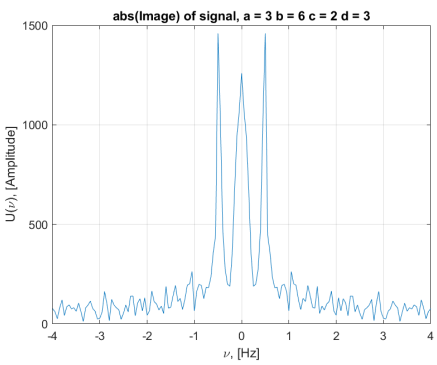
(б)



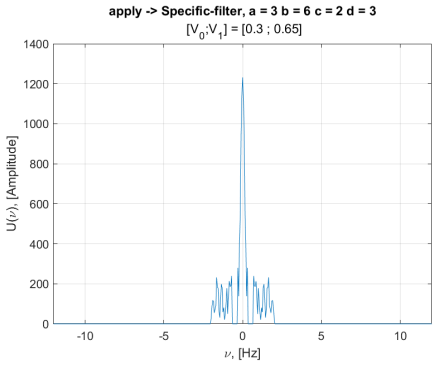
(в)

Рисунок 3 — Испытание 1

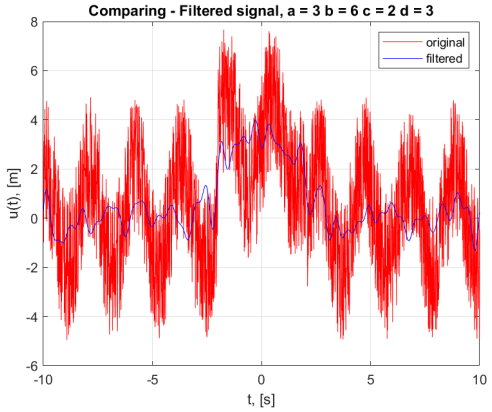
2.2.2 Испытание 2



(а)



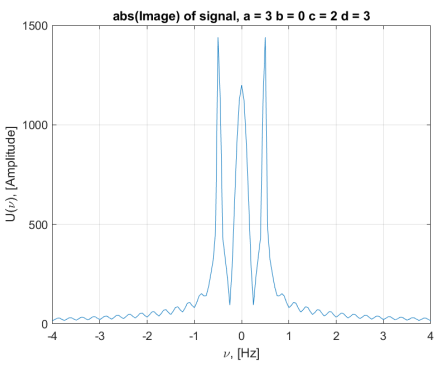
(б)



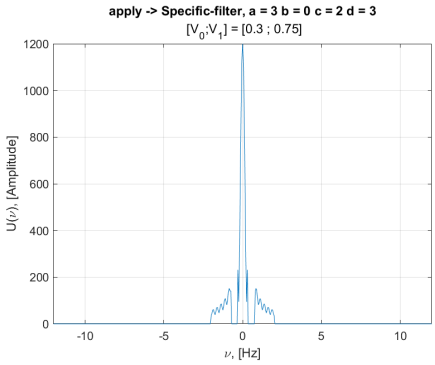
(в)

Рисунок 4 — Испытание 2

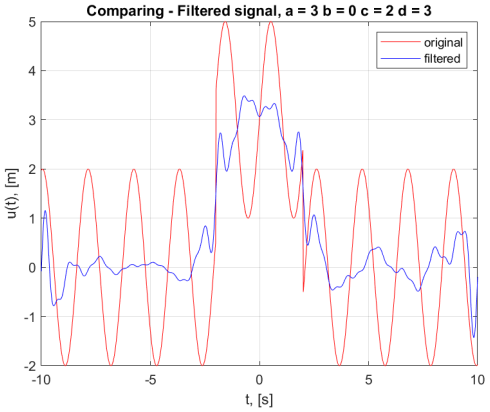
2.2.3 Испытание 3



(а)



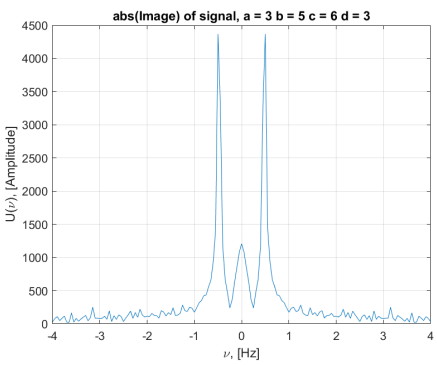
(б)



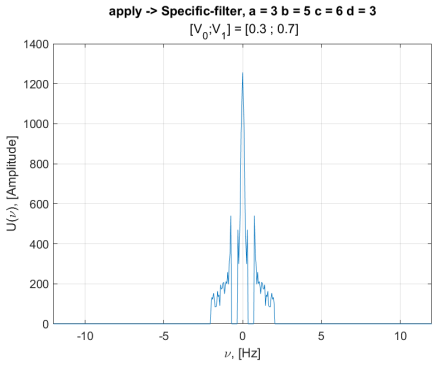
(в)

Рисунок 5 — Испытание 3

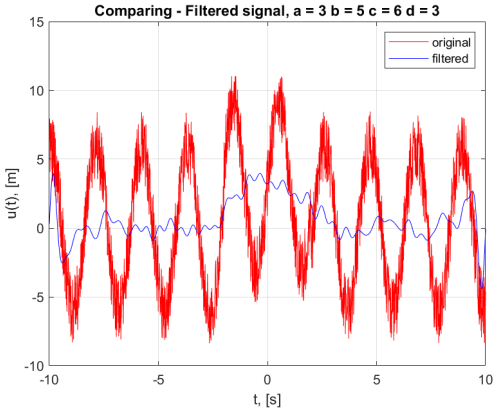
2.2.4 Испытание 4



(а)



(б)



(в)

Рисунок 6 — Испытание 4

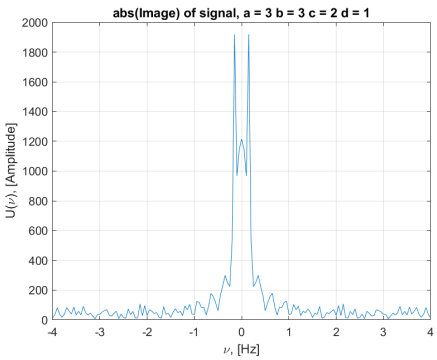
2.2.5 Выводы

Из формулы зашумленного сигнала понятно, что компонента b отвечает за случайный шум, а c и d - за гармонический, при этом c - амплитуда, а d - частота. Чем меньше шума было внесено в функцию, тем лучше будет получаться его убрать. Но даже при довольно больших значениях параметров b, c, d удалось убрать шум, не сильно исказив исходную функцию. И так получилось, что только *не* срезав достаточно низких частот, мы хорошо приближаемся к исходной гладкой функции.

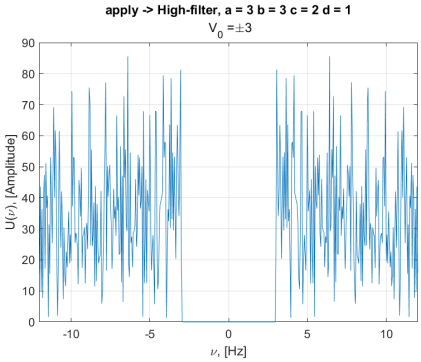
2.3 Убираем низкие частоты?

Рассмотрим фильтр, который обнуляет Фурье-образ на всех частотах в некоторой окрестности точки $\nu = 0$. Пропустим сигнал через такой фильтр и оценим результат.

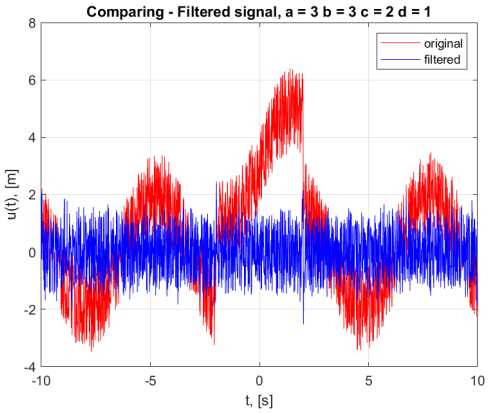
2.3.1 Испытание 1



(а)



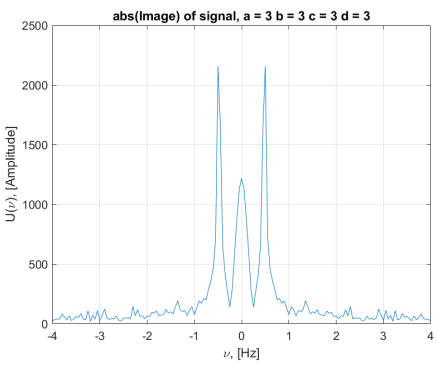
(б)



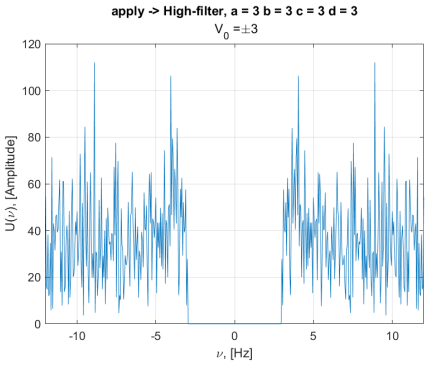
(в)

Рисунок 7 — Испытание 1

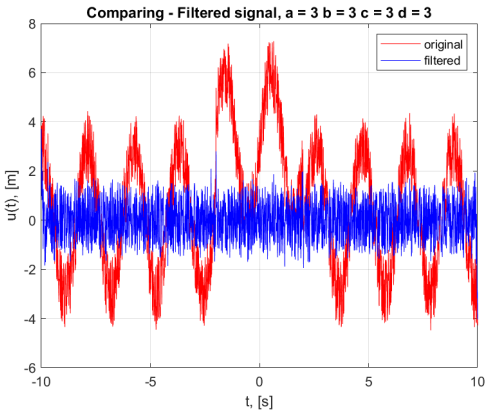
2.3.2 Испытание 2



(а)



(б)



(в)

Рисунок 8 — Испытание 2

2.3.3 Выводы

Видно, что от исходной функции остался только шум, и это логично, ведь мы убрали низкие частоты, которые, в большинстве, и соответствовали самой функции. Амплитуда шума как будто осталась неизменной, что ещё раз подтверждает нашу гипотезу. Однако, если взять слишком малый срез, то функция почти будет подобать оригинальной(второй случай).

3 ЗАДАНИЕ 2. ФИЛЬТРАЦИЯ ЗВУКА

Скачаем файл с гугл-диска, а после немного погуглим и выясним, что *голос человека лежит в диапазоне 85-3000Гц*, поэтому постараемся чистить всё, кроме этого промежутка. Очевидно, что нам понадобится два последовательно применённых фильтра, так и сделаем…

Уже после написания кода при прослушивании стало ясно, что при таком диапазоне особо шумы не пропали, поэтому я стал играться с нижней границей, и не прогадал - начиная ”басистые”шумы ушли, но если продолжить увеличивать нижнюю границу, то уже заметно будет страдать сам звук.

3.1 Графики звуковых сигналов

В итоге мы получим следующие графики(P.S. не забываем, что весь код в гитхабе лежит, но его там совсем немного, ведь это матлаб :D)

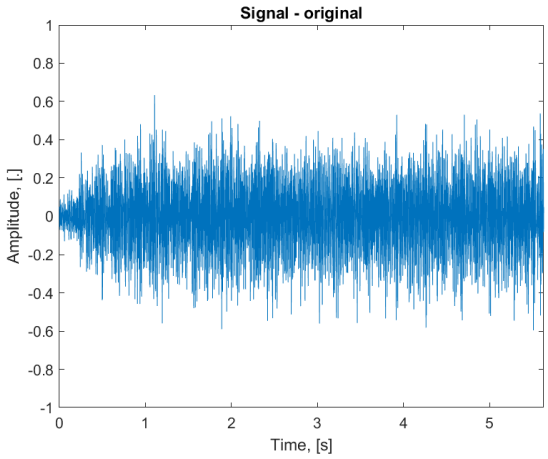


Рисунок 9 — Изначальный график сигнала

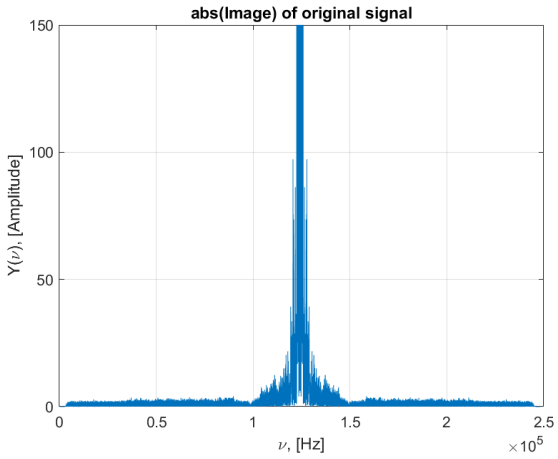


Рисунок 10 — Изначальный Фурье-образ сигнала

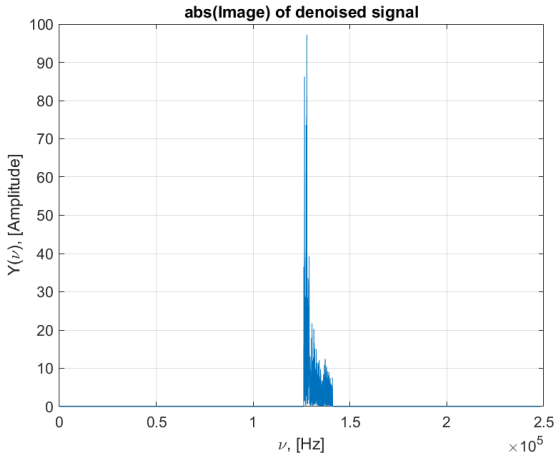


Рисунок 11 — Применили два фильтра - изменение Фурье-образа

Не забывайте, что с помощью функции матлаба `sound(y,f)` - можно нативно проиграть звук после преобразований, чтобы сравнить ”до”и ”после”.

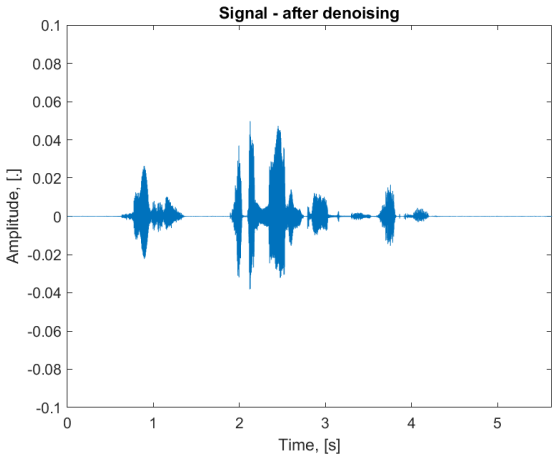


Рисунок 12 — Очищенный вид сигнала