Санкт-Петербургский Государственный Политехнический Университет
Институт Компьютерных Наук и Технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчёт по лабораторной работе №10
на тему
Линейный стационарные системы

Работу выполнил
Студент группы 3530901/80203
Курняков П.М.
Преподаватель
Богач Н.В.

Санкт-Петербург, 2021 год

1 Настройка проекта

Перед тем как выполнять задания необходимо настроить проект и сделать все необходимые импорты:

```
from __future__ import print_function, division

import thinkdsp
import thinkplot

import numpy as np
import pandas as pd
import scipy.signal
import warnings
warnings.filterwarnings('ignore')

PI2 = 2 * np.pi

np.set_printoptions(precision=3, suppress=True)
%matplotlib inline
```

Рис. 1: 2

2 Упражнение номер №1

Убедиться что дополнение нулями устраняет лишнюю ноту в начале фрагмента

Я усекаю оба сигнала до 2^16 элементов, а затем обнуляю их до 2^17. Использование степени двойки делает алгоритм DFT наиболее эффективным.

Вот импульсный отклик:

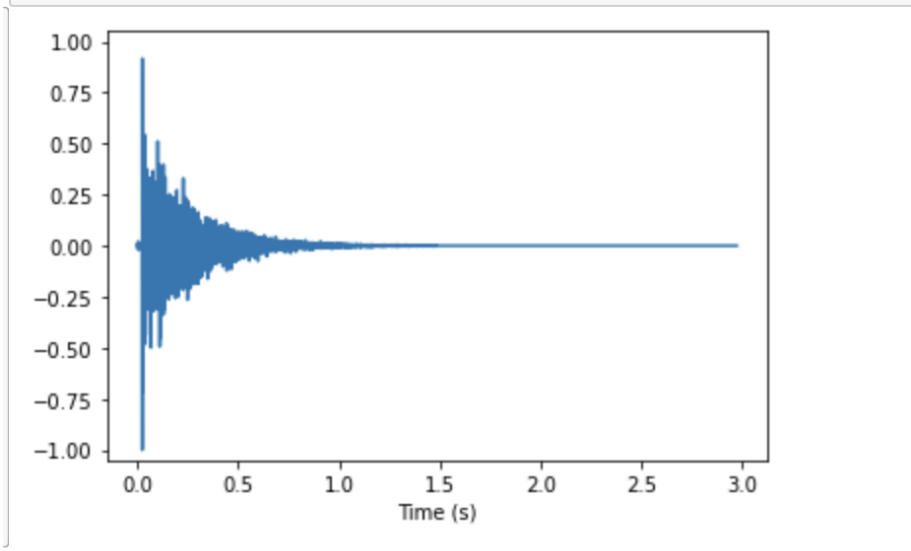


Рис. 2: 2

Выведем спектр:

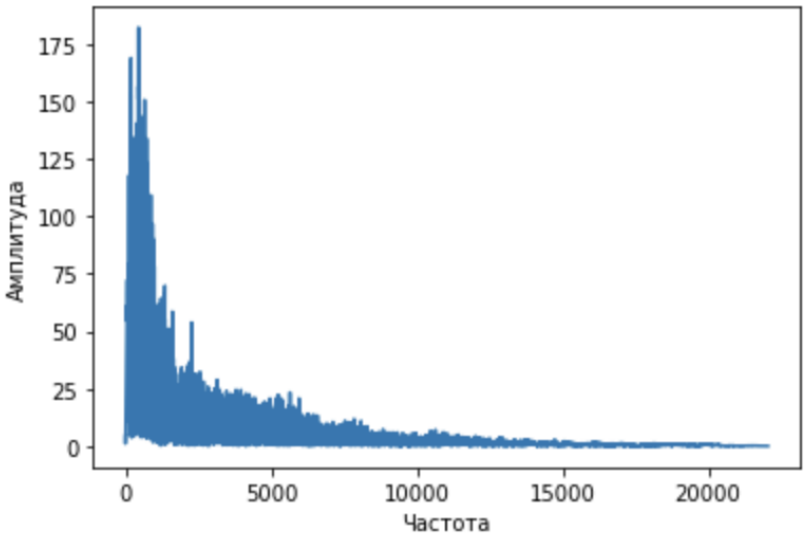


Рис. 3: 2

Рассмотрим еще один сигнал:

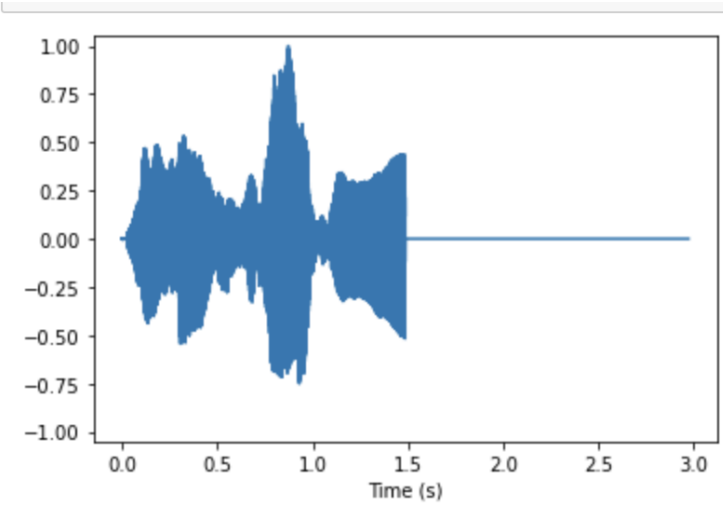


Рис. 4: 2

Теперь умножим DFT сигнала на передаточную функцию и преобразуем обратно в волну:

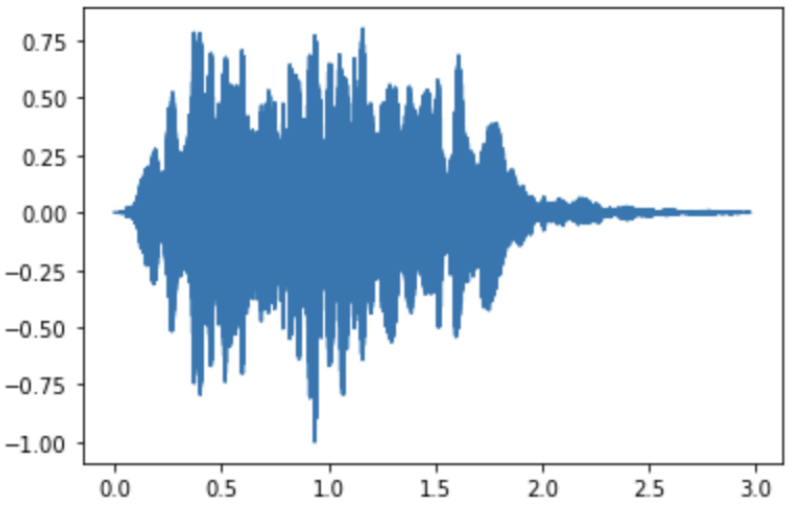
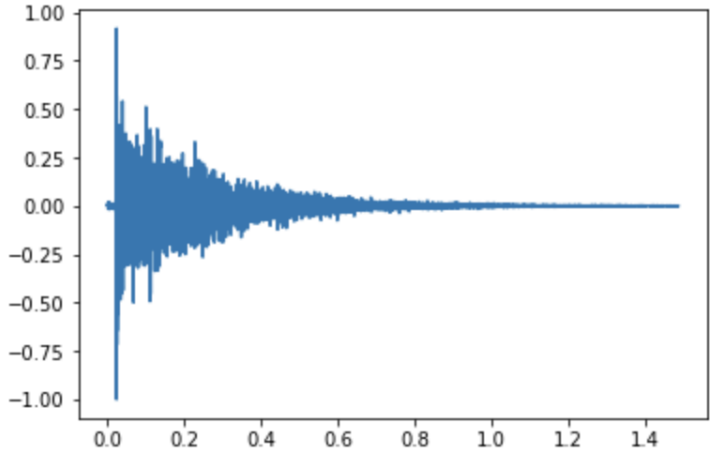


Рис. 5: 2

Результат не выглядит так, как будто он оборачивается вокруг:
Мы должны получить такие же результаты от np.convolve и scipy.signal.fftconvolve.
Обрежем нулевой отступ:

2]:
```
response.truncate(2**16)
response.plot()
```



3]:
```
violin.truncate(2**16)
violin.plot()
```

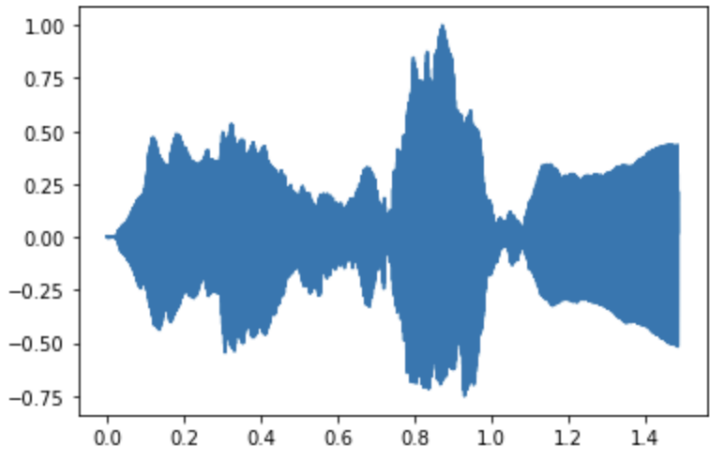


Рис. 6: 2

Сравним с np.convolve:

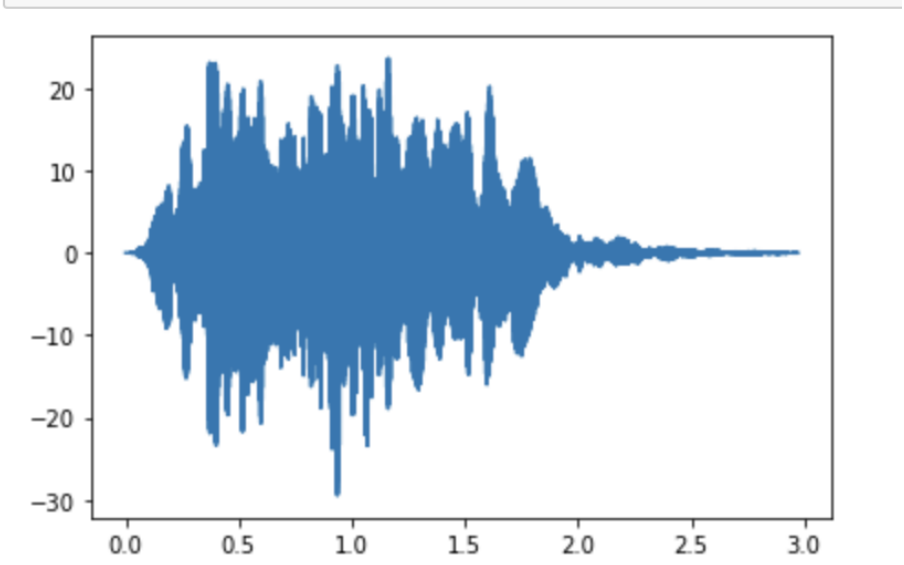


Рис. 7: 2

Как мы можем наблюдать результаты очень похожи.

```
6]: len(output), len(output2)

6]: (131072, 131071)
```

Рис. 8: 2

Но результаты не совсем одинаковой длины.

scipy.signal.fftconvolve делает то же самое, но, как следует из названия, он использует DFT, поэтому он значительно быстрее:

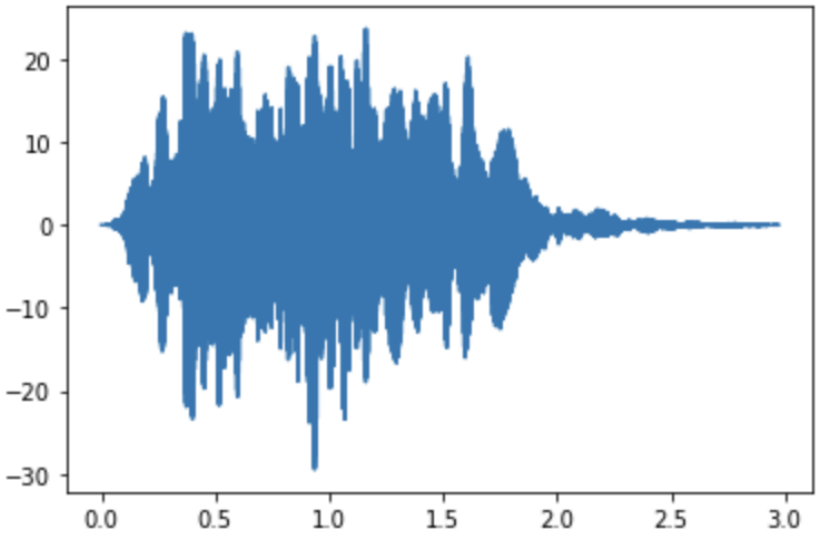


Рис. 9: 2

Результат такой же.

3 Упражнение номер №2

Необходимо смоделировать двумя способами в том пространстве где была измерена импульсная характеристика.

Возьмем звук из репозитория:

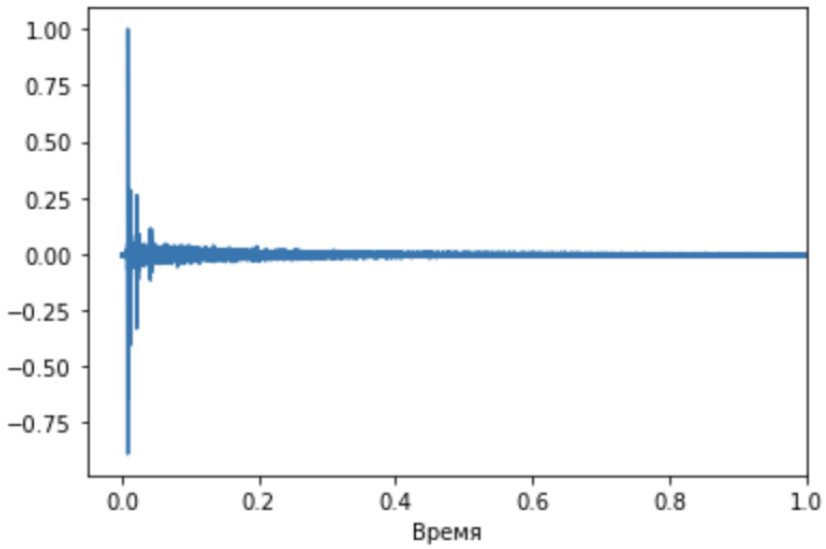


Рис. 10: 2

DFT импульсной характеристики - это передаточная функция:

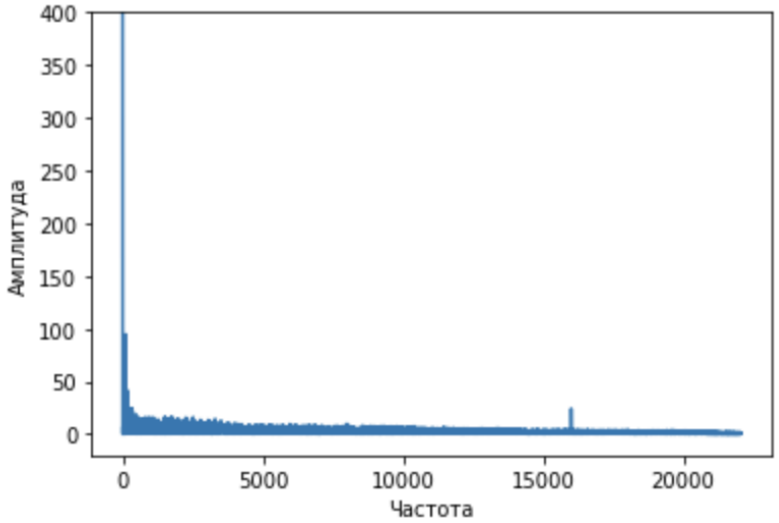


Рис. 11: 2

Рассмотрим график в логарифмической метрики:

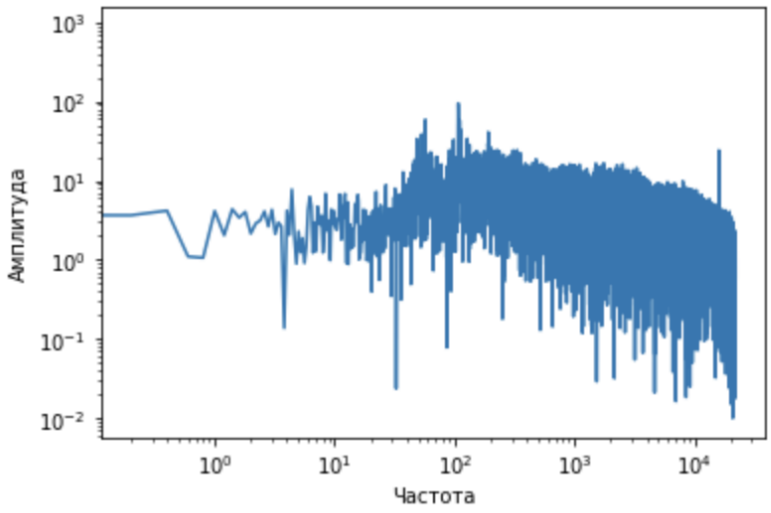


Рис. 12: 2

Теперь мы можем смоделировать звучание записи, если бы ее воспроизвели в одной комнате и записали бы таким же образом.

Возьмем еще одну запись из репозитория:

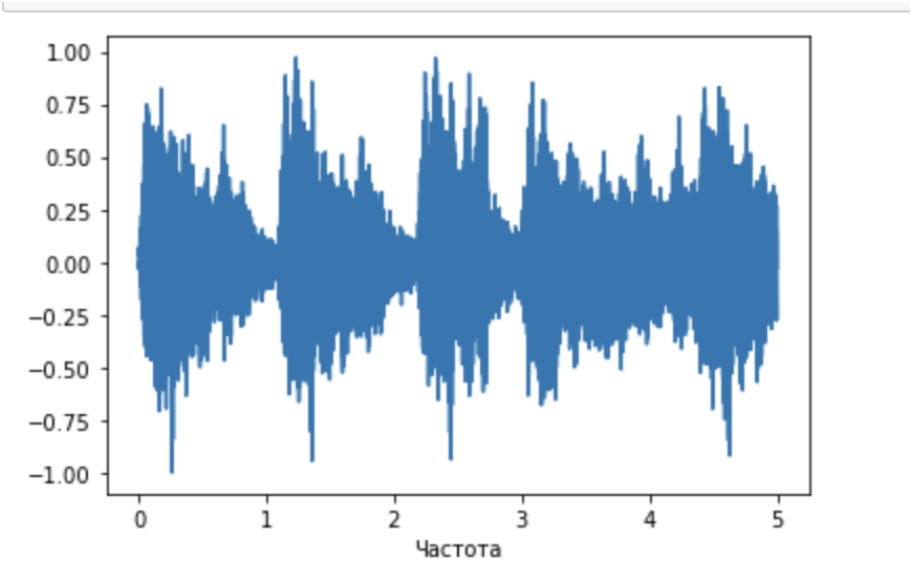


Рис. 13: 2

Теперь мы вычисляем DFT записи скрипки. Обрежем запись скрипки до той же длины, что и импульсная характеристика:

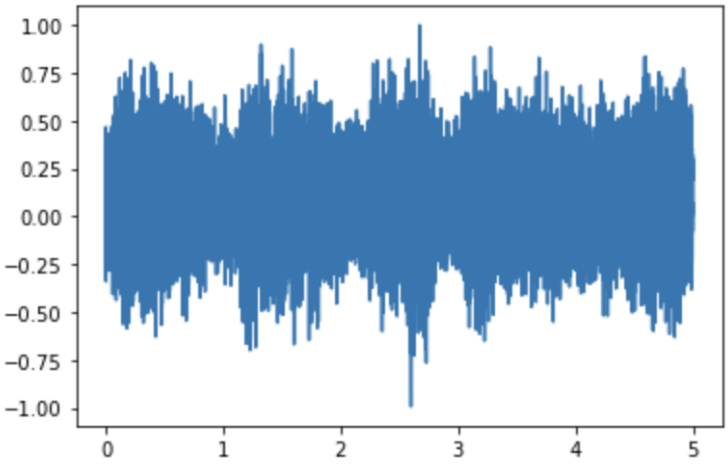
```
]: len(spectrum.hs), len(transfer.hs)
]: (110251, 110251)

]: spectrum.fs
]: array([    0. ,     0.2,     0.4, ..., 22049.6, 22049.8, 22050. ])

]: transfer.fs
]: array([    0. ,     0.2,     0.4, ..., 22049.6, 22049.8, 22050. ])
```

Рис. 14: 2

Теперь мы можем умножить в частотной области и преобразовать обратно во временную область. Проведем сравнение:




```
wave.plot()
```

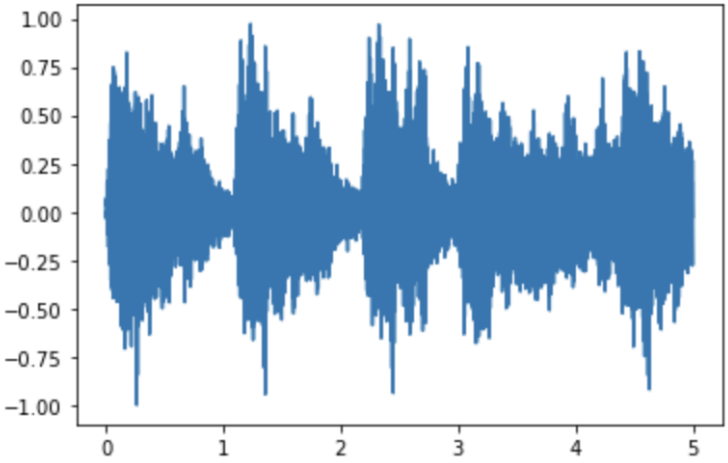


Рис. 15: 2

Теперь, когда мы распознаем эту операцию как свертку, мы можем вычислить ее с помощью метода convolve