Санкт-Петербургский государственный политехнический
университет Петра Великого

**Высшая школа интеллектуальных систем и
суперкомпьютерных технологий**

Лабораторная работа

Линейные стационарные системы

Работу выполнил студент
3-го курса, группа 3530901/80201
Солянкин Илья Андреевич

Преподаватель:
Богач Наталья Владимировна

Санкт-Петербург 2021

Содержание

Список иллюстраций

Листинги

1 Часть №1: chap10.ipynb

В первой части десятой лабораторной работы нам необходимо просмотреть весь блокнот chap10.ipynb, после чего заменить его, чтобы устранить лишнюю ноту в начале фрагмента.

Возмем исходный сигнал и выведем его на экран:

```
response = read_wave('180960__kleeb__gunshot.wav')
start = 0.12
response = response.segment(start=start)
response.shift(-start)

response.truncate(2**16)
response.zero_pad(2**17)

response.normalize()
response.plot()
decorate(xlabel='Time (s)')
```

Листинг 1: Получение сигнала

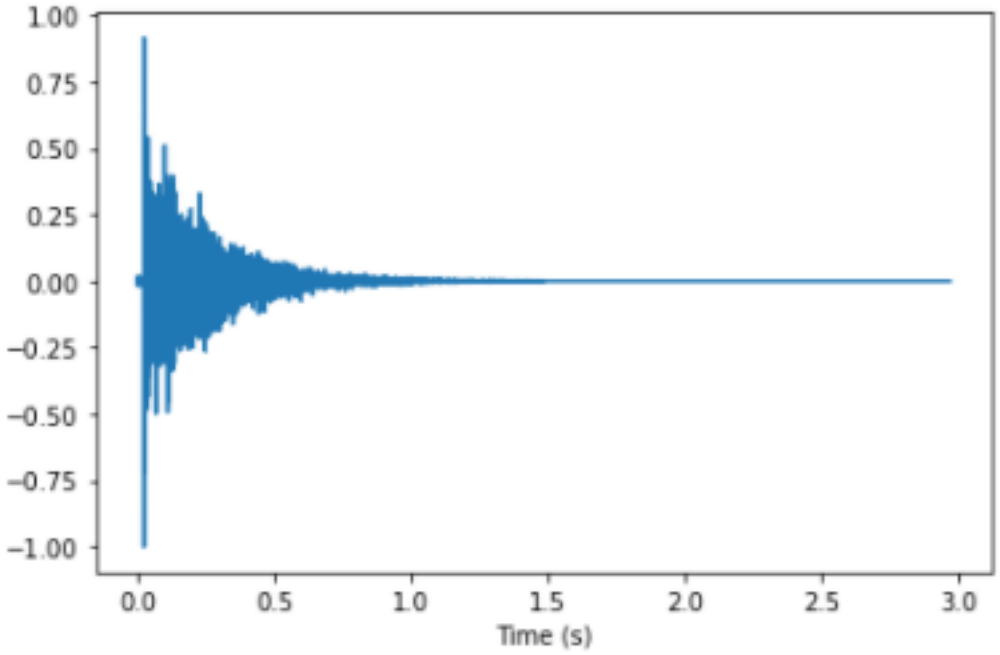

Рис. 1: Начальный сигнал

После этого получим спектр данного сигнала:

```
transfer = response.make_spectrum()
transfer.plot()
decorate(xlabel='Frequency (Hz)', ylabel='Amplitude')
```

Листинг 2: Получение спектра первого сигнала

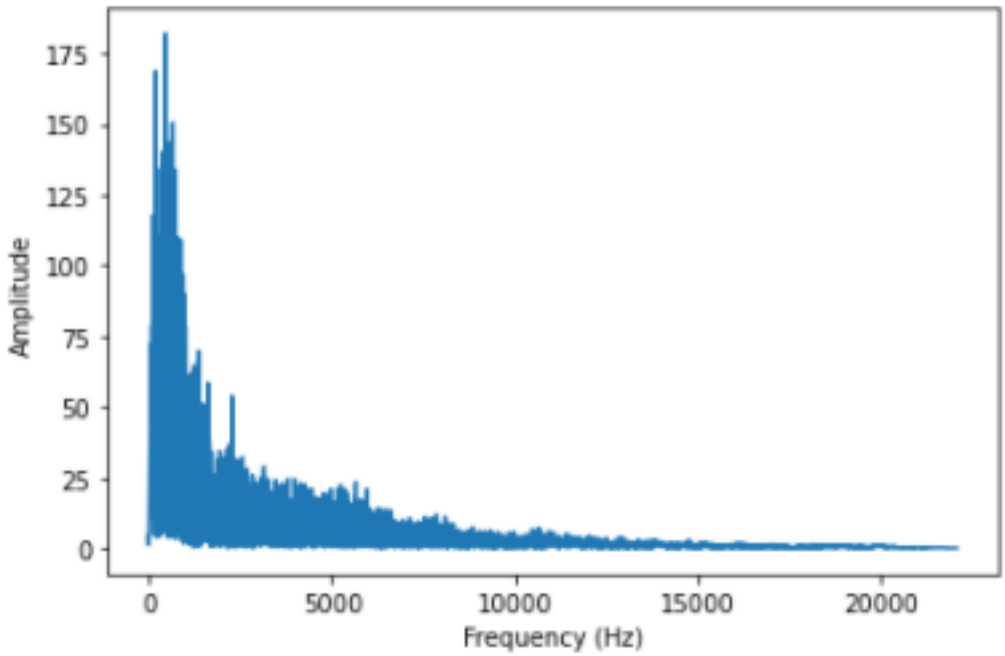

Рис. 2: Спектр первого сигнала

Теперь возьмем второй сигнал со скрипкой:

```
violin = read_wave('92002__jcveliz__violin-origional.wav')

start = 0.11
violin = violin.segment(start=start)
violin.shift(-start)

violin.truncate(2**16)
violin.zero_pad(2**17)

violin.normalize()
violin.plot()
decorate(xlabel='Time (s)')
```

Листинг 3: Получение второго сигнала

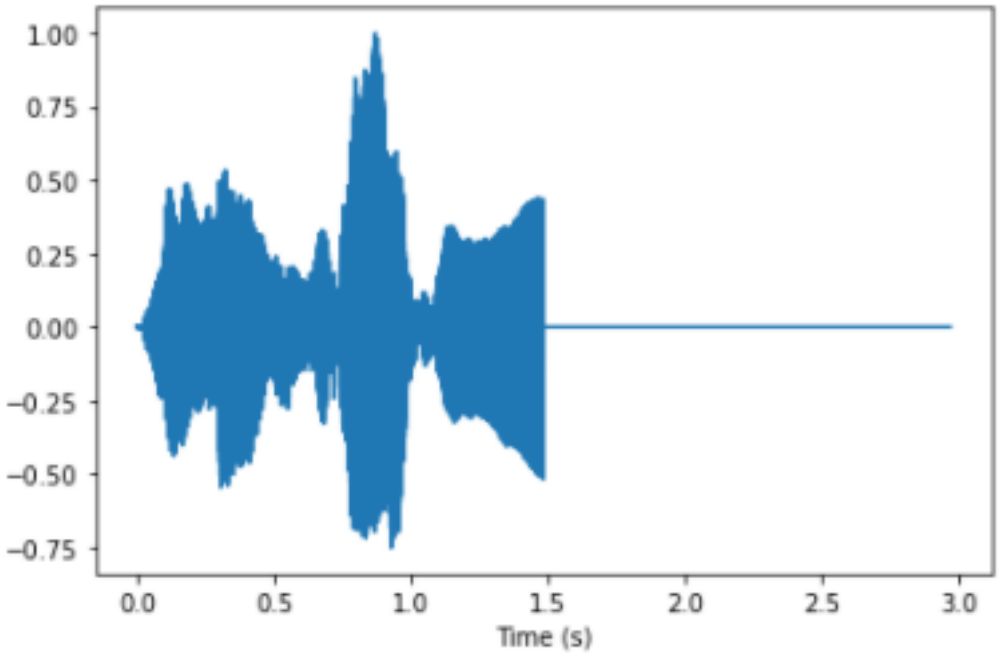

Рис. 3: Начальный второй спектр

И так же получим для его спектр:

```
spectrum = violin.make_spectrum()
spectrum.plot()
```

Листинг 4: Получение спектра второго сигнала

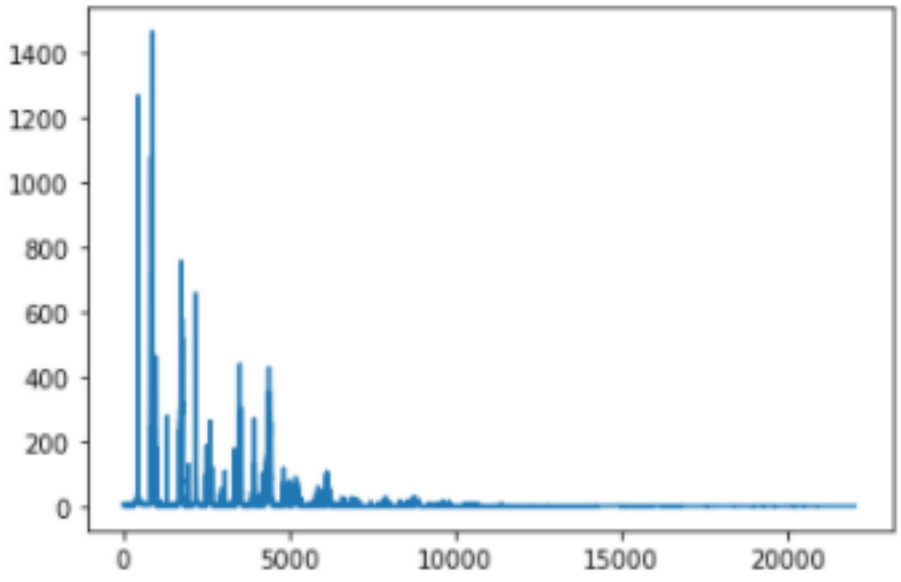

Рис. 4: Спектр второго сигнала

После этого усножим ДПФ сигнала на передаточную функцию, после чего преобразуем обратно в волну и выведем полученный сигнал на экран:

```
    output = (spectrum * transfer).make_wave()
    output.normalize()
    output.plot()
```

Листинг 5: Преобразование сигнала

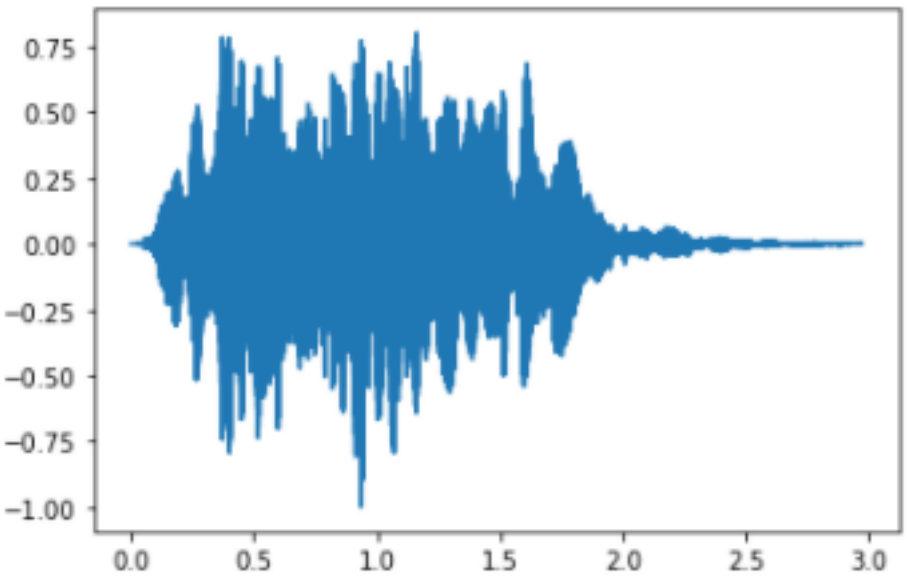

Рис. 5: Полученный сигнал

Нам осталось представить полученный сигнал в виде аудио и сравнить его с изначальным:

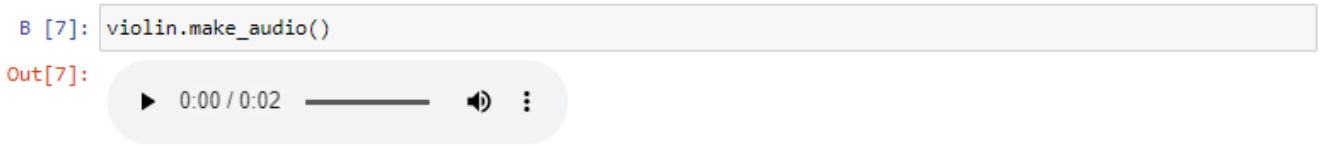

Рис. 6: Изначальный сигнал

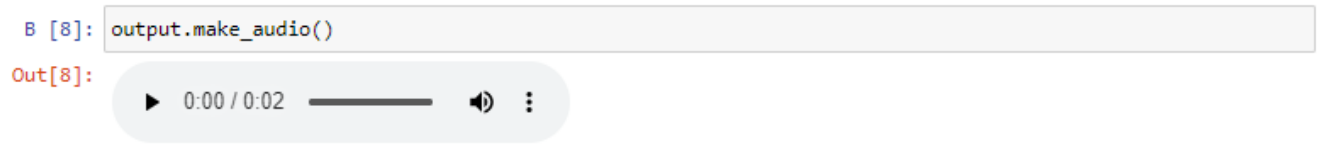

Рис. 7: Полученный сигнал

В ходе сравнения стало понятно, что исчезла нота, стоящая в начале фрагмента.

2 Часть №2: Изменение звучания сигнала

Во втором пункте десятой лабораторной работы нам необходимо смоделировать двумя способами звучание записи в том пространстве, где была измерена импульсная характеристика, как сверткой самой записи с импульсивной характеристикой, так и умножением ДПФ записи на вычислительный фильтр.

Скачаем файл со звуком лопнувшего шарика и сразу выведем его на экран:

```
response = read_wave('balloon2.wav')

start = 0
duration = 1
response = response.segment(duration=duration)
response.shift(-start)

response.normalize()
response.plot()
decorate(xlabel='Time (s)')
```

Листинг 6: Получение сигнала

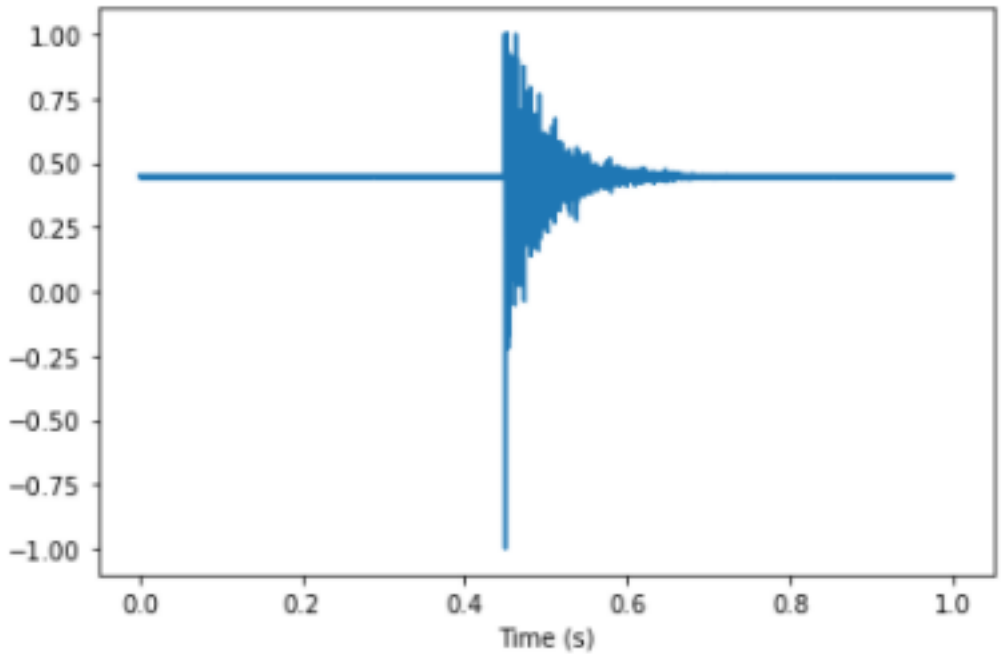

Рис. 8: Полученный сигнал

Теперь получим спектр данного сигнала:

```
transfer = response.make_spectrum()
transfer.plot()
decorate(xlabel='Frequency (Hz)', ylabel='Amplitude')
```

Листинг 7: Получение спектра сигнала

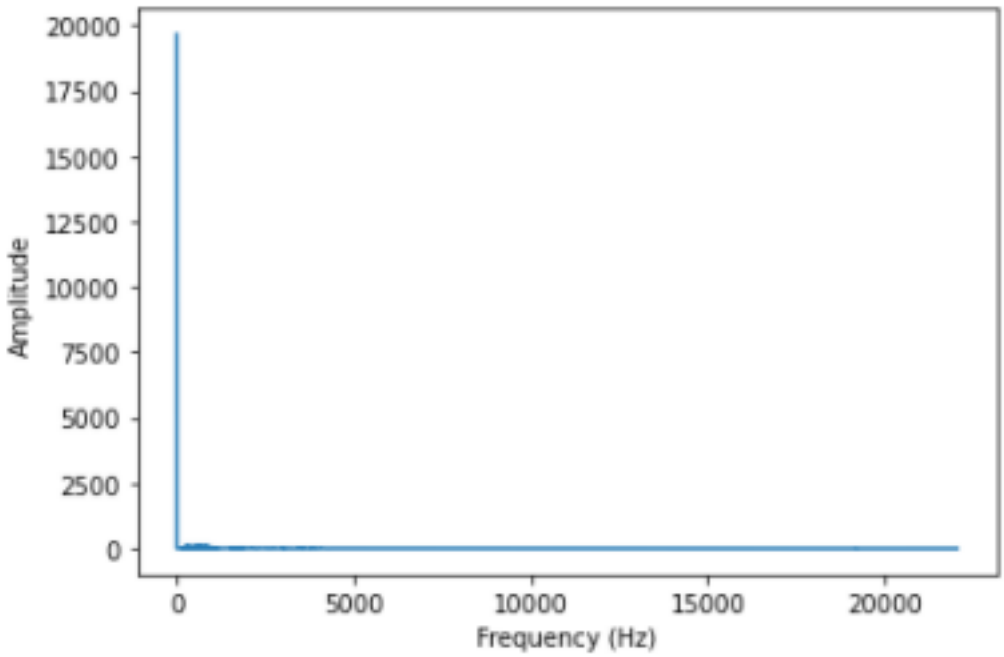

Рис. 9: Спектр сигнала

И, наконец, представим его в логарифмическом масштабе:

```
transfer.plot()
decorate(xlabel='Frequency (Hz)', ylabel='Amplitude', xscale='log', yscale='
log')
```

Листинг 8: Получение спектра сигнала в логарифмическом масштабе

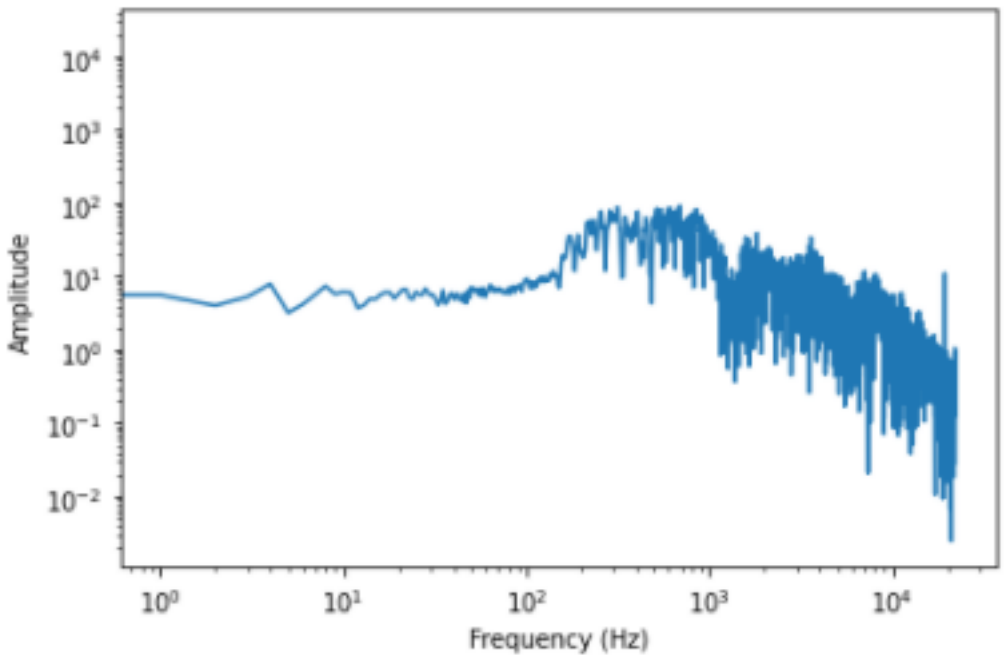

Рис. 10: Спектр сигнала в логарифмическом масштабе

Также можно смоделировать звучание записи, если она будет воспроизводиться в одной комнате.

Для этого вычислим ДПФ использованной в прошлом пункте записи скрипки:

```
    wave = read_wave('92002__jcveliz__violin-origional.wav')

    start = 0.0
    wave = wave.segment(start=start)
    wave.shift(-start)

    wave.truncate(len(response))
    wave.normalize()
    wave.plot()
    decorate(xlabel='Time (s)')
```

Листинг 9: Получение сигнала скрипки

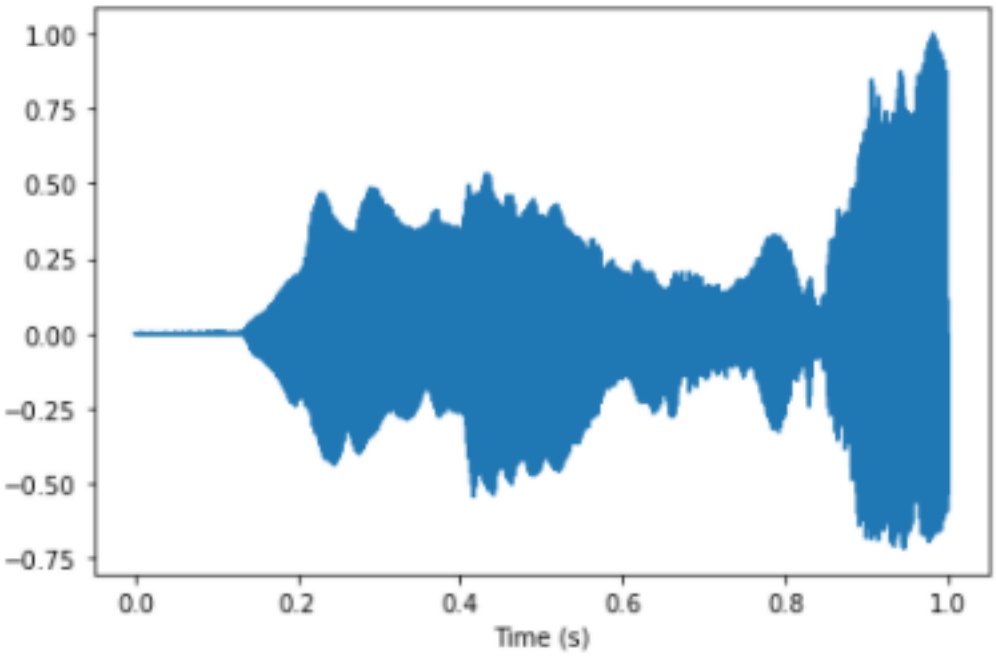

Рис. 11: Полученный сигнал скрипки

Теперь ображем запись скрипки до той же длины, что и испульсная характеристика:

```
In [16]: len(spectrum.hs), len(transfer.hs)
Out[16]: (22051, 22051)

In [17]: spectrum.fs
Out[17]: array([0.0000e+00, 1.0000e+00, 2.0000e+00, ..., 2.2048e+04, 2.2049e+04,
                2.2050e+04])

In [18]: transfer.fs
Out[18]: array([0.0000e+00, 1.0000e+00, 2.0000e+00, ..., 2.2048e+04, 2.2049e+04,
                2.2050e+04])
```

Рис. 12: Обрезание записи скрипки

После этого выполним умножение в частотной области и преобразуем обратно во временную область, после чего выведем полученный результат на экран:

```
output = (spectrum * transfer).make_wave()
output.normalize()
output.plot()
```

Листинг 10: Получение результата

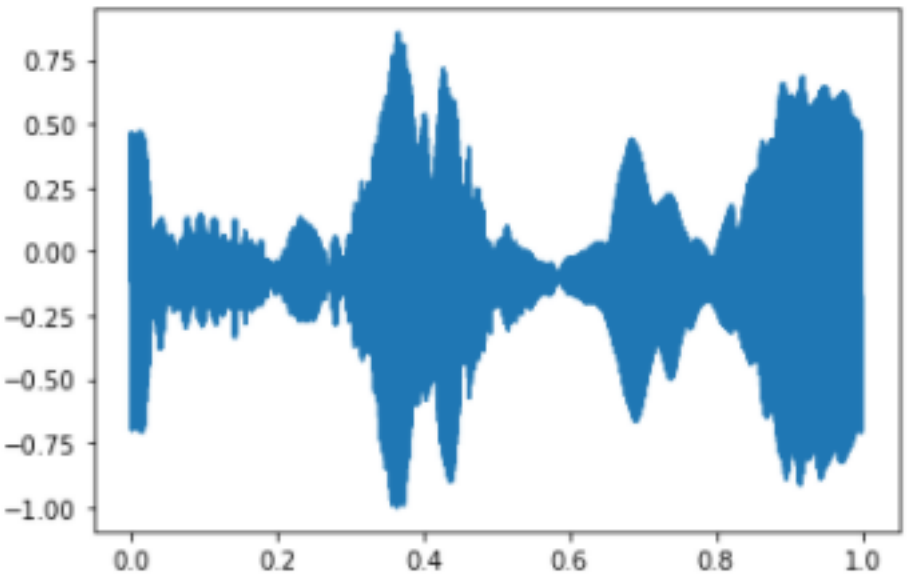

Рис. 13: Полученный результат

Теперь нам осталось только представить полученный сигнал в виде аудио, после чего воспользуемся методом `convolve` для свертки:

Рис. 14: Сравнение результата с иходной записью

В результате выполнения данного пункта мы получили чистый и довольно реалистичный звук игры на скрипке внутри помещения.

3 Выводы

В результате выполнения данной лабораторной работы нами были получены знания о свертке сигналов. Нами была модифицирована запись игры на скрипке, в которой была убрана первая нота. Кроме того мы смоделировали звучание записи в помещении.