Лабораторная работа № 10. Линейные стационарные системы.

3530901/80201, Шелаев Н. Р.

31 мая 2021 г.

Оглавление

1 Сигналы и системы **5**

2 Окна и фильтры **7**

3 Акустическая характеристика **9**

4 Системы и свертка **15**

5 Упражнения **19**
5.1 Задание 1 19
5.2 Задание 2 25

6 Вывод **30**

Список иллюстраций

1.1 Сигнал импульса 6
1.2 Его спектр 6

2.1 ДПФ двухэлементного скользящего среднего 7
2.2 Импульсная реакция 8

3.1 Сегмент записи выстрела 9
3.2 Спектр звука выстрела 10
3.3 Спектр в логарифмическом масштабе 11
3.4 Сегмент звука скрипки 12
3.5 Спектр этого звука 12
3.6 Оригинальный звук 13
3.7 Преобразованная запись 13
3.8 Спектр преобразованного звука 14

4.1 Два выстрела 15
4.2 Теперь это уже не похоже на звук выстрела 16
4.3 Произвольный сигнал 17
4.4 Чем-то похоже на звук двух последовательных выстрелов . 17
4.5 Сравнение спектра до и после свертки 18
4.6 Результат свертки 18

5.1 Исправленный звук выстрела 20
5.2 И его спектр 20
5.3 Исправленный сигнал скрипки 21
5.4 Умножение на передаточную функцию 22
5.5 Звук выстрела без нулей 23
5.6 Сигнал скрипки тоже без нулей 23
5.7 Результат свертки 24
5.8 Результат свертки с помощью другой функции 25
5.9 Сигнал импульса 26
5.10 Спектр импульса 26

5.11 Спектр импульса в логарифмическом масштабе 27
5.12 Сигнал записи игры на трубе 28
5.13 Оригинальная запись 28
5.14 Изменённая запись сигнала 29

Листинги

1.1 Строим импульс 5
1.2 Исследуем полученный импульс 5
2.1 Найдем двухэлементное скользящее среднее 7
2.2 Умножение передаточной функции на спектр импульса . . 7
3.1 Звук выстрела 9
3.2 Строим спектр звука выстрела 10
3.3 Звук скрипки 11
3.4 Вычисление выходного сигнала 12
4.1 Создадим звук двух последовательных выстрелов 15
4.2 Очень много последовательных выстрелов 16
4.3 Произвольный входной сигнал 16
4.4 Генерация сдвинутых версий импульсной характеристики . 17
4.5 Используем метод свертки 18
5.1 Сначала возьмём звук выстрела 19
5.2 Теперь попробуем с сигналом скрипки 20
5.3 Перемножаем спектры 21
5.4 Убираем нули в конце сигналов 22
5.5 Свертка сигнала скрипки 23
5.6 Свертка сигнала скрипки с помощью FFT 24
5.7 Сначала попробуем какой-то звук 25
5.8 Запись игры на трубе 27
5.9 Перемножаем спектры для этого сигнала 28

Глава 1

Сигналы и системы

Линейная стационарная система - это система, которая линейна и неизменна во времени.

```
from thinkdsp import Wave

impulse = np.zeros(8)
impulse[0] = 1
wave = Wave(impulse, framerate = 8)
impulse_spectrum = wave.make_spectrum(full=True)

```

Листинг 1.1: Строим импульс

```
impulse = np.zeros(10000)
impulse[0] = 1
wave = Wave(impulse, framerate = 10000)
wave.plot()
wave.make_spectrum().plot()

```

Листинг 1.2: Исследуем полученный импульс

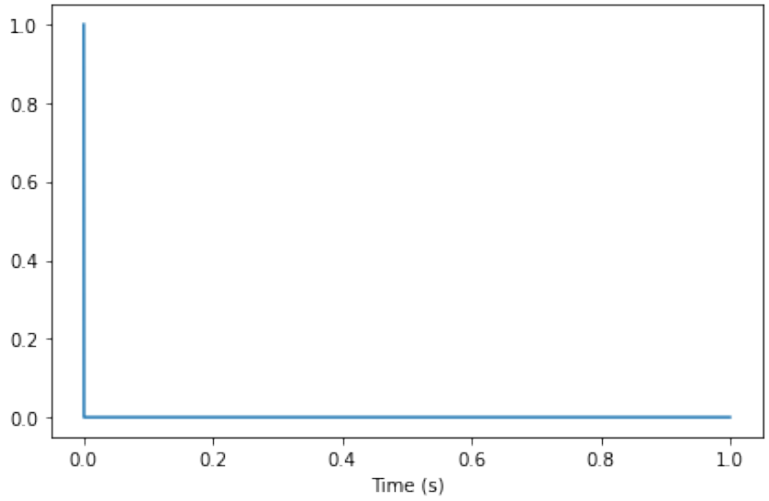

Рис. 1.1: Сигнал импульса

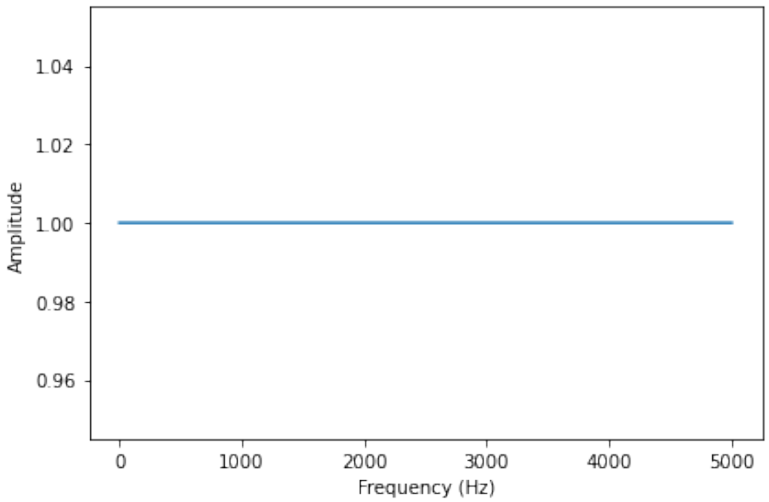

Рис. 1.2: Его спектр

Глава 2

Окна и фильтры

Вычисление двухэлементного скользящего среднего.

```
window_array = np.array([0.5, 0.5, 0, 0, 0, 0, 0, 0,])
window = Wave(window_array, framerate = 8)
filtr = window.make_spectrum(full = True)

```

Листинг 2.1: Найдем двухэлементное скользящее среднее

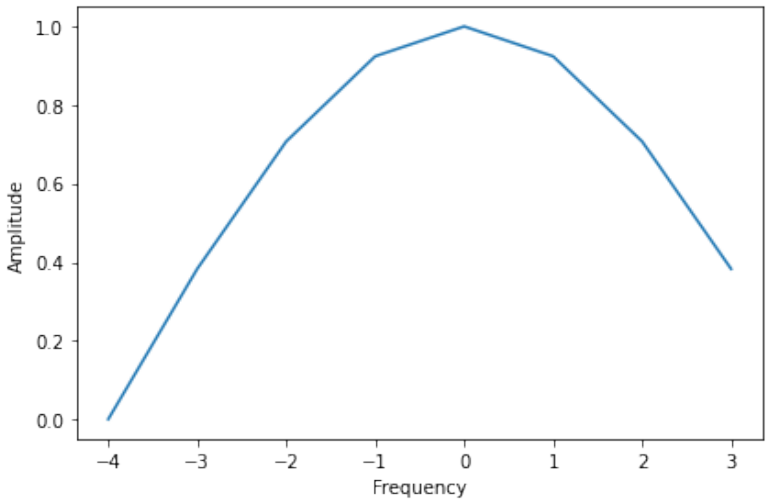

Рис. 2.1: ДПФ двухэлементного скользящего среднего

```
product = impulse_spectrum * filtr
filtered = product.make_wave()

```

Листинг 2.2: Умножение передаточной функции на спектр импульса

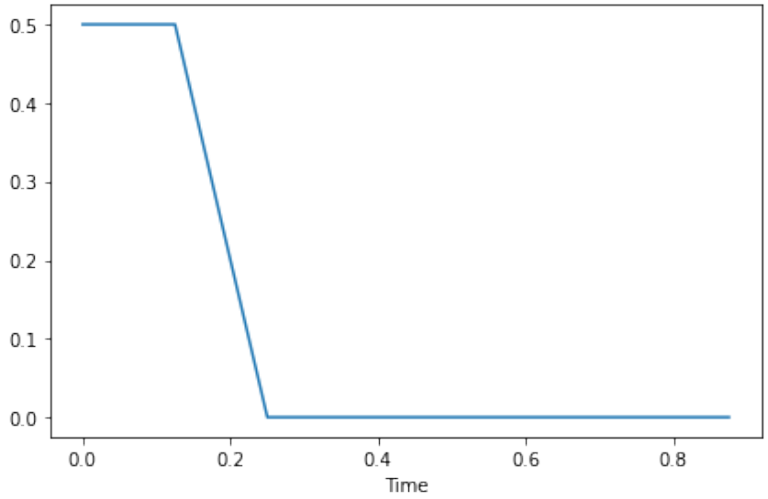

Рис. 2.2: Импульсная реакция

Запись импульсной характеристики достаточна для характеристики всей системы, потому что это `IDFT` для передаточной функции.

Глава 3

Акустическая характеристика

Проведём исследование акустической характеристики некоторого пространства (например, комнаты).

```
from thinkdsp import read_wave

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

```

Листинг 3.1: Звук выстрела

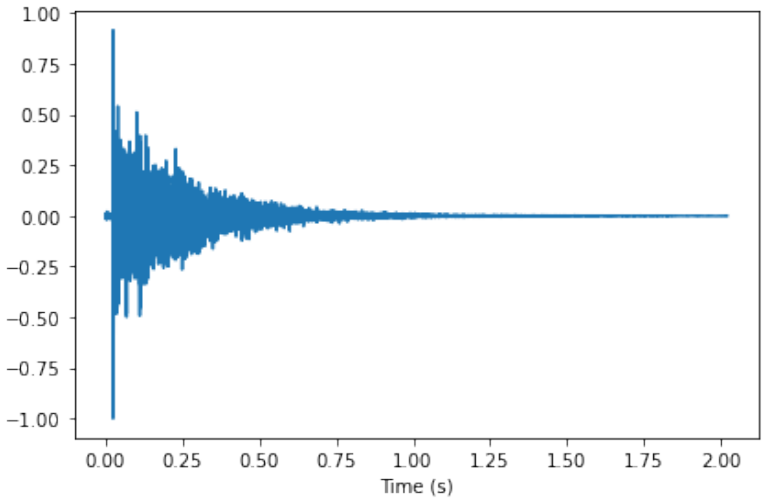

Рис. 3.1: Сегмент записи выстрела

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

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

```

Листинг 3.2: Строим спектр звука выстрела

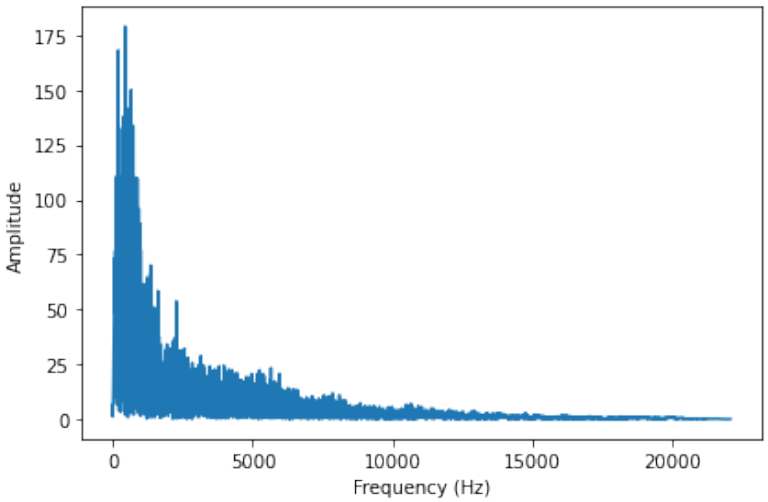

Рис. 3.2: Спектр звука выстрела

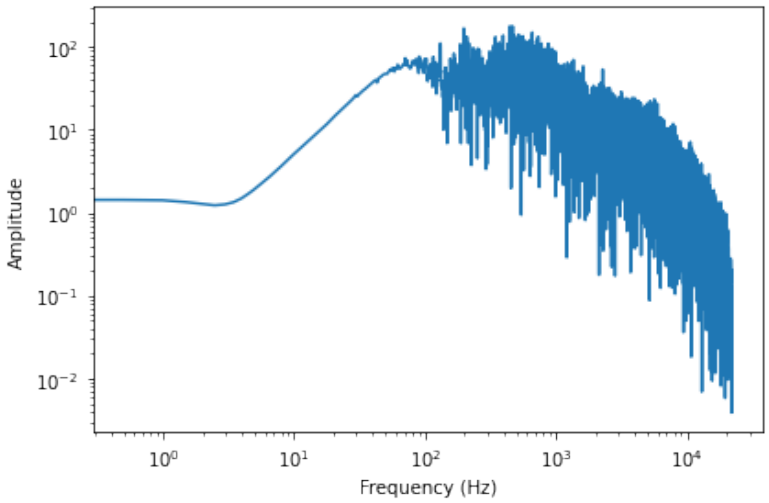

Рис. 3.3: Спектр в логарифмическом масштабе

Спектр соответствует отклику комнаты. Каждая частота в спектре представлена комплексным числом в виде амплитуды и фазы. Этот спектр называется Передаточной функцией.

```
    violin = read_wave('92002__jcveliz__violin-origional.
wav')
    start = 0.11
    violin = violin.segment(start = start)
    violin.shift(-start)
    violin.truncate(len(response))
    violin.normalize()
    violin.plot()
    spectrum = violin.make_spectrum()
    spectrum.plot()

```

Листинг 3.3: Звук скрипки

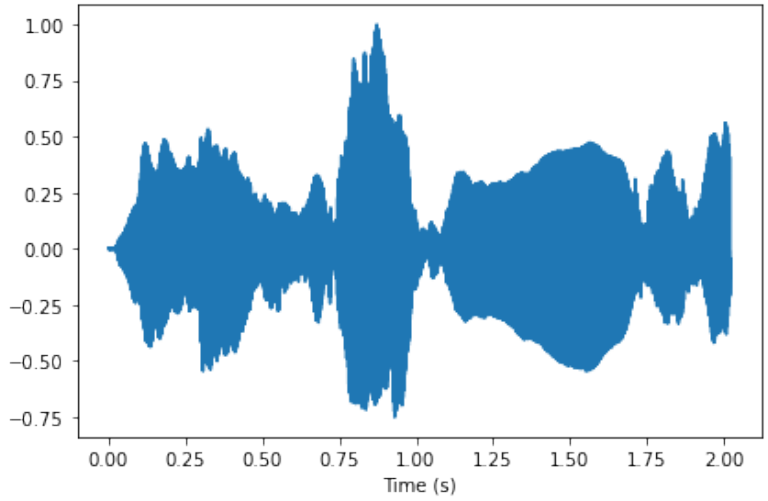

Рис. 3.4: Сегмент звука скрипки

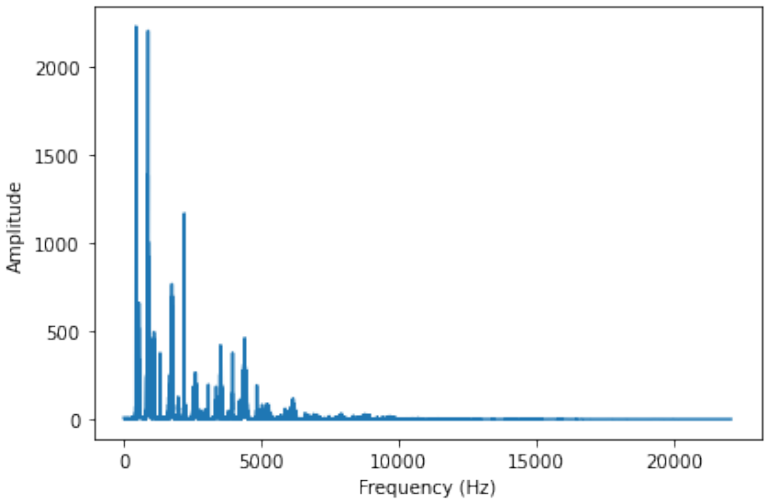

Рис. 3.5: Спектр этого звука

```
output = (spectrum * transfer).make_wave()
violin.plot()
output.plot()
spectrum = output.make_spectrum()
spectrum.plot()

```

Листинг 3.4: Вычисление выходного сигнала

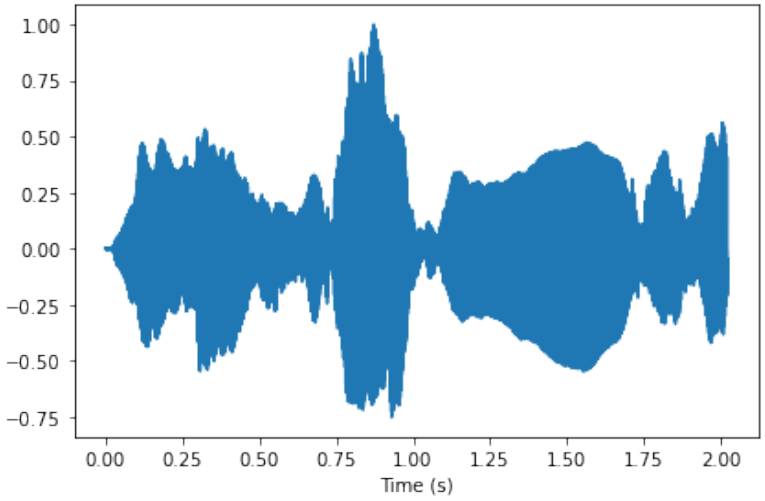

Рис. 3.6: Оригинальный звук

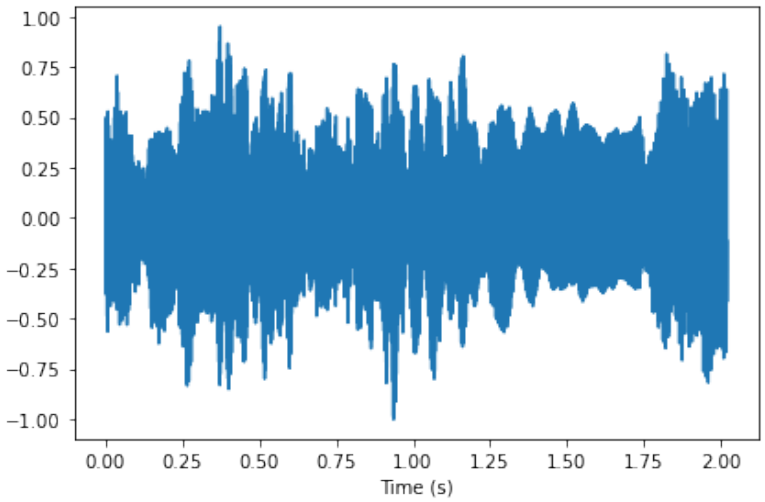

Рис. 3.7: Преобразованная запись

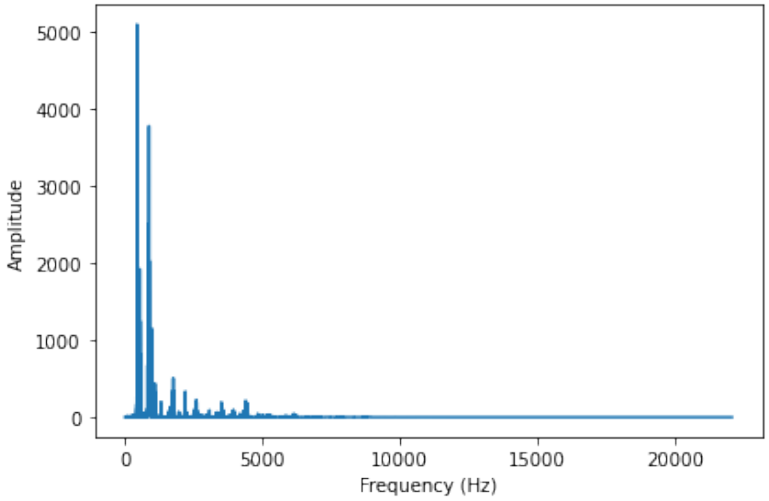

Рис. 3.8: Спектр преобразованного звука

Разница между этими двумя звуками хорошо различима на слух.

Глава 4

Системы и свертка

Выход системы - свертка входа и отклика системы.

```
    def shifted_scaled(wave, shift, factor):
        res = wave.copy()
        res.shift(shift)
        res.scale(factor)
        return res

    response2 = response + shifted_scaled(response, 1, 0.5)
    response2.plot()

```

Листинг 4.1: Создадим звук двух последовательных выстрелов

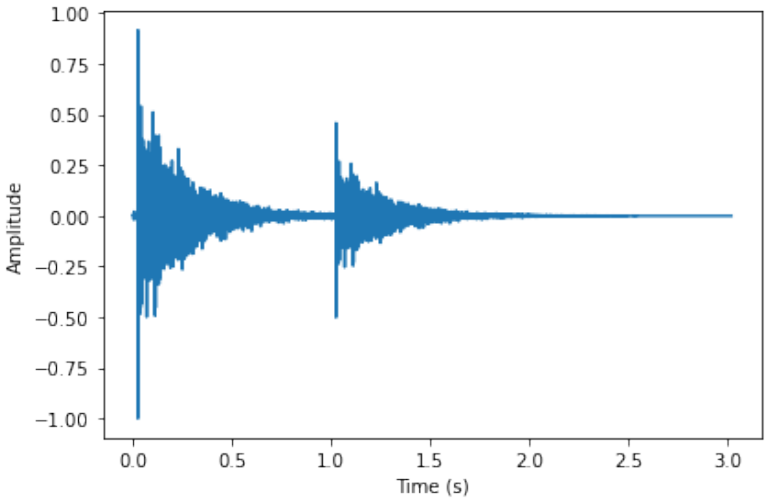

Рис. 4.1: Два выстрела

```
    dt = 1 / 441
    total = 0
    for k in range(220):
        total += shifted_scaled(response, k * dt, 1.0)

    total.normalize()
    total.plot()

```

Листинг 4.2: Очень много последовательных выстрелов

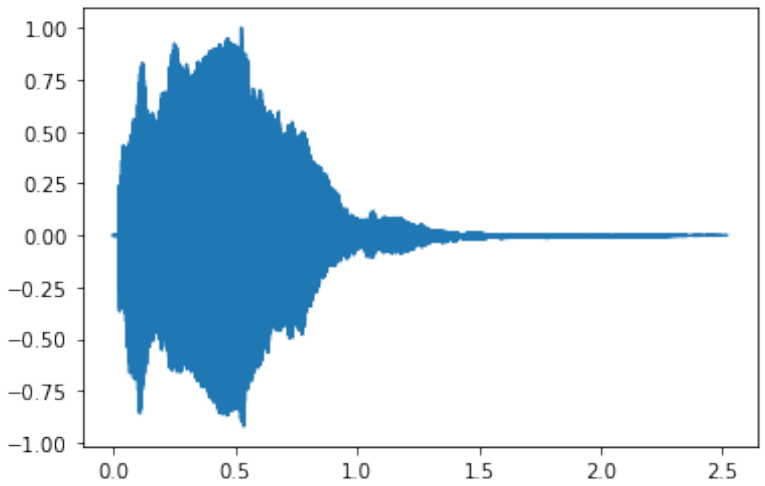

Рис. 4.2: Теперь это уже не похоже на звук выстрела

```
    from thinkdsp import SawtoothSignal

    signal = SawtoothSignal(freq = 441)
    wave = signal.make_wave(duration=0.2, framerate=
response.framerate)
    wave.plot()

```

Листинг 4.3: Произвольный входной сигнал

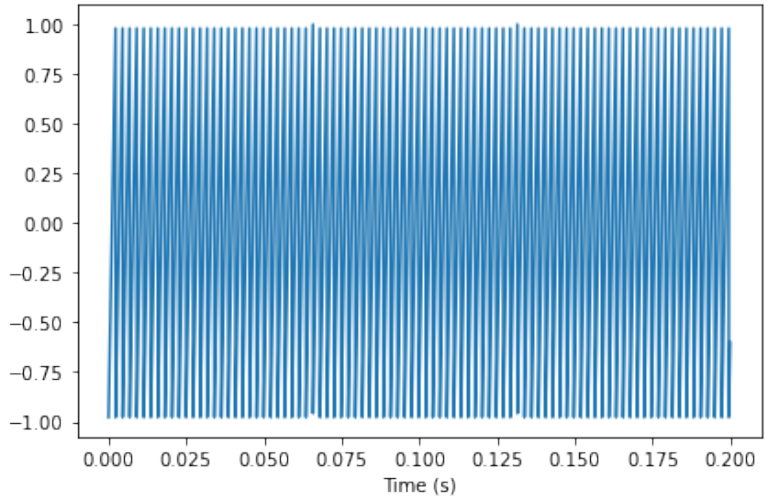

Рис. 4.3: Произвольный сигнал

```
    total = 0
    for t, y in zip(wave.ts, wave.ys):
        total += shifted_scaled(response, t, y)

    total.normalize()
    total.plot()

```

Листинг 4.4: Генерация сдвинутых версий импульсной характеристики

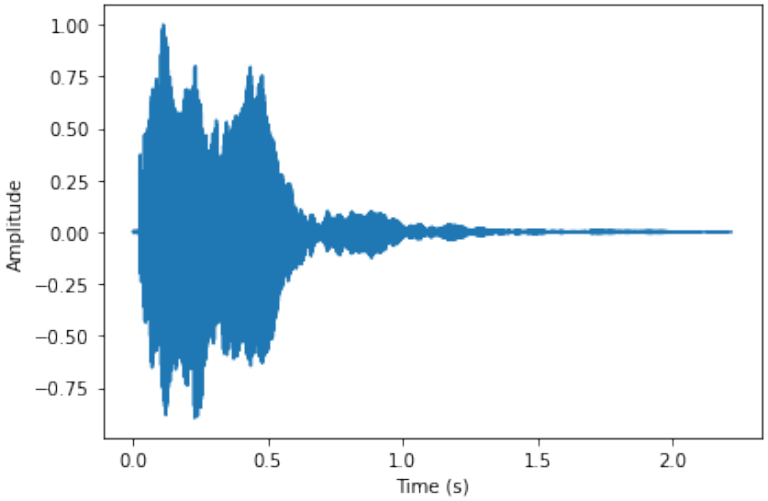

Рис. 4.4: Чем-то похоже на звук двух последовательных выстрелов

```
high = 5000
wave.make_spectrum().plot(high = high, color = '0.7')
segment = total.segment(duration = 0.2)
segment.make_spectrum().plot(high = high)
convolved2 = violin.convolve(response)
convolved2.plot()

```

Листинг 4.5: Используем метод свертки

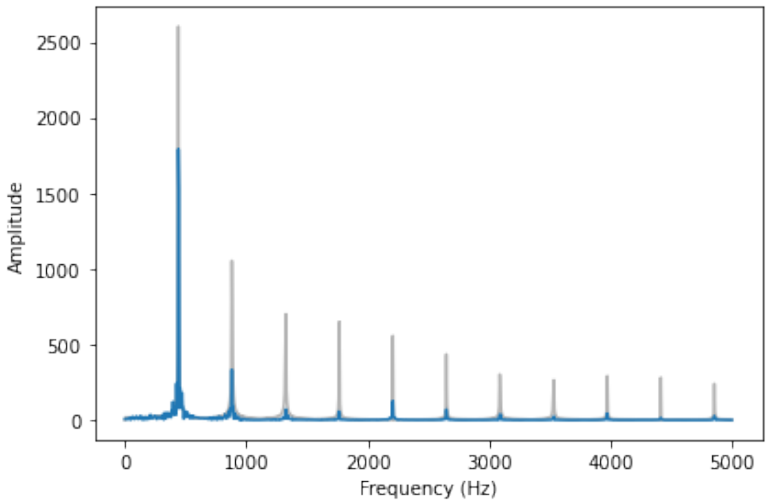

Рис. 4.5: Сравнение спектра до и после свертки

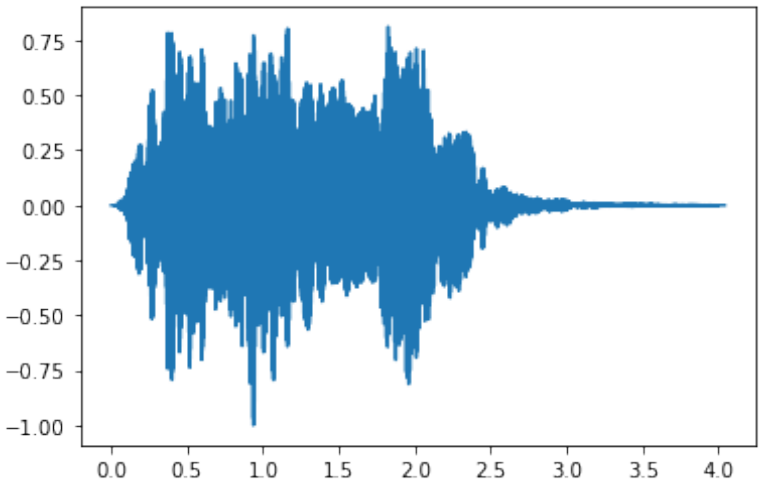

Рис. 4.6: Результат свертки

Глава 5

Упражнения

5.1 Задание 1

Устранение лишней ноты в начале фрагмента сигнала через дополнение нулями.

```
1       response = read_wave('180960__kleeb__gunshot.wav')
2       start = 0.12
3       response = response.segment(start=start)
4       response.shift(-start)
5       response.truncate(2 ** 16)
6       response.zero_pad(2 ** 17)
7       response.normalize()
8       response.plot()
9       transfer = response.make_spectrum()
10      transfer.plot()
11
```

Листинг 5.1: Сначала возьмём звук выстрела

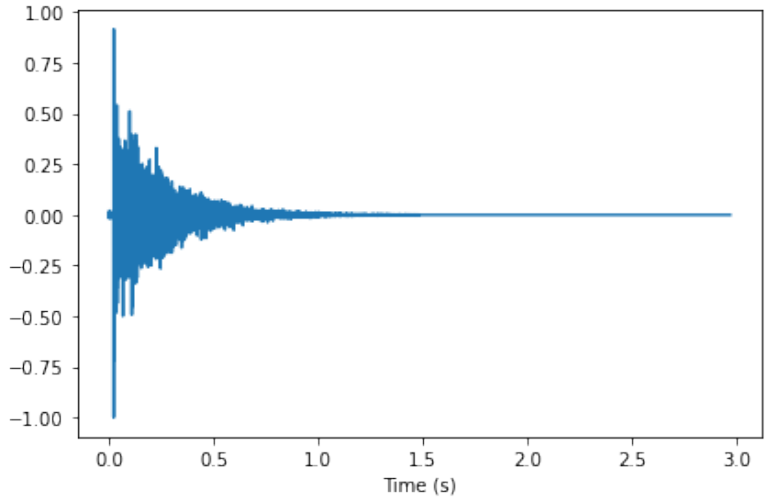

Рис. 5.1: Исправленный звук выстрела

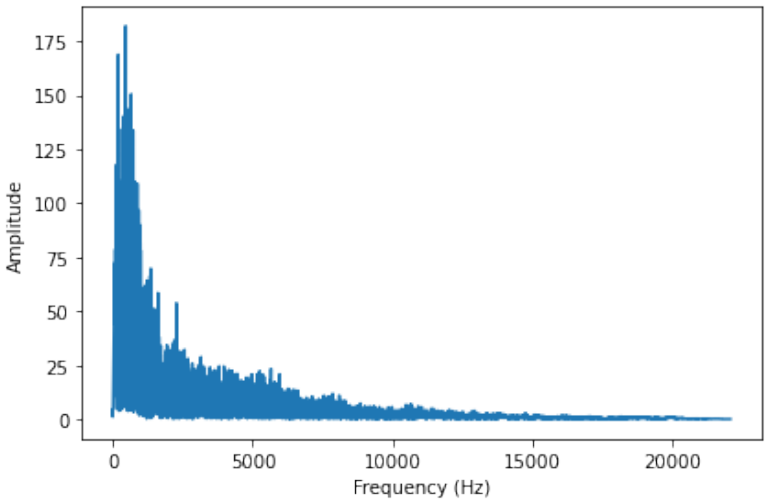

Рис. 5.2: И его спектр

```
    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()
```

Листинг 5.2: Теперь попробуем с сигналом скрипки

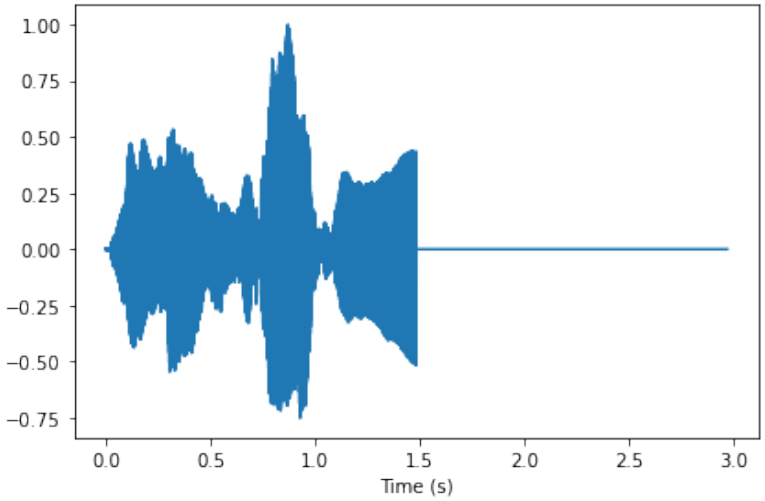

Рис. 5.3: Исправленный сигнал скрипки

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

Листинг 5.3: Перемножаем спектры

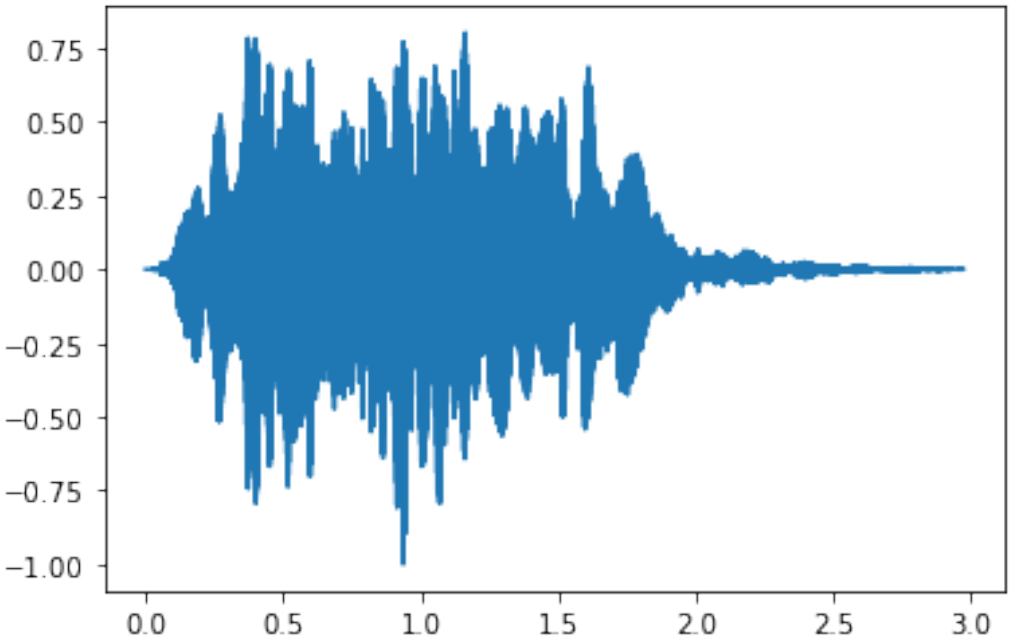

Рис. 5.4: Умножение на передаточную функцию

```
response.truncate(2 ** 16)
response.plot()
violin.truncate(2 ** 16)
violin.plot()

```

Листинг 5.4: Убираем нули в конце сигналов

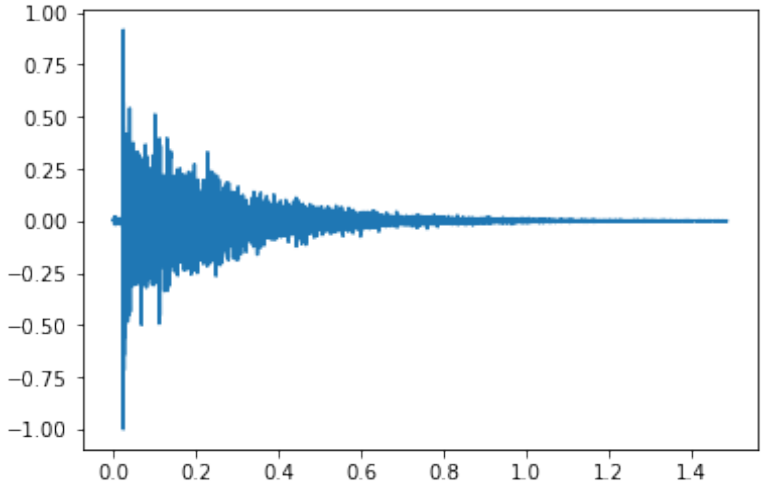

Рис. 5.5: Звук выстрела без нулей

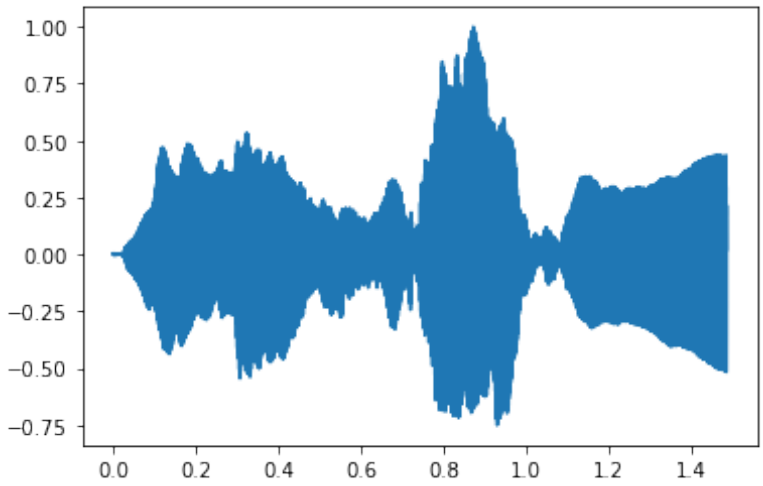

Рис. 5.6: Сигнал скрипки тоже без нулей

```
output2 = violin.convolve(response)
output2.plot()

```

Листинг 5.5: Свертка сигнала скрипки

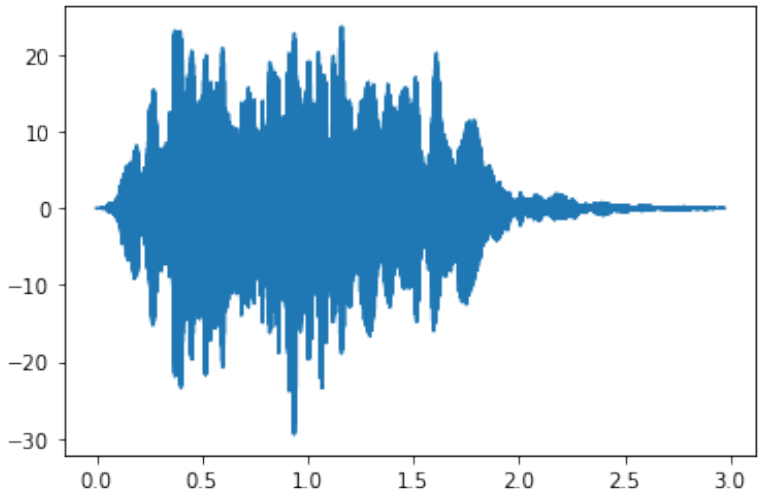

Рис. 5.7: Результат свертки

Результаты похожи по графикам и звуку, но `len(output2) = len(output) - 1`.

```
1       import scipy.signal
2       from thinkdsp import Wave
3
4       ys = scipy.signal.fftconvolve(violin.ys, response.ys)
5       output3 = Wave(ys, framerate = violin.framerate)
6       output3.plot()
7
```

Листинг 5.6: Свертка сигнала скрипки с помощью FFT

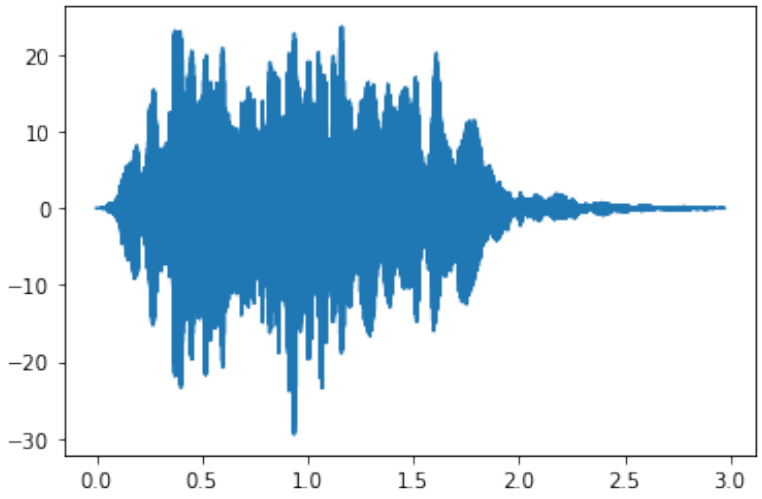

Рис. 5.8: Результат свертки с помощью другой функции

Разница между результатами работы первой и второй функций близка к 0, но вторая функция работает гораздо быстрее.

5.2 Задание 2

Моделирование звучания записи в том пространстве, где была измерена импульсная характеристика.

```
1       response = read_wave('stalbans_a_mono.wav')
2       response = response.segment(duration = 5)
3       response.shift(-0)
4       response.normalize()
5       response.plot()
6       transfer = response.make_spectrum()
7       transfer.plot()
8       decorate(xlabel = 'Frequency (Hz)', ylabel='Amplitude')
9
10      transfer.plot()
11      decorate(xlabel = 'Frequency (Hz)', ylabel='Amplitude',
    xscale = 'log', yscale = 'log')
12
```

Листинг 5.7: Сначала попробуем какой-то звук

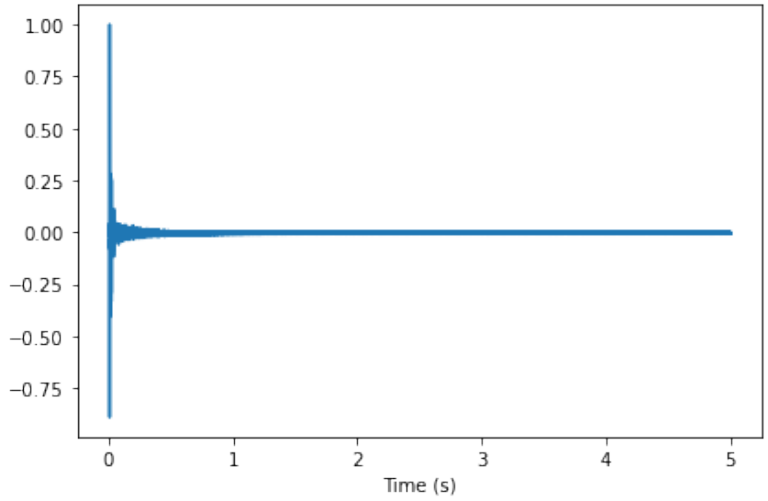

Рис. 5.9: Сигнал импульса

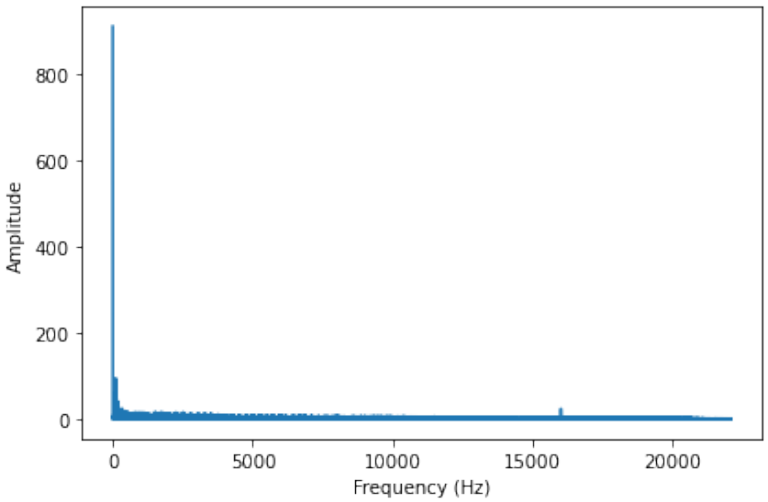

Рис. 5.10: Спектр импульса

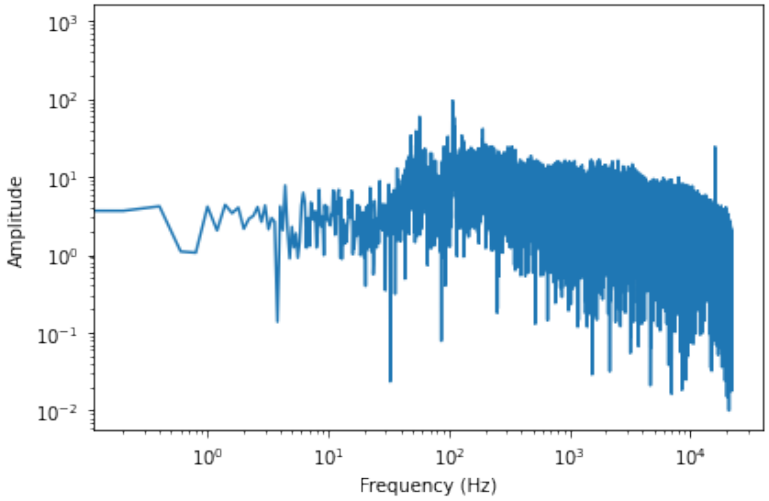

Рис. 5.11: Спектр импульса в логарифмическом масштабе

Теперь мы можем смоделировать, как звучала бы запись, если бы она воспроизводилась в той же комнате и записывалась таким же образом.

```
wave = read_wave('170255__dublie__trumpet.wav')
start = 0.0
wave = wave.segment(start = start)
wave.shift(-start)
wave.truncate(len(response))
wave.normalize()
wave.plot()

```

Листинг 5.8: Запись игры на трубе

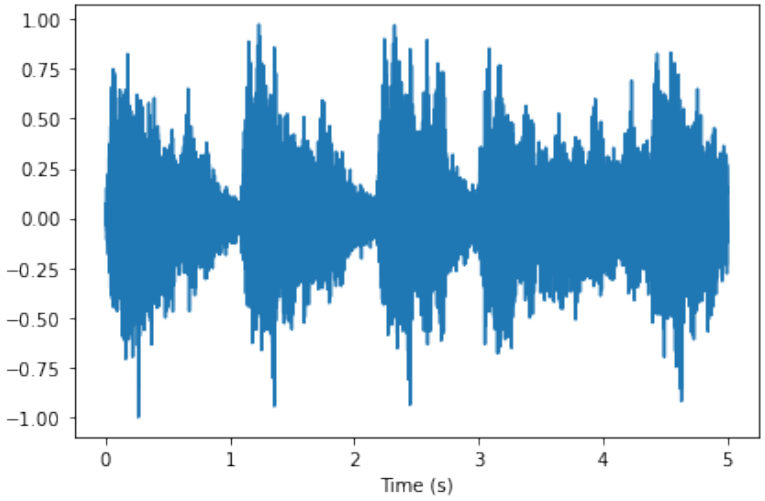

Рис. 5.12: Сигнал записи игры на трубе

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

```

Листинг 5.9: Перемножаем спектры для этого сигнала

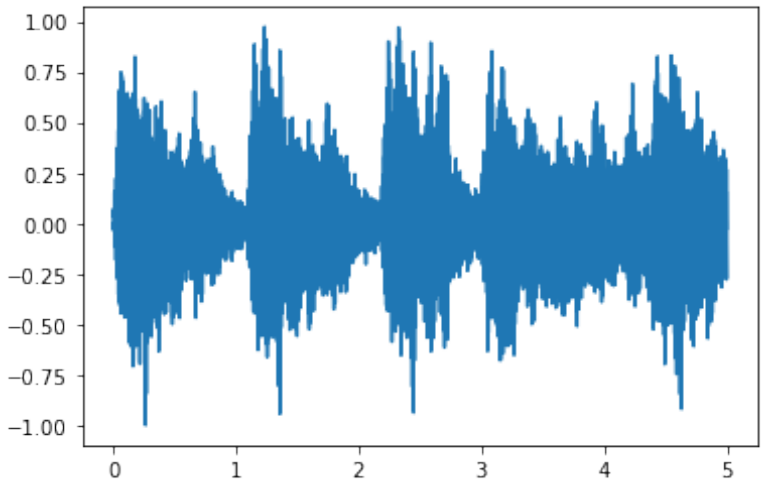

Рис. 5.13: Оригинальная запись

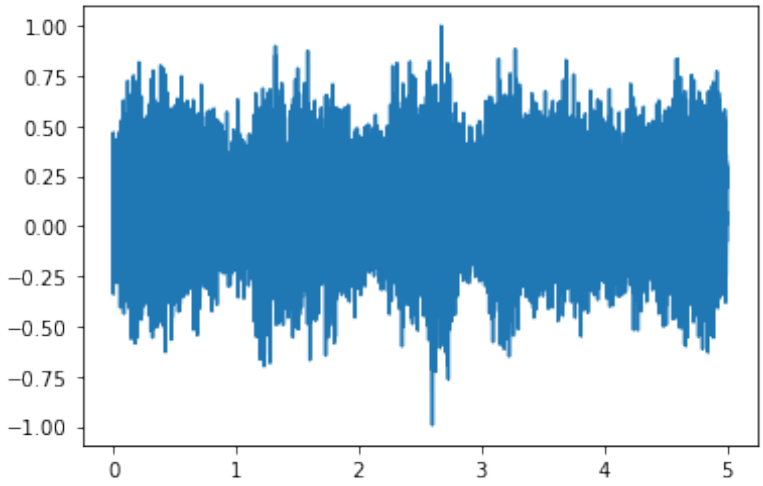

Рис. 5.14: Изменённая запись сигнала

Теперь кажется, что запись игры на трубе делали там же, где записывали импульс. Аналогичное решение можно получить, применив встроенную функцию свертки `fftconvolve`.

Глава 6

Вывод

В данной работе мы познакомились с линейными стационарными системами и акустическими характеристиками пространства. Также использовали различные методы для получения выхода системы по входному сигналу и отклику этой системы.