Лабораторная работа 10

Луняк Николай

2 мая 2021 г.

Оглавление

Список иллюстраций

Листинги

Глава 1

Импорты

«Полезные штуки» на все случаи жизни.

```
from thinkdsp import Signal, Sinusoid, SquareSignal,
    TriangleSignal, SawtoothSignal, ParabolicSignal
from thinkdsp import normalize, unbias, PI2, decorate
from thinkdsp import Chirp
from thinkdsp import read_wave
from thinkdsp import Spectrum, Wave,
    UncorrelatedGaussianNoise, Spectrogram
from thinkdsp import Noise
from thinkdsp import CubicSignal
from thinkdsp import zero_pad

import numpy as np
import pandas as pd

from matplotlib import pyplot as plt

import thinkstats2

from scipy.stats import linregress

import scipy
import scipy.fftpack

import scipy.signal

from ipywidgets import interact, interactive, fixed
import ipywidgets as widgets

loglog = dict(xscale='log', yscale='log')

```

```
29 PI2 = np.pi * 2
```

Листинг 1.1: Импорты

Глава 2

Дополняем нулями

В разделе 10.4 была рассмотрена операция *линейной* свертки, а в 10.3 выполнялась *циклическая*. «Превратить» ее в линейную можно было бы при помощи встаки нулей в конец.

Возьмем уже рассмотренные звуки выстрела и скрипки, обрежем их до 2^16 и дополним нулями до 2^17.

```
response = read_wave('Sounds/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 2.1: Сигнал выстрела

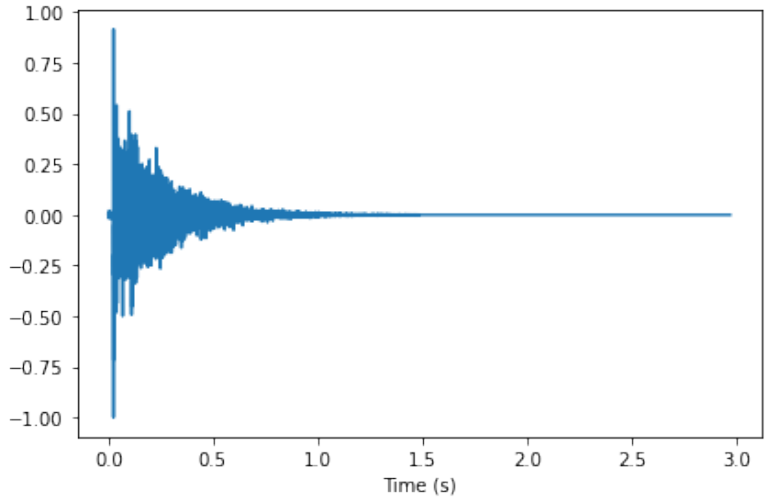

Рис. 2.1: Вытрел, дополненный нулями

Вот его спектр.

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

Листинг 2.2: Спектр выстрела

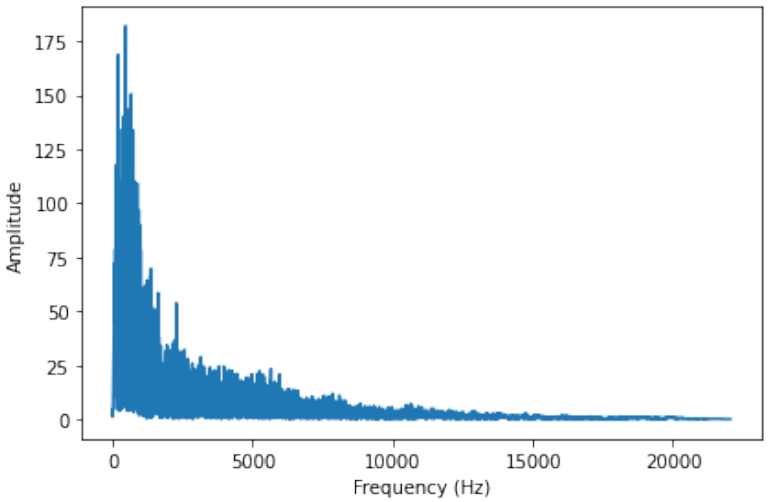

Рис. 2.2: Спектр

Загружаем скрипку.

```
violin = read_wave('Sounds/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 2.3: Сигнал скрипки

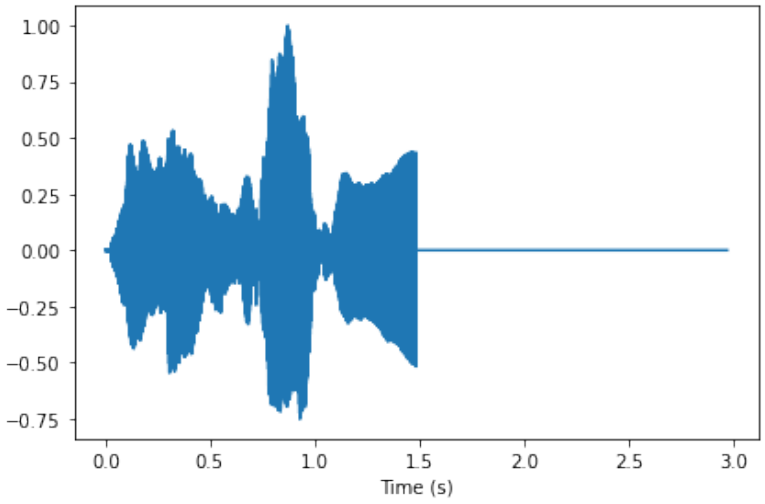

Рис. 2.3: Сигнал скрипки с нулями

Подействуем спектром передаточной функцией системы на скрипку (содержит свойства комнаты, где был записан выстрел).

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

Листинг 2.4: Перемножаем спектры

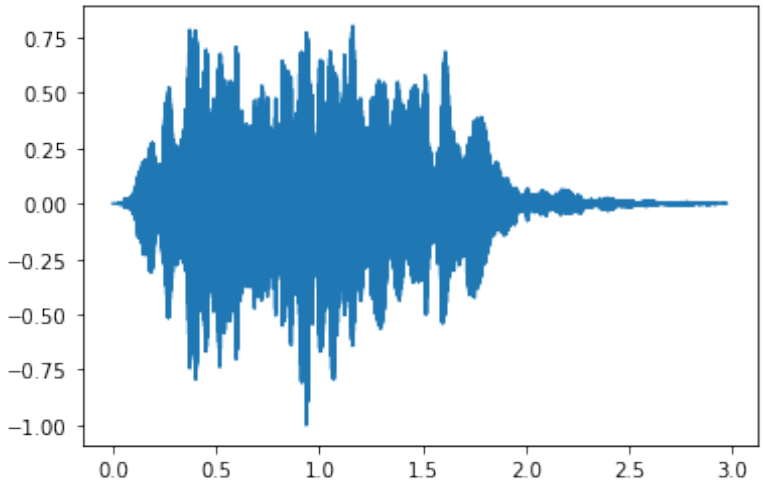

Рис. 2.4: Результат воздействия

Теперь проделаем то же самое, но при помощи свертки. Сначала уберем нули, которые мы вставили в конец.

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

Листинг 2.5: Убираем нули в звуке выстрела

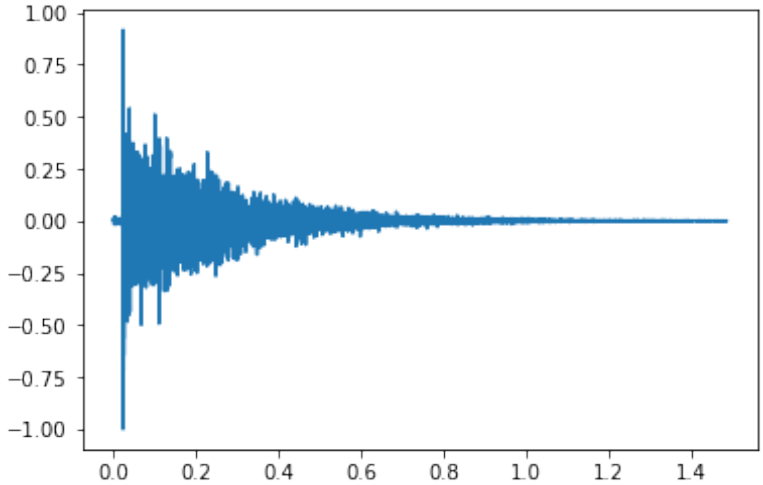

Рис. 2.5: Выстрел без дополнительных нулей

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

```
violin.plot()
```

Листинг 2.6: Убираем нули в звуке скрипки

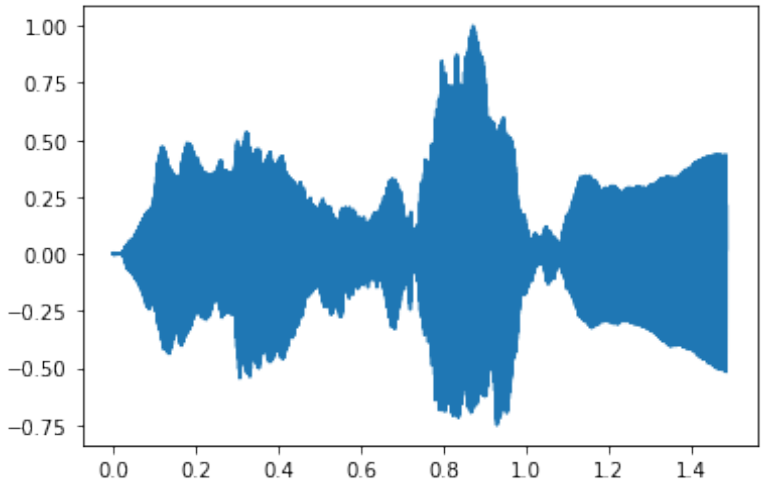

Рис. 2.6: Скрипка без дополнительных нулей

Выполняем свертку.

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

Листинг 2.7: Свертка

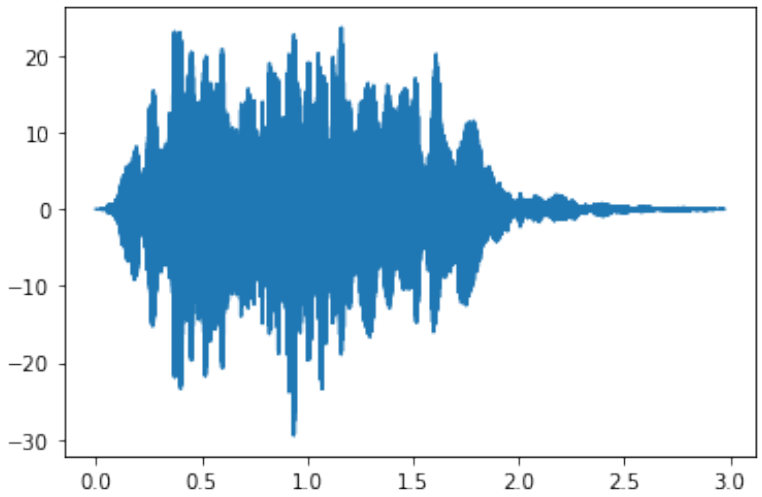

Рис. 2.7: Результат свертки

Разницы в картинках не видно. Если сравнить результат на слух, уловить какую-то разницу тоже не удастся. Единственный нюанс заключается лишь в том, что `len(output)` и `len(output2)` равны 131072 и 131071 соответственно.

Глава 3

Симулируем какой-нибудь свой звук

В качестве импульса возьмем это: https://www.openair.hosted.york.ac.uk/?page_id=425, а подопытным звуком пусть будет: https://freesound.org/people/JarredGibb/sounds/233099/.

```
response = read_wave('Sounds/s1r2_0_1.wav')

start = 0.08
duration = 1
response = response.segment(start=start, duration=duration)
response.shift(-start)

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

Листинг 3.1: Загрузка импульса

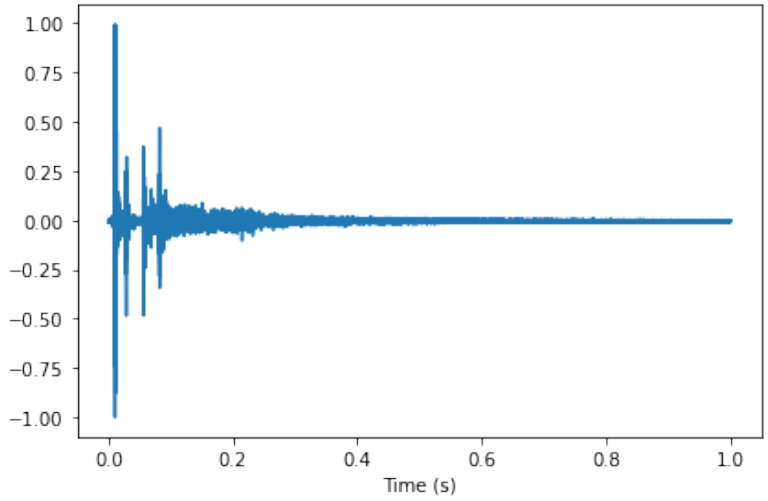

Рис. 3.1: Импульс

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

Листинг 3.2: Спектр импульса

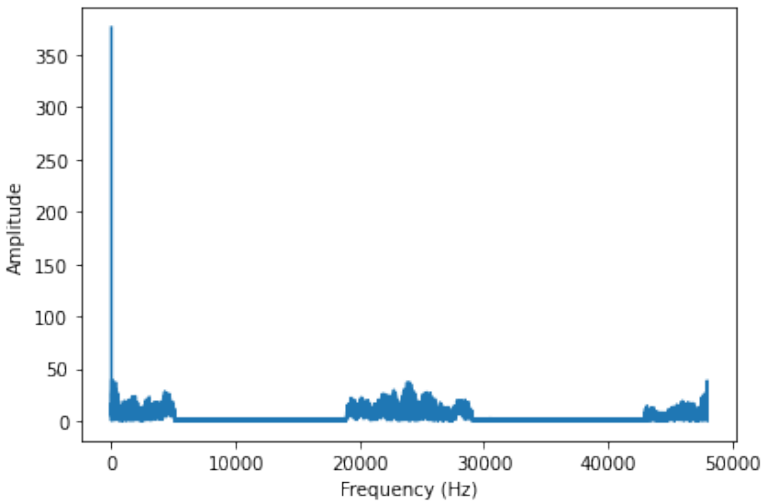

Рис. 3.2: Спектр

Теперь возьмем наш второй звук.

```
wave = read_wave('Sounds/233099__jarredgibb__goose-horn-1-96
    khz.wav')
```

```
2
3 start = 0.28
4 wave = wave.segment(start=start)
5 wave.shift(-start)
6
7 wave.truncate(len(response))
8 wave.normalize()
9 wave.plot()
10 decorate(xlabel='Time (s)')
```

Листинг 3.3: Зугрузка кряка

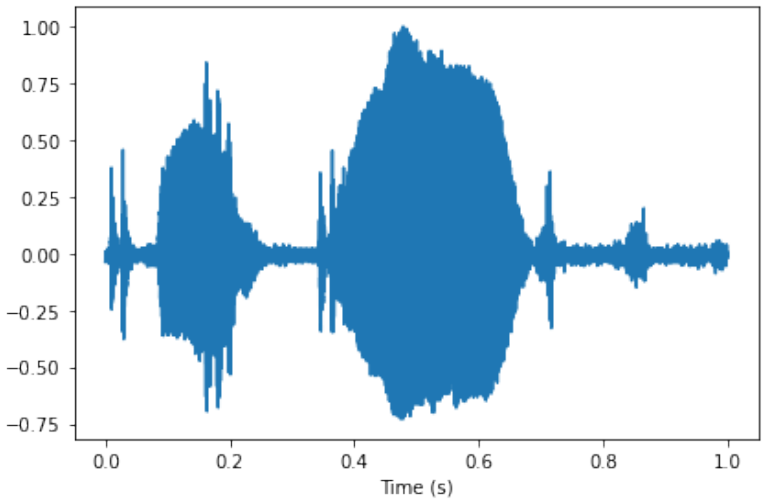

Рис. 3.3: Кряк

Применим "эффект".

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

Листинг 3.4: Трансформация

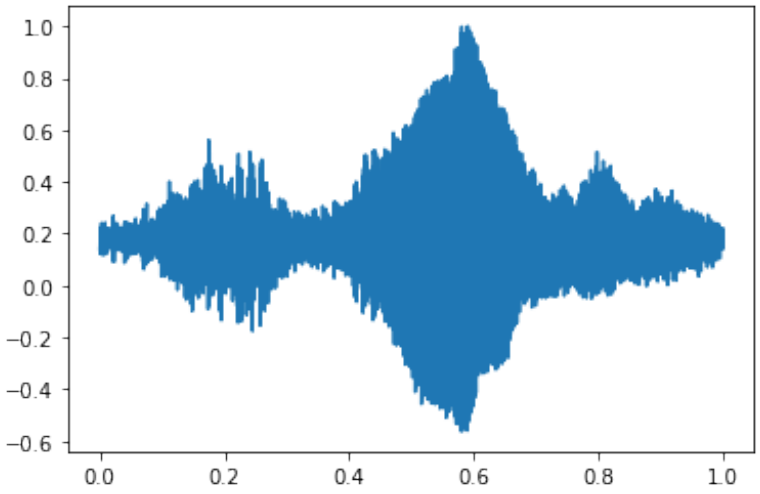

Рис. 3.4: Спектр

Если прослушать теперь кряк "до"и кряк "после то кажется, что второй произошел там же, где записывали импульс. Аналогичное решение получаем и через свертку:

```
ys = scipy.signal.fftconvolve(wave.ys, response.ys)
convolved2 = Wave(ys, framerate=wave.framerate)
convolved2.normalize()
convolved2.make_audio()
```

Листинг 3.5: Трансформация (x2)