Лабораторная работа 11

Луняк Николай

3 мая 2021 г.

Оглавление

Список иллюстраций

Листинги

Глава 1

Запуск `chap11.ipynb`

Нас просят запустить исходный файл `chap11.ipynb`, послушать, как звучат разные звуки. Тут не все так просто.

При запуске одной ячейки у меня дважды «умирало» ядро в notebook'е.

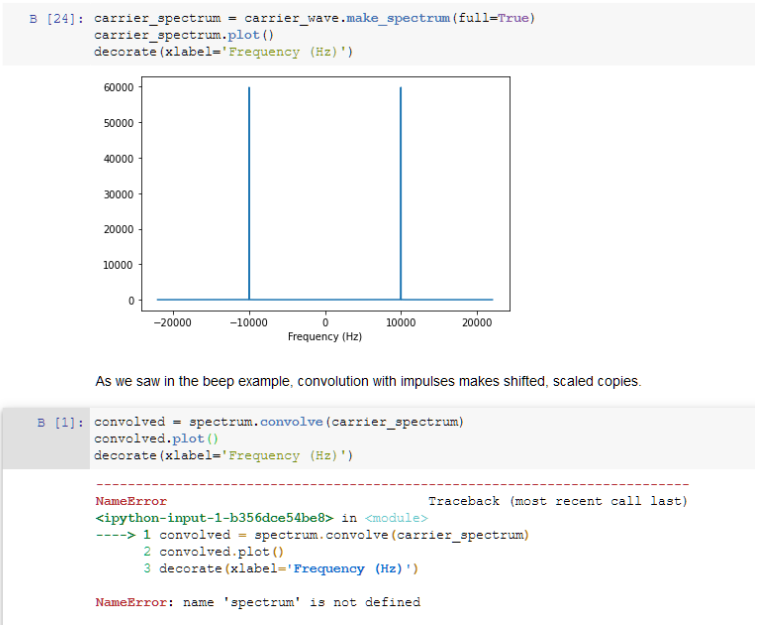

Рис. 1.1: Опасное место

Все ячейки выше работают нормально, а при запуске этой в консоли летят ошибки.

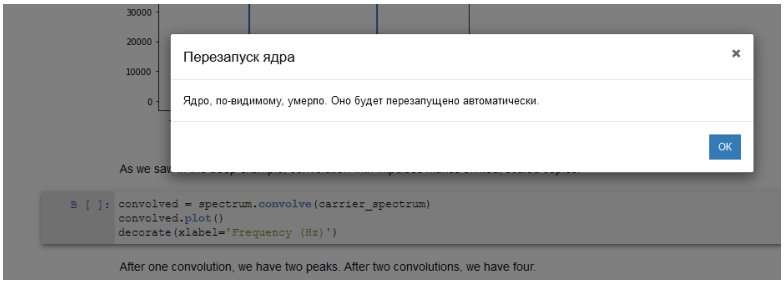

Рис. 1.2: Ошибка

Попробую сменить ядро «Python 3» на «Python 3.8.1 64-bit»... неа, не помогает.

Попробуем заменить в реализации `convolve()` версию `numpy` на `scipy`:

```
from thinkdsp import Spectrum

import scipy.signal

def convolve2(self, other):
    assert all(self.fs == other.fs)

    if self.full:
        hs1 = np.fft.fftshift(self.hs)
        hs2 = np.fft.fftshift(other.hs)
        hs = scipy.signal.fftconvolve(hs1, hs2, mode='same')
        hs = np.fft.ifftshift(hs)

    return Spectrum(hs, self.fs, self.framerate, self.full)

convolved = convolve2(spectrum, carrier_spectrum)
convolved.plot()
decorate(xlabel='Frequency (Hz)')
```

Листинг 1.1: Замена

Да, так работает! Придется везде внизу переписывать реализацию тогда.

Глава 2

Видеоролик

В этом задании нам предложено посмотреть: `https://youtu.be/cIQ9IXSUzuM`. Вообще, выглядит вполне как достойный контент, а больше всего меня заинтересовал неплоский dithering, который еще и приглушить можно так, что мои уши уже будут неспособны воспринять его.

Глава 3

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 3.1: Импорты

Глава 4

Боремся со спектральными копиями

Наша задача теперь - сымитировать сэмплирование, из-за чего появятся спектральные копии (ведь это же воздействие последовательностью импульсов), а затем избавиться от этих копий при помощи `low_pass()`, потому что они появятся в предсказуемых местах.

```
wave = read_wave('Sounds/263868__kevcio__amen-break-a-160-bpm.wav')
wave.normalize()
wave.plot()
wave.make_audio()
```

Листинг 4.1: Ударные

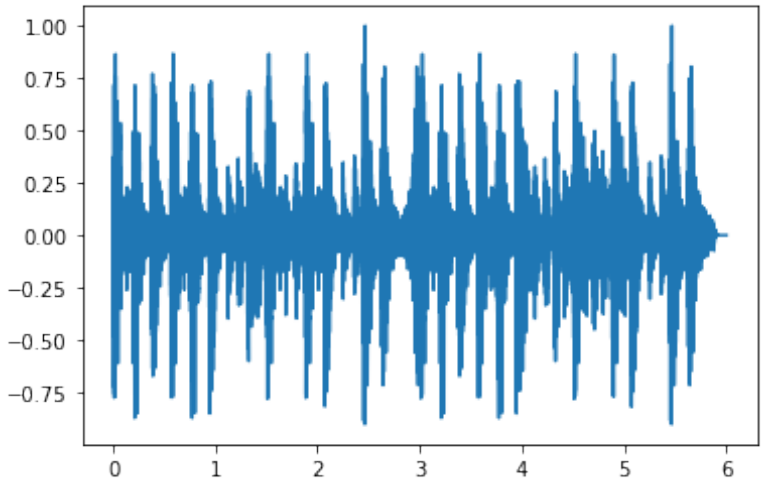

Рис. 4.1: Исходный сигнал

Этот сигнал будет как бы «настоящим» звуком.

Вот его спектр.

```
spectrum = wave.make_spectrum(full=True)
spectrum.plot()
```

Листинг 4.2: Спектр

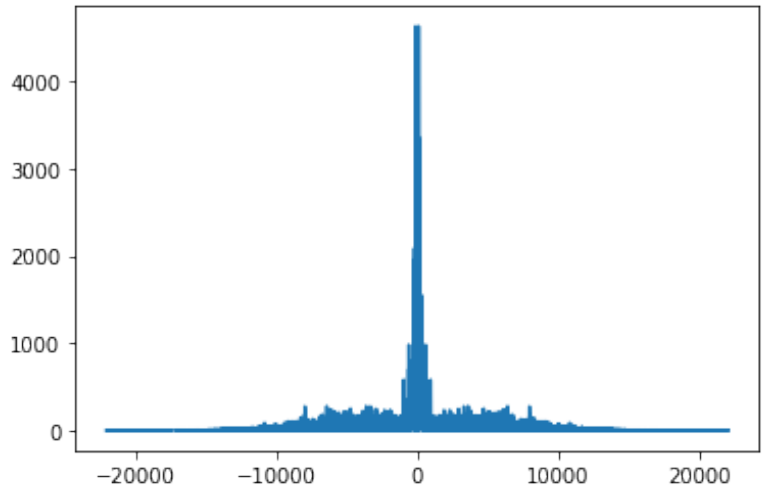

Рис. 4.2: Настоящий спектр

Сначала просто обрежем спектр:

```
factor = 5
framerate = wave.framerate / factor
cutoff = framerate / 2 - 1

spectrum.low_pass(cutoff)
spectrum.plot()

filtered = spectrum.make_wave()
filtered.make_audio()
```

Листинг 4.3: Сокращенный спектр

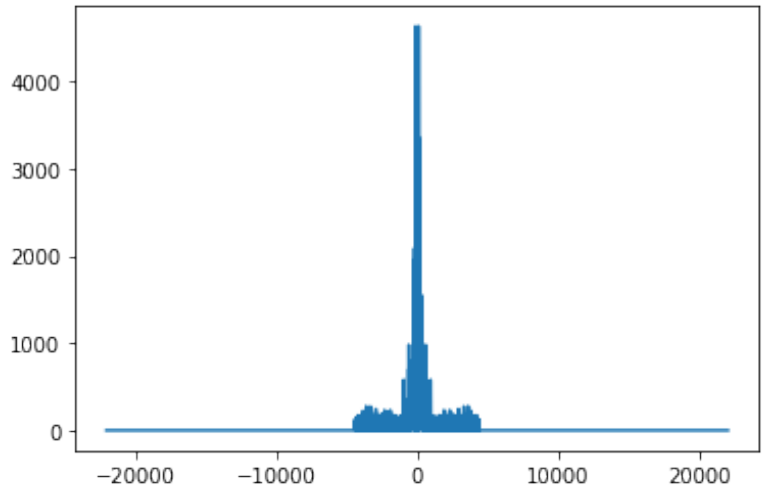

Рис. 4.3: Сокращенный Спектр

Теперь проведем симуляцию сэмплирования.

```
def sample(wave, factor):
    """Simulates sampling of a wave.

    wave: Wave object
    factor: ratio of the new framerate to the original
    """
    ys = np.zeros(len(wave))
    ys[::factor] = np.real(wave.ys[::factor])
    return Wave(ys, framerate=wave.framerate)

sampled = sample(filtered, factor)

sampled_spectrum = sampled.make_spectrum(full=True)
sampled_spectrum.plot()

sampled.make_audio()
```

Листинг 4.4: Сэмплирование

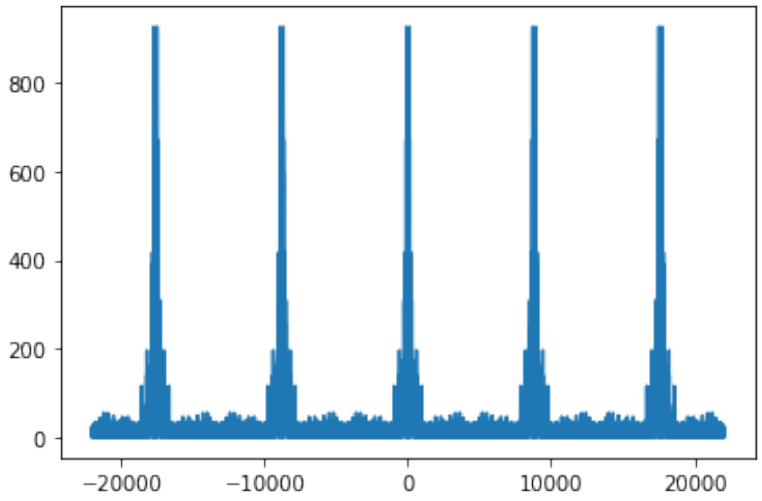

Рис. 4.4: Спектр результата сэмплирования

Обрежем его теперь по аналогии с тем, как мы это делали в первый раз и сравним с прошлым вариантом.

```
sampled_spectrum.low_pass(cutoff)
sampled_spectrum.scale(factor)
spectrum.plot()
sampled_spectrum.plot()
```

Листинг 4.5: Подчищаем спектр

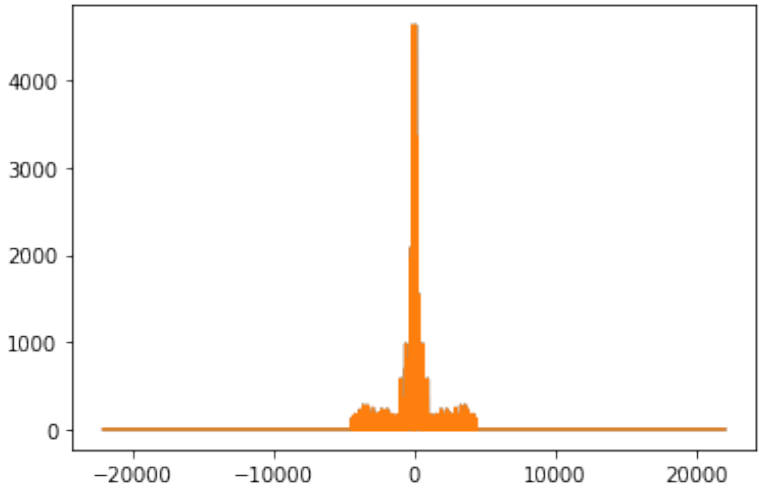

Рис. 4.5: Сравнение

Они не только выглядят идентично, но и звучат одинаково.