Лабораторная работа №2
Гармоники

Крынский Павел

26 мая 2021 г.

Оглавление

1 Упражнение 2.1 **4**

2 Упражнение 2.2 **5**
2.1 Написание класса и проверка сигнала 5
2.2 Спектр звука 6

3 Упражнение 2.3 **8**

4 Упражнение 2.4 **10**
4.1 Создание сигнала 10
4.2 Нулевой компонент спектра 11
4.3 Изменение нулевого компонента 11

5 Упражнение 2.5 **12**
5.1 Создание функции 12
5.2 Проверка работоспособности функции 12

6 Упражнение 2.6 **14**

7 Выводы **17**

Список иллюстраций

2.1 Полученный пилообразный звук 6
2.2 Спектр сегмента звука 6

3.1 Спектр сегмента звука 8

4.1 Визуализация созданного сигнала 10
4.2 Визуализация ускоренного звука 11

5.1 Сравнение спектров 13

6.1 Спектр сигнала 14
6.2 Спектр сигнала 15
6.3 Сравнение спектров 16

Листинги

2.1 Класс SawtoothSignal 5
2.2 Визуализация пилообразного звука 5
2.3 Спектр звука 6
3.1 Создание прямоугольного сигнала 8
3.2 Воспроизведение прямоугольного сигнала 9
3.3 Создание и воспроизведение сигнала с пониженной частотой 9
4.1 Создание треугольного сигнала 10
4.2 Вывод нулевого компонента 11
4.3 Смещение спектра и его визуализация 11
5.1 Создание функции 12
5.2 Создание сигнала и его воспроизведение 12
5.3 Сравнение спектров 12
5.4 Воспроизведение отфильтрованного звука 13
6.1 Создание сигнала и визуализация его спектра 14
6.2 Сравнение гармоник 14
6.3 Сегмент звука 15

Глава 1

Упражнение 2.1

В данном упражнении нас просят открыть `chap02.ipynb`, прочитать пояснения и запустить примеры. Поэтому здесь я изучил все примеры с комментариями и позапускал их.

Глава 2

Упражнение 2.2

2.1 Написание класса и проверка сигнала

Для данного упражнения нужно написать класс под названием `SawtoothSignal` и за основу просят взять `Signal`, но в главе примеры на основе `Sinusoid`, поэтому я буду использовать `Sinusoid`.

```
class SawtoothSignal(thinkdsp.Sinusoid):
    def evaluate(self,ts):
        cycles = self.freq * ts + self.offset / thinkdsp.PI2
        frac , _ = np.modf(cycles)
        ss = self.amp * frac
        return ss
```

Листинг 2.1: Класс SawtoothSignal

Убедимся, что все работает корректно.

```
ss = SawtoothSignal()
ss_wave = ss.make_wave(ss.period*3 , framerate = 20000)
ss_wave.plot()
```

Листинг 2.2: Визуализация пилообразного звука

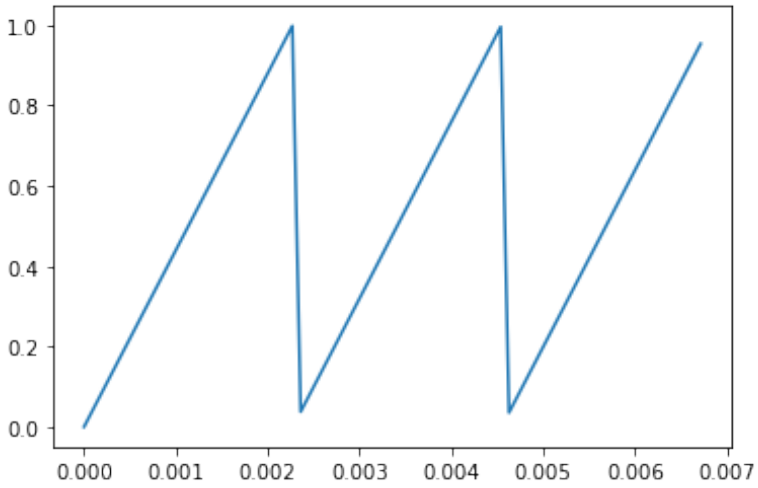

Рис. 2.1: Полученный пилообразный звук

2.2 Спектр звука

Теперь рассмотрим спектр нашего пилообразного звука.

```
spectr = ss_wave.make_spectrum()
spectr.plot()
```

Листинг 2.3: Спектр звука

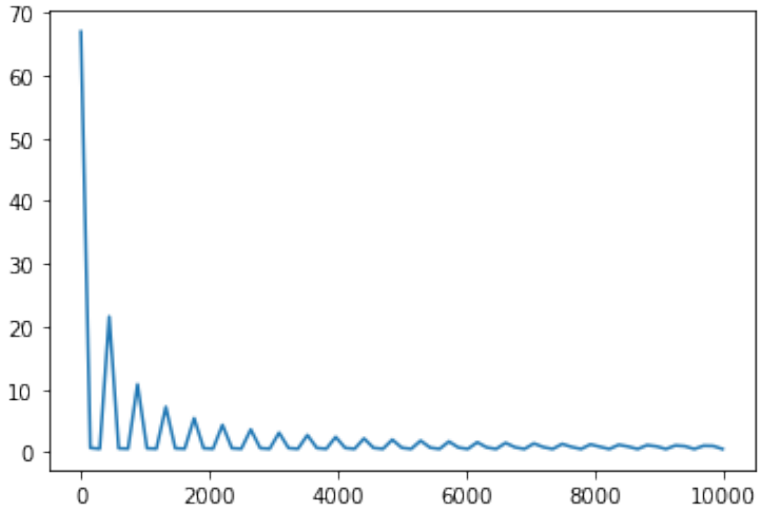

Рис. 2.2: Спектр сегмента звука

В сравнении с треугольной волной пилообразная форма уменьшается практически аналогично, но включает как четные, так и нечетные гармоники.

Глава 3

Упражнение 2.3

Нам нужно создать прямоугольный сигнал 1100 Гц и вычислить его спектр. Создадим прямоугольный сигнал 1100 Гц и вычислим его спектр.

```
signal = thinkdsp.SquareSignal(1100)
wave = signal.make_wave(duration=0.5, framerate=10000)
spectr = wave.make_spectrum()
spectr.plot()
```

Листинг 3.1: Создание прямоугольного сигнала

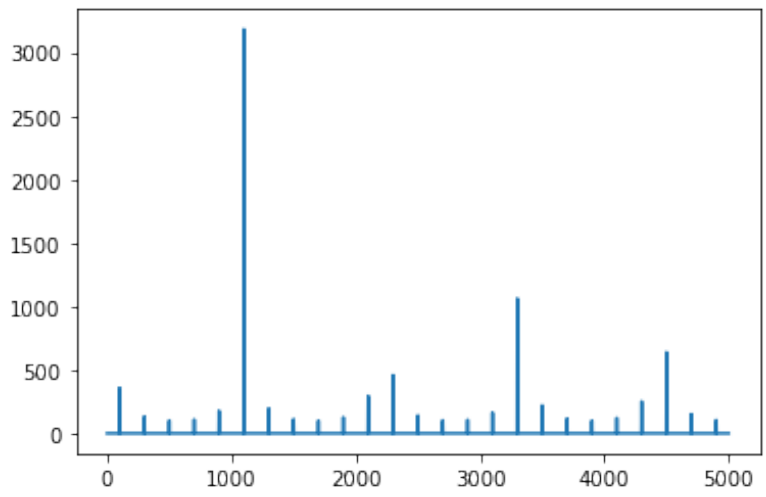

Рис. 3.1: Спектр сегмента звука

Основная и первая гармоника находятся в нужном месте, но вторая гармоника, которая должна быть 5500 Гц, смещается на 4500 Гц. Третья, которая должна быть 7700 Гц, находится на 2300 Гц и так далее.

Прослушаем полученный звук.

```
wave.make_audio()
```

Листинг 3.2: Воспроизведение прямоугольного сигнала

Когда мы слушаем полученную волну, можем слышать эти aliasing-гармоники, поскольку низкий тон имеет частоту 300 Гц. Создадим такой сигнал и прослушаем его. Разница присутствует и данные частоты можно расслышать.

```
thinkdsp.SinSignal(300).make_wave(duration=0.5,
    framerate=10000).make_audio()
```

Листинг 3.3: Создание и воспроизведение сигнала с пониженной частотой

Глава 4

Упражнение 2.4

4.1 Создание сигнала

Создадим треугольный сигнал с частотой 440 Гц и `wave` длительностью 0,01 секунды.

```
signal = thinkdsp.TriangleSignal()
wave = signal.make_wave(duration=0.01)
wave.plot()
```

Листинг 4.1: Создание треугольного сигнала

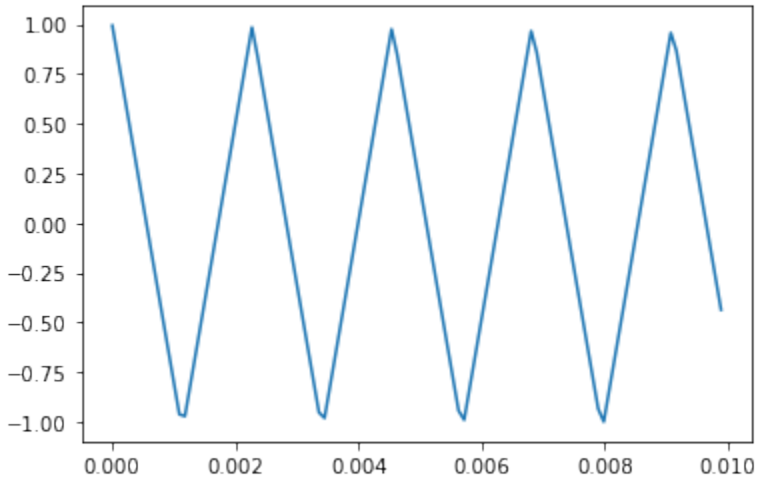

Рис. 4.1: Визуализация созданного сигнала

4.2 Нулевой компонент спектра

Первый элемент спектра - комплексное число, близкое к нулю. Если мы добавим в компонент нулевой частоты какое-то число, то это приведет к добавлению вертикального смещения спектра.

```
spectr = wave.make_spectrum()
spectr.hs[0]
```

Листинг 4.2: Вывод нулевого компонента

На выходе получили `(1.0436096431476471e-14+0j)`.

4.3 Изменение нулевого компонента

Теперь установим смещение равное 100 и увидим, что сигнал сместился по вертикали.

```
spectr.hs[0] = 100
spectr.make_wave().plot()
```

Листинг 4.3: Смещение спектра и его визуализация

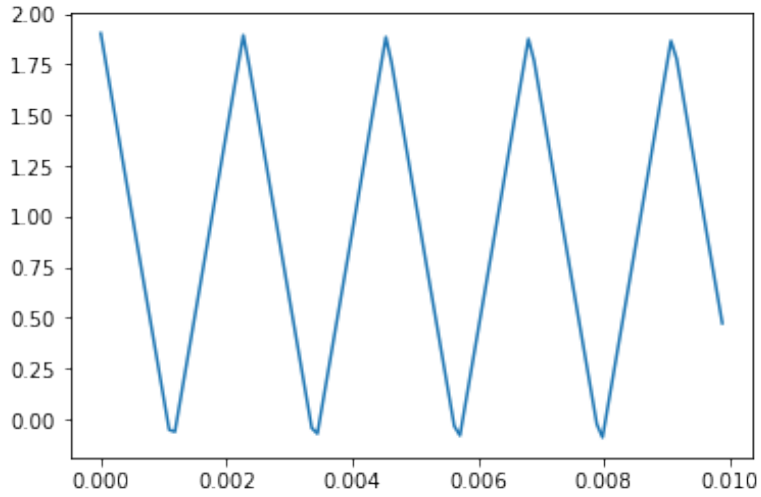

Рис. 4.2: Визуализация ускоренного звука

Глава 5

Упражнение 2.5

5.1 Создание функции

Напишем функцию, которая принимает `Spectrum` в качестве парметра и иизменяет его делением каждого элемента `hs` на соответсвующую частоту из `fs`.

```
def modif(spectr):
    for it in range (1 , len(spectr.hs)):
        spectr.hs[it] = spectr.hs[it] / spectr.fs[it]
    spectr.hs[0] = 0
```

Листинг 5.1: Создание функции

5.2 Проверка работоспособности функции

Создадим прямоугольный сигнал и прослушаем его.

```
wave = thinkdsp.SquareSignal(freq=400).make_wave(duration = 0.5)
wave.make_audio()
```

Листинг 5.2: Создание сигнала и его воспроизведение

Теперь выведем только что созданный спектр, а также изменим его при помощи нашей функции и посмотрим на результат.

```
spectrum = wave.make_spectrum()
spectrum.plot(high = 10000 ,color = 'red' )
modif(spectrum)
spectrum.scale(400)
```

```
spectrum.plot(high = 10000 )
```

Листинг 5.3: Сравнение спектров

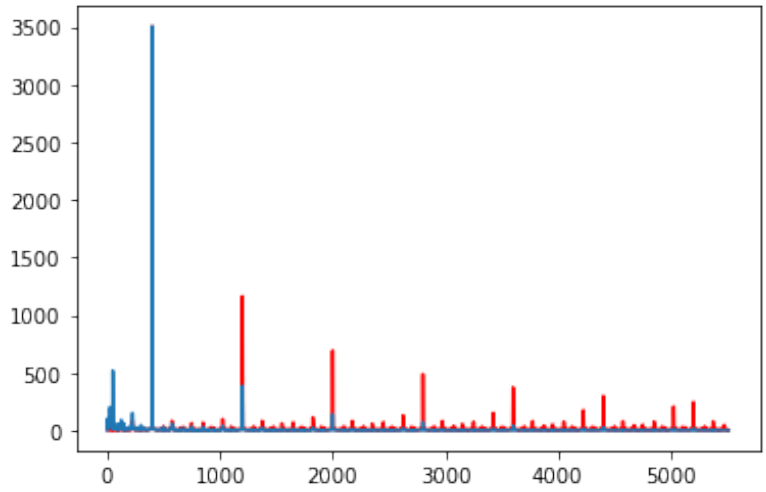

Рис. 5.1: Сравнение спектров

Фильтр подавляет гармоники, поэтому он действует как фильтр низких частот. Звук слышется почти как синусоида.

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

Листинг 5.4: Воспроизведение отфильтрованного звука

Глава 6

Упражнение 2.6

Создадим пилообразный сигнал и выведем его спектр.

```
signal = thinkdsp.SawtoothSignal(freq=500)
wave = signal.make_wave(duration=0.5, framerate=20000)
wave.make_audio()
spectr = wave.make_spectrum()
spectr.plot()
```

Листинг 6.1: Создание сигнала и визуализация его спектра

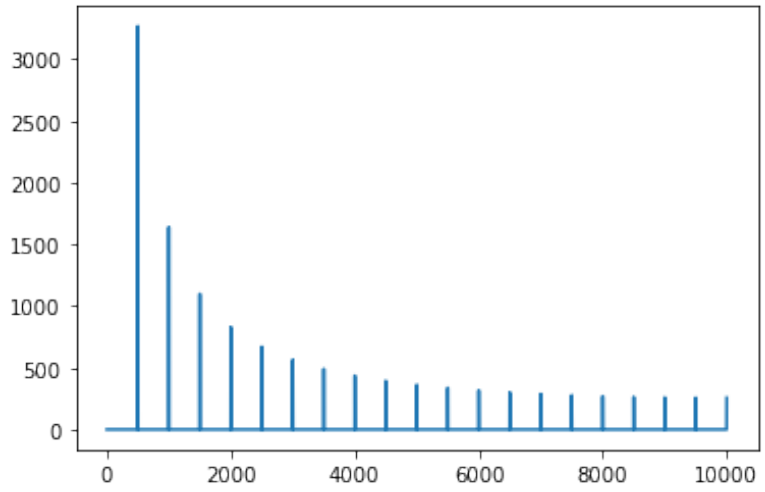

Рис. 6.1: Спектр сигнала

На рисунке видно, что гармоники уменьшаются как $1/f$.

```
spectr.plot(color='red')
```

```
modif(spectr)
spectr.scale(500)
spectr.plot()
```

Листинг 6.2: Сравнение гармоник

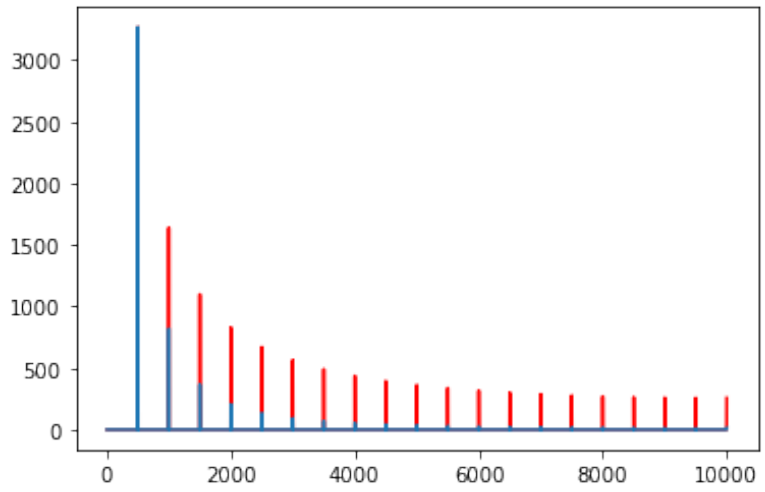

Рис. 6.2: Спектр сигнала

Теперь гармоника схожа с синусоидой.

```
wave.segment(duration=0.01).plot()
```

Листинг 6.3: Сегмент звука

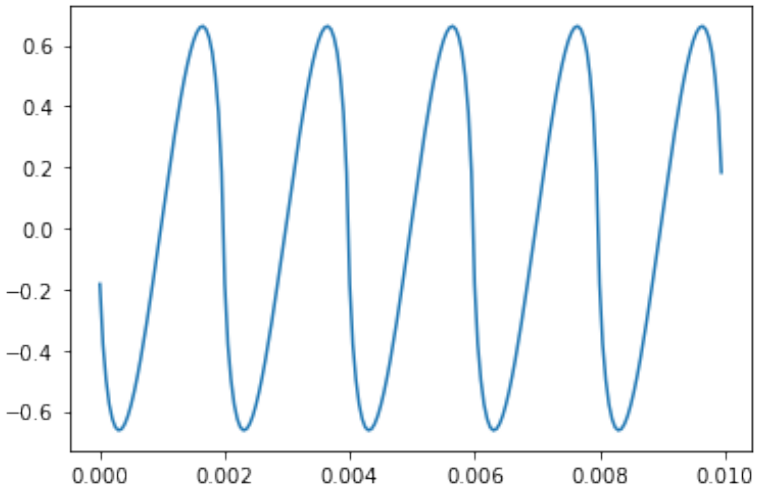

Рис. 6.3: Сравнение спектров

Глава 7

Выводы

Во время выполнения лабораторной работы получены навыки работы с новыми сигналами и их гармониками, а именно с треугольными, прямоугольными и пилообразными сигналами. Также рассмотрено одно из наиболее важных явлений в цифровой обработке сигналов - биения.