

Лабораторная работа №2  
Гармоники

Кобыжев Александр

10 марта 2021 г.

# Оглавление

<b>1</b>	<b>Упражнение 2.1</b>	<b>4</b>
<b>2</b>	<b>Упражнение 2.2</b>	<b>5</b>
2.1	Написание класса и проверка сигнала . . . . .	5
2.2	Спектр звука . . . . .	6
<b>3</b>	<b>Упражнение 2.3</b>	<b>9</b>
<b>4</b>	<b>Упражнение 2.4</b>	<b>11</b>
4.1	Создание сигнала . . . . .	11
4.2	Нулевой компонент спектра . . . . .	12
4.3	Изменение нулевого компонента . . . . .	12
<b>5</b>	<b>Упражнение 2.5</b>	<b>13</b>
5.1	Создание функции . . . . .	13
5.2	Проверка работоспособности функции . . . . .	13
<b>6</b>	<b>Упражнение 2.6</b>	<b>15</b>
<b>7</b>	<b>Выводы</b>	<b>18</b>

## Список иллюстраций

2.1	Полученный пилообразный звук . . . . .	6
2.2	Спектр сегмента звука . . . . .	7
2.3	Сравнение пилообразных и прямоугольных сигналов . . . .	7
2.4	Сравнение пилообразных и треугольных сигналов . . . . .	8
3.1	Спектр сегмента звука . . . . .	9
4.1	Визуализация созданного сигнала . . . . .	11
4.2	Визуализация ускоренного звука . . . . .	12
5.1	Сравнение спектров . . . . .	14
6.1	Спектр сигнала . . . . .	15
6.2	Спектр сигнала . . . . .	16
6.3	Сравнение спектров . . . . .	17

# Листинги

2.1	Класс SawtoothSignal . . . . .	5
2.2	Визуализация пилообразного звука . . . . .	5
2.3	Прослушивание звука . . . . .	6
2.4	Спектр звука . . . . .	6
2.5	Сравнение пилообразных и прямоугольных сигналов . . . . .	7
2.6	Сравнение пилообразных и треугольных сигналов . . . . .	8
3.1	Создание прямоугольного сигнала . . . . .	9
3.2	Воспроизведение прямоугольного сигнала . . . . .	10
3.3	Создание и воспроизведение сигнала с пониженной частотой . . . . .	10
4.1	Создание треугольного сигнала . . . . .	11
4.2	Вывод нулевого компонента . . . . .	12
4.3	Смещение спектра и его визуализация . . . . .	12
5.1	Создание функции . . . . .	13
5.2	Создание сигнала и его воспроизведение . . . . .	13
5.3	Сравнение спектров . . . . .	13
5.4	Воспроизведение отфильтрованного звука . . . . .	14
6.1	Создание сигнала и визуализация его спектра . . . . .	15
6.2	Сравнение гармоник . . . . .	15
6.3	Сегмент звука . . . . .	16

# Глава 1

## Упражнение 2.1

В данном упражнении нас просят открыть `chap02.ipynb`, прочитать пояснения, а также запустить примеры.

# Глава 2

## Упражнение 2.2

### 2.1 Написание класса и проверка сигнала

Для данного упражнения нужно написать класс под названием `SawtoothSignal`, который, как сказано по заданию, должен расширять `Signal`, но в коде, который нам предлагается для изучения используется в качестве основы `Sinusoid`, поэтому я также решил написать класс на его основе.

```
1 class SawtoothSignal(thinkdsp.Sinusoid):
2     def evaluate(self, ts):
3         cycles = self.freq * ts + self.offset / PI2
4         frac, _ = np.modf(cycles)
5         ys = thinkdsp.normalize(thinkdsp.unbias(frac), self.amp)
6         return ys
```

Листинг 2.1: Класс `SawtoothSignal`

После написания класса удостоверимся, что конечный сигнал является пилообразным.

```
1 sawtooth = SawtoothSignal(100).make_wave(duration=0.25,
2     framerate=20000)
3 sawtooth.plot()
```

Листинг 2.2: Визуализация пилообразного звука

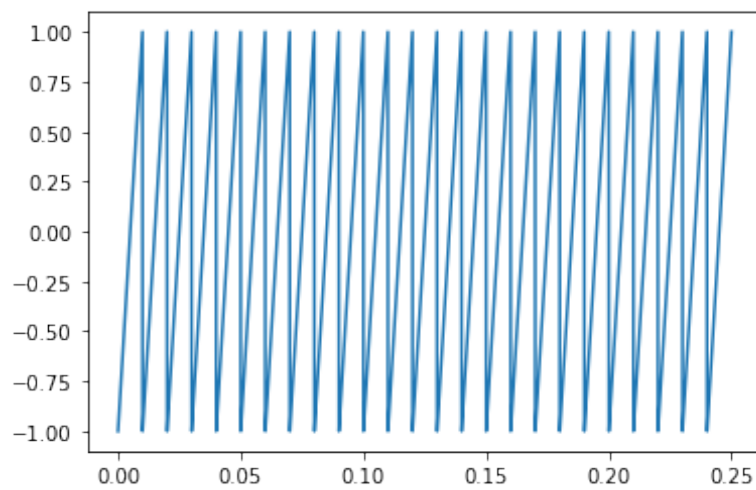


Рис. 2.1: Полученный пилообразный звук

Далее попробуем прослушать полученный звук. Он напоминает звук работающего трансформатора на электрической станции.

```
1 sawtooth.make_audio()
```

Листинг 2.3: Прослушивание звука

## 2.2 Спектр звука

Теперь рассмотрим спектр нашего пилообразного звука.

```
1 sawtooth.make_spectrum().plot()
```

Листинг 2.4: Спектр звука

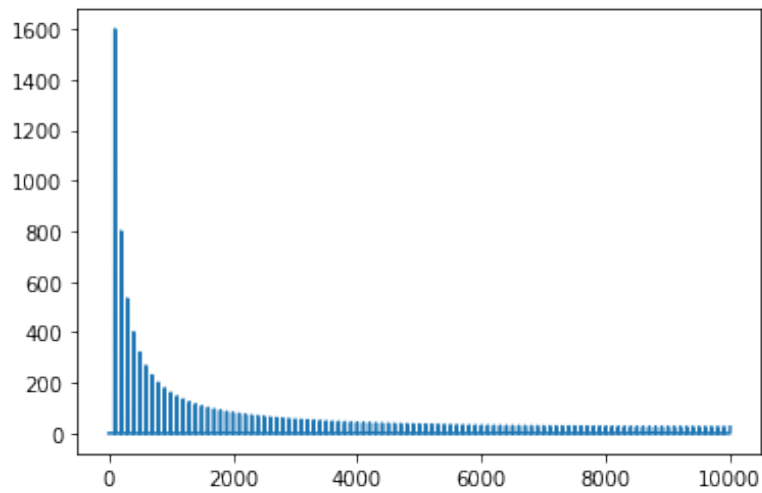


Рис. 2.2: Спектр сегмента звука

Рассмотрим соотношения гармонических структур наших пилообразных сигналов с прямоугольными и треугольными. Мы видим, что пилообразные гармоники спадают медленнее, чем прямоугольные.

```

1 sawtooth.make_spectrum().plot(color='red')
2 square = thinkdsp.SquareSignal(amp=0.5).make_wave(duration=0.25,
    framerate=20000)
3 square.make_spectrum().plot(color='blue')

```

Листинг 2.5: Сравнение пилообразных и прямоугольных сигналов

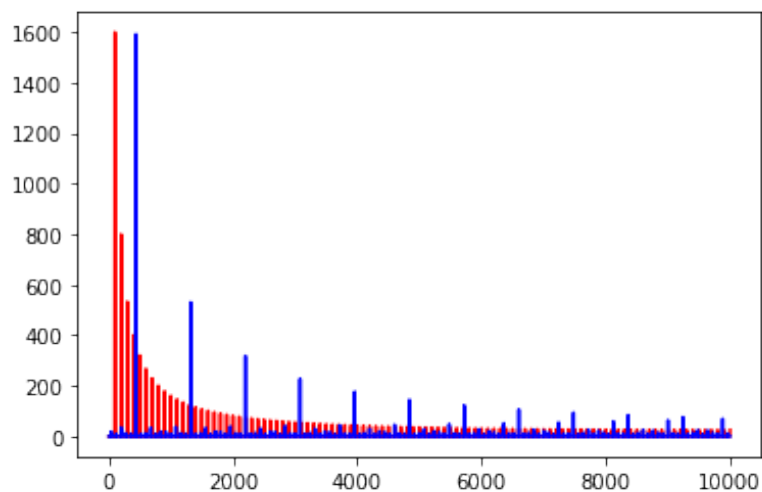


Рис. 2.3: Сравнение пилообразных и прямоугольных сигналов



В результате получается, что по сравнению с треугольной волной пилообразная форма уменьшается практически аналогично, но включает как четные, так и нечетные гармоники.

```
1 sawtooth.make_spectrum().plot(color='red')
2 triangle =
    thinkdsp.TriangleSignal(amp=0.79).make_wave(duration=0.25,
    framerate=20000)
3 triangle.make_spectrum().plot(color='blue')
```

Листинг 2.6: Сравнение пилообразных и треугольных сигналов

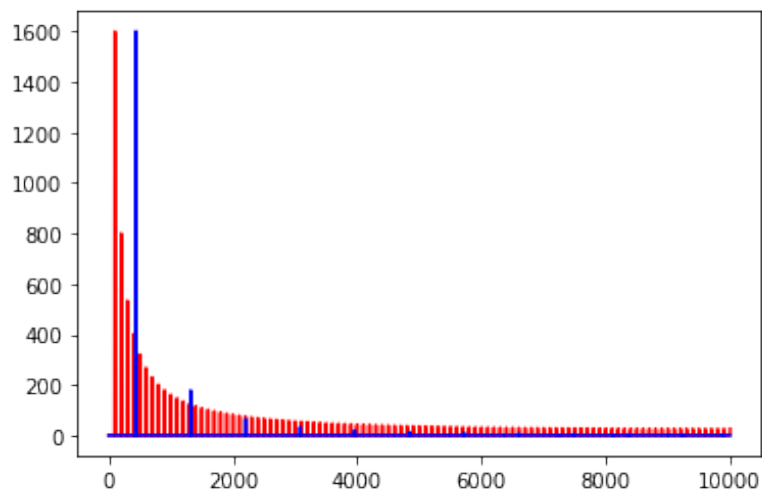


Рис. 2.4: Сравнение пилообразных и треугольных сигналов

## Глава 3

### Упражнение 2.3

Создадим прямоугольный сигнал 1100 Гц и вычислим его спектр.

```
1 square = thinkdsp.SquareSignal(1100).make_wave(duration=0.5,  
    framerate=10000)  
2 square.make_spectrum().plot()
```

Листинг 3.1: Создание прямоугольного сигнала

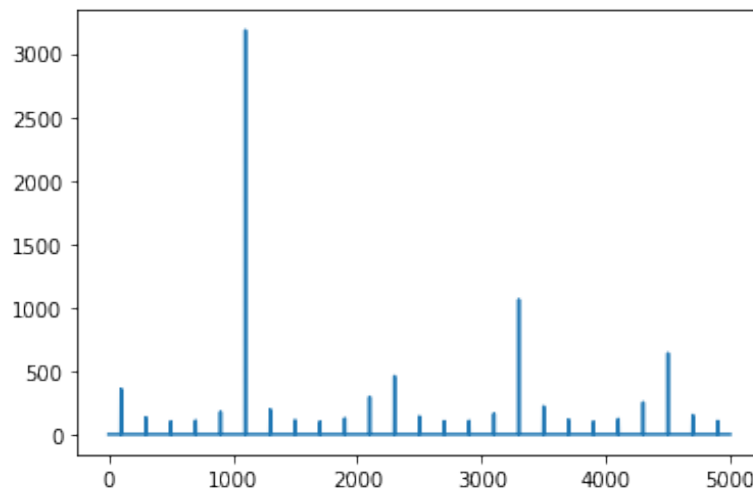


Рис. 3.1: Спектр сегмента звука

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.

```
1 square.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.

```
1 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.

```
1 triangle = thinkdsp.TriangleSignal(440).make_wave(duration=0.01)
2 triangle.plot()
```

Листинг 4.1: Создание треугольного сигнала

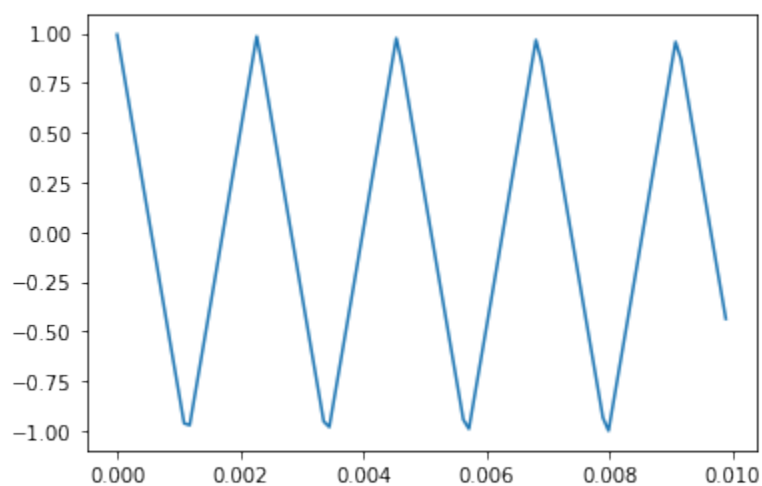


Рис. 4.1: Визуализация созданного сигнала

## 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.

```
1 spectrum = triangle.make_spectrum()  
2 spectrum.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.

```
1 spectrum.hs[0] = 100  
2 spectrum.make_wave().plot()
```

Листинг 4.3: Смещение спектра и его визуализация

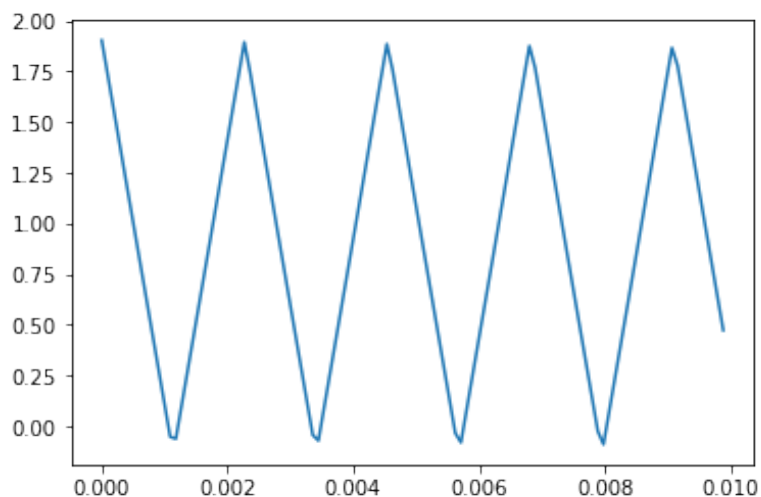


Рис. 4.2: Визуализация ускоренного звука

# Глава 5

## Упражнение 2.5

### 5.1 Создание функции

Напишем функцию, которая принимает `Spectrum` в качестве параметра и изменяет его делением каждого элемента `hs` на соответствующую частоту из `fs`.

```
1 def filter_spectrum(spectrum):  
2     spectrum.hs /= spectrum.fs  
3     spectrum.hs[0] = 0
```

Листинг 5.1: Создание функции

### 5.2 Проверка работоспособности функции

Создадим прямоугольный сигнал и прослушаем его.

```
1 wave = thinkdsp.SquareSignal(freq=440).make_wave(duration=0.5)  
2 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.

```
1 high = 10000  
2 spectrum = wave.make_spectrum()  
3 spectrum.plot(high=high, color='red')  
4 filter_spectrum(spectrum)  
5 spectrum.scale(440)
```

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

Листинг 5.3: Сравнение спектров

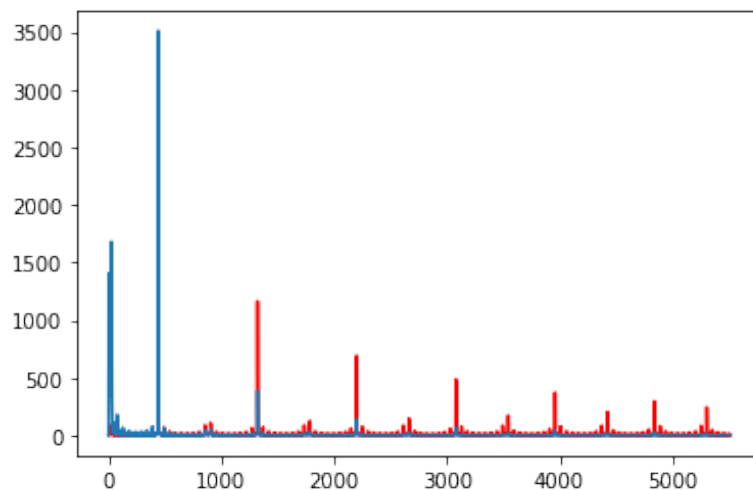


Рис. 5.1: Сравнение спектров

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

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

Листинг 5.4: Воспроизведение отфильтрованного звука

## Глава 6

### Упражнение 2.6

Создадим пилообразный сигнал и выведем его спектр.

```
1 freq = 500
2 signal = thinkdsp.SawtoothSignal(freq=freq)
3 wave = signal.make_wave(duration=0.5, framerate=30000)
4 wave.make_audio()
5 spectrum = wave.make_spectrum()
6 spectrum.plot()
```

Листинг 6.1: Создание сигнала и визуализация его спектра

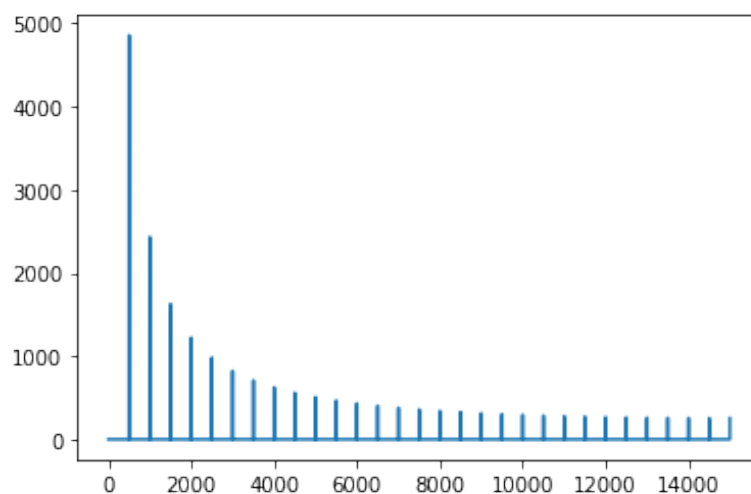


Рис. 6.1: Спектр сигнала

На рисунке видно, что гармоники уменьшаются как  $1/f^2$ .



```

1 spectrum.plot(color='red')
2 filter_spectrum(spectrum)
3 spectrum.scale(freq)
4 spectrum.plot()

```

Листинг 6.2: Сравнение гармоник

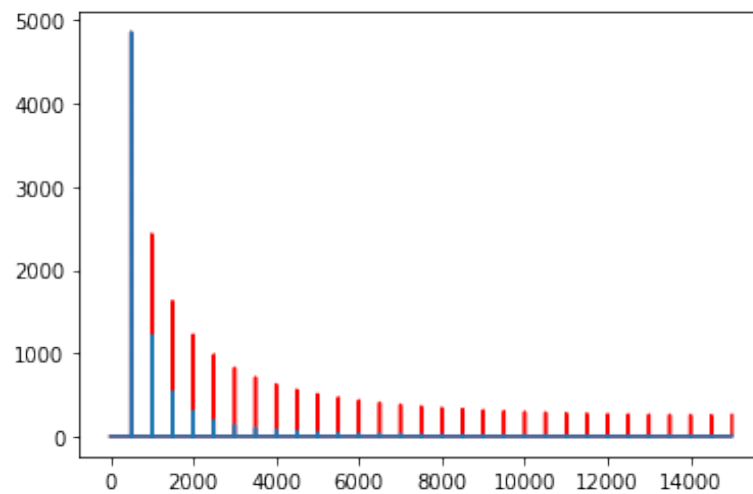


Рис. 6.2: Спектр сигнала

Теперь гармоника схожа с синусоидой.

```

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

```

Листинг 6.3: Сегмент звука

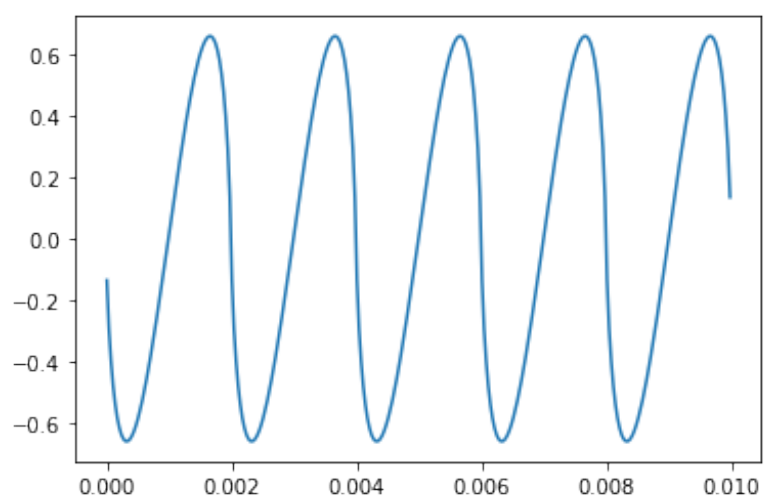


Рис. 6.3: Сравнение спектров

## 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.