Лабораторная работа №3
Апериодические сигналы

Крынский Павел

26 мая 2021 г.

Оглавление

Список иллюстраций

Листинги

Глава 1

Упражнение 3.1

1.1 Пример утечки

В данном упражнении нам нужно открыть `chap01.ipynb`, прочитать пояснения и запустить примеры. Поэтому здесь я изучил все примеры с комментариями и позапускал их.
Также нужно заменить окно Хэмингтона одним из окон, предосталяемых NumPy и посмотреть, как они влияют на утечку. Заменим окно Хэмминга.

```
for window_func in [np.kaiser, np.bartlett]:
    wave = signal.make_wave(duration)
    if window_func.__name__ == "kaiser":
        wave.ys *= window_func(len(wave.ys) , 8)
    else:
        wave.ys *= window_func(len(wave.ys))
    spectrum = wave.make_spectrum()
    spectrum.plot(high=880)
```

Листинг 1.1: Создание других окон

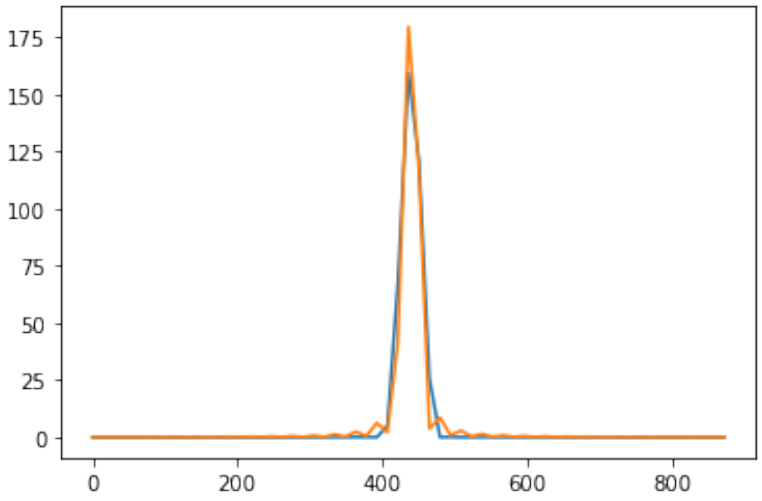

Рис. 1.1: Спектр созданных окон

Можно видеть, что окна нормально уменьшают утечку.

Глава 2

Упражнение 3.2

2.1 Написание класса SawtoothChirp

Напишем класс `SawtoothChirp`, который переопределяет `evaluate` для генерации пилообразного сигнала.

```
class SawtoothChirp(thinkdsp.Chirp):
    def evaluate(self , ts):
        freqs = np.linspace(self.start, self.end, len(ts) - 1)
        dts = np.diff(ts)
        dphis = (2*np.pi) * freqs * dts
        phases = np.cumsum(dphis)
        phases = np.insert(phases, 0 , 0)
        frac, _ = np.modf(phases)
        ys = self.amp * frac
        return ys
```

Листинг 2.1: Класс SawtoothChirp

2.2 Проверка работоспособности

```
sc = SawtoothChirp(start = 200 , end = 400)
wave = sc.make_wave(duration = 1 , framerate=10000)
sp = wave.make_spectrogram(256)
sp.plot()
```

Листинг 2.2: Проверка

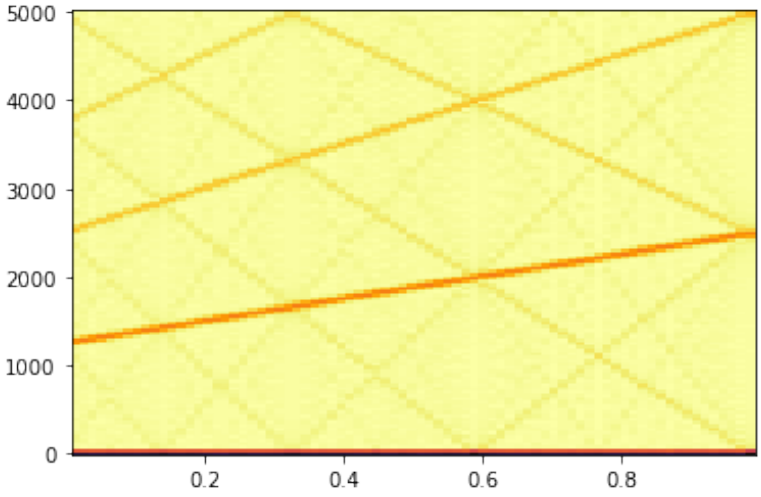

Рис. 2.1: Спекторграмма

Глава 3

Упражнение 3.3

Создаем сигнал, как просят в задании.

```
signal = SawtoothChirp(2500 , 3000)
wave = signal.make_wave(duration = 1 , framerate = 20000)
wave.make_audio()
```

Листинг 3.1: Создание сигнала

Теперь посмотрим на получившийся спектр.

```
cut_wave = wave.make_spectrum()
cut_wave.high_pass(10)
cut_wave.plot()
```

Листинг 3.2: Визуализация спектра

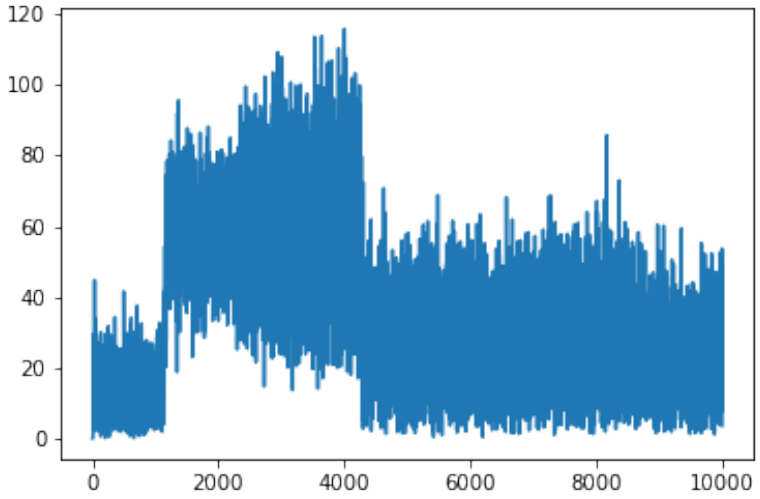

Рис. 3.1: Спектр созданного звука

Глава 4

Упражнение 3.4

Для данного задания я взял предложенный вариант файла.

```
wave = thinkdsp.read_wave('72475__rockwehrmann__glissup02.wav')
wave.make_audio()
```

Листинг 4.1: Загрузка

Теперь рассмотрим спектрограмму.

```
wave.make_spectrogram(512).plot(high=5000)
```

Листинг 4.2: Спектр

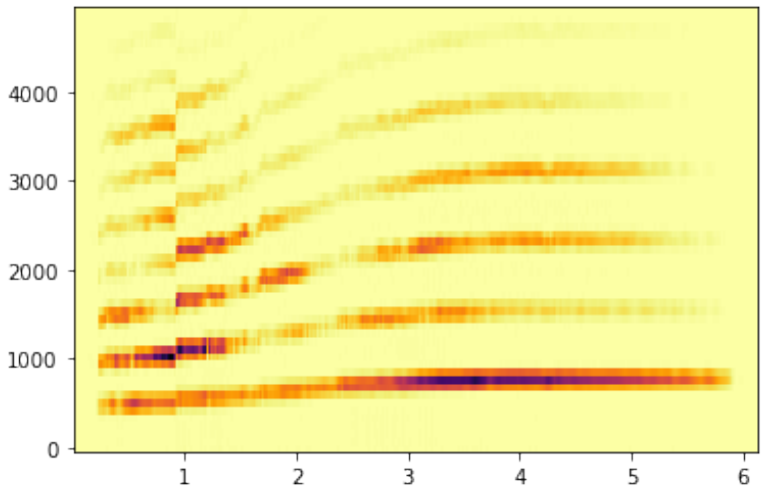

Рис. 4.1: Спектр

Глава 5

Упражнение 3.5

5.1 Создание класса

Написанный класс представляет собой тромбоноподобный сигнал с переменной частотой.

```
class TromboneGliss(thinkdsp.Chirp):
    def _evaluate(self, ts):
        l1, l2 = 1.0 / self.start, 1.0 / self.end
        lengths = np.linspace(l1, l2, len(ts)-1)
        freqs = 1 / lengths
        return self._evaluate(ts, freqs)
```

Листинг 5.1: Создание класса

5.2 Создание звука и его спектрограмма

Создадим первую часть звука от C3 до F3, где C3 - 262 Гц, а F3 - 349 Гц.

```
class TromboneGliss(thinkdsp.Chirp):
    def _evaluate(self, ts):
        l1, l2 = 1.0 / self.start, 1.0 / self.end
        lengths = np.linspace(l1, l2, len(ts) - 1)
        freqs = 1 / lengths
        return self._evaluate(ts,freqs)

signal = TromboneGliss(262,349)
waves = signal.make_wave(duration=1)
waves.apodize()
```

```
waves.make_audio()
```

Листинг 5.2: Создание первой части звука

Теперь создадим вторую часть звука от F3 до C3.

```
signal = TromboneGliss(349,262)
waves2 = signal.make_wave(duration=1)
waves2.apodize()
waves2.make_audio()
```

Листинг 5.3: Создание второй части звука

Затем соединим обе части в один полноценный звук.

```
wave = waves | waves2
wave.make_audio()
```

Листинг 5.4: Соединение двух частей звука

Теперь сделаем спектрограмму и расммотрим её.

```
sp = wave.make_spectrogram(1024)
sp.plot(high=1000)
```

Листинг 5.5: Спектрограмма звука

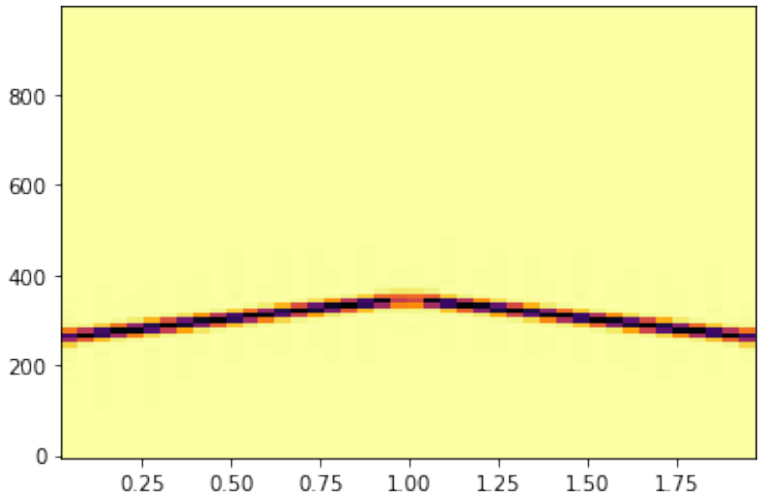

Рис. 5.1: Спектрограмма глиссандо на трамбоне

Глава 6

Упражнение 3.6

В интернете я нашёл звуки гласных.

```
wave = thinkdsp.read_wave('87778__marcgascon7__vocals.wav')
wave.make_audio()
```

Листинг 6.1: Загрузка и прослушивание звука

Посмотрим на них.

```
sg = wave.segment(start = 0 , duration = 7)
sg.plot()
```

Листинг 6.2: Участок

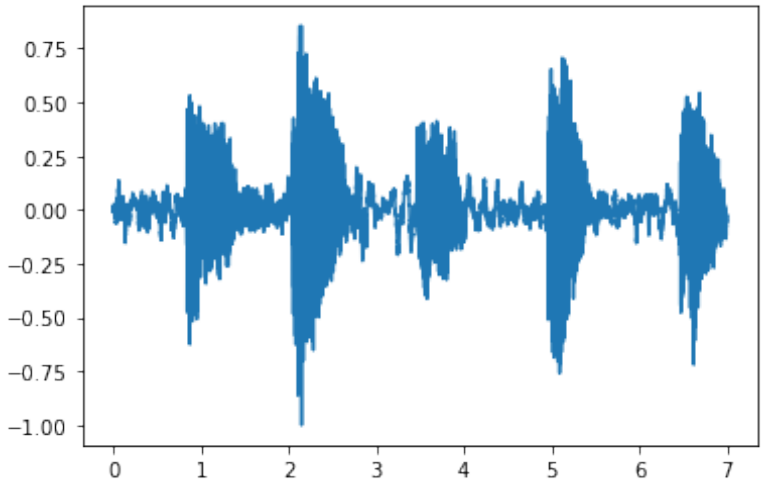

Рис. 6.1: Участок

```
wave.make_spectrogram(1024).plot(high=1000)
```

Листинг 6.3: Спектограмма

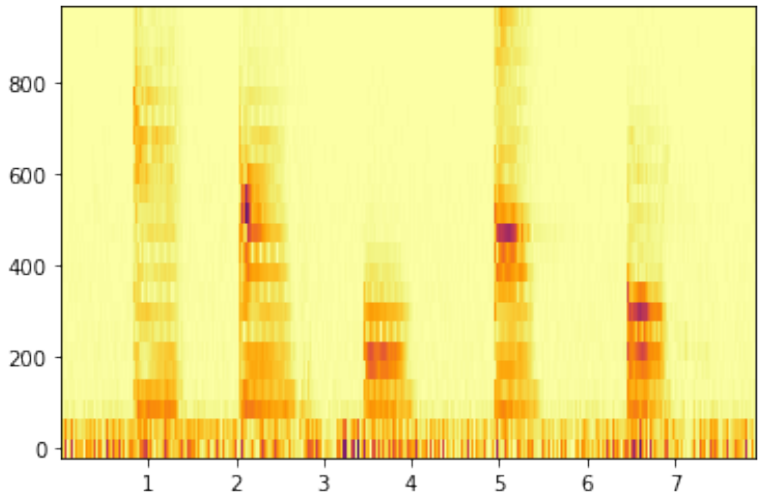

Рис. 6.2: Спектограмма

Нам важно то, что некоторые из участков темнее, а некоторые светлее (в рамках одного столбца), что соответствует спектрам соответствующих гласных.

Глава 7

Выводы

Во время выполнения лабораторной работы получены навыки работы с апериодическими сигналами, частотные компоненты которых изменяются во времени, то есть практически все звуковые сигналы. Также рассмотрены спектрограммы - распространённый способ визуализации апериодических сигналов.