Лабораторная работа 3

Чернышев Ярослав

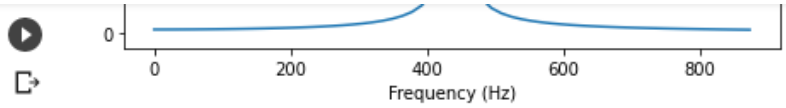
30 мая 2021 г.

Оглавление

Глава 1

Задание 3.1

В данном задании от меня требуется изучить готовые примеры из chap03.ipynb, а также ознакомиться с результатами работы с ним.



The following figure shows the effect of 4 different windows.

```
[5]  for window_func in [np.bartlett, np.blackman, np.hamming, np.hanning]:
         wave = signal.make_wave(duration)
         wave.ys *= window_func(len(wave.ys))

         spectrum = wave.make_spectrum()
         spectrum.plot(high=880, label=window_func.__name__)

     decorate(xlabel='Frequency (Hz)')
```

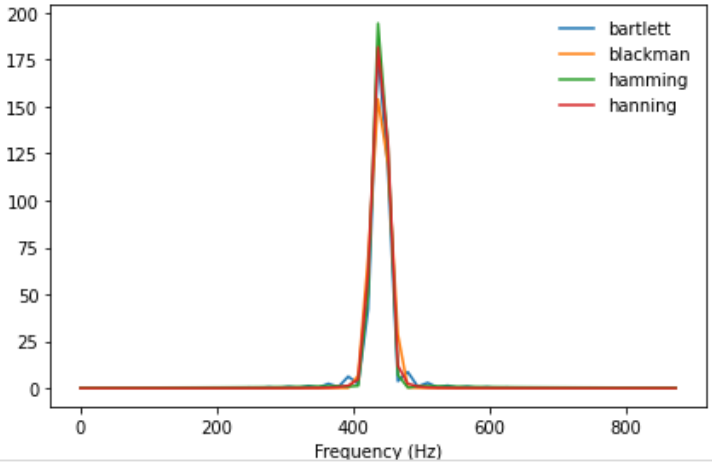


Рис. 1.1: Фрагмент работы

Глава 2

Задание 3.2

В этом задании требуется создать пилообразные Chirp и поэкспериментировать с ними.

Первым делом, напишем соответствующий класс:

```
  class SawtoothChirp(Chirp):
    def evaluate(self, ts):
        freqs = np.linspace(self.start, self.end, len(ts) -
    1)
        dts = np.diff(ts)
        dphis = PI2 * freqs * dts
        phases = np.cumsum(dphis)
        phases = np.insert(phases, 0, 0)
        cycles = phases / PI2
        frac, _ = np.modf(cycles)
        ys = self.amp * frac
        return ys

saw = SawtoothChirp(start=220, end=440)
wave = saw.make_wave(duration=1, framerate=11025)
wave.segment(start=0, duration=0.02).plot()
decorate(xlabel='Time (s)')

```

Листинг 2.1: class Sawtooth

С данными настройками получили следующий результат:

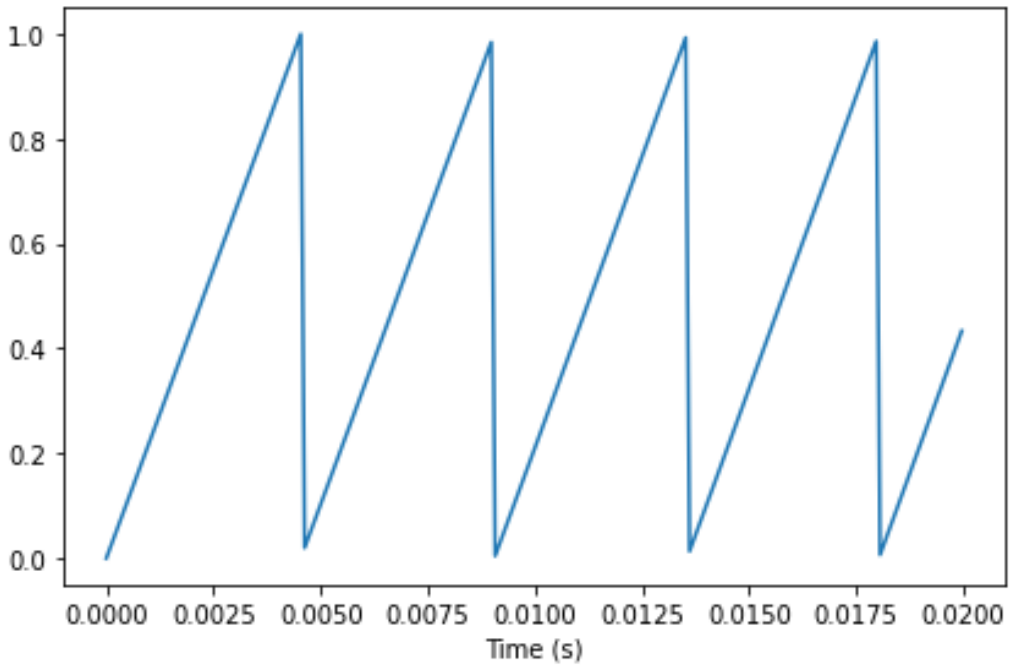


Рис. 2.1: Результат

Ниже приведён сегмент с 0.98 по 1:

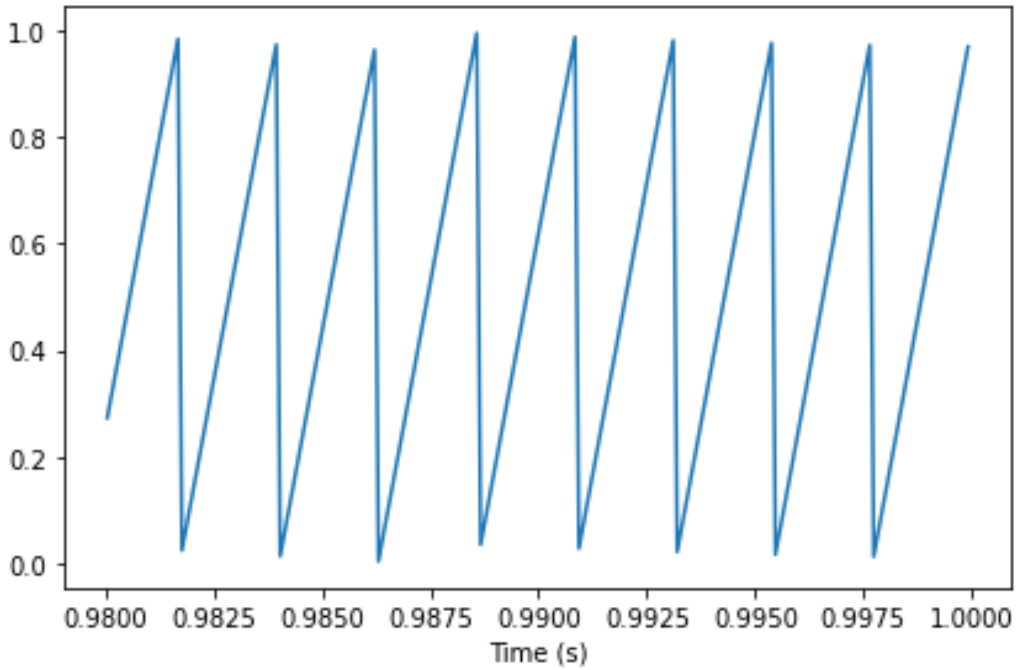


Рис. 2.2: Результат

Глава 3

Задание 3.3

В этом задании требуется создать пилообразный Chirp с заданной частотой и вывести график.

```
  signal = SawtoothChirp(start=2500, end=3000)
wave = signal.make_wave(duration=1, framerate=20000)
wave.segment(start=0.9, duration=0.02).plot()
decorate(xlabel='Time')

```

Листинг 3.1: class Sawtooth

Получили следующую волну:

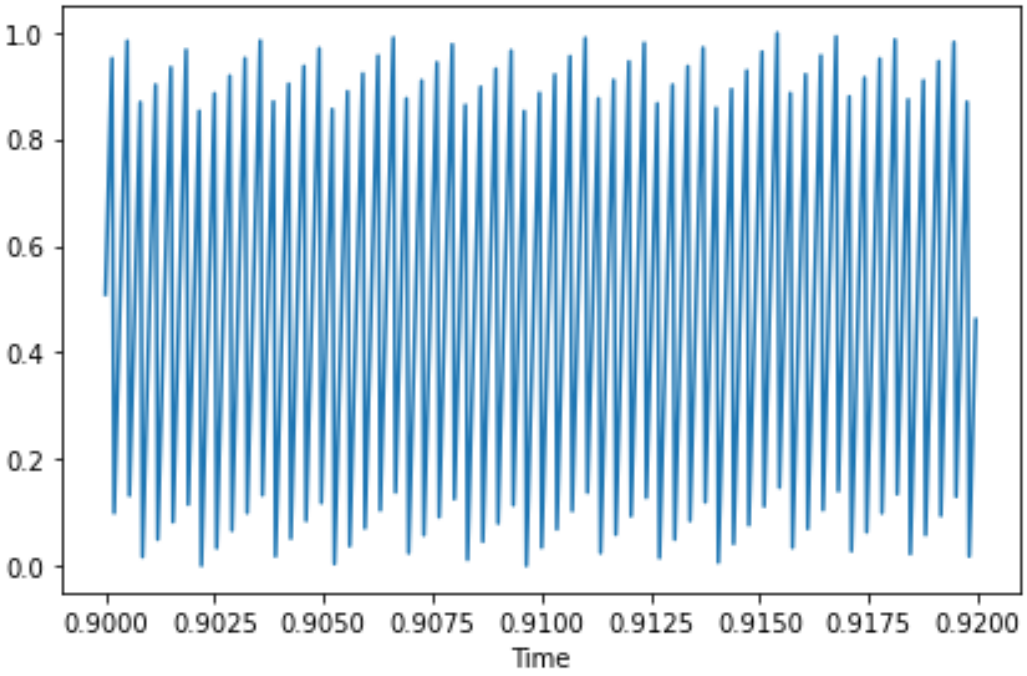


Рис. 3.1: Искомая волна

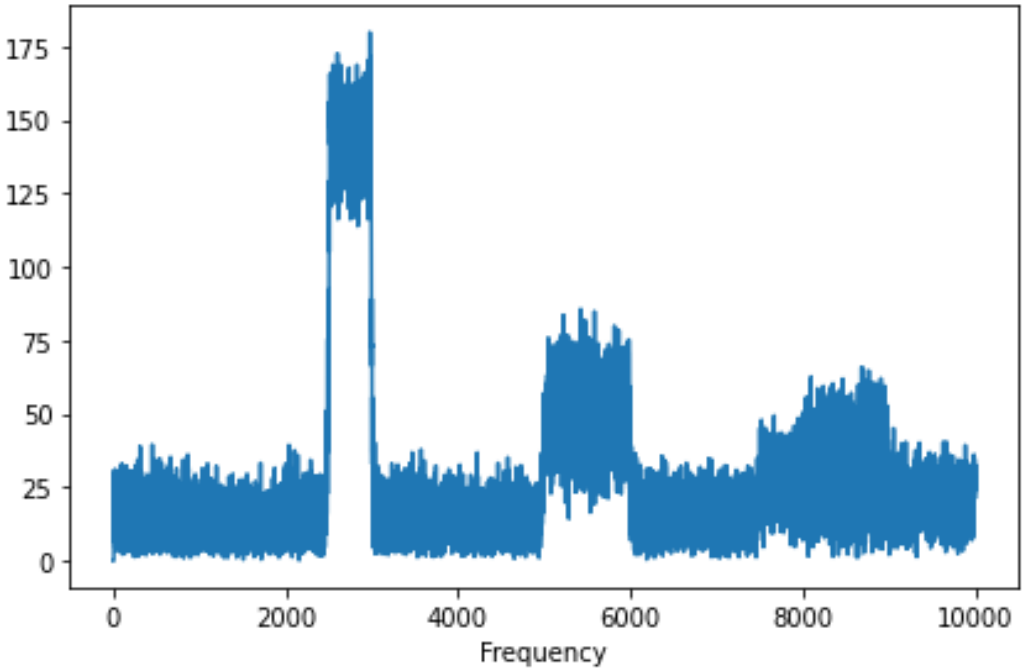


Рис. 3.2: Её спектр

Я предполагал уменьшение амплитуды на кратных основным частотах.

Глава 4

Задание 3.4

Требуется поэксперементровать с глиссандо. Загружаем подходящий аудиофайл:

```
from thinkdsp import read_wave

wave = read_wave('rhapblue11924.wav')
segment = wave.segment(start=1.35, duration=0.45)
segment.plot()
```

Листинг 4.1: Загрузка аудио

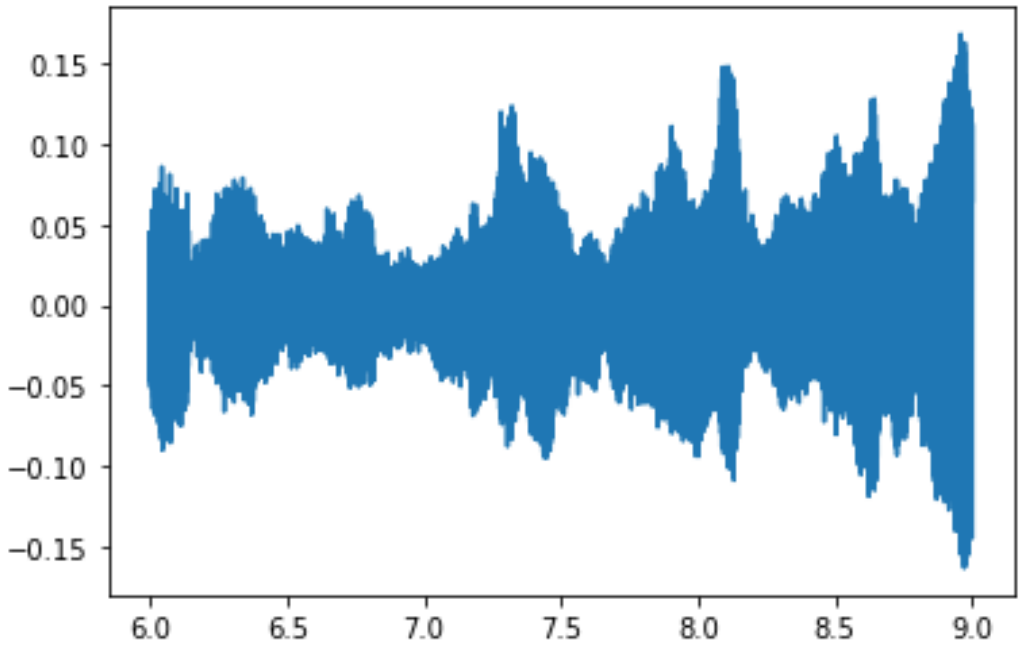


Рис. 4.1: Искомая волна

И строим спектрограмму:

```
spectrogram = segment.make_spectrogram(256)
print('Time resolution (s)', spectrogram.time_res)
print('Frequency resolution (Hz)', spectrogram.freq_res)
spectrogram.plot(high=3000)
decorate(xlabel='Time(s)', ylabel='Frequency (Hz)')
```

Листинг 4.2: stretch

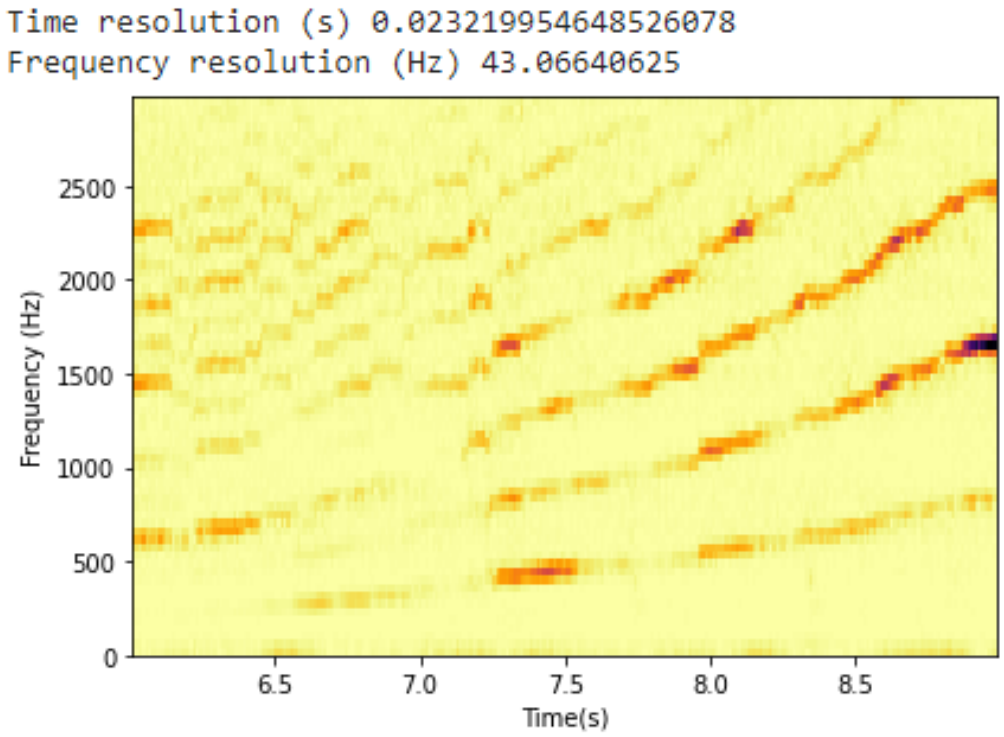


Рис. 4.2: Искомая спектрограмма

Глава 5

Задание 3.5

Требуется написать поэксперементировать с глиссандо тромбона. Соответствующий класс:

```
class MyTromboneGliss(Chirp):
    def _get_frequency(self, ts):
        S, E = self.start, self.end

        return E / (1 + ts * (E/S - 1))

    def evaluate(self, ts):
        l_C3, l_F3 = 2, 1

        freqs = self._get_frequency(1 - 2 * np.abs(ts[:-1] -
    0.5))

        dts = np.diff(ts)
        dphis = PI2 * freqs * dts

        phases = np.cumsum(dphis)
        phases = np.insert(phases, 0, 0)

        ys = self.amp * np.cos(phases)
        return ys

signal = MyTromboneGliss(start=262, end=349)
wave = signal.make_wave(duration=1, framerate=11025)
wave.segment(start=0, duration=0.04).plot()
```

Листинг 5.1: TromboneGliss

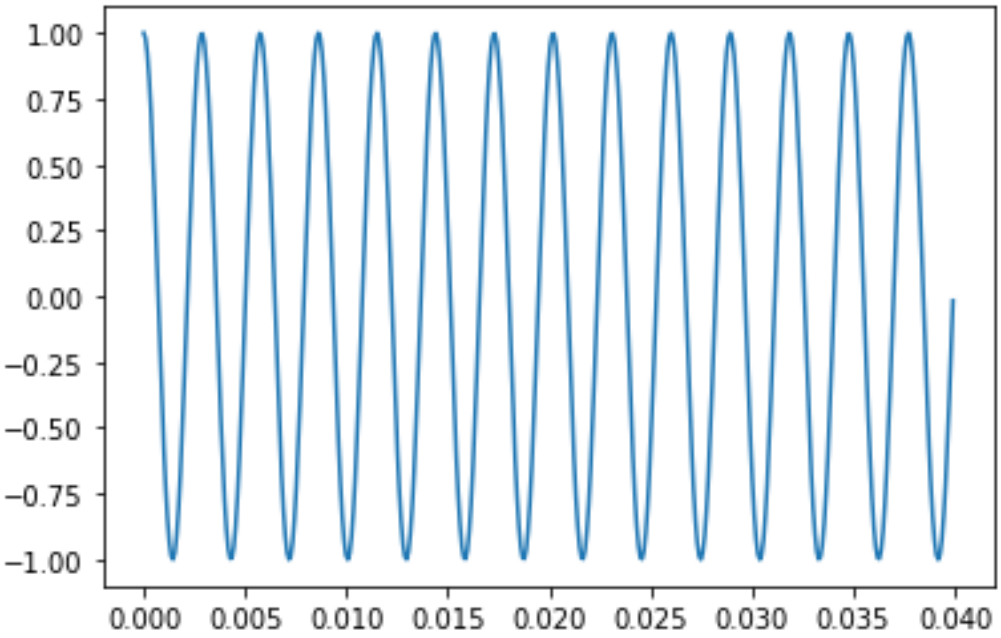


Рис. 5.1: Волна

Далее, аналогично, получаем спектр и спектрограмму:

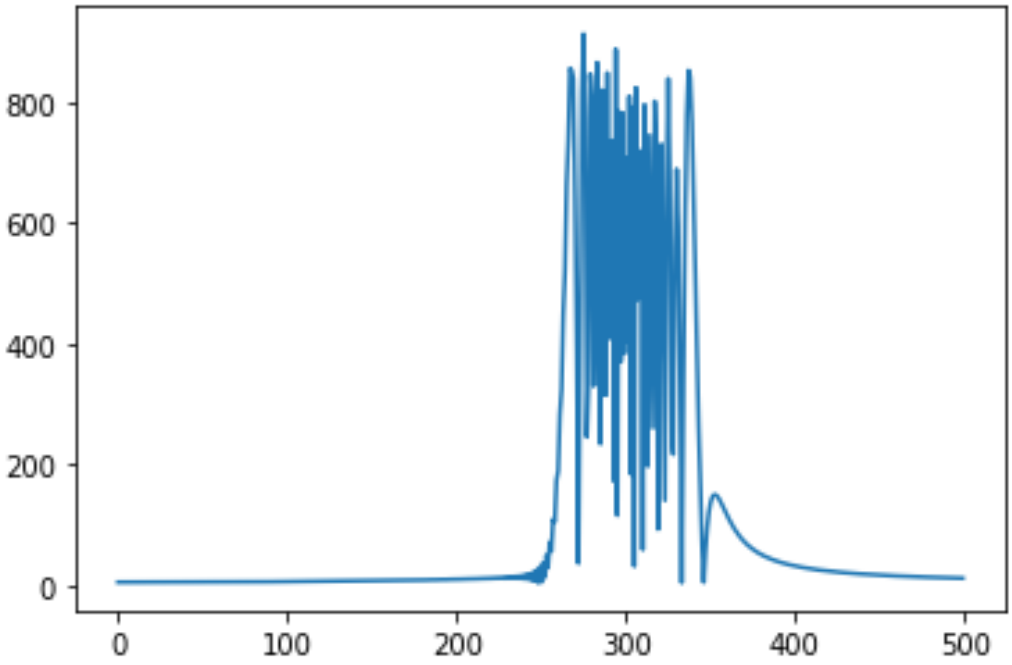


Рис. 5.2: Спектр

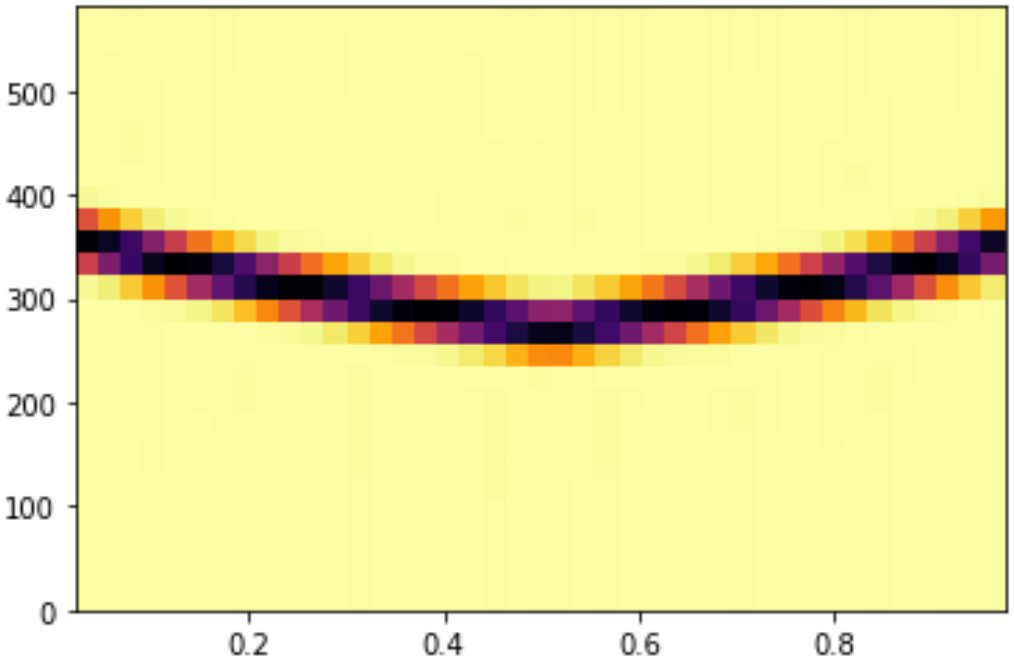


Рис. 5.3: Спектрограмма

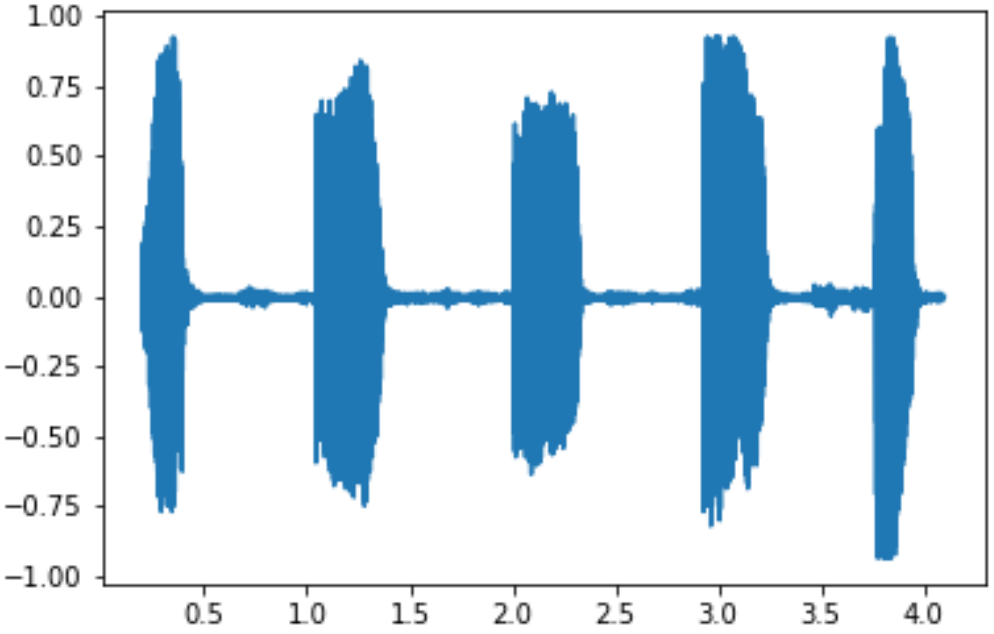
Глава 6

Задание 3.6

Требуется с помощью спектрограммы определить гласные звуки.

```
wave = read_wave('523055__cbelloso__vocales-vowels-man-woman-kid-girl.wav')
segment = wave.segment(start=0.2, duration=3.9)
segment.plot()
```

Листинг 6.1: Поиск гласных




```
segment.make_spectrogram(256).plot()
```

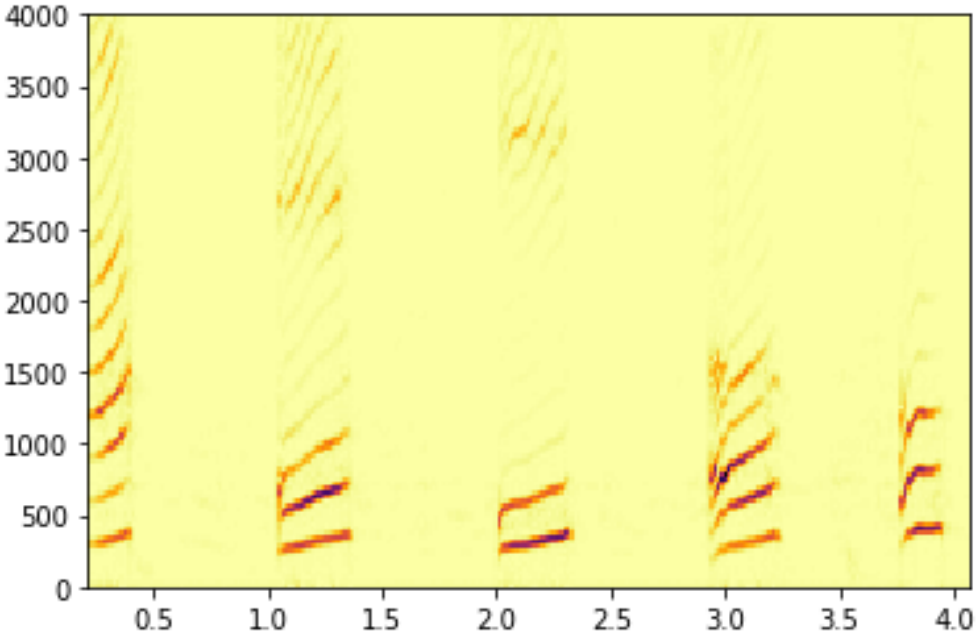


Рис. 6.1: Волны и спектрограмма

На двух графиках можно видеть, что с пощью спектрограммы, действительно, можно определять гласные.