ThinkDSP. Лабораторная 3. Апериодические сигналы.

Шерепа Никита

28 апреля 2021 г.

Содержание

Список иллюстраций

Листинги

1 Упражнение 3.1

1. **Задание**

   Запустите и прослушайте примеры из блокнота *chap03.ipynb*. В примере с утечкой замените окно Хэмминга одним из окон, предоставляемых *NumPy*, и посмотрите, как они влияют на утечку.

2. **Ход работы**

   Для начала создадим утечку

```
from thinkdsp import decorate
from thinkdsp import SinSignal

signal = SinSignal(freq=440)
duration = signal.period * 30.25
wave = signal.make_wave(duration)
spectrum = wave.make_spectrum()

spectrum.plot(high=880)
decorate(xlabel='Frequency (Hz)')
```

Листинг 1: Пример утечки

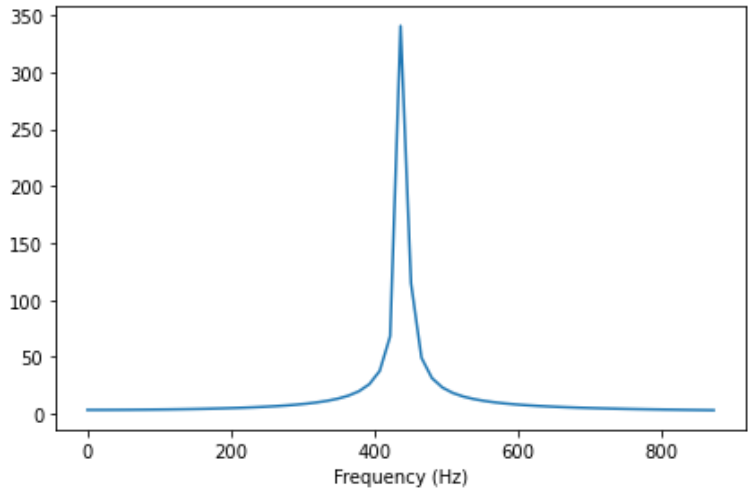

Рис. 1: Визуализация утечки

Теперь заменим окно Хэмминга на другие 4 окна из *NumPy*

```
import numpy as np
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)')
```

Листинг 2: Создание 4х новых окон

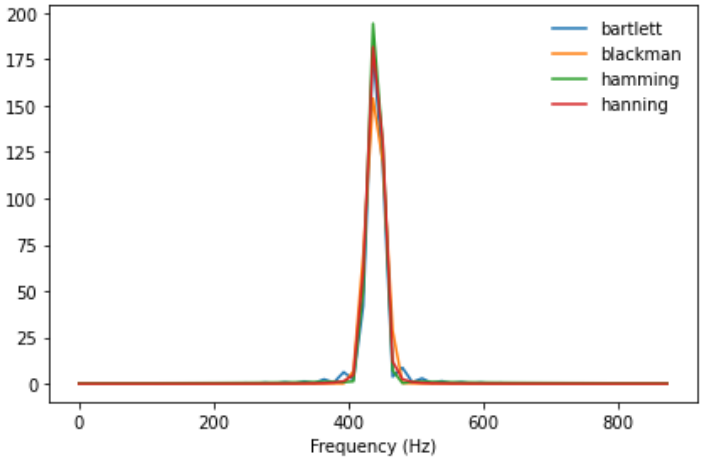

Рис. 2: Сравнение 4х новых окон с окном Хэмминга

Из графика видно, что все 4 окна успешно справляются с уменьшением утечки.

2 Упражнение 3.2

1. **Задание**

   Напишите класс, называемый *SawtoothChirp*, расширяющий *Chirp* и переопределяющий *evaluate* для генерации пилообразного сигнала с линейно увеличивающейся (или уменьшающейся) частотой. Подсказка: надо совместить функции *evaluate* из *Chirp* и *SawtoothSignal* Нарисуйте эскиз спектрограммы этого сигнала, а затем распечатайте ее. Эффект биений должен быть очевиден, а если сигнал внимательно прослушать, то биения можно и услышать.

2. **Ход работы**

   Напишем класс *SawtoothChirp*

```
from thinkdsp import Chirp
from thinkdsp import normalize, unbias

PI2 = 2 * np.pi

class SawtoothChirp(Chirp):

def evaluate(self, ts):
freqs = np.linspace(self.start, self.end, len(ts))
dts = np.diff(ts, prepend=0)
dphis = PI2 * freqs * dts
phases = np.cumsum(dphis)
cycles = phases / PI2
frac, _ = np.modf(cycles)
ys =  normalize(unbias(frac), self.amp)
return ys
```

Листинг 3: Класс *SawtoothChirp*

   Теперь попробуем с помощью него сгенерировать пилообразный сигнал

```
signal = SawtoothChirp(start=10, end=1000)
wave = signal.make_wave(duration=1, framerate=10000)
wave.apodize()
wave.make_audio()
```

Листинг 4: Генерация пилообразного сигнала

Получился возрастающий электронный звук, похожий на какой-нибудь электрический эффект из советского кино.

Теперь построим спектрограмму звука

```
sp = wave.make_spectrogram(512)
sp.plot()
decorate(xlabel='Time (s)', ylabel='Frequency (Hz)')
```

Листинг 5: Построение спектрограммы звука

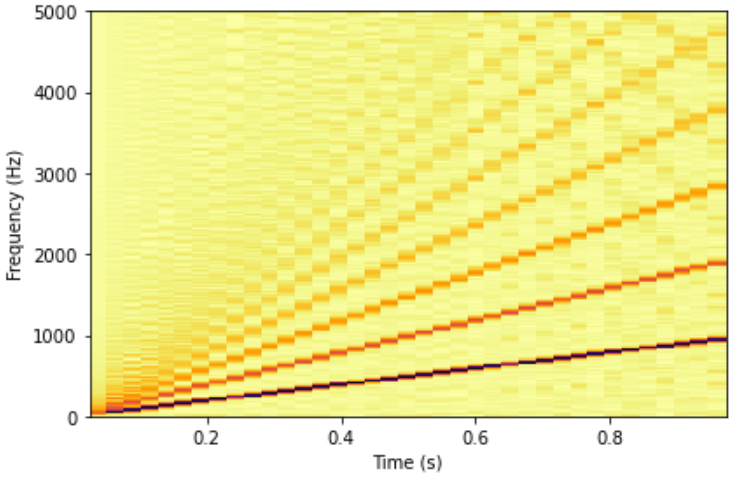

Рис. 3: Спектрограмма звука

Как видим, идет четкое повышение частоты со временем, как и в записи.

3 Упражнение 3.3

1. **Задание**

   Создайте пилообразный чирп, меняющийся от 2500 до 3000 Гц, и на его основе сгенерируйте сигнал длительностью 1 с и частотой кадров 20 кГц. Нарисуйте, каким примерно будет *Spectrum*. Затем распечатайте *Spectrum* и посмотрите, правы ли вы.

2. **Ход работы**

   Создадим пилообразный чирп и на его основе сгенерируем сигнал

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

Листинг 6: Чирп и сигнал

   Получился очень острый нарастающий звук, который можно было бы использовать в каком нибудь старом советском фильме про космос.

   Теперь посмотрим на его спектр

```
wave.make_spectrum().plot()
decorate(xlabel='Frequency (Hz)')
```

Листинг 7: Создание спектра

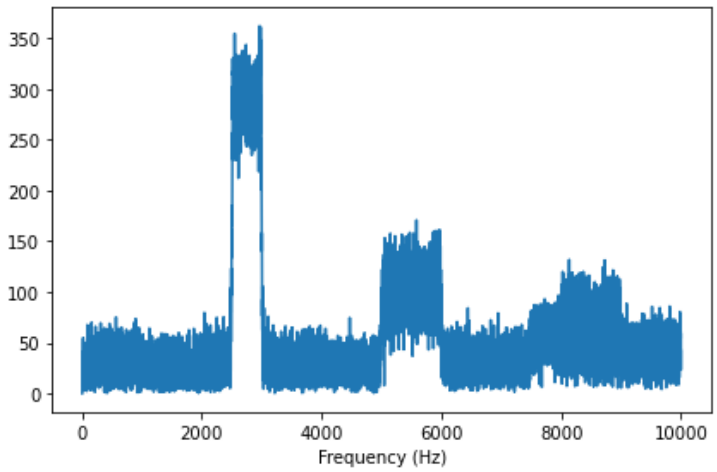

Рис. 4: Спектр

Видно, что частота отчеливо меняется, что и слышно в записи звука.

4 Упражнение 3.4

1. **Задание**

   В музыкальной терминологии глиссандо - это нота, меняющаяся от одной высоты до другой, то есть своеобразный чирп. Найдите или запишите звук глиссандо и распечатайте спектрограмму первых нескольких секунд.

2. **Ход работы**

   В выбрал фрагмент из одного из выступлений Витаса.

```
wave = read_wave('res/vitas.wav')
wave.make_audio()

wave.make_spectrogram(512).plot(high=5000)
decorate(xlabel='Time (s)', ylabel='Frequency (Hz)')
```

Листинг 8: Воспроизведение глиссандо и построение спектрограммы

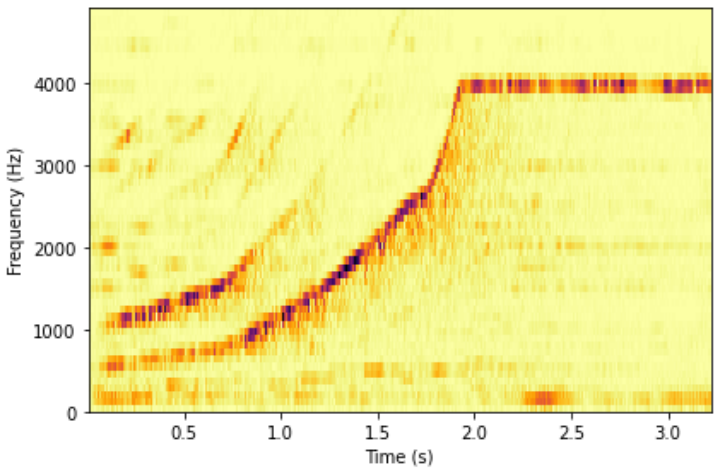

Рис. 5: Спектрограмма глиссандо

Четко видно резкое повышение частоты.

5 Упражнение 3.5

1. **Задание**

   Тромбонист играет глиссандо, непрерывно дуя в мундштук и двигая кулису тромбона. При этом общая длина турбы меняется, а играемая нота обратно пропорциональна этой длине.

   Если предположить, что музыкант двигает кулису с постоянной скоростью, как будет меняться во времени частота?

   Напишите класс, называемый *TromboneGliss*, расширяющий *Chirp* и предоставляющий *evaluate*. Создайте сигнал, имитирующий глиссандо на тромбоне от C3 до F3, и обратно до C3. C3 - 262 Гц F3 - 349 Гц

   Напечатайте спектрограмму полученного сигнала. На что похоже глиссандо на тромбоне - на линейный или же экспоненциальный чирп?

2. **Ход работы**

   Напишем класс *TromboneGliss*

```
        class TromboneGliss(Chirp):

        def evaluate(self, ts):
        l1, l2 = 1.0 / self.start, 1.0 / self.end
        lengths = np.linspace(l1, l2, len(ts))
        freqs = 1 / lengths

        dts = np.diff(ts, prepend=0)
        dphis = PI2 * freqs * dts
        phases = np.cumsum(dphis)
        ys = self.amp * np.cos(phases)
        return ys
```

Листинг 9: Класс *TromboneGliss*

   Теперь создадим первую, убывающую, часть звука

```
        low = 262
        high = 349
        signal = TromboneGliss(high, low)
        wave1 = signal.make_wave(duration=1)
        wave1.apodize()
```

```
wave1.make_audio()
```

Листинг 10: Создание убывающей части звука

Теперь создадим вторую, возрастающую, часть звука

```
signal = TromboneGliss(low, high)
wave2 = signal.make_wave(duration=1)
wave2.apodize()
wave2.make_audio()
```

Листинг 11: Создание возрастающей части звука

Теперь соединим две части

```
wave = wave1 | wave2
wave.make_audio()
```

Листинг 12: Соединение двух частей

И построим спектрограмму получившегося звука

```
sp = wave.make_spectrogram(1024)
sp.plot(high=1000)
decorate(xlabel='Time (s)', ylabel='Frequency (Hz)')
```

Листинг 13: Создание спектрограммы получившегося звука

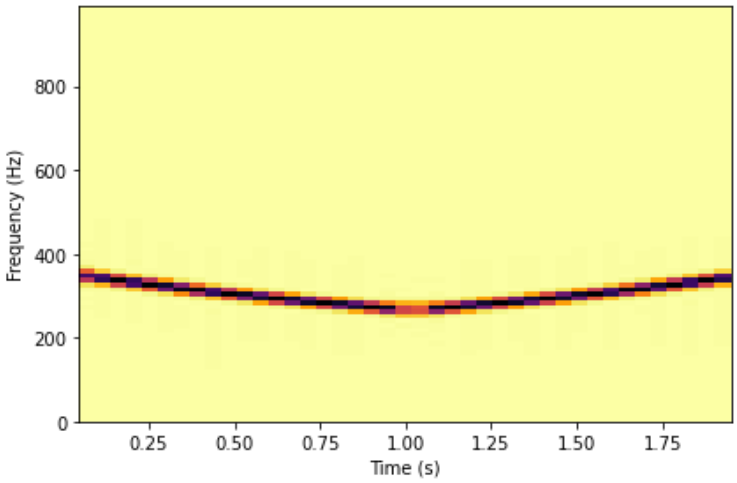

Рис. 6: Спектрограмма получившегося звука

Четко видно убывание и возрастание. Также получившийся сигнал похож на линейный чирп.

6 Упражнение 3.6

1. **Задание**

   Сделайте или найдите запись серии гласных звуков и посмотрите на спектрограмму. Сможете ли вы различить разные глассные?

2. **Ход работы**

   Я взял звуки из како-то детской передачи по изучению гласных. Построим спектрограмму.

```
wave = read_wave('res/vowels.wav')
wave.make_audio()

wave.make_spectrogram(1024).plot(high=1000)
decorate(xlabel='Time (s)', ylabel='Frequency (Hz)')
```

Листинг 14: Построение спектрограммы гласных звуков

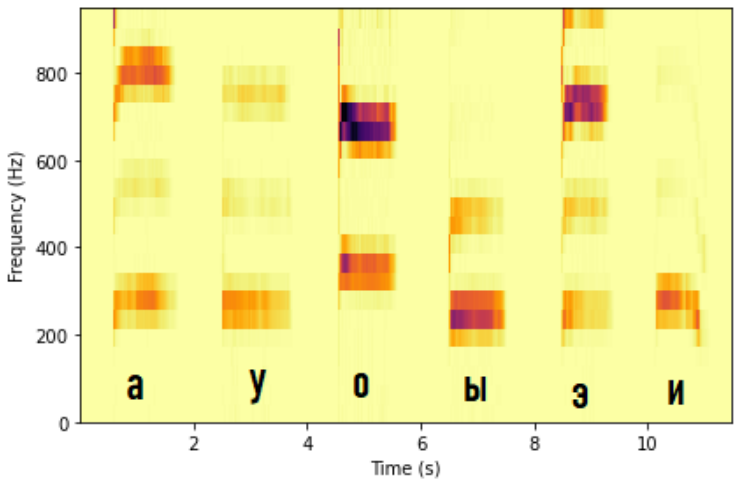

Рис. 7: Спектрограмма гласных звуков

Видно, что участки некоторых глассных темнее, а некоторых - светлее.

Посмотрим на спектры каждой гласной.

```
high = 1000

segment = wave.segment(start=0, duration=2)
```

```
4           segment.make_spectrum().plot(high=high)
```

Листинг 15: Спектр буквы а

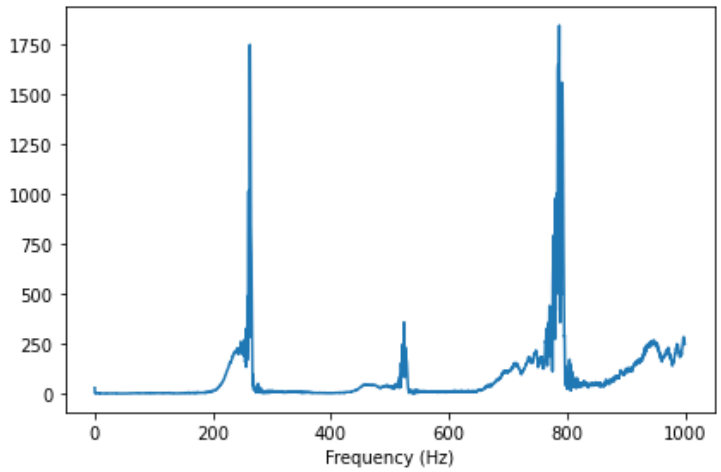

Рис. 8: Спектр буквы а

```
1           segment = wave.segment(start=2, duration=2)
2           segment.make_spectrum().plot(high=high)
3           decorate(xlabel='Frequency (Hz)')
```

Листинг 16: Спектр буквы у

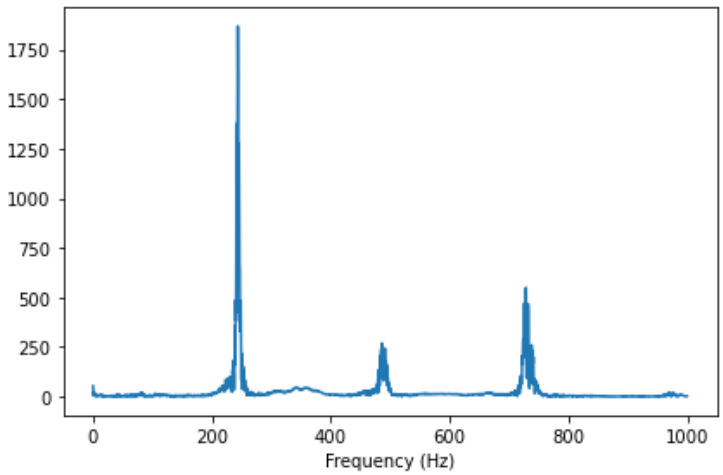

Рис. 9: Спектр буквы у

```
segment = wave.segment(start=4, duration=2)
segment.make_spectrum().plot(high=high)
decorate(xlabel='Frequency (Hz)')
```

Листинг 17: Спектр буквы о

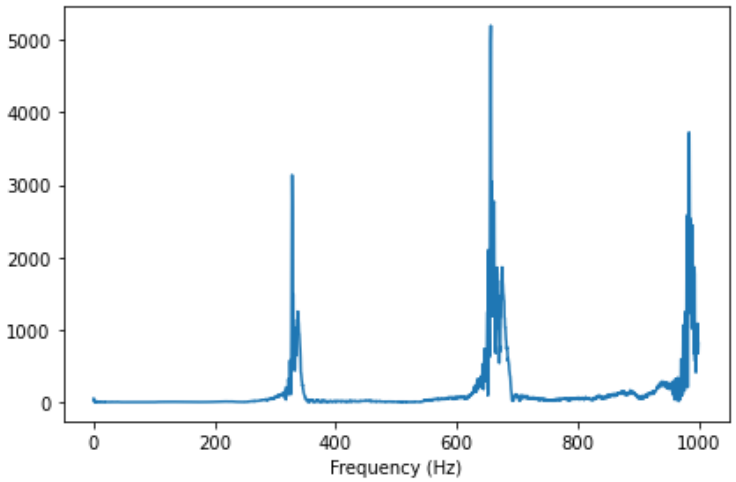

Рис. 10: Спектр буквы о

```
segment = wave.segment(start=6, duration=2)
segment.make_spectrum().plot(high=high)
decorate(xlabel='Frequency (Hz)')
```

Листинг 18: Спектр буквы ы

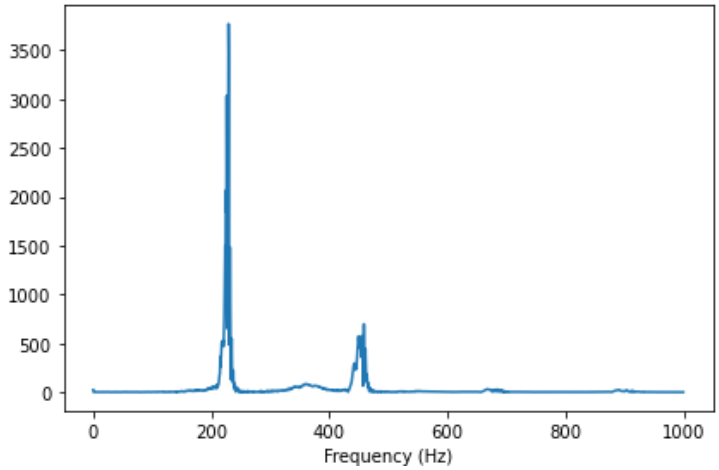

Рис. 11: Спектр буквы ы

```
segment = wave.segment(start=8, duration=2)
segment.make_spectrum().plot(high=high)
decorate(xlabel='Frequency (Hz)')
```

Листинг 19: Спектр буквы э

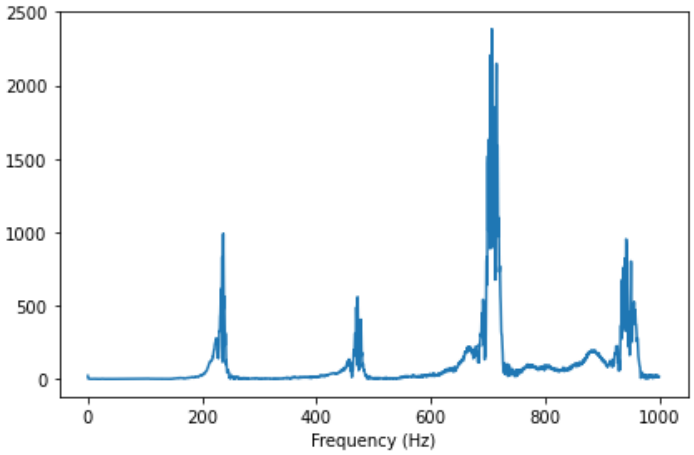

Рис. 12: Спектр буквы э

```
segment = wave.segment(start=10, duration=2)
segment.make_spectrum().plot(high=high)
decorate(xlabel='Frequency (Hz)')
```

Листинг 20: Спектр буквы и

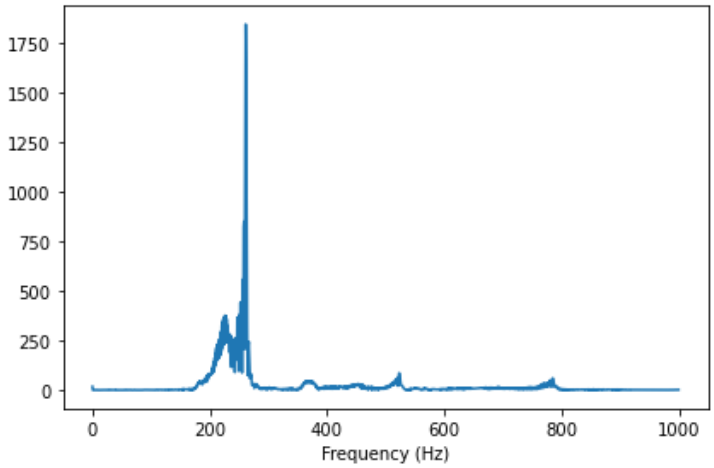

Рис. 13: Спектр буквы и

Из графиков видно, что спектр каждой гласной имеет разную высоту. Чем выше спектр, тем темнее спектрограмма.

7 Вывод

В результате выполнения лабораторной работы получены навыки работы с апериодическими сигналами - сигналами, частотные компоненты которых изменяются во времени, чирпами - сигналами с переменной частотой. Также получены навыки построения спектрограмм и их анализ.