

Санкт-Петербургский государственный политехнический
университет Петра Великого.

**Высшая школа интеллектуальных систем и
суперкомпьютерных технологий**

ЛАБОРАТОРНАЯ РАБОТА

Апериодические сигналы.

Работу выполнила студентка:

_____ А. И. Луцкевич
« ____ » _____ 2021 г.

Преподаватель лабораторных
работ:

_____ Н. В. Богач
« ____ » _____ 2021 г.

Санкт-Петербург, 2021 г.

Суть работы 3.1:

Прочитать пояснения и запустить примеры.

Был прочитан весь параграф, были запущены все примеры и просмотрены все результаты.

Суть работы 3.2:

Напишите класс, называемый SawtoothChirp, расширяющий Chirp и переопределяющий evaluate для генерации пилообразного сигнала с линейно увеличивающейся/уменьшающейся частотой. Нарисуйте эскиз спектрограммы этого сигнала, а затем распечатайте ее.

Создадим класс SawtoothChirp. Он будет наследником класса Chirp, изменен только метод evaluate. Метод evaluate был написан совмещением метода evaluate из Chirp и SawtoothSignal: freqs - частота на каждом интервале (с помощью np.linspace, который возвращает массив из n-1 значений между start и end); dts - разница между соседними элементами ts (длина в секундах); dphis - насколько меняется фаза за каждый интервал (по формуле $\Delta\phi = 2\pi f(t)\Delta t$, где freqs - $f(t)$, dts - Δt); phases - полная фаза на всех отрезках времени (сумма dphis); cycles, frac, _ и ys - из функции evaluate класса SawtoothSignal:

```
In [4]: class SawtoothChirp(Chirp):
        def evaluate(self, ts):
            freqs = np.linspace(self.start, self.end, len(ts))
            dts = np.diff(ts, prepend=0)
            dphis = 2 * np.pi * freqs * dts
            phases = np.cumsum(dphis)
            cycles = phases / (2 * np.pi)
            frac, _ = np.modf(cycles)
            ys = normalize(unbias(frac), self.amp)
            return ys
```

Создадим сигнал в две октавы длительностью 1 секунда, частотой 6000 и прослушаем его:

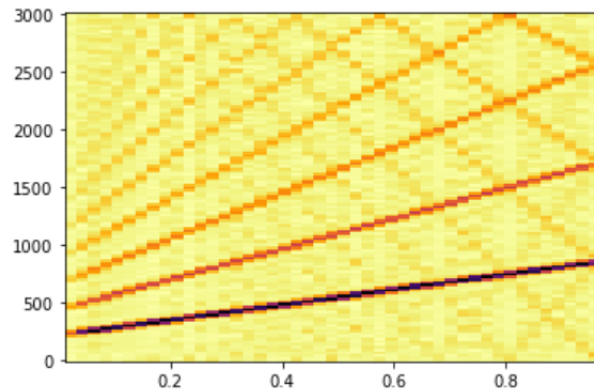
```
In [16]: s = SawtoothChirp(start=220, end=880)
         wave = s.make_wave(duration=1, framerate=6000)
         wave.apodize()
         wave.make_audio()
```

Out[16]:

▶ 0:01 / 0:01 ————— 🔊 ⋮

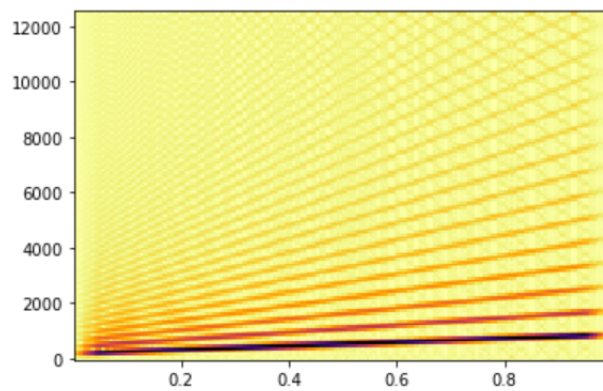
Также необходимо посмотреть как выглядит спектрограмма:

```
In [23]: spec_graph = wave.make_spectrogram(256)
spec_graph.plot()
```



Видно, что сглаженные гармоники откакивают от частоты складывания, это звучит как шипение на фоне. Попробуем увеличить частоту кадров до 25000:

```
In [25]: spec_graph = wave.make_spectrogram(256)
spec_graph.plot()
```



Картинка изменилась, стала более сглаженной.

Суть работы 3.3:

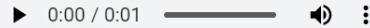
Создайте пилообразный чирп, меняющийся от 2500 до 3000 Гц, и на его основе сгенерируйте сигнал длительностью 1 с и частотой 20кГц. Распечатайте Spectrum.:

Создадим пилообразный чирп и ожидаем увидеть следующую спектрограм-

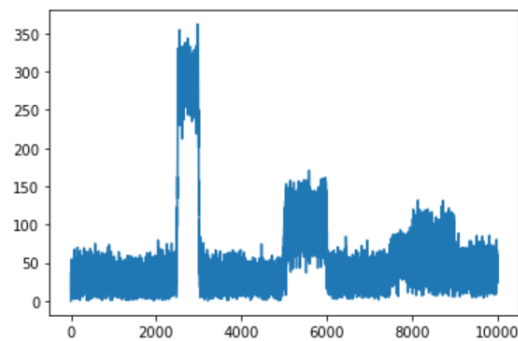
му - самое большое значение от 2500 до 3000 Гц, первая гармоника должна быть от 5000 до 6000 Гц и вторая гармоника (значение меньше) должна быть от 7500 до 9000 Гц.

```
In [26]: s = SawtoothChirp(start=2500, end=3000)
         wave = s.make_wave(duration=1, framerate=20000)
         wave.make_audio()
```

Out[26]:



```
In [27]: wave.make_spectrum().plot()
```



Все ожидания подтвердились.

Суть работы 3.4:

Найдите или запишите звук глиссандо и распечатайте спектрограмму первых нескольких секунд

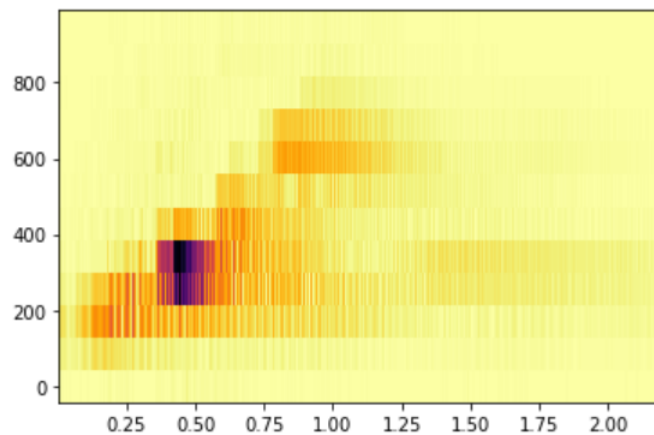
Мною был взят звук глиссандо на арфе: была создана wave и распечатана спектрограмма:

```
In [34]: from thinkdsp import read_wave
wave = read_wave('1.wav')
wave.make_audio()
```

Out[34]:



```
In [35]: wave.make_spectrogram(512).plot(high=1000)
```



Суть работы 3.5:

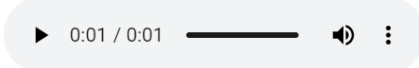
Тромбонист играет глиссандо, непрерывно дует в мундштук и двигая кулису тромбона. При этом общая длина трубы меняется, а играемая нота обратно пропорциональна этой длине. Если предположить, что музыкант двигает кулису с постоянной скоростью, как будет меняться частота со временем? Напишите класс под названием TromboneGliss, который расширяет Chirp и предоставляет evaluate. Создайте сигнал, имитирующий глиссандо на тромбоне от C3 до F3, и обратно до C3. C3-262 Гц; F3-349 Гц. Постройте спектрограмму полученного сигнала. На что похоже глиссандо на тромбоне - на линейный или экспоненциальный чирп?

Создадим сначала класс TromboneGliss, который является наследником класса Chirp и переопределим метод evaluate (будет вычислен диапазон с помощью self.start и self.end):

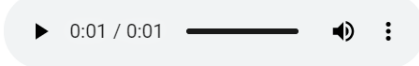
```
In [36]: class TromboneGliss(Chirp):
def evaluate(self, ts):
    len1, len2 = 1.0 / self.start, 1.0 / self.end
    lengths = np.linspace(len1, len2, len(ts))
    freqs = 1 / lengths
    dts = np.diff(ts, prepend=0)
    dphis = 2 * np.pi * freqs * dts
    phases = np.cumsum(dphis)
    ys = self.amp * np.cos(phases)
    return ys
```

Создадим сигнал от 349 Гц до 262 Гц длительностью 1 секунда и второй сигнал от 262 Гц до 349 Гц:

```
In [39]: s = TromboneGliss(349, 262)
wave = s.make_wave(duration=1)
wave.apodize()
wave.make_audio()
```

Out[39]: 

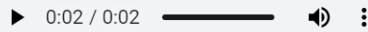
```
In [40]: s = TromboneGliss(262, 349)
wave1 = s.make_wave(duration=1)
wave1.apodize()
wave1.make_audio()
```

Out[40]: 

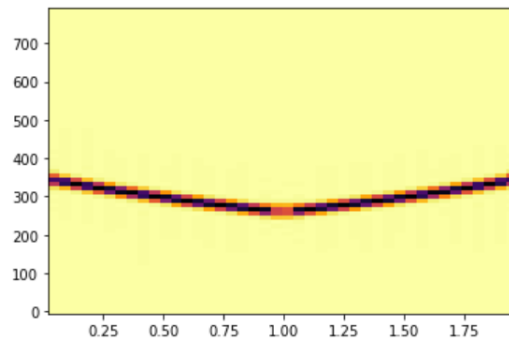
Соединим эти два сигнала в один и напечатаем спектрограмму:

```
In [41]: wave = wave | wave1  
wave.make_audio()
```

Out[41]:



```
In [45]: spec_gram = wave.make_spectrogram(1024)  
spec_gram.plot(high=800)
```



Судя по картинке глissандо на тромбоне похоже на линейный чирп.

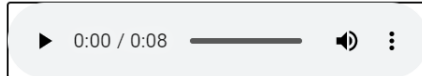
Суть работы 3.6:

Сделайте запись серии гласных звуков и посмотрите на спектрограмму. Сможете ли вы различить разные гласные звуки?

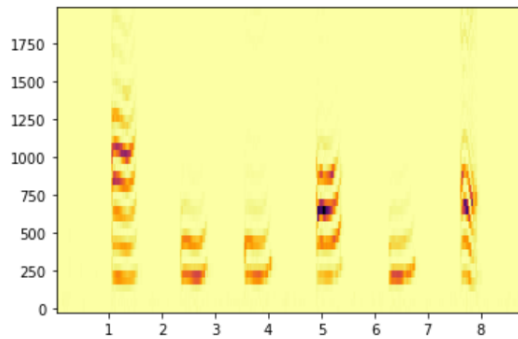
Я сделала запись 6 гласных звуков: а, у, ы, о, и, э. Построим спектрограмму:

```
In [68]: wave = read_wave('gl.wav')
wave.make_audio()
```

Out[68]:

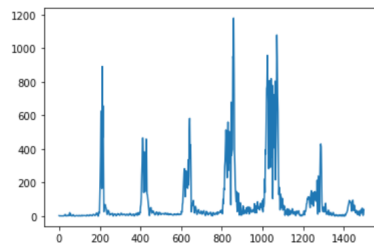


```
In [70]: wave.make_spectrogram(1024).plot(high=2000)
```



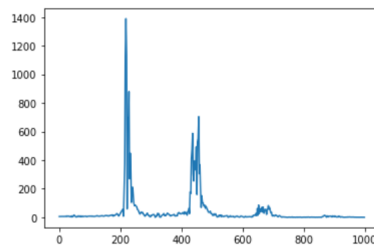
Выберем сегмент со звуком "а". Фундаментальная частота 820 Гц. Следующие высокие пики находятся на частоте 1300 Гц и 200 Гц:

```
In [90]: segment = wave.segment(start=1, duration=0.5)
segment.make_spectrum().plot(high=1500)
```



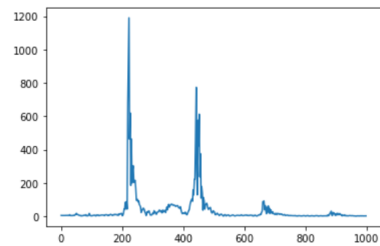
На звуке "у" фундаментальная высота 200 Гц:

```
In [91]: segment = wave.segment(start=2.3, duration=0.5)
segment.make_spectrum().plot(high=1000)
```



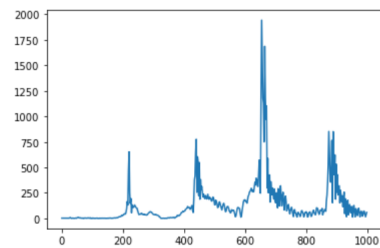
На звуке "ы" фундаментальная высота тоже 200 Гц:


```
In [92]: segment = wave.segment(start=3.5, duration=0.5)
segment.make_spectrum().plot(high=1000)
```



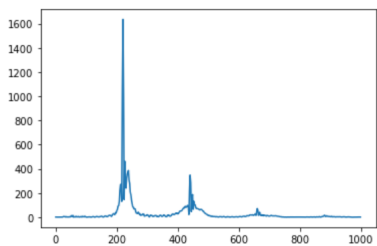
На звуке "о" фундаментальная высота 630 Гц:

```
In [93]: segment = wave.segment(start=4.9, duration=0.5)
segment.make_spectrum().plot(high=1000)
```



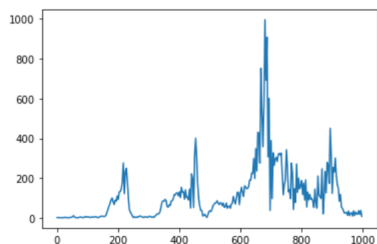
На звуке "и" фундаментальная высота 200 Гц:

```
In [94]: segment = wave.segment(start=6.2, duration=0.5)
segment.make_spectrum().plot(high=1000)
```



На звуке "э" фундаментальная высота 640 Гц:

```
In [95]: segment = wave.segment(start=7.6, duration=0.3)
segment.make_spectrum().plot(high=1000)
```



С помощью спектрограммы я, к сожалению, не смогу различить гласные звуки.

Заключение:

По итогу выполнения данной лабораторной работы я теперь имею представление как устроены апериодические сигналы, понимаю различия между линейным чирпом и экспоненциальным, узнала, что можно сделать глиссандо с помощью программных средств и как оно выглядит (раньше была знакома с этим термином только с точки зрения игры на инструменте), а также сделала собственную запись ласных звуков и посмотрела на их отличия с точки зрения спектрограмм.