Лабораторная работа 1

Чернышев Ярослав

16 мая 2021 г.

Оглавление

Глава 1

Задание 1.1

В данном задании от меня требуется изучить готовые примеры из chap01.ipynb, а также ознакомиться с результатами работы с ним.

Signals

Instantiate cosine and sine signals.

```
[ ]  from thinkdsp import CosSignal, SinSignal

     cos_sig = CosSignal(freq=440, amp=1.0, offset=0)
     sin_sig = SinSignal(freq=880, amp=0.5, offset=0)
```

Plot the sine and cosine signals. By default, `plot` plots three periods.

```
[ ]  from thinkdsp import decorate

     cos_sig.plot()
     decorate(xlabel='Time (s)')
```

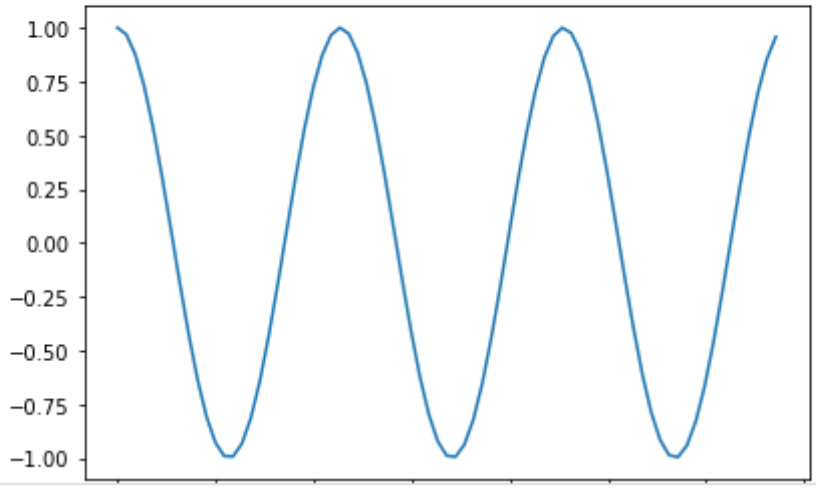

Рис. 1.1: Фрагмент работы с Signal

Можно сделать вспомнить теретическую часть, а именно:

Сигнал - изменяющаяся во времени величина.

Волна - значения сигнала, взятые за определённую последовательность моментов времени.

Спектр - разложение сигнала в виде суммы синусод с разными частотами.

Это совпадает с увиденным мною при выполнении образцов кода.

Глава 2

Задание 1.2

В этом задании от меня требуется загрузить звук, выбрать полусекундный сегмент с примерно постоянной высотой, проанализировать и сделать выводы.

Первым делом, выделим интересующий сегмент:

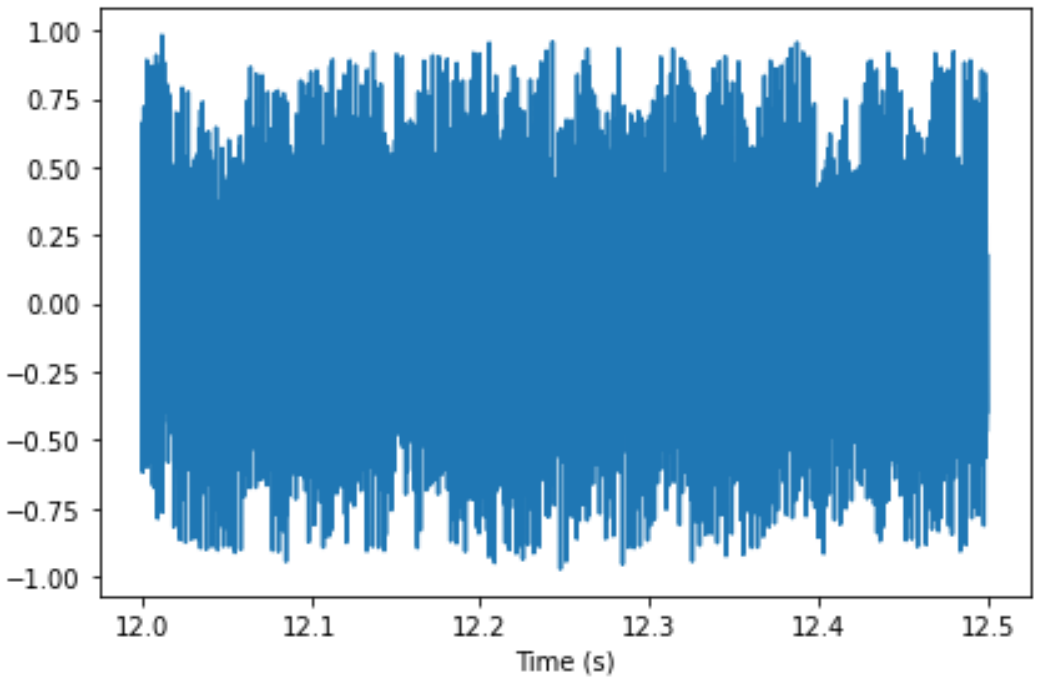

Рис. 2.1: Исходный звук

Как можно видеть, это он действительно имеет приблизительно постоянную высоту.

Теперь раскладываем выделенный сегмент в спектр:

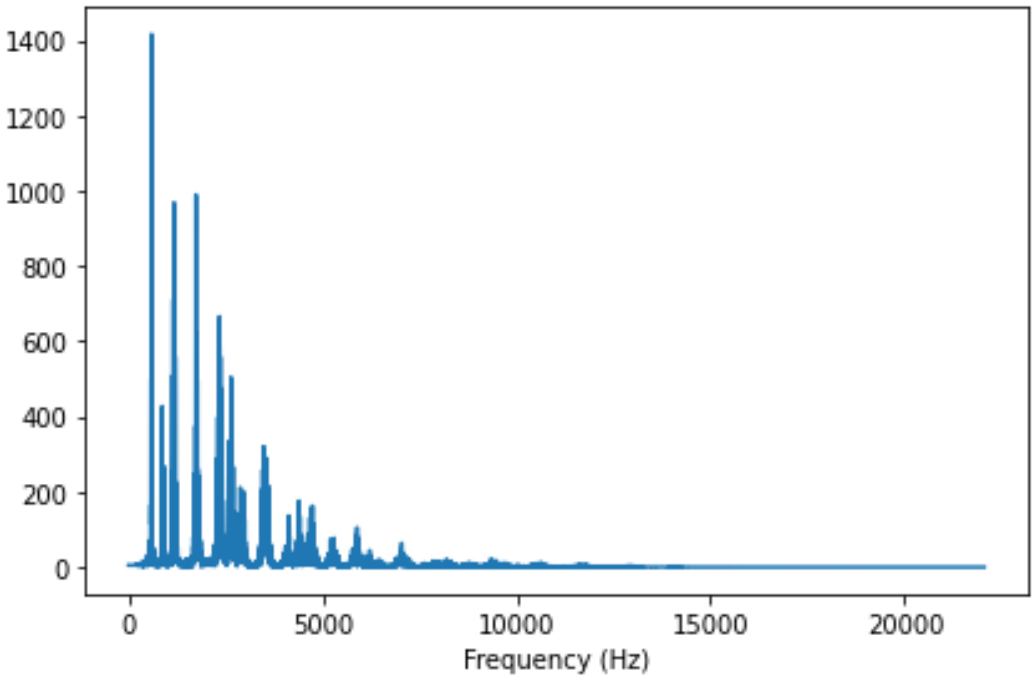

Рис. 2.2: Спектр

Как можно видеть на графике больше всего гармоник с низкими частотами. Попробуем отфильтровать часть гармоник. Для этого воспользуемся функцией `low_pass`:

```
spectrum.low_pass(5000)
spectrum.plot(high=5000)
decorate(xlabel='Frequency (Hz)')

```

Листинг 2.1: Применение `low_pass`

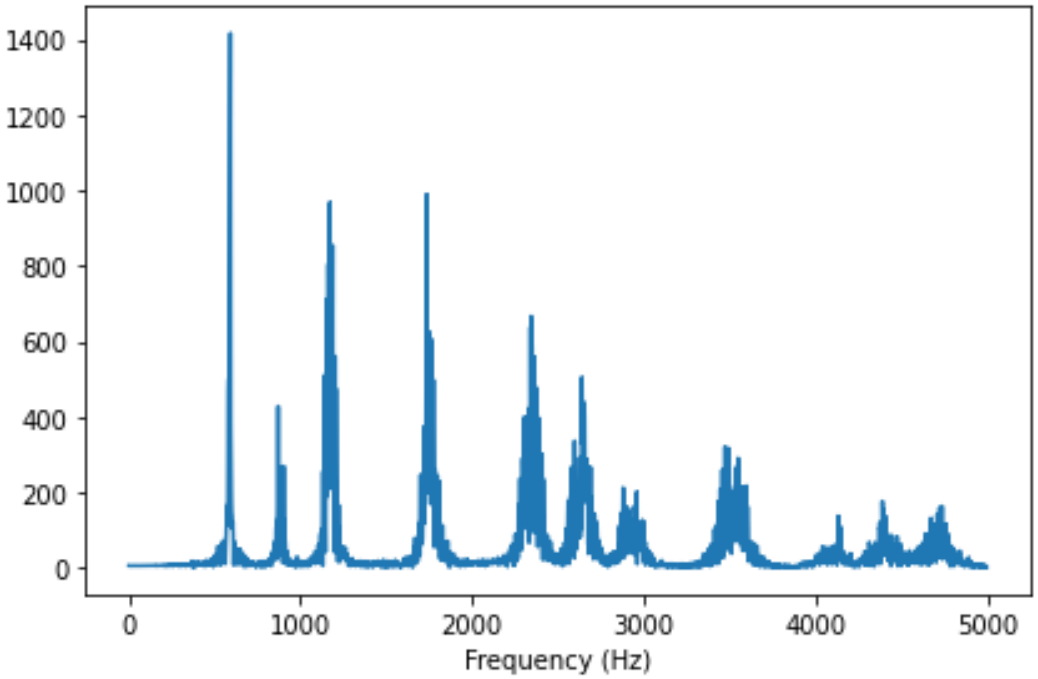

Рис. 2.3: Результат фильтрации

Теперь преобразуем получивший спектр обратно в волну:

```
filtered = spectrum.make_wave()
filtered.normalize()
filtered.plot()
decorate(xlabel='Time (s)')
```

Листинг 2.2: Обработанный звук

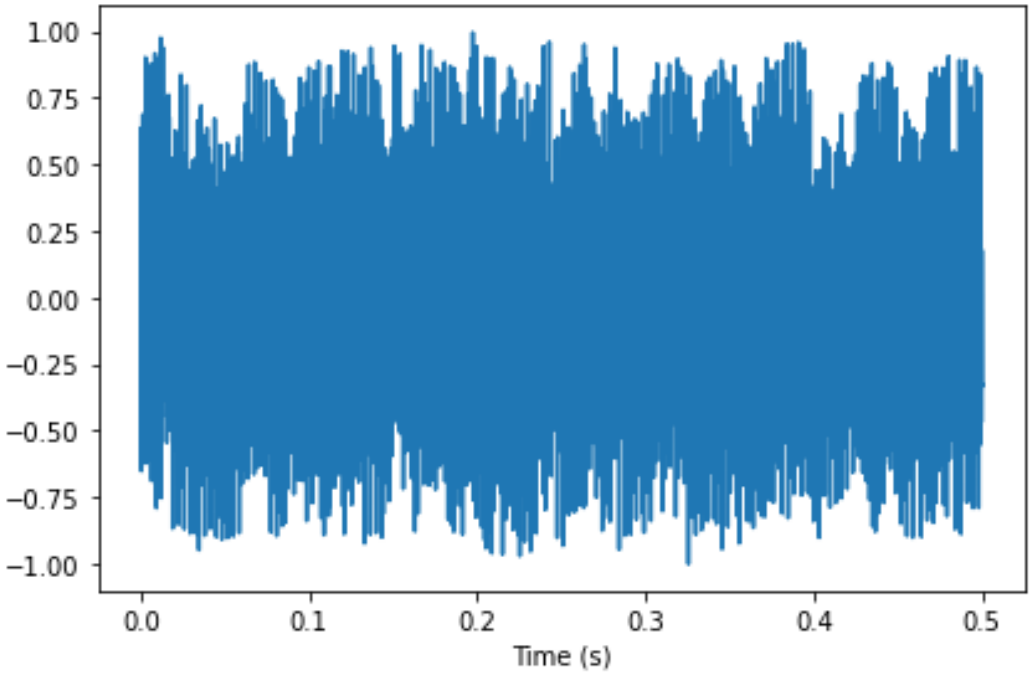

Рис. 2.4: Результат

Изменения по сравнению с оригиналом малозаметны; можно сказать, что звук стал более приглушённым.

Глава 3

Задание 1.3

В этом задании я сложил разные синусоиду и косинусоиду:

```
from thinkdsp import CosSignal, SinSignal

cosSig = CosSignal(freq=440,     amp=1.0, offset=0)
sinSig = SinSignal(freq=440*3, amp=0.75, offset=0)

cosSig.plot()
decorate(xlabel='Time (s)')

sinSig.plot()
decorate(xlabel='Time (s)')
```

Листинг 3.1: Создание сигналов

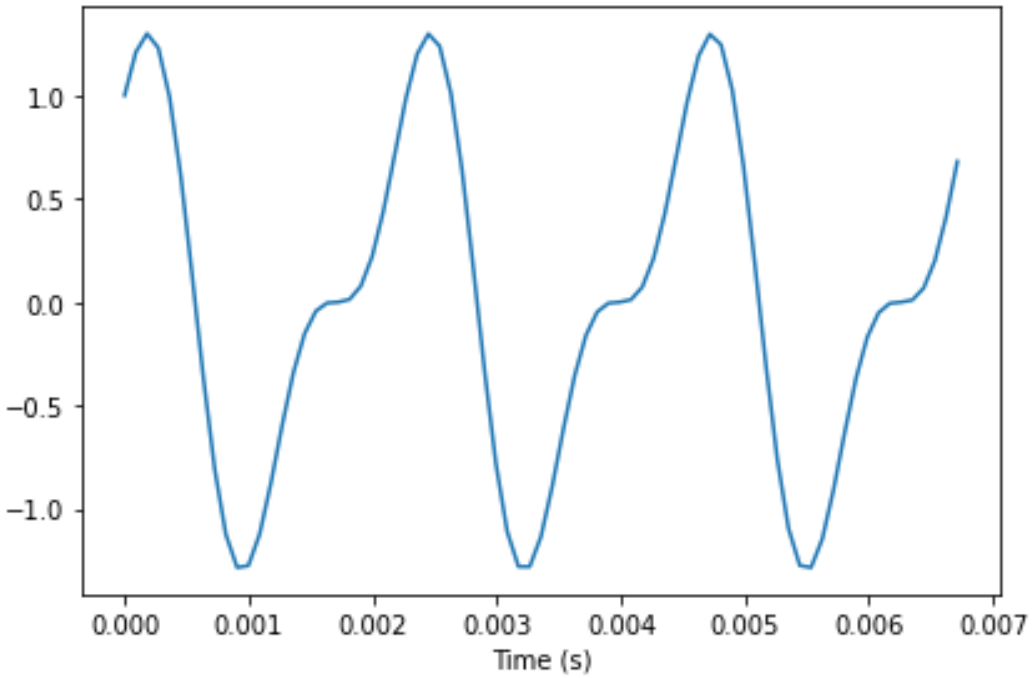

Рис. 3.1: Построенные сигналы

После этого, суммируем эти сигналы:

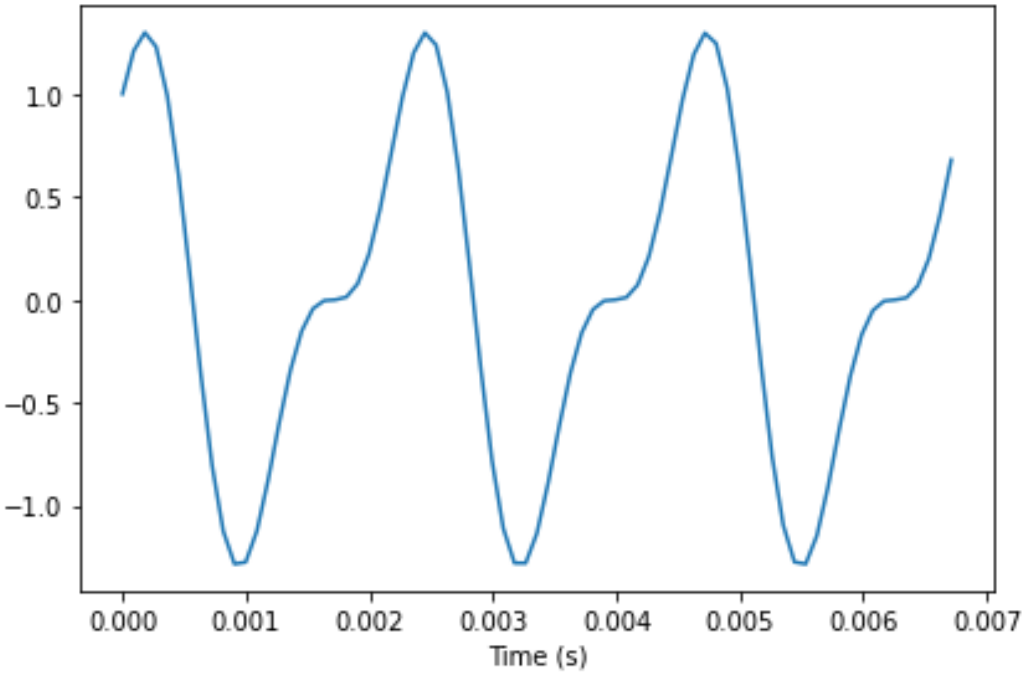

Рис. 3.2: Сумма сигналов

Также, можно взгялнуть на их спектр:

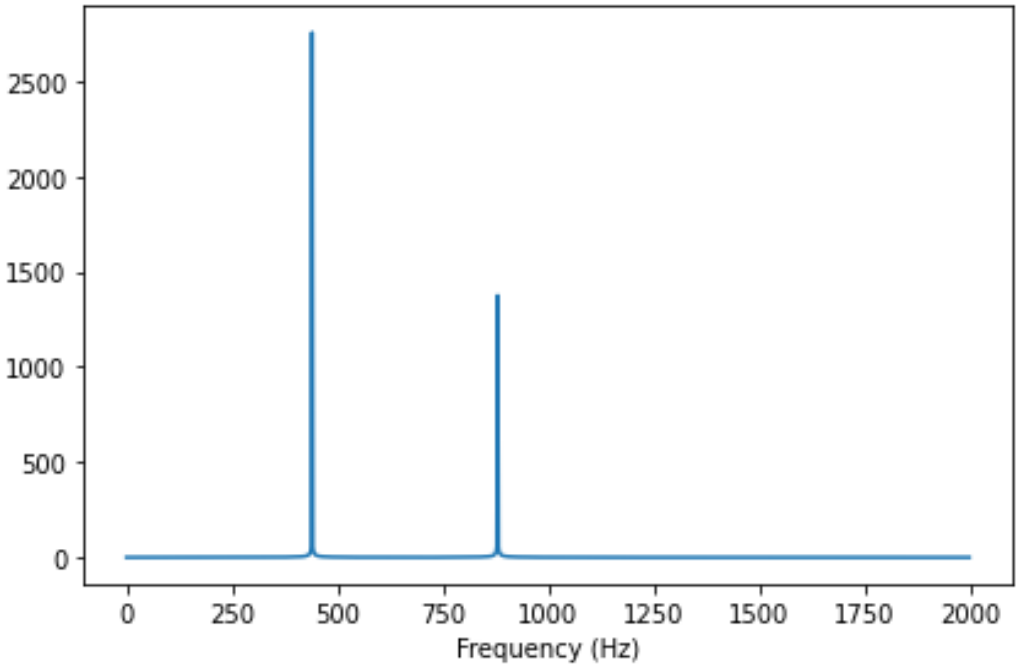

Рис. 3.3: Спектр суммы сигналов

Теперь прибавим к нашей сумме некратный CosSignal(freq=440 * 3.1416, amp=1.0, offset=0). В результате получаем следующий график:

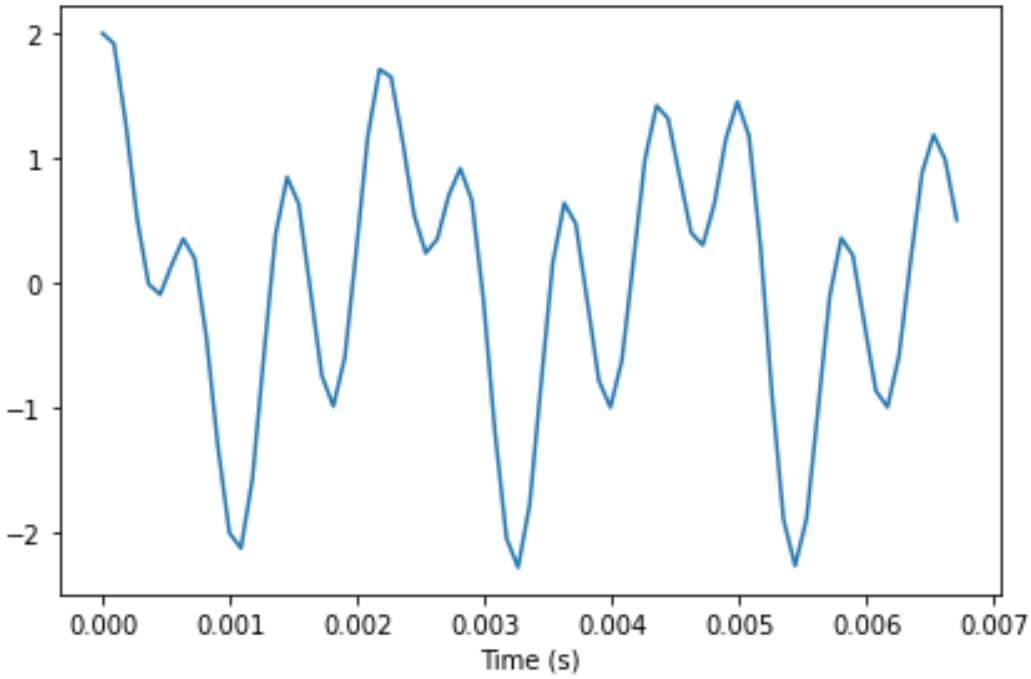

Рис. 3.4: Сумма с некратным сигналом

Этой волне соответствует следующий спектр:

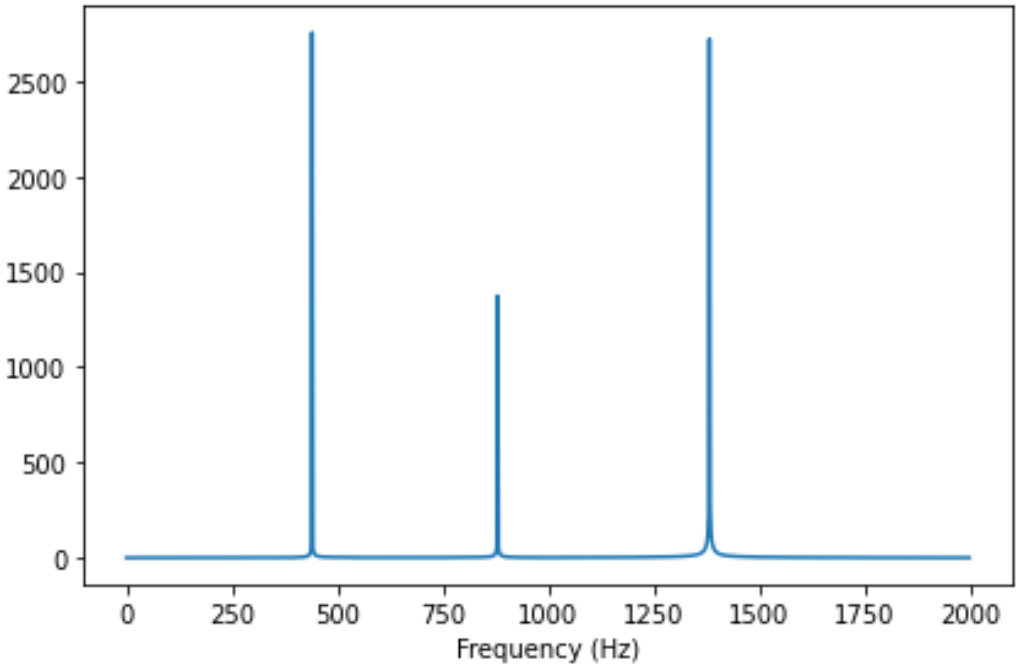

Рис. 3.5: Спектр суммы сигналов

Глава 4

Задание 1.4

Требуется написать функцию, ускоряющую или замедляющую волну в зависимости от параметра stretchFactor:

```
1 def stretch(wave, stretchFactor):
2     wave.ts /= stretchFactor
3     wave.framerate *= stretchFactor
```

Листинг 4.1: stretch

При указании stretchFactor больше единицы сигнал будет ускоряться, а при stretchFactor меньшем единицы - замедляться.