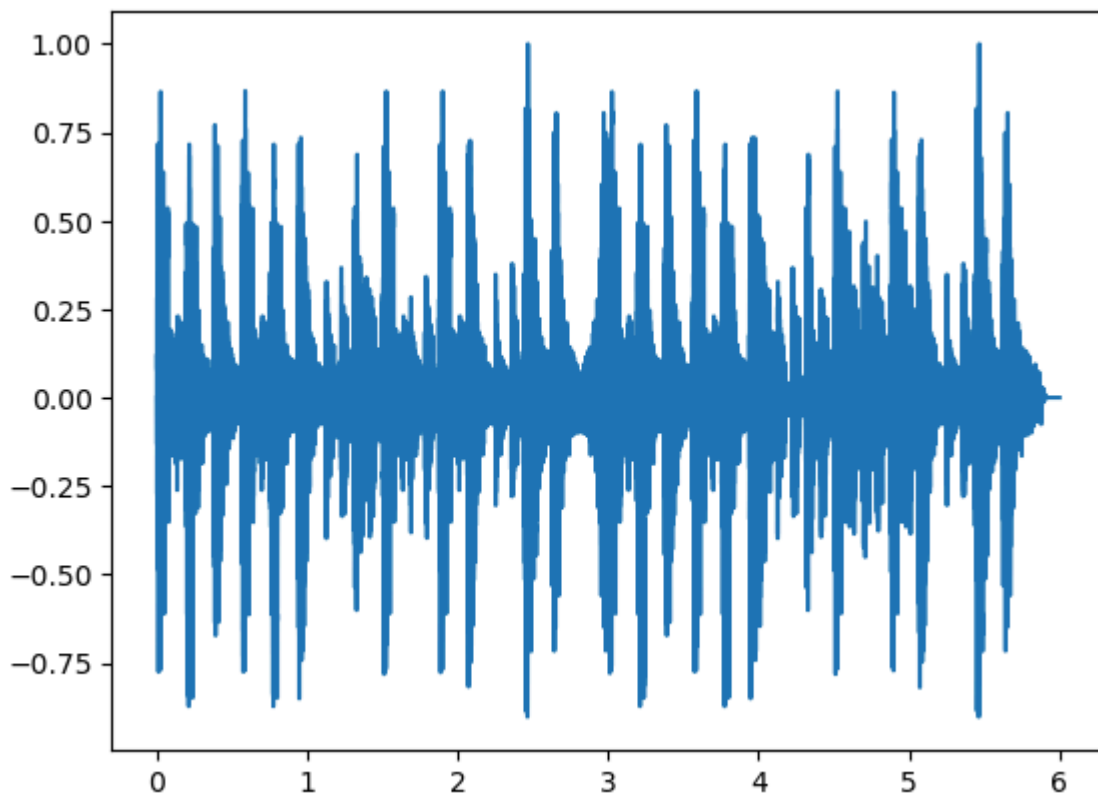


Лабораторная работа 11

Упражнение 11.3

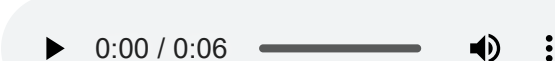
Выше показано, что при взятии выборок из сигнала при слишком низкой частоте кадров составляющие, большие частоты заворота дадут биения. В таком случае эти компоненты не отфильтруешь, поскольку они неотличимы от более низких частот. Полезно отфильтровать эти частоты до выборки; фильтр НЧ, используемый для этой цели, называется фильтр сглаживание. Вернитесь к примеру "Соло на барабанах", примените фильтр НЧ до выборки, а затем, опять же с помощью фильтра НЧ, удалите спектральные копии, вызванные выборкой. Результат должен быть идентичен отфильтрованному сигналу.

```
In [ ]: from thinkdsp import *  
  
wave = read_wave('263868__kevcio__amen-break-a-160-bpm.wav')  
wave.normalize()  
wave.plot()
```



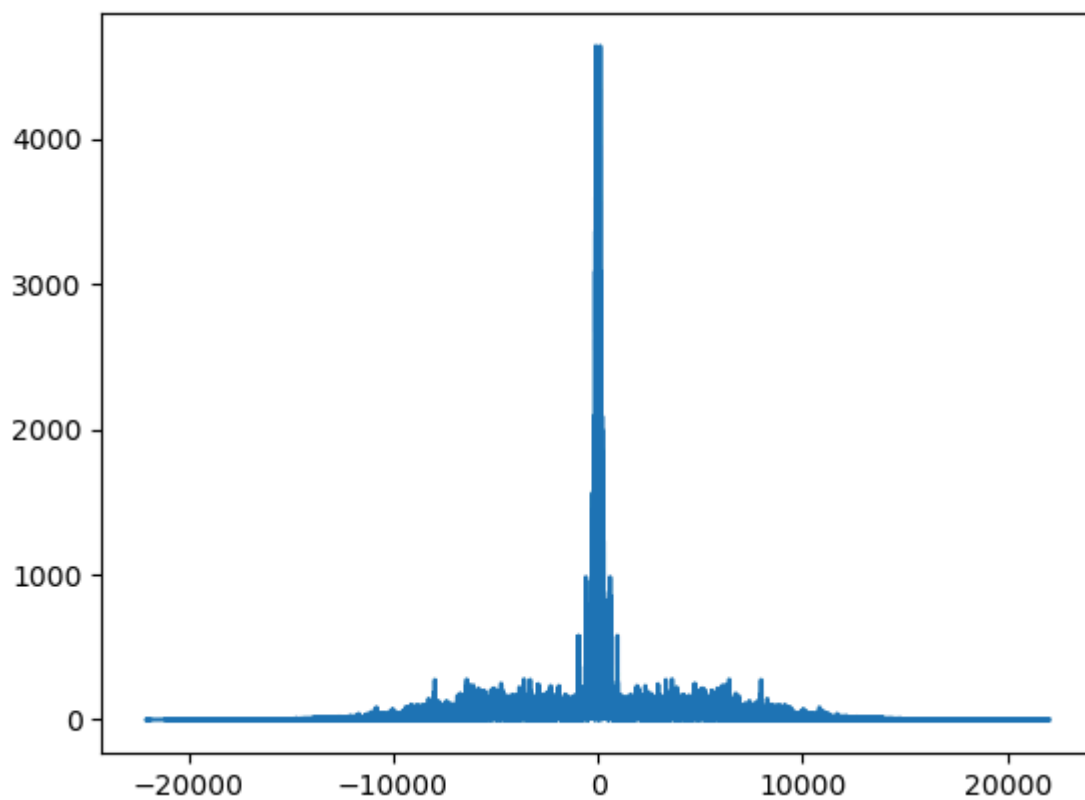
```
In [ ]: print(wave.framerate)  
wave.make_audio()
```

44100

Out []: 

Сигнал записан с частотой кадров 44100 Гц

```
In [ ]: spectrum = wave.make_spectrum(full=True)  
spectrum.plot()
```

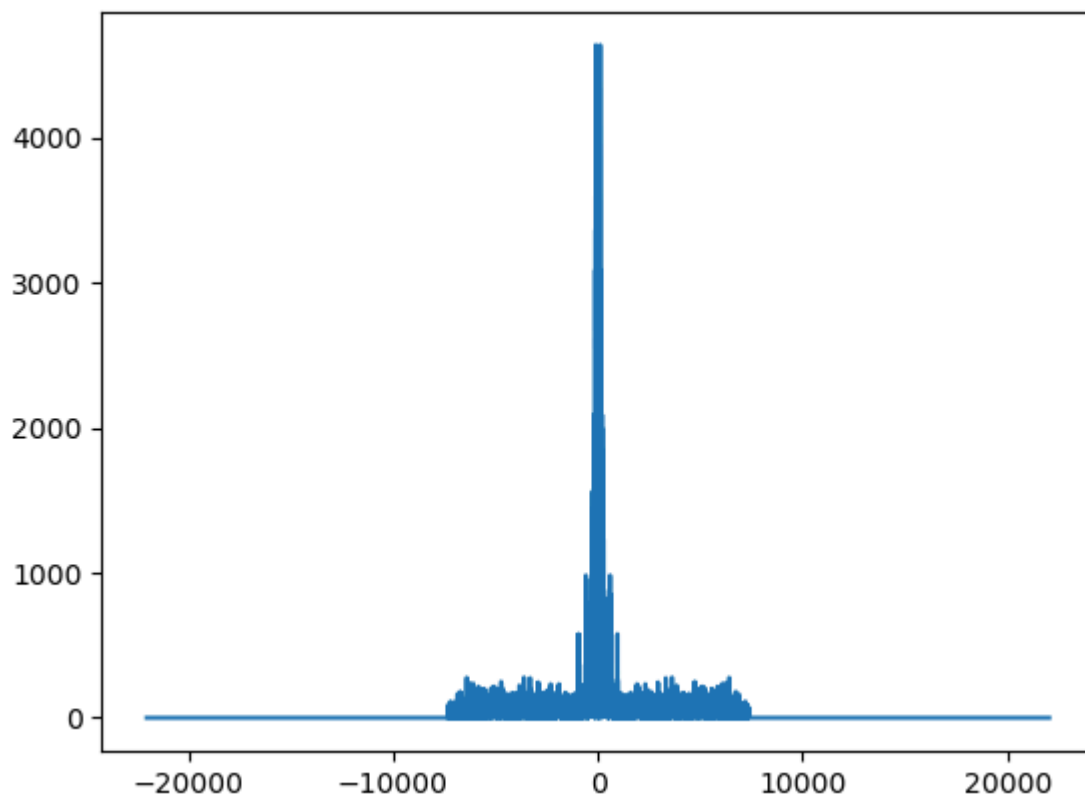


Уменьшим частоту записи в 3 раза. Также вычислим новую частоту заворота

```
In [ ]: framerate = wave.framerate / 3
        cutoff = framerate / 2 - 1
```

Перед взятием выборки отфильтруем частоты выше новой частоты заворота

```
In [ ]: spectrum.low_pass(cutoff)
        spectrum.plot()
```



```
In [ ]: filtered = spectrum.make_wave()
        filtered.make_audio()
```

Out []:

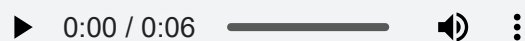


Функция взятия выборок

```
In [ ]: def sample(wave, factor):  
        ys = np.zeros(len(wave))  
        ys[::factor] = np.real(wave.ys[::factor])  
        return Wave(ys, framerate=wave.framerate)
```

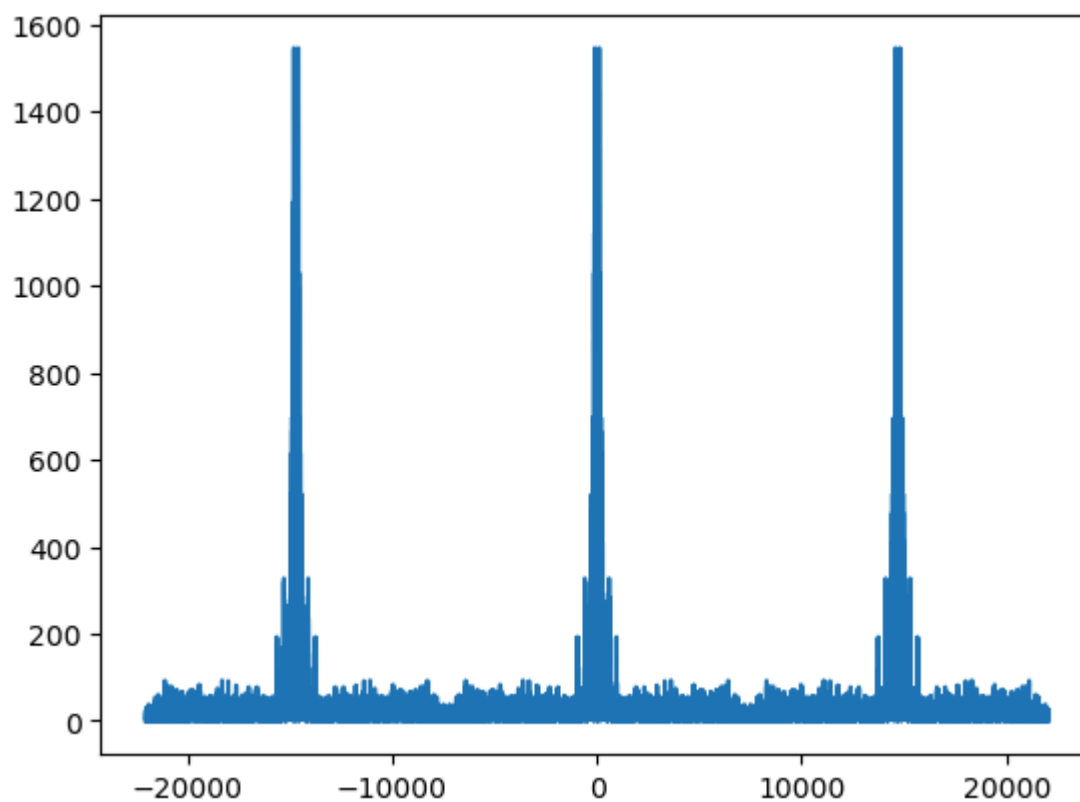
```
In [ ]: sampled = sample(filtered, 3)  
        sampled.make_audio()
```

Out []:

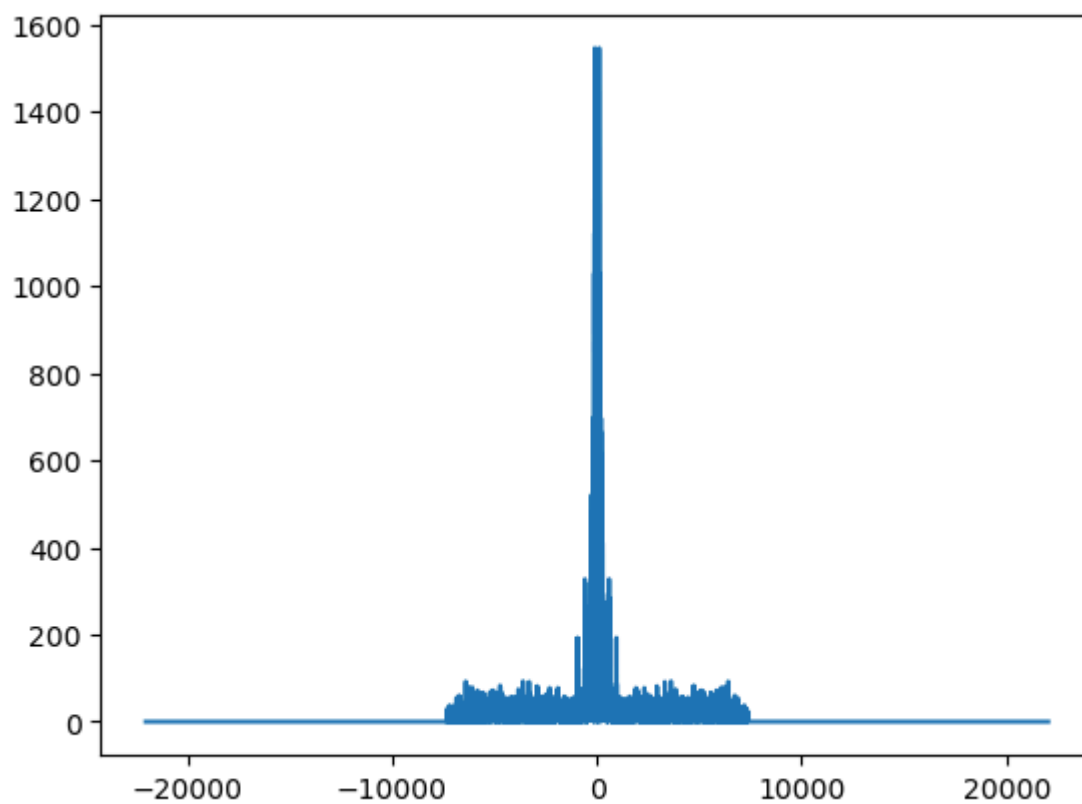


В результате можем услышать некий звон. Это слышно копии спектра в области высоких частот

```
In [ ]: sampled_spectrum = sampled.make_spectrum(full=True)  
        sampled_spectrum.plot()
```

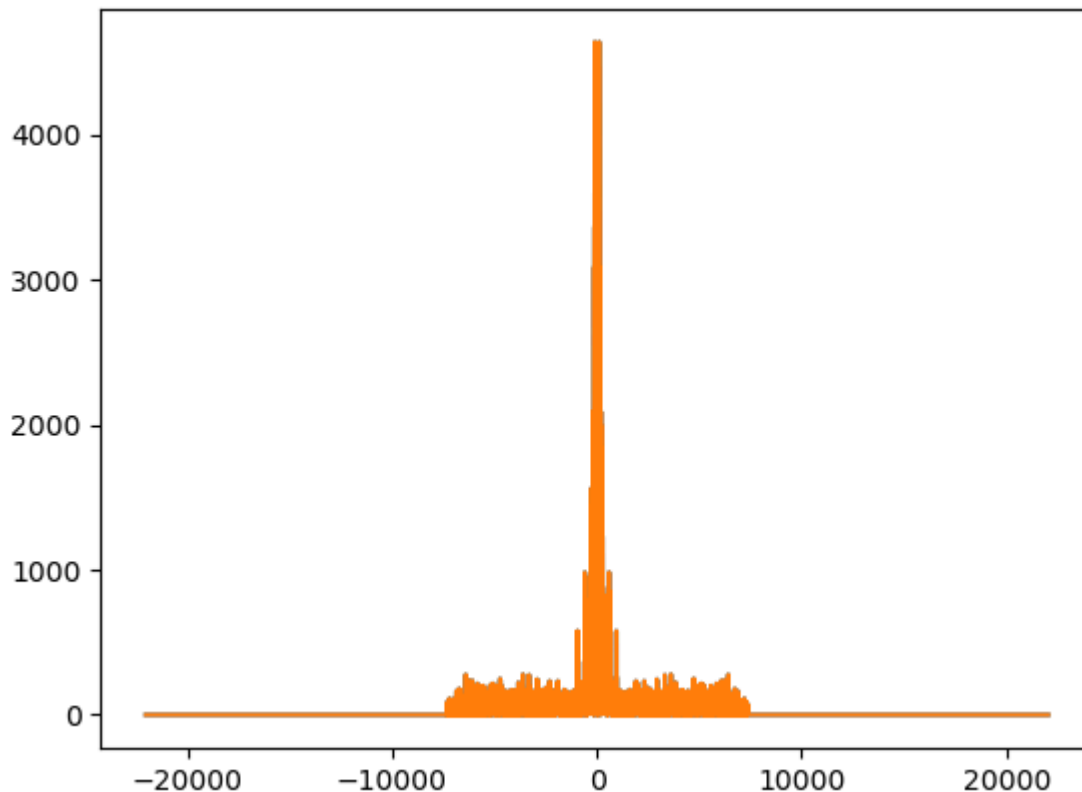


```
In [ ]: sampled_spectrum.low_pass(cutoff)  
        sampled_spectrum.plot()
```



В процессе взятия выборок и фильтрации, была потеряна энергия. Можем восстановить ее, усилив выборку

```
In [ ]: sampled_spectrum.scale(3)
        spectrum.plot()
        sampled_spectrum.plot()
```

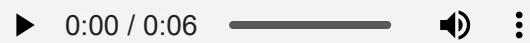


Разница между семплированным спектром и оригинальным отфильтрованным почти не заметна

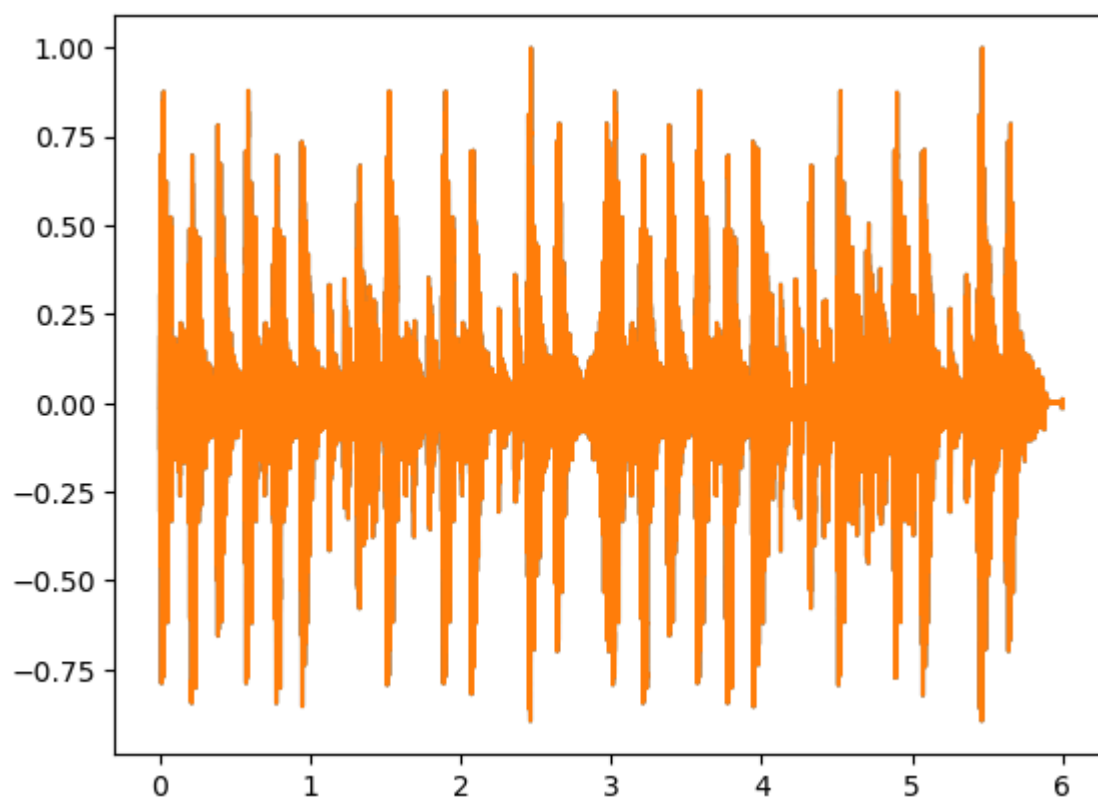
После фильтрации и семплирования можем вернуться к сигналу

```
In [ ]: interpolated = sampled_spectrum.make_wave()
        interpolated.make_audio()
```

Out[]:



```
In [ ]: filtered.plot()  
interpolated.plot()
```



Также мало заметна разница между сигналом отфильтрованным и семплированным

In []:

In []: