Filip Kuś

**Sprawozdanie   
Projekt 1 – Analiza dźwięków**

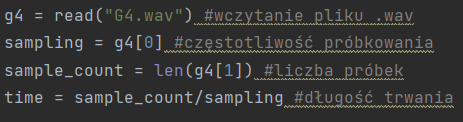
**Program 1**

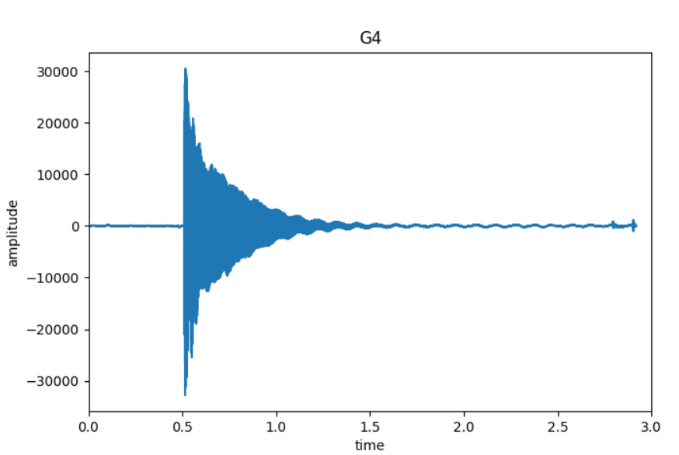
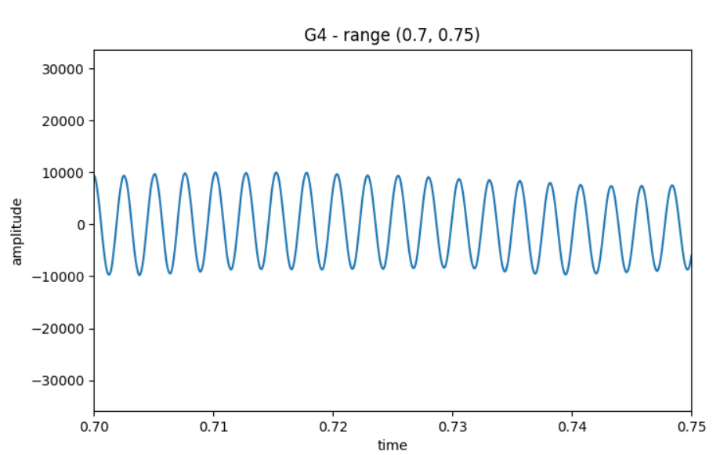
Zaimportowano niezbędne biblioteki.

Stworzono funkcje:

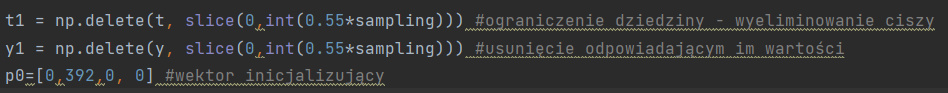
* **plots** – służąca wyświetlaniu wykresu po podaniu dziedziny, wartości i danych do jego opisu i zakresu,
* **fourier** – przyjmująca czas i wartości, zwracająca nasilenie danych częstotliwości (FFT),
* **func** – konstruująca gasnącą sinusoidę na podstawie kilku parametrów.

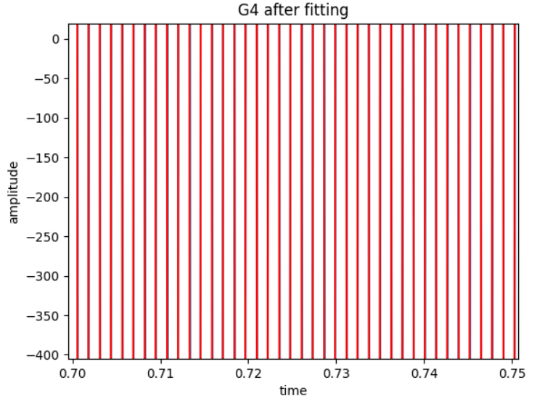
Wczytano plik ***G4.wav*** dzięki funkcji ***scipy.io.wavfile.read***. Stworzono zmienne niezbędne do wykonywania dalszych operacji.

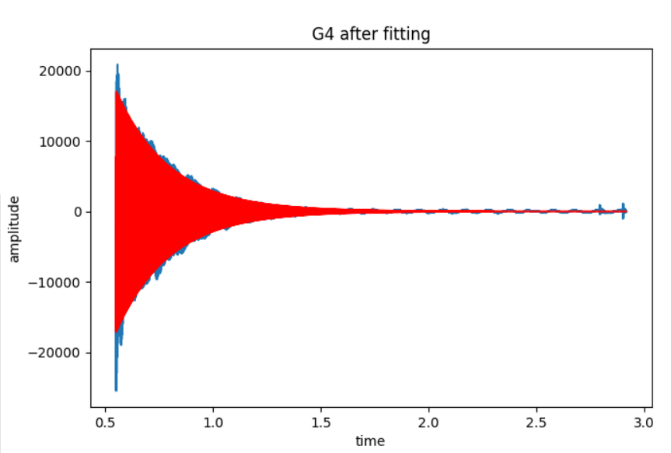


Za pomocą funkcji **plots** wyświetlono dwa wykresy obrazujące fale dźwiękową:

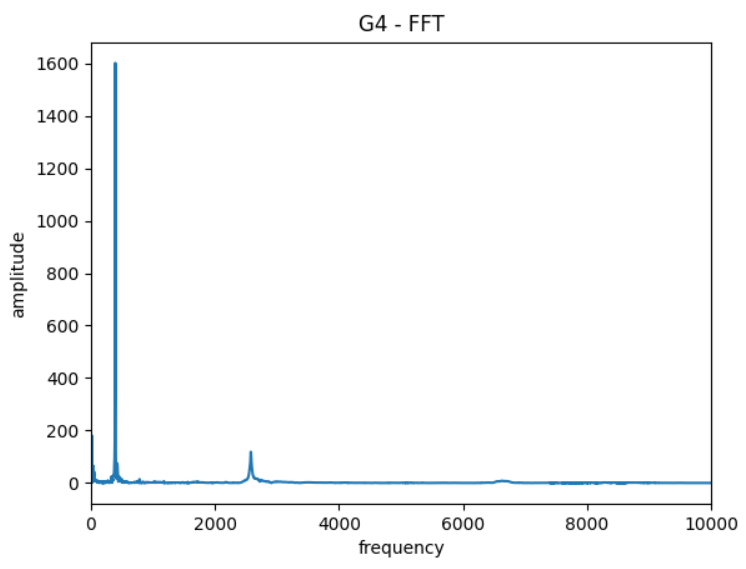
Obliczona ręcznie na podstawie wykresu po prawej stronie częstotliwość wyniosła 400 Hz (40 przejść przez miejsca zerowe w ciągu 0.05 s).

Następnie wykonano dopasowanie poprzedniego wykresu. Zredukowano dziedzinę, tak by nie uwzględniać ciszy z samego początku nagrania oraz ustalono wektor inicjalizujący w taki sposób, by fitowanie nie zwracało funkcji stałej równej 0, przez co algorytm zwraca zadowalający efekt.





Obliczona w ten sam sposób, jak w poprzednim przypadku, częstotliwość wyniosła dokładnie tyle samo (400 Hz).

Ostatnią najważniejszą częścią było wyświetlenie FFT dzięki funkcji ją obliczającej (**fourier**) i **plots**. Tak stworzony wykres zaprezentował zdecydowanie najdokładniejszą częstotliwość na poziomie 392.5 Hz.

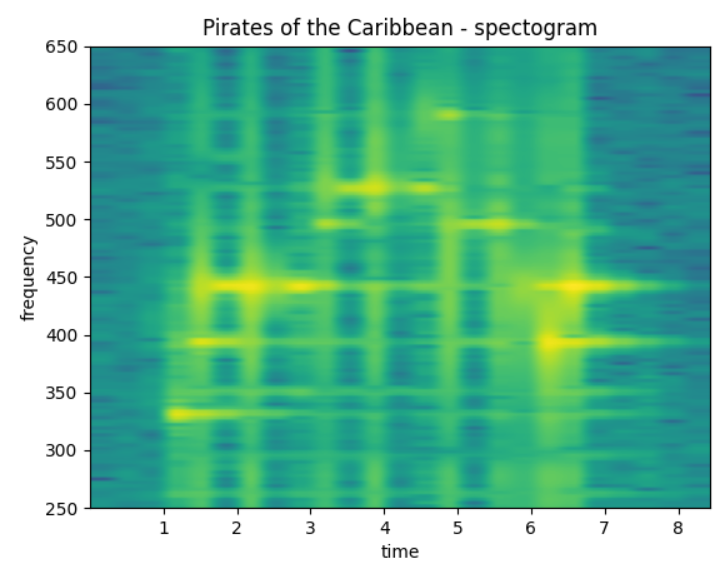
Porównując do siebie częstotliwości uzyskane na trzy różne sposoby oraz konsultując je z modelową częstotliwością, jaka jest właściwa dźwiękowi G4, nie sposób nie zauważyć, że Transformata Fouriera jest zdecydowanie najdokładniejszą metodą liczenia częstotliwości. Wystarczy umieć się tylko posłużyć algorytmem ją obliczającym, a rezultat zostanie zwrócony w bardzo przystępnej formie. W przypadku dwóch pierwszych rozwiązań należało najpierw wybrać przedział, na którym mają być dokonane obliczenia, zbadać, ile długości fali mieści się w danym przedziale czasu, a następnie obliczyć ich stosunek i doprowadzić go do postaci w Hz. Takie rozwiązanie ma jednak fundamentalną wadę: w poszczególnych przedziałach liczba długości fali będzie się różnić, co wpłynie na częstotliwość, która również będzie się zmieniać. Także bez zastosowania wygodnego algorytmu obliczającego miejsca zerowe (np. metody połowienia) ręczne liczenie na zadanym przedziale jest mało wygodne.

**Program 2**

Zaimportowano niezbędne biblioteki.

Wczytano plik *piraci.wav* i po uzyskaniu odpowiednich danych stworzono za pomocą *plt.specgram* spektrogram.





Odczytane (z pewną niedokładnością) ze spektrogramu dla NFFT=15000 (parametr, który, subiektywnie, umożliwia najlepszy odczyt) dźwięki:

|  |  |  |
| --- | --- | --- |
| **Czas [s]** | **Częstotliwość [Hz]** | **Dźwięk** |
| 1.1 | 330 | E |
| 1.5 | 395 | G |
| 2.2 | 440 | A |
| 2.5 | 440 | A |
| 2.8 | 440 | A |
| 3.2 | 495 | H |
| 3.8 | 525 | C |
| 4.1 | 525 | C |
| 4.5 | 525 | C |
| 4.8 | 590 | D |
| 5.1 | 495 | H |
| 5.9 | 495 | H |
| 6.2 | 440 | A |
| 6.2 | 390 | G |
| 6.6 | 440 | A |