DSP ASSIGNMENT: RECOGNISE AN INSTRUMENT IN AN ORCHESTRA

Digital Signal Processing – IoT Minor

Joshua van ‘t Kruis
Gavin Geurtsen
Janire Pampin Rubio

10.01.2023

Table of content:

1. Introduction

The assignment of DSP (Digital Signal Processing) consists of applying what had been learned in class about filters and Fourier Transform to detect certain instruments on an audio sound.

We as a group have decided to do so, in the song “He’s a pirate” from the film Pirates of the Caribbean, played by an orchestra.

Firstly, we talk about the theoretical background of the project by defining the music instruments, describing the sound waves and explaining the filters. Secondly, all the steps taken in order to achieve the goal of this assignment are explained, including code. Thirdly, a conclusion and recommendation of the project are given. Lastly the references and appendix are displayed.

2. Theoretical background

In this section of the report theoretical aspects of the assignment are explained. This includes how music instruments make sound and how an instrument can be extracted from an audio signal using filters.

2.1. Music Instruments

Music instruments are any object that can be used to make sound. So as to make sound, they have to create pressure waves in the air. Depending on the method used, musical instruments can be classified in three different groups:

- Wind instruments: These instruments produce sound by vibrating a column of air, using a reed or the lips. Within this category instruments such as the trumpet or the flute can be found.
- String instruments: These instruments produce sound from vibrating strings, using a plectrum or the fingers. Within this category instruments such as the violin or the piano can be found.
- Percussion instruments: These instruments produce sound by being struck or scraped, using hands or percussion mallets for example. Within this category instruments such as the drums or the txalaparta can be found.

The sound an instrument produces is determined by many factors, including the type of material used to make the instrument, the size and shape of the instrument, and the way in which the instrument is played.

Characteristics of musical instruments:

- Pitch: Term that describes how high or low sounds are.
- Frequency range: Frequency range of the sound a certain instrument can create.
- Timbre: The specific tone or quality that a certain instrument or voice has.

2.2. Physics of the sound

A sound wave is the pattern of disturbance produced by the movement of energy traveling as it travels through a medium (like air or water) as it propagates away from the source.

Main characteristics of sound waves:

- Wavelength: The distance between two successive crests or troughs of a wave.
- Period: Time required to complete a cycle.
- Amplitude: The amount of energy present in a wave.
- Frequency or pitch: Number of cycles completed in one second.
- Velocity: Distance travelled by the sound wave per unit of time.

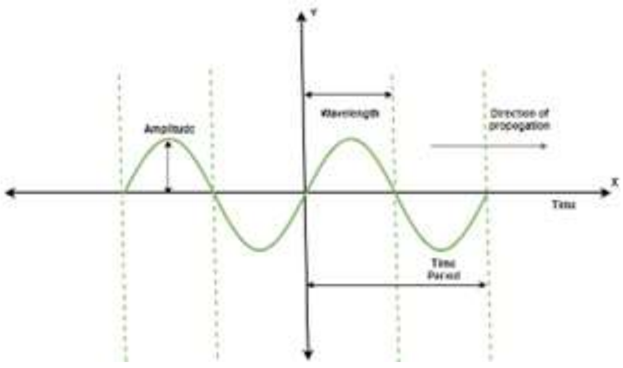


Figure 1: Parameters of a sound wave.

One of the ways to calculate the power of an audio signal is through the RMS. The root mean square power (or RMS power) is the square root of the average of the square of the power of the sound signal over a given duration. It is calculated using the following formula:

$$P_{RMS} = \sqrt{\frac{1}{n}(x_1^2 + x_1^2 + \cdots + x_n^2)}$$

This parameter has been used to detect when an instrument is being played.

2.3. Digital Filters

The definition of a filter is that of a circuit capable of keeping certain frequencies while erasing others. Depending on what it amplifies or attenuates, many different filters can be found.

The basic types of filters are:

- High-pass: Attenuates all frequencies below the cut frequency given.
- Low-pass: Attenuates all frequencies above the cut frequency given.
- Band-pass: Attenuates all frequencies outside the frequency band given.
- Band-stop: Attenuates all frequencies of frequency band given.

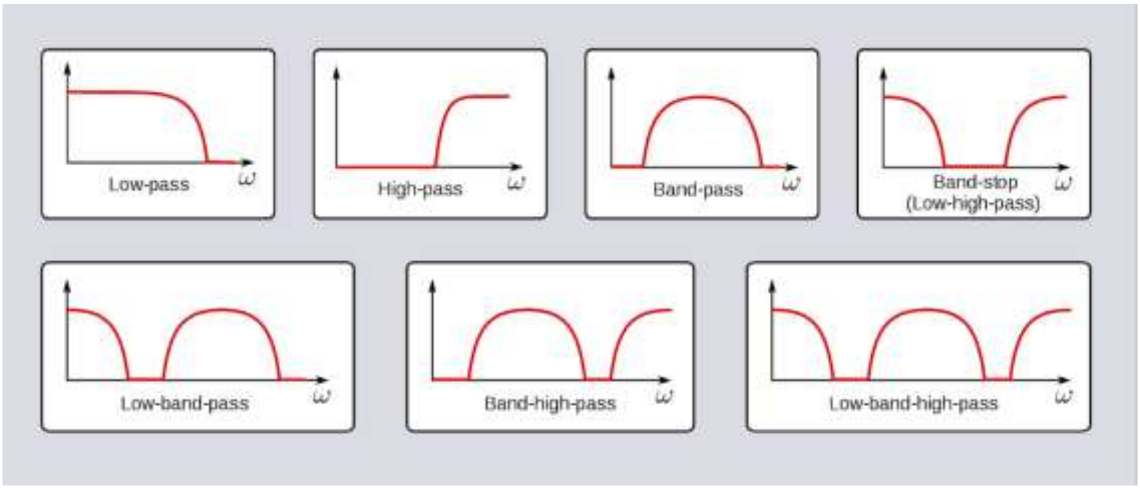


Figure 2: Different type filters.

All filters are characterized with the following parameters (Davis, 2017):

- Cutoff frequency (fc): The frequency at which the output signal has one-half of the power of the input signal. Also called -3dB, because that is the equivalent of the power in decibels.
- Bandwidth: Range of frequencies that the filter passes, in case of the band-stop, rejects.
- Rolloff: The slope of the filter's response in the *transition region* between the pass-band and stop-band.
- Ripple: The variation in the attenuation of the input signal in the filter's pass-band or stop-band.

2.4. Butterworth filter

The Butterworth filter is a signal processing filter designed to have a frequency response as flat as possible in the passband. From 0 Hz to the cut-off frequency at -3 dB, a flat frequency response can be obtained mathematically without ripple. The first-order filter rolls off towards zero at a rate of -20 dB/decade if the frequency exceeds the cut-off frequency. The rate of a roll-off period increases when the order of the filter is increased. And for next order, it is -40 dB/decade.

In the image below the Frequency Response of Butterworth Filter can be seen:

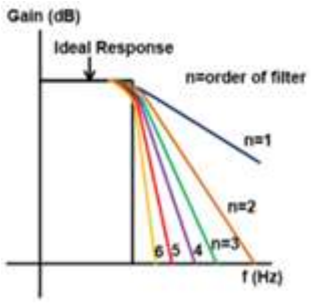


Figure 3: Butterworth frequency response funcion with different orders.

3. Assignment

The group has chosen the assignment of creating a python program that can recognize instruments in an orchestra. To accomplish this task, the group will utilize techniques and theories from the DSP course. Upon completing the program, the group will carefully review the results to ensure that they meet the requirements of the assignment.

3.1. Steps taken

The first meeting of the project group was dedicated to conducting research on how to isolate individual instruments from an audio signal and selecting the orchestra that would be used for this project. This research provided the necessary foundation for the project and allowed us to plan the next steps.

Following this, each member of the group was given a task to focus on programming a solution to isolate a specific instrument. By dividing the work in this way, we were able to make progress on the project. The instruments were divided as follows. (*Tech Stuff - Frequency Ranges*, n.d.)

Student	Instrument	Frequency range(in Hz)
Janire	Violin	196 - 10000
Gavin	Drums(timpani)	90 - 180
Joshua	Trumpet	165 - 988

Table 1: Distribution of the instrument per member of the group.

3.2. Software implementation

In this section the software developed during the project will be discussed. The assignment was developed in a jupyter notebook and the following subtopics will be discussed: included libraries, methods and the developed notebook.

3.2.1. Utilized libraries

During the realization of the project three python libraries were used to realize the assignment. A brief overview of the used library will be given and their functionality within the code:

- Firstly, NumPy is a library that is used to work with large, multi-dimensional arrays of numerical data, along with a large collection of mathematical functions to operate on these arrays. For this project NumPy is used to represent and manipulate the audio data and filter coefficients as arrays, and to perform various operations on these arrays such as indexing, filtering, and clipping.

- Secondly Scipy, a library built on top of NumPy and extends its arithmetic capabilities to be able to perform more complex tasks such as signal processing. In this project SciPy is used to read and write .wav files, apply a Butterworth bandpass filter to the audio signal, and compute the frequency response of the filter.

- Matplotlib is a 2-dimensional plotting library, in this project it was mainly used to create plots of the raw audio signal, spectrogram of the audio, and frequency response of the filter.

- Librosa is also a plotting library. It is more powerful and its spectrogram has higher quality than matplotlib. However, it takes up to 3 minutes to load the audio signal's spectrogram.

- Pydub is a library that is used to Manipulate audio with a simple and easy high-level interface. It has been used to play the audio in the notebook and also to split the audio signal in different slots.

3.2.2. Imported methods

There are two important functions that will be explained as to why and how they contribute to the assignment. Both functions are taken from the scipy library and are named butter and lfilter.

Scipy.signal.butter, this function takes two frequency bounds, order number and subsequently returns two arrays which represent the filter coefficients of a Butterworth filter. Where a and b are pre-determined filter coefficients and these are used to generate the required transfer functions. See figure in the appendix for a table with predetermined polynomials.

scipy.signal.lfilter, the equation beneath describes the transfer function used by the function lfilter, the input arrays a and b are filter coefficients calculated by the function scipy.signal.butter.

$$H(s) = \frac{B[0] \; + \; B[1]s + \ldots + B[M]s}{A[0] \; + \; A[1]s + \ldots + A[N]s}$$

The numerator polynomial B(s) represents the gain of the system at different frequencies, while the denominator polynomial A(s) represents the phase shift of the system at different frequencies. As described in the function documentation (*scipy.signal.lfilter — SciPy v1.10.0 Manual*, n.d.).

3.2.3. Developed notebook

The purpose of the developed Jupyter notebook is to identify an instrument by analysing the frequency ranges in which it is present. The notebook begins by loading a .wav file and filters out all frequencies except for the range associated with the targeted instrument. The notebook then determines whether the instrument is being played by measuring the power of the remaining frequencies.

The input variables that are essential for filtering in this Jupyter notebook are the order number, low cut frequency, and high cut frequency. These variables determine the range of frequencies that will be passed through the bandpass filter. The order number, in particular, determines the sharpness (roll-off factor) of the filter and can affect the output signal. A high order number results in a sharper filter, which means that more frequencies outside of the target range will be attenuated.

The output of the script is an audio signal that has been filtered to only include frequencies within the specified range. This filtered signal is then plotted on a graph showing the amplitudes as a function of frequency and a spectrogram. Then, the Jupyter notebook will generate a .wav file containing the filtered audio signal. Lastly, to recognize an instrument playing, the audio signal is divided into segments and the power of a segment is read out and if above a determined amount it will be recognized as playing. This process will repeat over the whole signal.

Link to the Jupyter Notebook.

3.3. Results

When executing the notebook on the wav file "pirates-of-the-caribean.wav", it will display the following spectrogram of the signal before filtering has been applied. The spectrogram is a visual representation of the spectrum of frequencies of a signal as a function of time. The range of the original audio goes from 0Hz to 10.000kHz.

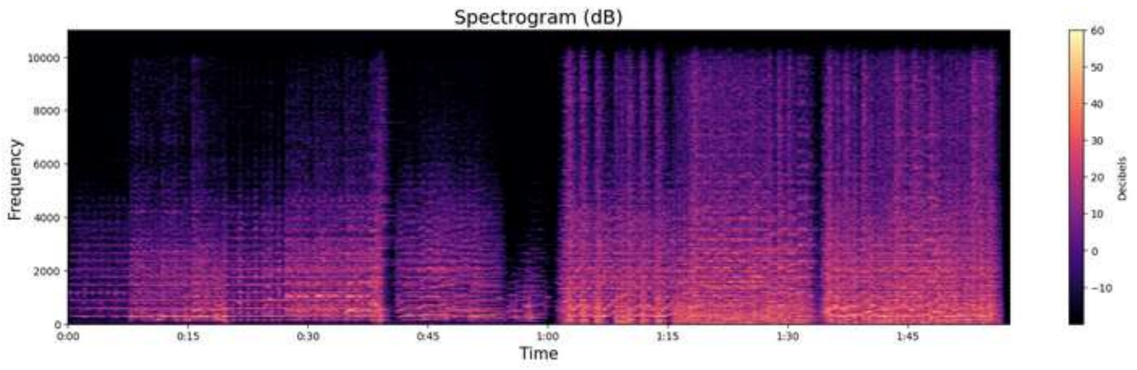


Figure 4: Spectrogram of input .wav file.

To extract the instruments, three different filters have been used. The filter A is for the violin, filter B is for the timpani and C is for the trumpet. Each one has different cutoff frequencies. In the case of the violin, its frequency range takes nearly the whole audio. Nevertheless, we have only selected the frequencies above 3500 Hz, because the rest of instruments cannot reach them. In other words, if there is some power in this frequency range it is only because of a violin playing. In the case of the trumpet and the timpani, these instruments have a narrower spectrum, so we have selected their respective frequency ranges.

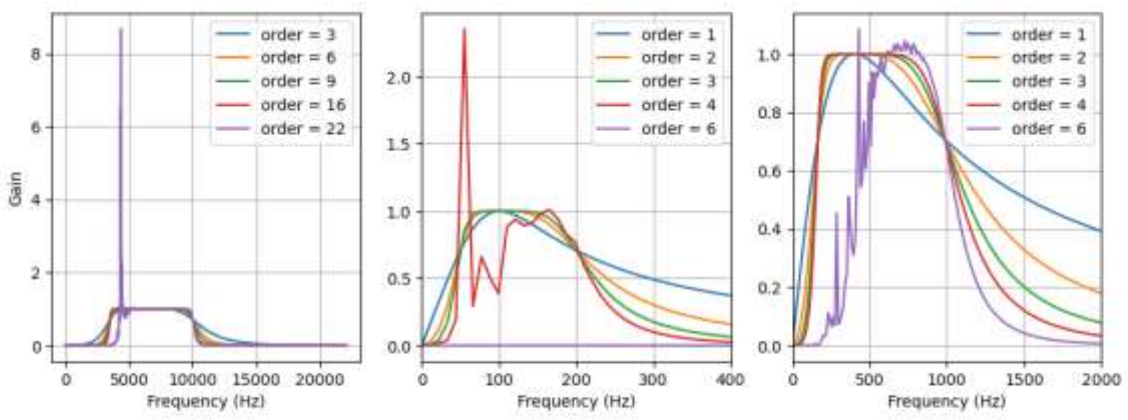


Figure 5: Filters used to extract different instruments. From left to right: instrument A, instrument B and instrument C.

In the above image appear the filters that have been use to extract the violin, timpani and trumpets. As can be seen in the images, from one order the filter begins to fail. That is why we have used the highest order without it becoming saturated. This seem to be a typical issue regarding digital filters. Due to the limited precision of floating-point numbers, high order filters with cutoff frequencies far below the Nyquist frequency frequently have unstable coefficients.

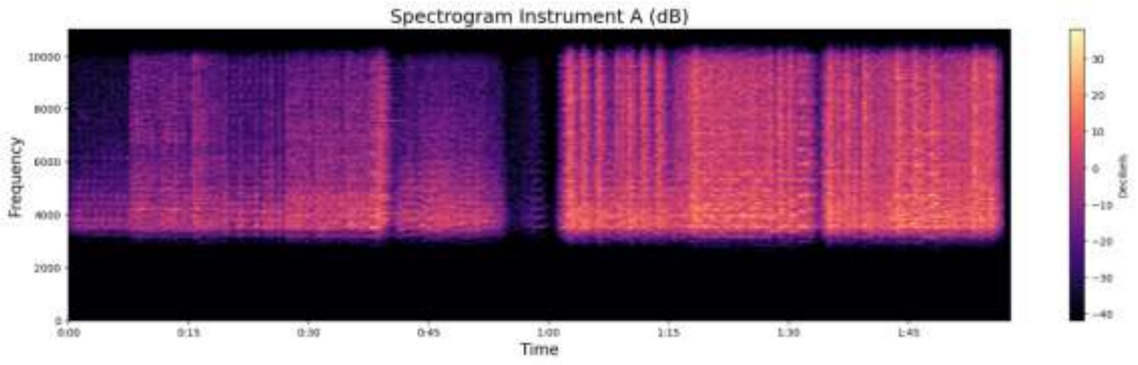


Figure 6: Spectrogram of the output signal of filter A.

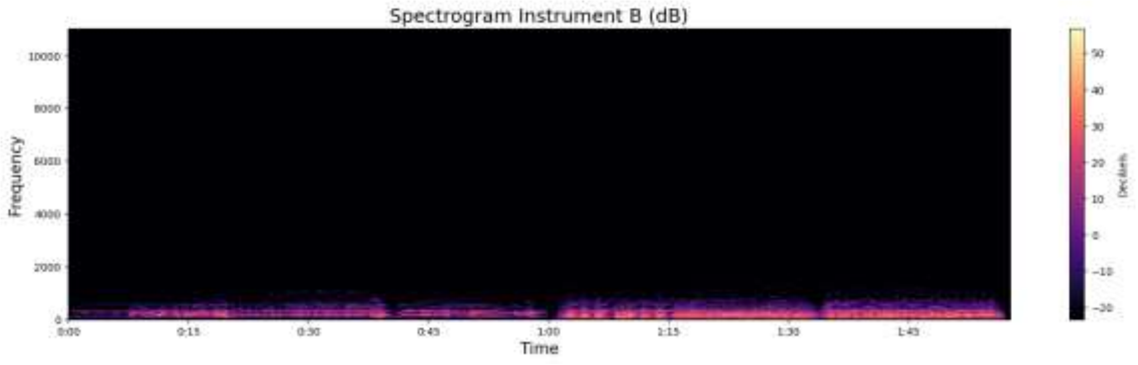


Figure 7: Spectrogram of the output signal of filter B.

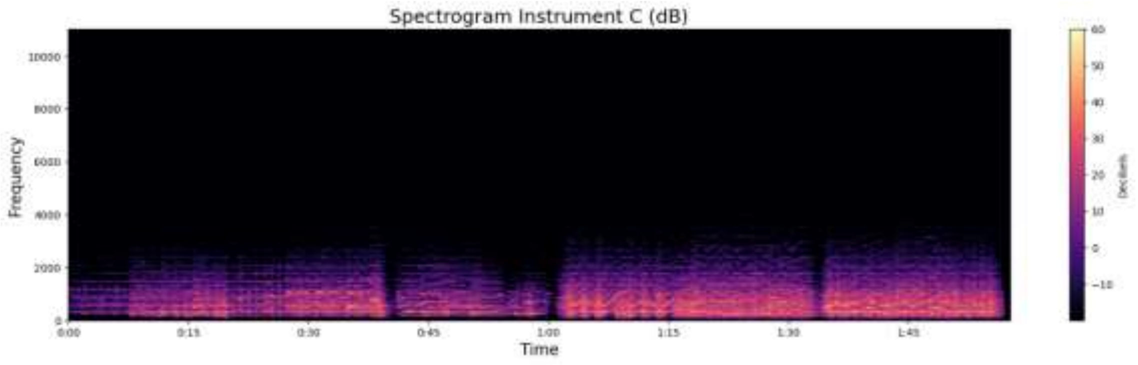
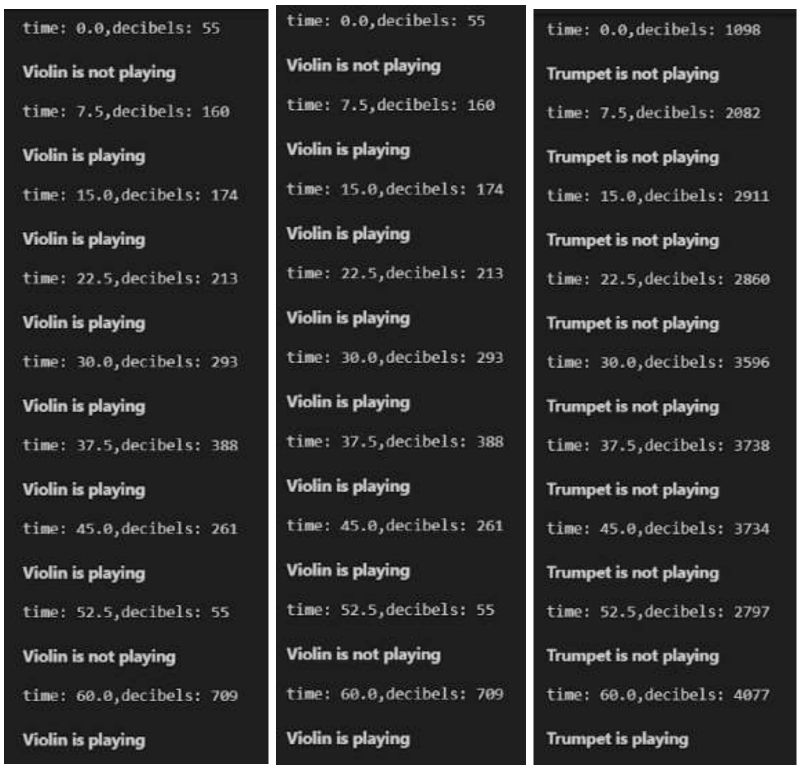


Figure 8: Spectrogram of the output signal of filter C.

The photographs above show that the filters have worked correctly, allowing only the following specified frequencies to pass: in the case of the violin 3500-10000Hz, in the case of the timpani 50-200Hz and in the case of the trumpet 165-988Hz.

The last step has been to determine a threshold of power which detect when each instrument is playing. For this purpose, we have looked carefully at the audio when each instrument is playing and it has been concluded that the limit of the violin is 150, that of the timpani is 2000 and that of the trumpet is 4000.



```
time: 0.0,decibels: 55
Violin is not playing
time: 7.5,decibels: 160
Violin is playing
time: 15.0,decibels: 174
Violin is playing
time: 22.5,decibels: 213
Violin is playing
time: 30.0,decibels: 293
Violin is playing
time: 37.5,decibels: 388
Violin is playing
time: 45.0,decibels: 261
Violin is playing
time: 52.5,decibels: 55
Violin is not playing
time: 60.0,decibels: 709
Violin is playing
```

```
time: 0.0,decibels: 55
Violin is not playing
time: 7.5,decibels: 160
Violin is playing
time: 15.0,decibels: 174
Violin is playing
time: 22.5,decibels: 213
Violin is playing
time: 30.0,decibels: 293
Violin is playing
time: 37.5,decibels: 388
Violin is playing
time: 45.0,decibels: 261
Violin is playing
time: 52.5,decibels: 55
Violin is not playing
time: 60.0,decibels: 709
Violin is playing
```

```
time: 0.0,decibels: 1098
Trumpet is not playing
time: 7.5,decibels: 2082
Trumpet is not playing
time: 15.0,decibels: 2911
Trumpet is not playing
time: 22.5,decibels: 2860
Trumpet is not playing
time: 30.0,decibels: 3596
Trumpet is not playing
time: 37.5,decibels: 3738
Trumpet is not playing
time: 45.0,decibels: 3734
Trumpet is not playing
time: 52.5,decibels: 2797
Trumpet is not playing
time: 60.0,decibels: 4077
Trumpet is playing
```

Figure 9: Recognition of instruments for segments of the audio signal.

4. Conclusion

The assignment was to recognize curtain instruments in an orchestra, for the implementation the project group realized that frequency ranges of instruments overlapped, which caused the filtering and recognition to be less reliable than desired.

5. Recommendation

While there were issues of frequency ranges overlapping, we believe this can be resolved by implementing AI or ML, the reason as to why this hadn't been applied is down to not being a relevant part of the DSP course.

6. Literature References

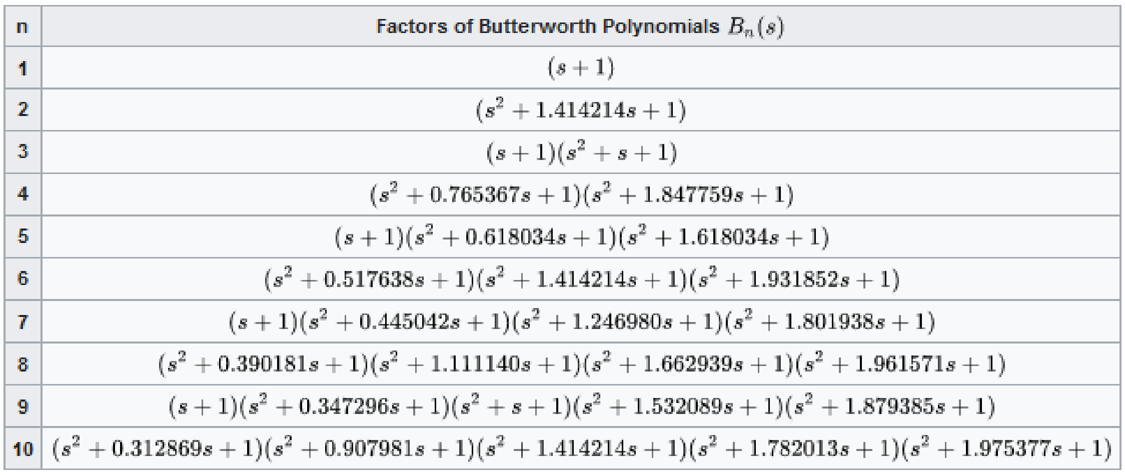
scipy.signal.lfilter — SciPy v1.10.0 Manual. (n.d.).
https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.lfilter.html

Electrical4U. (2021, April 16). *Butterworth Filter: What is it? (Design & Applications) | Electrical4U*. https://www.electrical4u.com/butterworth-filter/

Tech Stuff - Frequency Ranges. (n.d.). https://www.zytrax.com/tech/audio/audio.html

Davis, Nick. "An Introduction to Filters." *Allaboutcircuits.com*, 31 July 2017,
www.allaboutcircuits.com/technical-articles/an-introduction-to-filters/

7. Appendix



n	Factors of Butterworth Polynomials $B_n(s)$
1	$(s+1)$
2	$(s^2+1.414214s+1)$
3	$(s+1)(s^2+s+1)$
4	$(s^2+0.765367s+1)(s^2+1.847759s+1)$
5	$(s+1)(s^2+0.618034s+1)(s^2+1.618034s+1)$
6	$(s^2+0.517638s+1)(s^2+1.414214s+1)(s^2+1.931852s+1)$
7	$(s+1)(s^2+0.445042s+1)(s^2+1.246980s+1)(s^2+1.801938s+1)$
8	$(s^2+0.390181s+1)(s^2+1.111140s+1)(s^2+1.662939s+1)(s^2+1.961571s+1)$
9	$(s+1)(s^2+0.347296s+1)(s^2+s+1)(s^2+1.532089s+1)(s^2+1.879385s+1)$
10	$(s^2+0.312869s+1)(s^2+0.907981s+1)(s^2+1.414214s+1)(s^2+1.782013s+1)(s^2+1.975377s+1)$

Figure 10: Figure showing polynomials of a butterworth filter.