# Relatório de Processamento Digital de Sinais - Aplicação de filtros digitais

José Adrian, Robert Vinicius

September 23, 2024

## 1 Introdução

Este relatório tem como objetivo descrever o processo de aplicação de um filtro digital rejeita-banda em um sinal de áudio com ruído, utilizando um filtro Butterworth. O filtro rejeita uma faixa de frequências específica para remover o ruído presente no sinal. Será apresentado o código, gráficos de espectro de frequências, formas de onda e a análise da relação sinal-ruído (SNR).

## 2 Teoria

Filtros rejeita-banda são úteis para eliminar uma faixa de frequências específica, como ruídos ou interferências em um sinal. O filtro Butterworth utilizado neste projeto é conhecido por ter uma resposta em frequência plana na banda passante, minimizando ondulações indesejadas. A implementação do filtro é feita usando a função `butter` da biblioteca `scipy.signal`, que calcula os coeficientes do filtro para uma determinada ordem.

### 2.1 Relação Sinal-Ruído (SNR)

A relação sinal-ruído (SNR) é uma medida da qualidade de um sinal com ruído. Ela é definida como a razão entre a potência do sinal e a potência do ruído, e pode ser expressa em decibéis (dB) pela fórmula:

$$\text{SNR (dB)} = 10 \log_{10} \left( \frac{P_{\text{signal}}}{P_{\text{noise}}} \right) \tag{1}$$

## 3 Código Implementado

O código implementado carrega um arquivo de áudio com ruído, aplica um filtro rejeita-banda e calcula a potência do ruído antes e depois da filtragem. Nesta seção, cada função usada no código será explicada.

### 3.1 Função calcular_snr

A função `calcular_snr` é utilizada para calcular a relação sinal-ruído (SNR) em decibéis (dB). Ela recebe como parâmetros a potência do sinal útil e a potência do ruído, retornando a SNR. A fórmula utilizada para esse cálculo é:

$$SNR(dB) = 10 \log_{10} \left( \frac{P_{\text{signal}}}{P_{\text{noise}}} \right) \tag{2}$$

```
def calcular_snr(signal_power, noise_power):
    return 10 * np.log10(signal_power / noise_power)
```

### 3.2 Função potencia_sinal

A função `potencia_sinal` calcula a potência de um sinal de áudio, utilizando a fórmula da potência média de um sinal, que é a média dos quadrados das amplitudes dos samples. Isso é feito para estimar a energia do sinal ou do ruído ao longo do tempo.

```
def potencia_sinal(sinal):
    return np.mean(np.square(sinal))
```

### 3.3 Função butter_bandstop

A função `butter_bandstop` cria um filtro rejeita-banda Butterworth. Ela recebe como parâmetros as frequências de corte inferior e superior (lowcut e highcut), a frequência de amostragem do sinal (`fs`), e a ordem do filtro. A função converte as frequências de corte para a escala de Nyquist e usa a função `butter` da biblioteca `scipy.signal` para calcular os coeficientes do filtro.

```
def butter_bandstop(lowcut, highcut, fs, order=4):
    nyquist = 0.5 * fs
    low = lowcut / nyquist
    high = highcut / nyquist
    b, a = butter(order, [low, high], btype='bandstop')
    return b, a
```

O parâmetro `order` define a ordem do filtro, que controla a inclinação da resposta do filtro em torno das frequências de corte. Quanto maior a ordem, mais abrupta será a transição entre as bandas passantes e as bandas rejeitadas.

### 3.4 Função bandstop_filter

A função `bandstop_filter` aplica o filtro rejeita-banda ao sinal de áudio. Ela usa a função `butter_bandstop` para obter os coeficientes do filtro e, em seguida, aplica o filtro usando a função `lfilter`, também da biblioteca `scipy.signal`. O resultado é o sinal filtrado, onde as frequências dentro da faixa rejeitada (definida pelos parâmetros lowcut e highcut) são atenuadas.

```
def bandstop_filter(data, lowcut, highcut, fs, order=4):
    b, a = butter_bandstop(lowcut, highcut, fs, order=order)
    y = lfilter(b, a, data)
    return y
```

### 3.5 Fluxo Principal do Código

O código principal executa as seguintes etapas:

1. Carrega o sinal de áudio com ruído a partir de um arquivo usando a biblioteca `soundfile`.
2. Define os parâmetros do filtro rejeita-banda, incluindo as frequências de corte inferior e superior.
3. Aplica o filtro ao sinal de áudio utilizando a função `bandstop_filter`.
4. Calcula a relação SNR antes e depois da aplicação do filtro, utilizando as funções `potencia_sinal` e `calcular_snr`.
5. Exibe os valores de SNR e salva o áudio filtrado em um novo arquivo.

Listing 1: Código Python para Filtragem de Áudio

```
import numpy as np
from scipy.signal import butter, lfilter
import soundfile as sf
import matplotlib.pyplot as plt

# Caminho para o arquivo de udio com ru do
input_audio_noisy_path = 'audio_ruido.wav'
output_audio_filtered_path = 'audio_bandstop_filtered.wav'

# Carregar o sinal de udio com ru do
audio_noisy, sr = sf.read(input_audio_noisy_path)

# Definir a faixa de frequ ncias para o filtro rejeita-banda
# (2200 Hz a 10000 Hz)
lowcut = 2200.0
highcut = 10000.0

# Aplicar o filtro rejeita-banda para remover o ru do
audio_filtered = bandstop_filter(audio_noisy, lowcut, highcut, sr, order=15)

# Salvar o udio filtrado
sf.write(output_audio_filtered_path, audio_filtered, sr)

# Estimar o ru do usando os primeiros 0.5 segundos do udio original
```

```
inicio_ruido = 0
fim_ruido = int(.5 * sr)
ruido_est = audio_noisy[inicio_ruido:fim_ruido]

# Calcular a potncia do sinal til e do ruido
potencia_sinal_util = potencia_sinal(audio_noisy)
potencia_ruido_antes = potencia_sinal(ruido_est)
potencia_ruido_depois = potencia_sinal(audio_filtered[inicio_ruido:fim_ruido])

# Calcular SNR antes e depois da filtragem
snr_antes = calcular_snr(potencia_sinal_util, potencia_ruido_antes)
snr_depois = calcular_snr(potencia_sinal_util, potencia_ruido_depois)

print(f"SNR antes da filtragem: {snr_antes:.2f} dB")
print(f"SNR depois da filtragem: {snr_depois:.2f} dB")
```

# 4 Resultados

Os resultados da filtragem podem ser observados nos gráficos abaixo, onde é comparado o espectro de frequências do áudio original com ruído e do áudio filtrado, bem como as formas de onda antes e depois da filtragem.


Figure 1: Espectro de Frequências - Áudio com Ruído

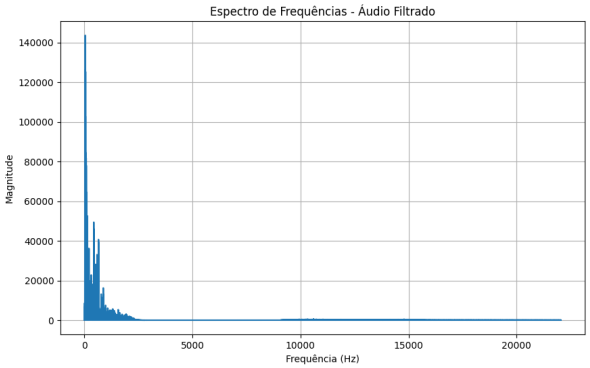


Figure 2: Espectro de Frequências - Áudio Filtrado

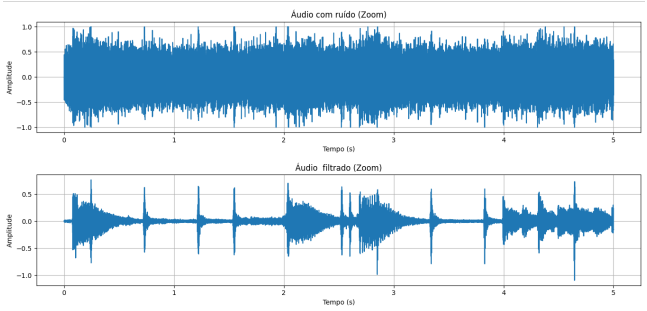


Figure 3: Formas de Onda - Áudio com Ruído e Áudio Filtrado (Zoom)

# 5 Conclusão

A aplicação do filtro rejeita-banda foi eficaz em reduzir o ruído presente no sinal de áudio. Isso pode ser verificado pelo aumento da relação SNR de 2.42 dB antes da filtragem para 7.75 dB após a filtragem. Os gráficos gerados mostram a redução na magnitude do ruído nas frequências rejeitadas, assim como a clareza do sinal filtrado comparado ao sinal original.