# KELOMPOK 5 SINYAL DAN SISTEM

Yusuf Andrian H1A024105

Henkel Raditya H1A024121

Adam H1A024125

Ghozi Alfalah H. H1A024139

## Chapter 3 : Fourier Series

### Analisis Spektrum

Pada percobaan ini, beberapa sampel suara dipilih dan dianalisis dengan bantuan aplikasi Audacity. Setiap sampel diberi perlakuan menggunakan filter High-pass, Low-pass, dan efek Equalizer (AM Radio). Dengan memanfaatkan fitur Plot Spectrum pada Audacity, dapat diamati perbedaan distribusi frekuensi dan karakter suara yang dihasilkan setelah diterapkan filter atau efek tertentu.

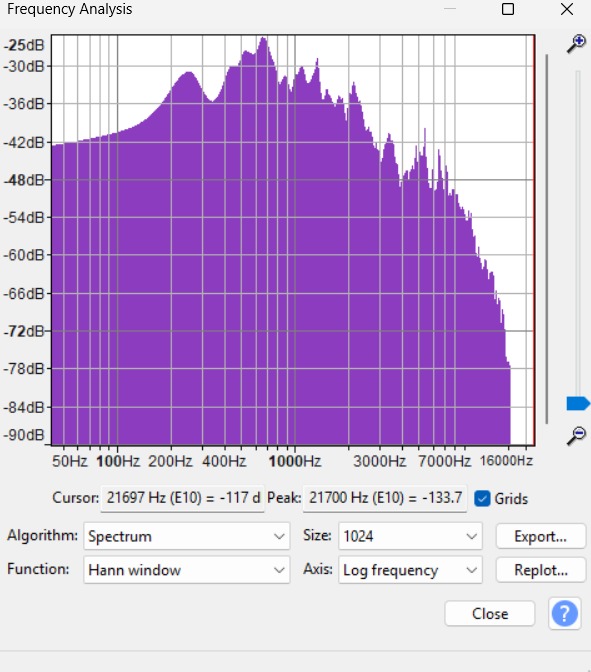
### Dasar Teori Singkat

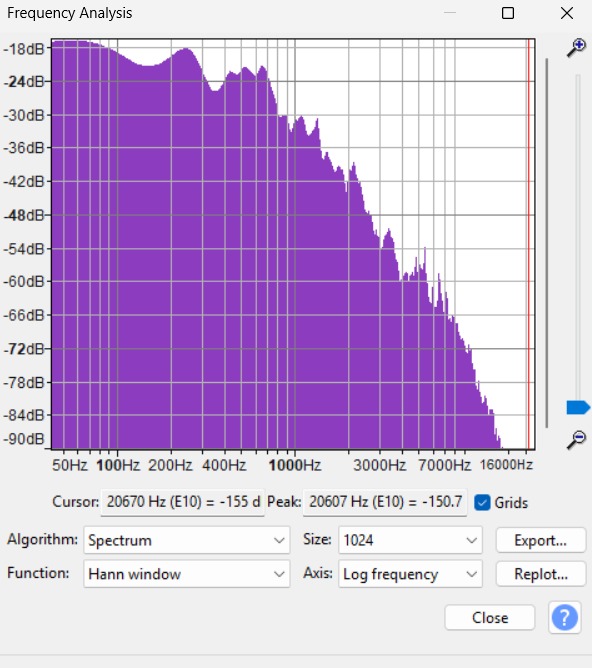
Audacity adalah perangkat lunak open-source yang digunakan untuk merekam, mengedit, dan menganalisis audio digital. Aplikasi ini menyediakan berbagai fitur dasar seperti perekaman, pemotongan, penggabungan audio, hingga analisis spektrum. Salah satu fitur yang penting adalah Plot Spectrum yang menampilkan grafik hubungan frekuensi terhadap amplitudo. Selain itu, Audacity juga menyediakan efek Equalizer dan filter (Low-pass, High-pass, Band-pass, dan lainnya) yang dapat digunakan untuk memodifikasi kualitas audio sesuai kebutuhan.

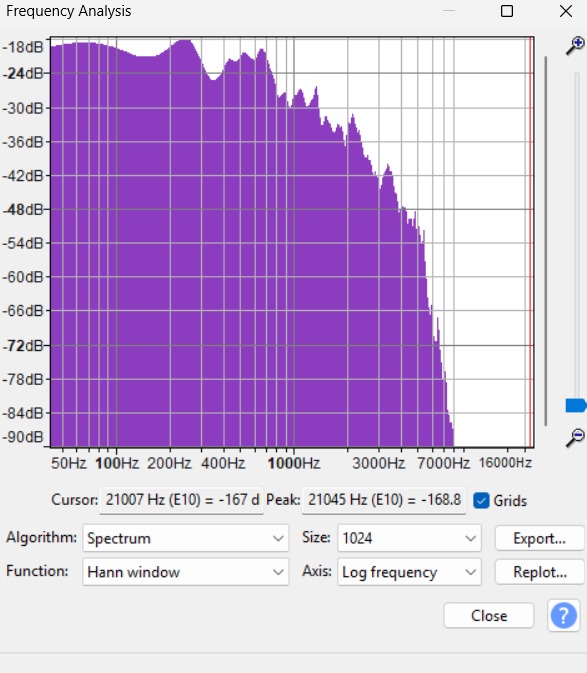
### Efek EQ (Equalization) dan Filter

Equalization (EQ) adalah proses pengaturan level (gain) pada rentang frekuensi tertentu dalam sinyal audio. EQ memungkinkan pengguna menonjolkan atau meredam frekuensi rendah (bass), menengah (mid), dan tinggi (treble). Filter audio sendiri merupakan bentuk penyaringan frekuensi:   
- High-pass filter (HPF): melewatkan frekuensi tinggi dan menahan frekuensi rendah.  
- Low-pass filter (LPF): melewatkan frekuensi rendah dan menahan frekuensi tinggi.  
- Equilizer AM Radio : memangkas frekuensi rendah maupun tinggi, sehingga hanya frekuensi menengah yang dipertahankan

### Sample 1 : Lagu Fly Me To The Moon

1. Grafik High-pass Filter

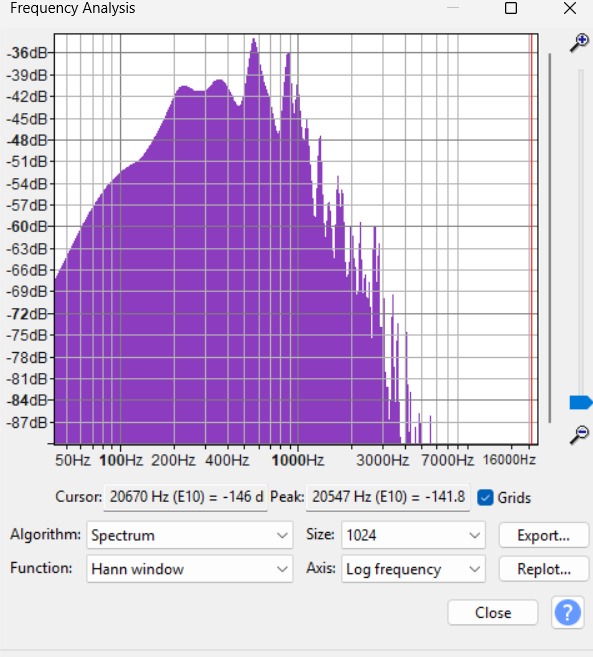
2. Grafik Low-pass Filter

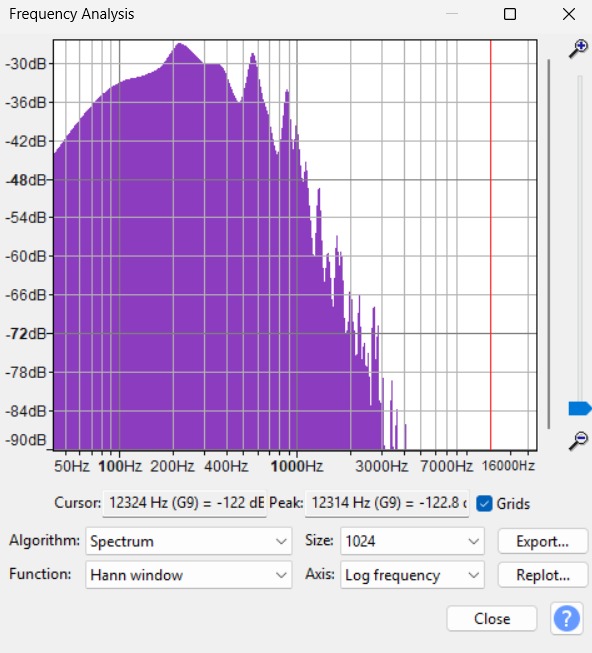
3. Grafik Equalizer AM Radio

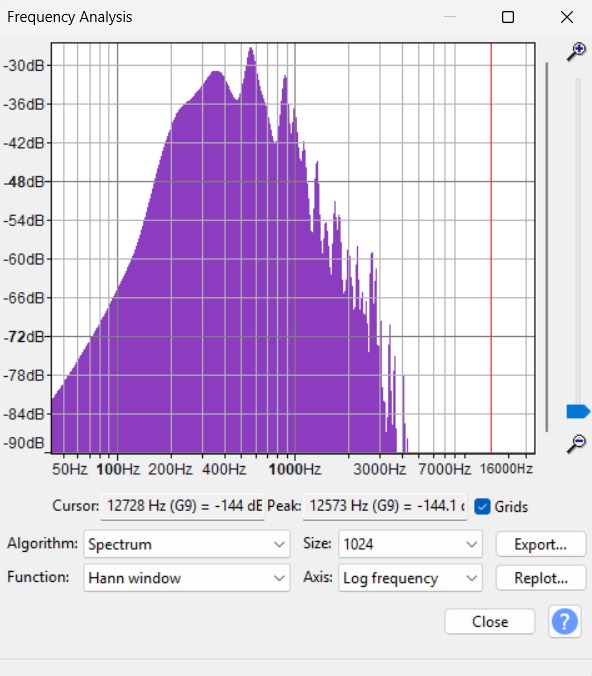
#### Pembahasan

Pada sampel lagu ini, penerapan High-pass filter membuat suara frekuensi rendah seperti bass berkurang, sehingga vokal dan instrumen dengan nada tinggi terdengar lebih jelas. Sebaliknya, dengan Low-pass filter, frekuensi tinggi dipotong sehingga lagu terdengar lebih 'boomy' dan dominan bass. Sementara itu, efek Equalizer AM Radio menghasilkan suara dengan karakteristik mirip radio lama, di mana frekuensi rendah dan tinggi melemah drastis sehingga hanya frekuensi menengah yang dominan. Hasilnya, lagu terdengar lebih tipis dan bernuansa retro.

### Sample 2 : Suara Piano (River Flows in You)

1. Grafik High-pass Filter

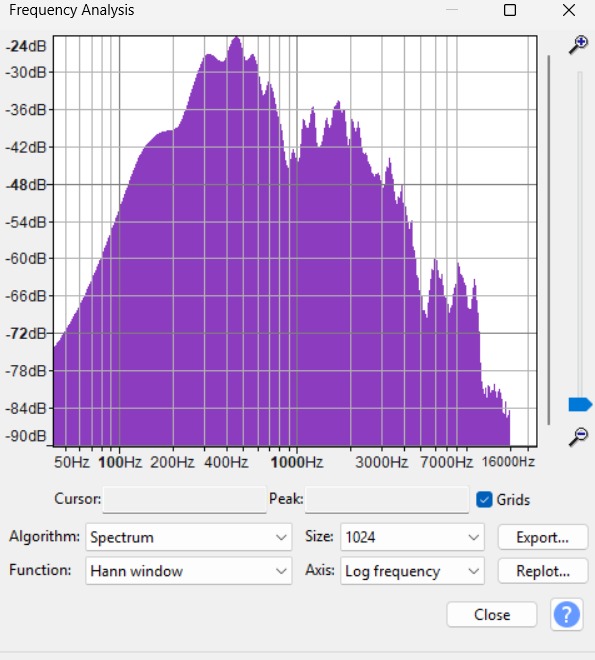
2. Grafik Low-pass Filter

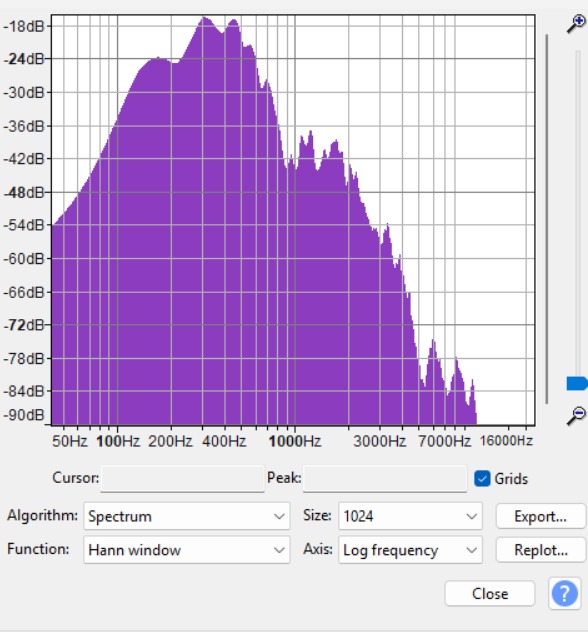
3. Grafik Equalizer AM Radio

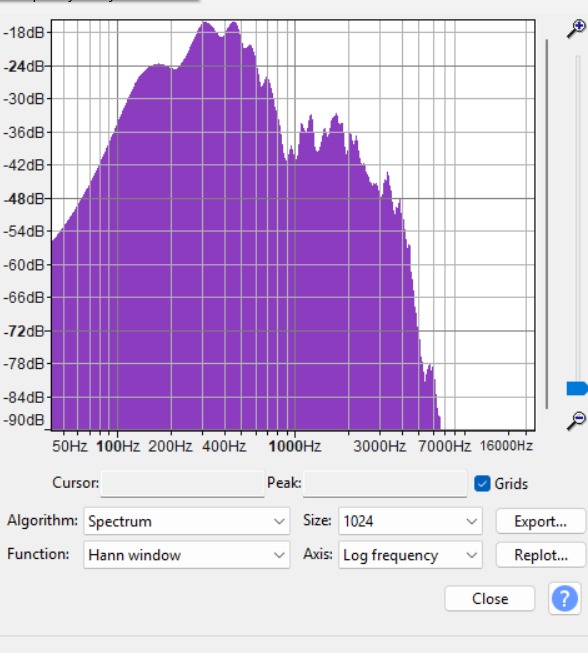
#### Pembahasan

Pada instrumen piano, penggunaan High-pass filter menghilangkan sebagian nuansa bass sehingga nada tinggi lebih dominan dan suara piano terdengar ringan. Low-pass filter sebaliknya menekankan bass dan mengurangi detail nada tinggi, menyebabkan suara terdengar lebih berat dan tebal. Dengan efek AM Radio, suara piano menjadi terbatas pada frekuensi menengah, memberi kesan seperti dimainkan melalui perangkat radio lama dengan kualitas yang lebih sempit.

### Sample 3 : Sound Meme

1. Grafik High-pass Filter

2. Grafik Low-pass Filter

3. Grafik Equalizer AM Radio

#### Pembahasan

Untuk sampel suara meme, High-pass filter menyingkirkan suara bass sehingga hasilnya lebih nyaring dan agak 'menusuk' di telinga. Low-pass filter justru membuat efek suara terdengar lebih dalam dengan detail frekuensi tinggi yang hilang. Efek AM Radio memberikan nuansa lucu dan retro karena hanya menekankan frekuensi menengah, sehingga suara terdengar datar dan sempit.

## Fourier Series Grafik

Selain analisis audio dengan filter, percobaan juga membuat sinyal periodik berupa gelombang sawtooth. Gelombang ini dibangkitkan dengan menggunakan bahasa pemrograman Python dan divisualisasikan menggunakan matplotlib. Sinyal sawtooth kemudian dapat dianalisis menggunakan metode Fourier Series untuk menunjukkan bahwa sinyal periodik dapat diuraikan menjadi penjumlahan komponen sinus dan kosinus dengan frekuensi kelipatan dari frekuensi dasar.

Berikut kode program pembuatan gelombang sawtooth:  
  
import numpy as np

import wave

import matplotlib.pyplot as plt

# ==== 1. Generator gelombang sawtooth ====

def sawtooth\_wave(freq=440.0, duration=10.0, sr=44100, amp=0.5):

    """

    Membuat gelombang sawtooth sederhana.

    - freq: frekuensi nada (Hz)

    - duration: durasi (detik)

    - sr: sample rate

    - amp: amplitudo (0–1)

    """

    t = np.linspace(0, duration, int(sr\*duration), endpoint=False)

    # rumus sawtooth = 2\*(t\*freq - floor(t\*freq + 0.5))

    y = amp \* (2 \* (t \* freq - np.floor(t \* freq + 0.5)))

    return sr, y

# ==== 2. Fungsi simpan WAV stereo ====

def write\_wav\_stereo(filename, sr, left, right):

    """

    Menyimpan data numpy array sebagai file WAV stereo 16-bit.

    """

    with wave.open(filename, "w") as f:

        f.setnchannels(2)       # stereo

        f.setsampwidth(2)       # 16-bit

        f.setframerate(sr)

        # ubah ke int16 dan interleave

        left\_int = np.int16(np.clip(left, -1, 1) \* 32767)

        right\_int = np.int16(np.clip(right, -1, 1) \* 32767)

        stereo = np.column\_stack((left\_int, right\_int)).ravel()

        f.writeframes(stereo.tobytes())

# ==== 3. Contoh penggunaan ====

sr, left = sawtooth\_wave(freq=220.0, duration=5.0, sr=44100, amp=0.8)

\_, right = sawtooth\_wave(freq=330.0, duration=5.0, sr=44100, amp=0.8)  # sedikit beda frekuensi

# Hitung panjang minimum antara kiri dan kanan

min\_len = min(len(left), len(right))

# Potong supaya sama panjang

left = left[:min\_len]

right = right[:min\_len]

# Baru simpan jadi file WAV stereo

write\_wav\_stereo("sawtooth\_stereo.wav", sr, left, right)

print("File 'sawtooth\_stereo.wav' berhasil dibuat!")

# ==== 4. Visualisasi ====

time\_axis = np.arange(0, 1000) / sr  # ambil 1000 sample pertama (≈ 0.02 detik)

plt.figure(figsize=(10,4))

plt.plot(time\_axis, left[:1000], label="Left channel (220 Hz)")

plt.plot(time\_axis, right[:1000], label="Right channel (330 Hz)", alpha=0.7)

plt.title("Gelombang Sawtooth Stereo")

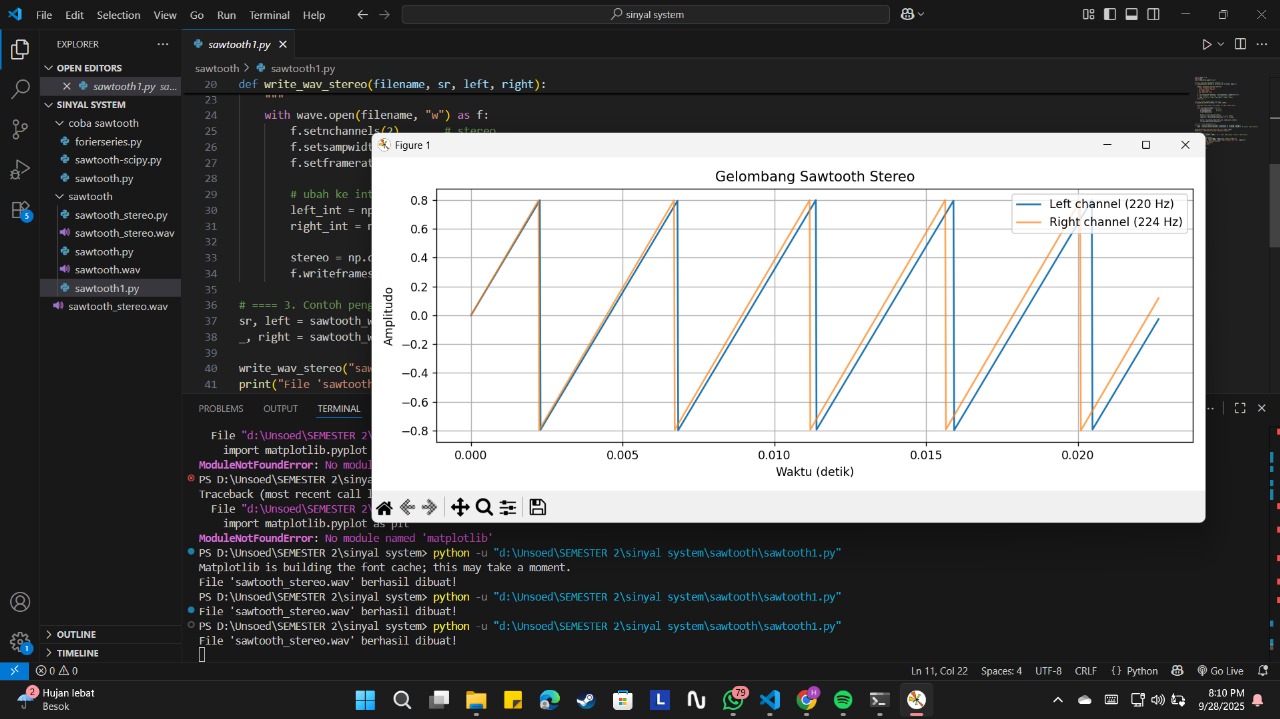
plt.xlabel("Waktu (detik)")

plt.ylabel("Amplitudo")

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()  
  
Lampiran :

#### Penjelasan

Kode program di atas membuat gelombang sawtooth dengan frekuensi tertentu dan menyimpannya dalam format WAV stereo. Setiap channel (kiri dan kanan) dapat diberikan frekuensi yang berbeda sehingga menghasilkan kombinasi suara harmonis. Visualisasi dilakukan dengan matplotlib, yang memperlihatkan bentuk khas gelombang sawtooth: naik secara linier lalu turun mendadak. Metode Fourier Series kemudian dapat digunakan untuk menganalisis gelombang ini menjadi harmonik-harmonik penyusunnya.