**Kelompok 8**

**Daffa Novandra S (H1A024109)**

**RAFAEL SIMANJUNTAK (H1A024117)**

**ARYA SAMANDIKA (H1A024109)**

**Chapter 3 : Fourier Series**

Analisis Spektrum

"Pada percobaan ini, sampel suara yang sudah disiapkan akan dianalisis spektrumnya menggunakan fitur Plot Spectrum pada aplikasi Audacity. Dengan fitur ini, dapat ditampilkan grafik hubungan antara frekuensi dan kekuatan suara dari sampel yang digunakan."

**Dasar Teori Singkat**  
**Audacity**  
Audacity merupakan perangkat lunak open-source yang berfungsi untuk merekam, mengedit, sekaligus menganalisis audio digital melalui perangkat keras seperti laptop. Aplikasi ini populer di bidang rekayasa audio, penelitian, hingga kegiatan akademik karena bersifat lintas platform, ringan, mudah diakses, dan memiliki fitur dasar yang lengkap. Dengan Audacity, pengguna dapat merekam suara, memotong atau menggabungkan audio, menambahkan efek, serta melakukan analisis spektrum sinyal dalam bentuk grafik.

**Efek EQ (Equalization) dan Filter**  
Equalization (EQ) adalah proses pengaturan gain pada frekuensi tertentu dalam sinyal audio, yang bertujuan untuk menonjolkan, mengurangi, atau menyeimbangkan karakter suara.

* Frekuensi rendah (bass): memberi kesan lebih berat atau mengurangi bunyi yang berlebihan.
* Frekuensi menengah (mid): berpengaruh pada kejernihan vokal dan instrumen.
* Frekuensi tinggi (treble): menambah detail atau mengurangi suara mendesis.

Filter audio sendiri digunakan untuk menyaring frekuensi tertentu. Contohnya:

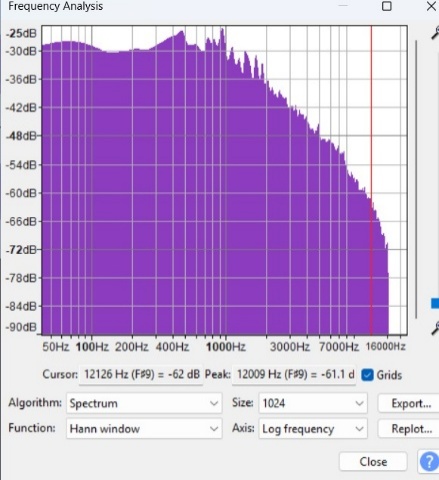
* **Low-pass filter (LPF):** melewatkan frekuensi rendah dan menahan frekuensi tinggi.
* **High-pass filter (HPF):** melewatkan frekuensi tinggi dan menahan frekuensi rendah.
* **Band-pass filter (BPF):** melewatkan frekuensi dalam rentang tertentu.
* **Notch filter:** meredam frekuensi sempit, misalnya untuk menghilangkan noise 50 Hz.

Dalam Audacity, efek EQ dan filter berfungsi untuk memperbaiki kualitas audio, mengurangi noise, serta membentuk karakter suara sesuai kebutuhan pengguna.

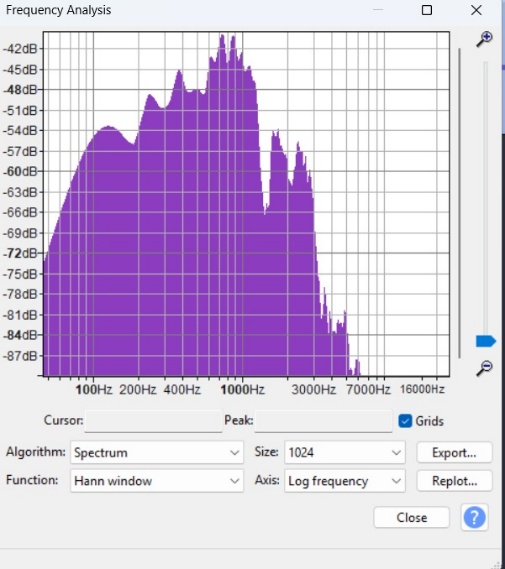
**Sampel 1 : Suara Rekaman**

**Graphics Plot**

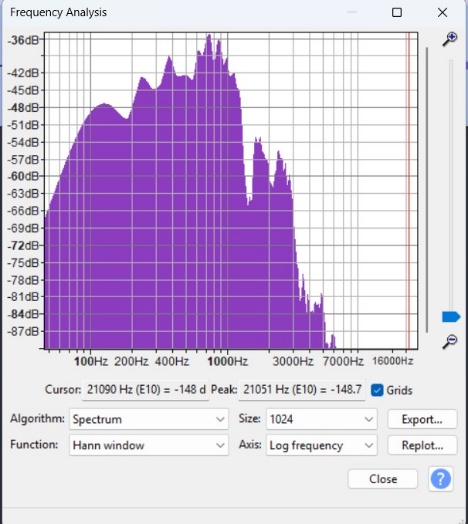
**1.Grafik** **High-pass filter**



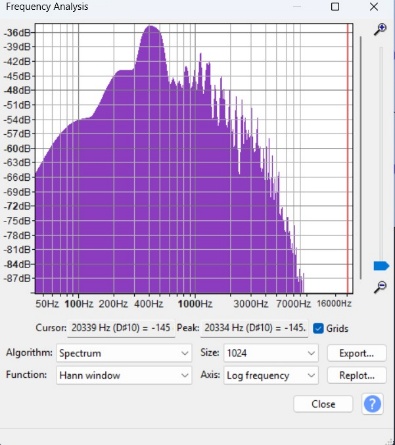
**2. Grafik Low-pass filter**



**3. Grafik Notch filter**



**4. grafik Shelf Filter**

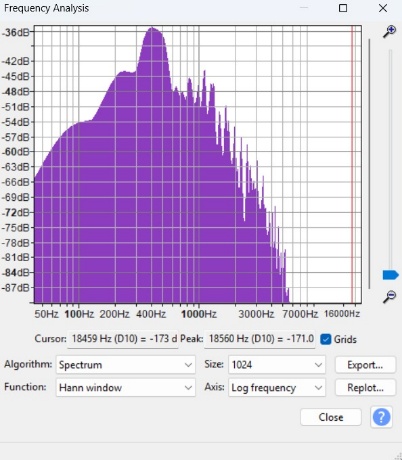


**Pembahasan**  
Penerapan efek **EQ Bass Boost** pada voice note menambah intensitas frekuensi rendah (bass). Akibatnya, suara piano terdengar lebih tebal dan hangat, dengan dominasi nada rendah yang semakin jelas. Pada tampilan spektrum, amplitudo di frekuensi rendah meningkat, sementara frekuensi tinggi cenderung tetap atau sedikit menurun. Efek ini efektif digunakan untuk memperkuat karakter bass pada instrumen dalam voice note.

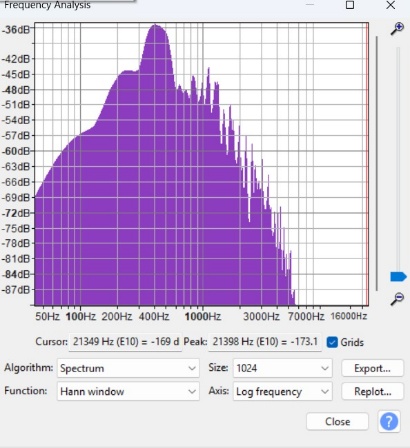
**SAMPEL 2 : SUARA LAGU JVKE-GOLDDEN HOWAR**

**Graphics Plot**

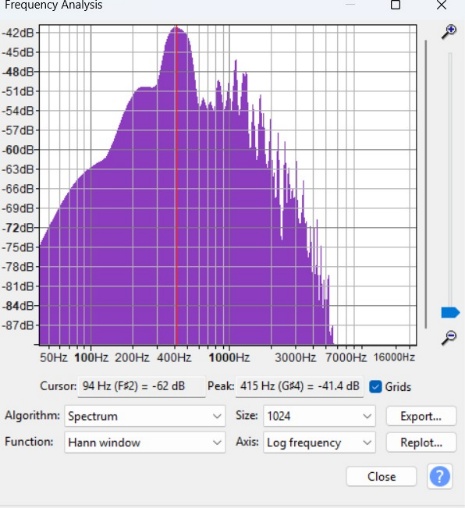
**1.Grafik** **High-pass filter**



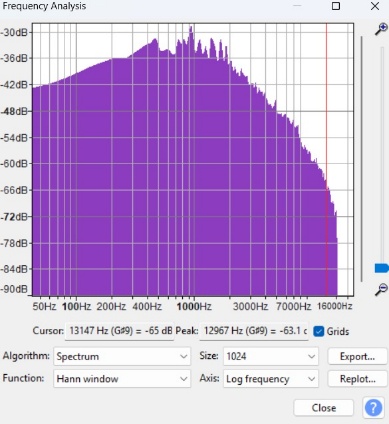
**2. Grafik Low-pass filter**



**3. Grafik Notch filter**



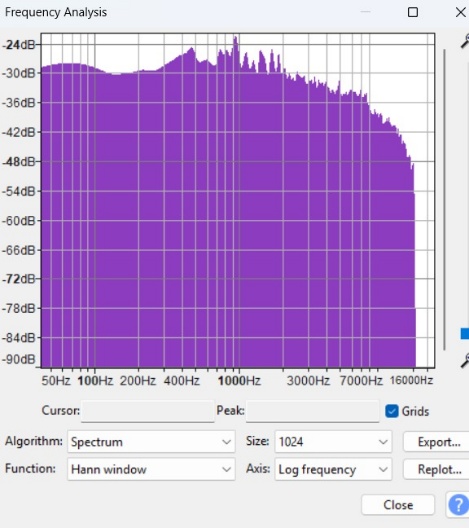
**4. grafik Shelf Filter**



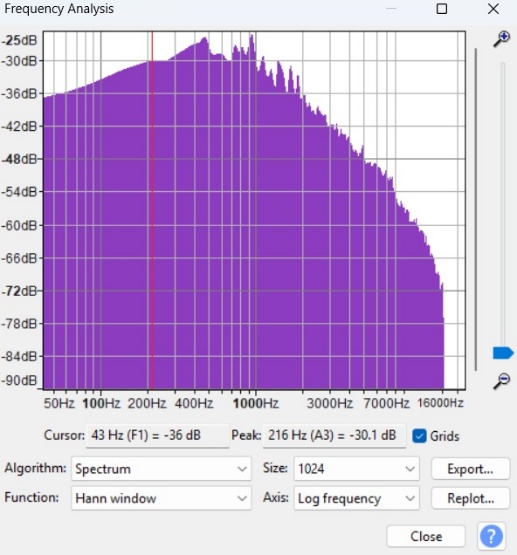
**Pembahasan  
Efek EQ Walkie-Talkie bekerja dengan memangkas frekuensi rendah maupun tinggi, sehingga hanya frekuensi menengah yang dipertahankan. Rekaman suara yang dihasilkan terdengar tipis dan menyerupai suara komunikasi radio, di mana vokal pada area tengah tetap jelas, namun detail bass dan treble hilang. Pada grafik spektrum, bagian frekuensi menengah terlihat dominan, sementara area rendah dan tinggi melemah.**

**Sampel 3 : Suara Lagu JJ**

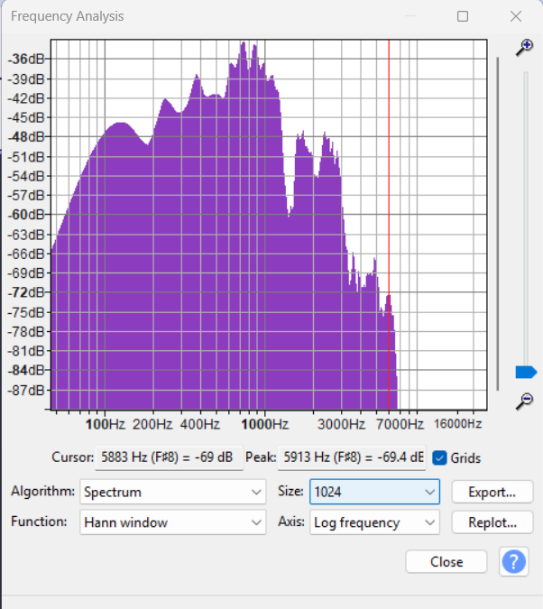
**1.Grafik** **High-pass filter**



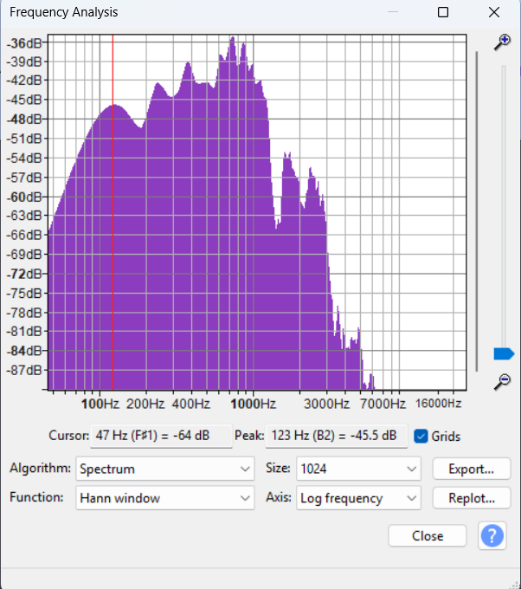
**2. Grafik Low-pass filter**



**3. Grafik Notch filter**

****

**4. grafik Shelf Filter**

****

**Pembahasan**

Efek EQ AM Radio menghasilkan karakter suara mirip radio lama dengan rentang frekuensi terbatas dan penekanan pada frekuensi menengah. Hasilnya, suara lagu terdengar bernuansa retro dan kurang detail, sehingga memberi kesan nostalgis. Pada spektrum, frekuensi rendah maupun tinggi berkurang drastis, sementara frekuensi menengah tampak dominan.

**Fourier Series Grafik**  
Dalam percobaan ini dibuat grafik gelombang *sawtooth* menggunakan MATLAB atau Python (Matplotlib). Sinyal tersebut kemudian dianalisis dengan metode **Fourier Series** untuk memecahnya menjadi komponen harmonik, sekaligus menuliskan persamaan Fourier Series yang diperoleh.

**Dasar Teori**  
**Fourier Series** adalah metode matematis yang merepresentasikan sinyal periodik sebagai hasil penjumlahan fungsi sinus dan kosinus dengan frekuensi kelipatan dari frekuensi dasarnya. Gagasan ini diperkenalkan oleh Joseph Fourier, yang membuktikan bahwa setiap sinyal periodik dapat diuraikan ke dalam komponen harmoniknya.

**Gelombang Sawtooth**  
Gelombang sawtooth merupakan jenis sinyal periodik yang umum digunakan dalam bidang elektronika, musik digital, dan sistem komunikasi. Ciri khasnya adalah bentuk menyerupai gigi gergaji, dengan pola kenaikan linier yang diikuti penurunan mendadak.

Program Gelombang Kotak

import numpy as np

import matplotlib.pyplot as plt

from scipy.signal import square

def run\_square(amplitude, frequency, duration, fs=1000):

"""

Fungsi untuk menampilkan gelombang kotak (square wave).

Parameters:

amplitude (float): Amplitudo gelombang

frequency (float): Frekuensi gelombang (Hz)

duration (float): Durasi sinyal (s)

fs (int): Sampling rate (default = 1000)

"""

# Vektor waktu

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

# Gelombang kotak

signal = amplitude \* square(2 \* np.pi \* frequency \* t)

# Plot

plt.figure(figsize=(8, 4))

plt.plot(t, signal, color='blue', label=f"Square Wave: A={amplitude}, f={frequency}Hz, T={duration}s")

plt.title("Square Wave Menggunakan Matplotlib")

plt.xlabel("Waktu (s)")

plt.ylabel("Amplitudo")

plt.grid(True)

plt.legend()

plt.show()

# Contoh penggunaan

run\_square(1, 5, 1) # Amplitudo = 1, Frekuensi = 5 Hz, Durasi = 1 detik

**Program Gelombang Sawtooth**

import numpy as np

import matplotlib.pyplot as plt

from scipy.signal import sawtooth

# Parameter

amp = 1 # Amplitudo

freq = 5 # Frekuensi (Hz)

dur = 1 # Durasi (s)

fs = 1000 # Sampling rate

# Vektor waktu

t = np.arange(0, dur, 1/fs)

# Gelombang sawtooth

sw = amp \* sawtooth(2 \* np.pi \* freq \* t)

# Plot

plt.figure(figsize=(8,4))

plt.plot(t, sw, color='red')

plt.title('Sawtooth Wave')

plt.xlabel('Time (s)')

plt.ylabel('Amplitude')

plt.grid(True)

plt.show()