BAB 3
METODE PENELITIAN

3.1 Metode Penelitian

Dalam melakukan kegiatan perancangan ini memerlukan peralatan berupa perangkat lunak dan perangkat keras. Perangkat lunak disini digunakan sebagai simulasi dalam melakukan kegiatan filter suara. Sedangkan perangkat keras sendiri digunakan pensimulasi dan pengukuran dalam pengambilan suara.

3.1.1 Perangkat Keras

Perangkat keras yang digunakan dalam melakukan pensimulasian disini hanya laptop, spesifikasi seprti pada Gambar 3.1.

Tabel 3. 1 Spesifikasi Perangkat Lunak.

Siste Operasi	*Windows 11 Home Single Language*
Processor	*Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz*
RAM	*4,00 GB (3,75 GB usable)*

3.1.2 Perangkat Lunak

1. Matlab 2016, perangkat lunak ini digunakan untuk simulasi filter suara pada *low pass filter* dengan frekuensi *cut off* 4000 Hz.
2. *Microsoft Word 2021,* perangkat lunak ini digunakan untuk mencatat hasil review journal dan daftar Pustaka untuk dilakukan perbandingan.

3.2 Alur Penelitian

Adapun beberapa tahapan dalam melakukan penelitian yang akan dilaksanakan dengan menentukan frekuensi *cut-off* pada *low pass filter* yang sudah ditetapkan yaitu 4 KHz, 10 KHz, dan 20 KHz dengan menggunakan metode *Elliptic.* Dengan adanya metode yang sudah ditentukan yaitu *Elliptic* pada *Low Pass Filter* pemfilteran dapat berjalan, yaitu frekuensi diatas *cut-off* di redam sedangkan dibawah frekuensi *cut-off* diloloskan.

Gambar 3. 1 Diagram Alir Alur Penelitian.

Pada Gambar 3.1 menjelaskan mengenai alur tahapan dalam melaksanakan penelitian yang akan dilakukan agar peneliti dapat melakukannya secara terurut. Melalui gambar diagram alir pemahaman dalam penelitian akan semakin jelas, proses keseluruhan dan pelaksanaan setiap langkah akan dijelaskan. Berikut adalah penjelasan dari setiap proses yang akan dilakukan saat melakukan penelitian berdasarkan Gambar 3.1.

3.2.1 Studi Literatur

Proses ini merupakan tahap awal dalam melakukan kegiatan penelitian. Metode yang digunakan di dalam mengumpulkan data melalui studi literatur adalah menggunakan pedoman buku ilmiah, jurnal, dan website yang masih berkaitan dengan bahan, komponen, dan metode saja yang dibutuhkan dalam melakukan simulasi *Low Pass Filter* pada metode *Elliptic*. Dengan berbagai sumber yang sudah didapat sebelumnya sehingga dapat melakukan analisa untuk diterapkan pada penelitian selanjutnya.

3.2.2 Penentuan Metode

Metode yang digunakan yaitu *Low Pass Filter* dengan menggunakan metode *Elliptic*, karena metode ini merupakan filter yang paling baik digunakan karena memiliki rentang frekuensi yang luas, sehingga dengan menggunakan metode ini diharapkan peredaman dapat dilakukan dengan cara melewatkan frekuensi rendah atau memblokir frekuensi tinggi, untuk nilai frekuensi *cut-off* pada *Low Pass* i ini diantara 4 KHz, 10 KHz, dan 20 KHz.

3.2.3 Penentuan Frekuensi

Dalam menentukan frekuensi *cut-off* memerlukan beberapa data melalui studi literatur tersebut mendapatkan nilai ketetapan frekuensi *cut-off* pada *Low Pass Filter* melalui beberapa journal yaitu 4 KHz atau lebih. Sedangkan dalam penelitian ini frekuensi *cut-off* yang digunakan untuk perbandingan yaitu 4 KHz, 10 KHz, dan 20 KHz.

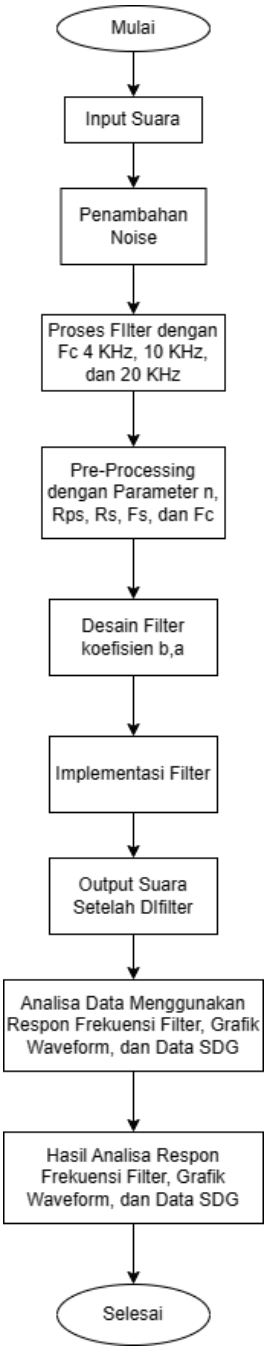
3.2.4 Pemfilteran Suara

Dalam menentukan frekuensi *cut-off* memerlukan beberapa data melalui studi literatur tersebut mendapatkan nilai ketetapan frekuensi *cut-off* pada *Low Pass Filter* melalui beberapa journal yaitu 4 KHz atau 4.5 KHz.

Pemfilteran dilakukan dengan *Low Pass Filter* dengan metode *Elliptic*, Filter jenis ini merupakan filter yang memiliki respon meloloskan sinyal dengan frekuensi dibawah frekuensi *cut-off* (Fc) dan meredam sinyal yang memiliki frekuensi diatas frekuensi *cut-off* yaitu diatas *4 KHz*.

3.3 PENGOLAHAN DATA

Untuk tahapan pengolahan data pada penelitian tugas akhir ini terdapat *block diagram* dan urainnya sebagai berikut :



Gambar 3. 2 Diagram Alir Pengolahan Data.

Pada Gambar 3.2 merupakan alur dalam pengolahan data atau proses dalam melakukan pemfilteran. Pengolahan data pada filter suara digital ini merupakan

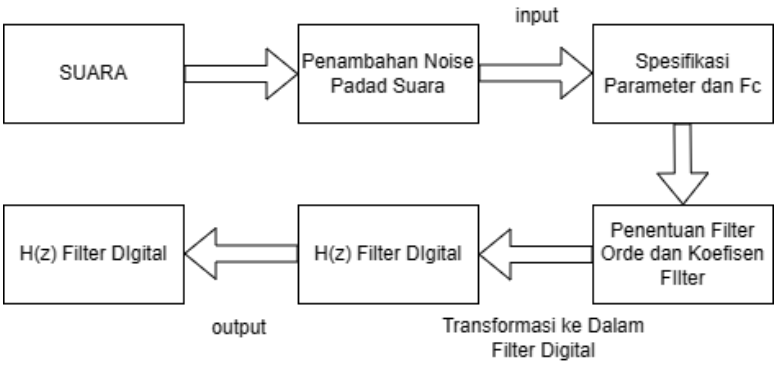
proses dimana mencari hasil output yang diinginkan dengan cara manipulasi atau transformasi data suara dalam domain digital. Filter suara digital ini juga dikatakan dapat mengubah karakteristik suara, seperti memperkuat atau melemahkan frekuensi tertentu, menghilangkan derau atau gangguan, serta menghasilkan efek khusus pada suara.

Berikut merupakan beberapa langkah umum dalam pengolahan data pada filter suara digital:

1. Perekaman Suara: Proses awal yang dilakukan pada tahap ini yaitu input suara kedalam *software* Matlab dengan mengimpor sample suara seperti potongan lagu yang terdapat pada web, suara yang akan digunakan berupa format .wav. Untuk memberikan efek *noise* diberikan tiga *sample noise* yang selanjutnya akan dilakukan pemfilteran, hal itu agar lebih mudah dalam melakukan perbandingan. Untuk membandingkan beberapa noise dan hasil filter tersebut dilakukan secara bergantian.
2. Pre-processing: Setelah menyiapkan *audio* yang akan difilter melanjutkan proses *pre-processing*. Pada Langkah *pre-processing* ini membuat Filter *Elliptic* LPF menggunakan Matlab. Filter *Elliptic low pass filter* adalah proses *filtering* pada data *audio* menggunakan *filter high pass* untuk menghilangkan *noise* pada frekuensi yang lebih rendah dari frekuensi *cut-off* pada filter *low pass*. Dalam program yang akan dilakukan terdapat jumlah parameter seperti orde (n), *frekuensi sampling* (fs), *ripple passband* (Rp) dalam dB, *ripple stopband* (Rs) dalam dB, Wn sendiri merupakan hasil pembagian antara frekuensi *cut-off* dan 0.5*fs yang besarnya antara 0 dan 1, dan terdapat *Ftype* (untuk *type* yang digunakan yaitu LPF). Hasil filter harus dipastikan bahwa sinyal *audio* telah dibersihkan dari noise pada frekuensi yang lebih rendah dari frekuensi *cut-off* pada *filter low pass.*
3. Desain Filter: Proses desain filter sendiri ini dapat melibatkan pemilihan jenis filter yang sesuai dengan tujuan pengolahan suara, seperti menentukan parameter filter seperti frekuensi *cut-off*. Pada tahapan desain filter pada filter *noise low pass filter* dengan metode *elliptic* yaitu parameter yang dibutuhkan seperti frekuensi *cut-off, ripple passband, attenuation* dan Wn.

4. Implementasi Filter: Filter yang sudah didesain selanjutnya diimplementasikan. Pengimpletasian filter ini sendiri menggunakan algoritma pemrosesan sinyal digital (DSP) atau perangkat lunak pengolahan suara pada data suara digital. Filter dapat diterapkan dalam domain waktu atau frekuensi, tergantung pada jenis filter yang digunakan.
5. Analisis Hasil: Setelah tahapan filter suara berhasil dan dapat diterapkan, analisa hasil dilakukan untuk menganalisa bahwa hasil pemfilteran tersebut dapat berfungsi sesuai yang diinginkan dengan menggunakan hasil seperti *respon frekuensi, waveform,* dan SDG. Grafik respon frekuensi menunjukkan seberapa baik sebuah perangkat audio dapat menghasilkan suara pada frekuensi tertentu. Waveform adalah bentuk gelombang yang menunjukkan variasi amplitudo atau kekuatan gelombang suara. *Subjective Difference Grade* (SDG) merupakan metode subjektif yang mengukur perbedaan persepsi kualitas suara antara dua sinyal oleh pendengar manusia.
6. Rata-rata SDG diperoleh dari beberapa responden, dengan membandingkan perubahan yang terjadi pada audio sebelum dan sesudah dilakukan pemfilteran, kemudian diberi nilai 1 hingga 5 Evaluasi dapat melibatkan perbandingan dengan suara asli, analisis spektral, atau pengukuran kualitas suara. Dalam penilaian tersebut responden diarahkan pada skala Tabel 2.4, bahwa semakin jernih hasil suara maka nilai skala dalam penilaian akan semakin tinggi. Hasil *audio* pemfilteran ada 3 yaitu Fc 1 dengan frekuensi *cut-off* 4 KHz, Fc 2 dengan frekuensi *cut-off* 10 KHz, dan Fc3 menggunakan frekuensi *cut-off* 20 KHz. Sehingga responden diberikan hasil ke-tiga *audio* tersebut untuk di nilai masing-masing dengan membandingkan audio asli yang telah ditambah *noise.*

Terdapat diagram blok pada penelitian ini. Blok diagram ini merupakan gambaran dasar mengenai sistem yang akan dirancang. Setiap bagian blok sistem memiliki fungsi masing-masing, dengan memahami gambar blok diagram maka sistem pemfilteran *low pass* dengan *elliptic* yang dirancang sudah dapat dibangun dengan baik. Adapun blok diagram yang akan dirancang seperti dicantumkan pada Gambar 3. 3.



Gambar 3. 3 Diagram Blok Pemfilteran

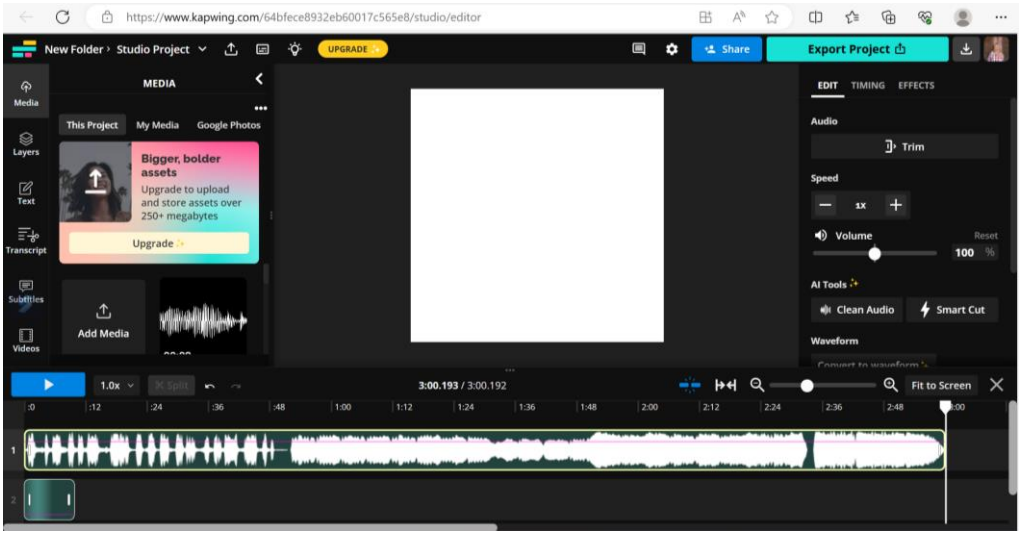
Dalam diagram blok di atas merupakan sistem pemfilteran *elliptic lowpass*, pengkonversian filter elliptic low pass analog ke dalam domain digital melibatkan serangkaian langkah untuk mengubah karakteristik filter dari domain kontinu menjadi domain diskret. Pertama, desain filter analog prototipe dilakukan berdasarkan spesifikasi yang diinginkan, seperti frekuensi cut-off, tingkat riple di passband, dan penolakan di stopband. Langkah berikutnya melibatkan transformasi dari domain analog ke domain digital. Setelah transformasi, fungsi transfer filter analog diubah menjadi fungsi transfer filter digital. Frekuensi dalam domain digital dinormalisasi agar sesuai dengan karakteristik diskret. Filter digital prototipe kemudian didesain berdasarkan fungsi transfer yang dihasilkan dari transformasi. Orde filter digital mungkin perlu disesuaikan agar respons frekuensinya sesuai dengan filter analog. Setelah desain, filter digital diimplementasikan dalam perangkat keras atau perangkat lunak sesuai dengan kebutuhan aplikasi. Filter yang diimplementasikan harus diuji dan divalidasi untuk memastikan respons frekuensi dan karakteristik lainnya sesuai dengan spesifikasi yang diinginkan.

BAB 4
ANALISIS DAN PEMBAHASAN

Dalam bab 4 ini membahas tentang implementasi dan bagaimana pengujian dilakukan untuk mengevaluasi kinerja filter *audio low pass filter* menggunakan metode *elliptic* untuk mendapatkan hasil grafik *respon frekuensi, grafik waveform,* dan rata-rata skala nilai *Subjective Difference Grade* (SDG).

4.1 Pengambilan Audio

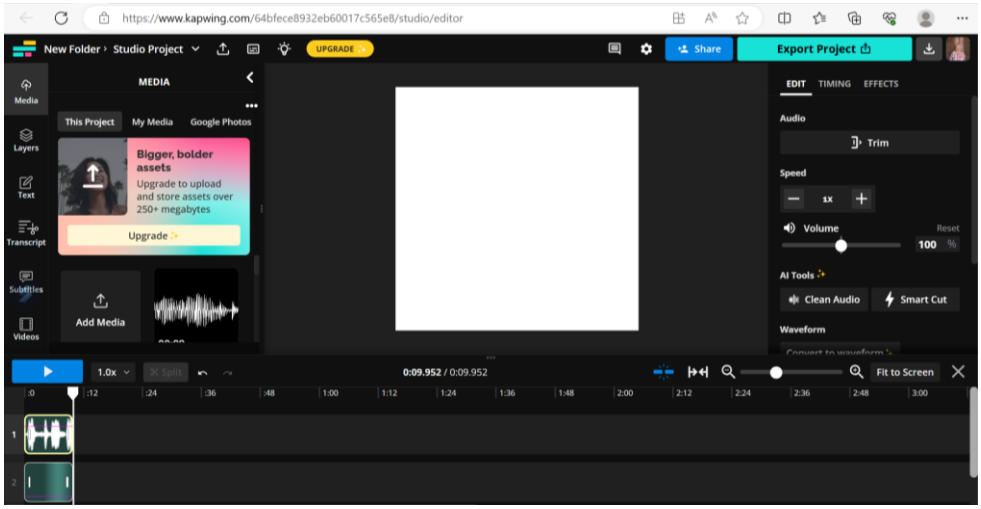
Dalam pengambilan *audio* sendiri menggunakan *audio* pembacaan teks Proklamasi yang dideklarasikan oleh pak ir. Soekarno pada tahun 1945. *Audio* tersebut dapat ditemukan dalam web. Untuk melakukan pemfilteran *noise* pada *audio* dengan simulai software Matlab hanya bagian awal durasi sekitar 9 detik. Simulasi ini sengaja ditambahkan *noise* agar pemfilteran lebih terlihat, *audio noise* sedndiri juga dapat ditemukan pada web. Dalam melakukan penambahan atau penggabungan audio asli dan *audio noise* terebut dilakukan pada web www.kapwing.com.



Gambar 4. 1 Audio Sebelum di Potong.

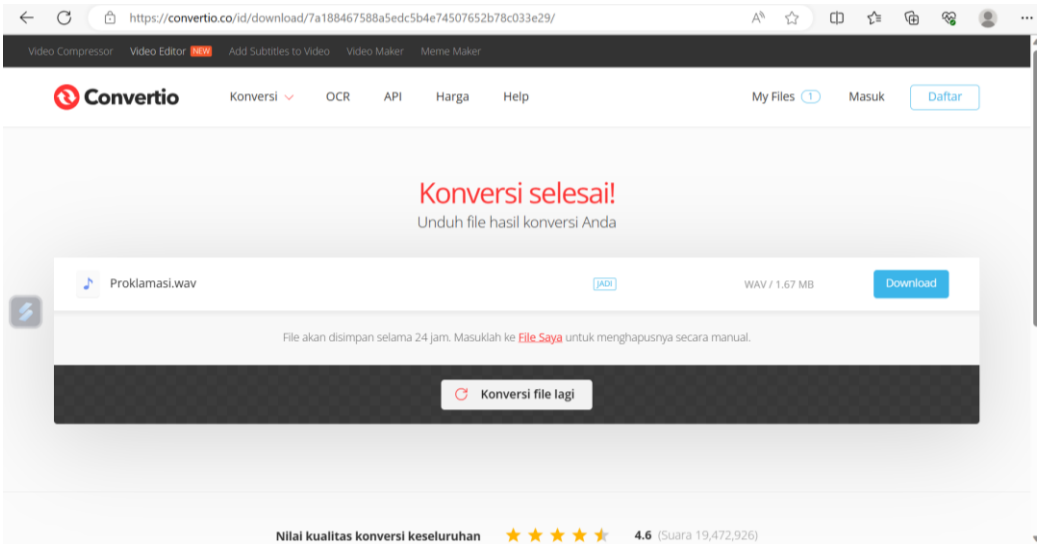
Pada Gambar 4.1 terdapat dua grafik *audio* bagian atas dan bawah yang memiliki durasi *audio* yang berbeda. Bagian atas merupakan audio asli dari *audio* deklarasi Proklamasi yang sudah di unduh sebelumnya yang memiliki durasi *audio*

3:00.193 sedangkan grafik *audio* bagian bawah merupakan grafik *audio noise* yang memiliki durasi 0:09.952.



Gambar 4. 2 Audio Sesudah di Potong.

Dapat dilihat dari Gambar 4.1 dan Gambar 4.2 ada perbandingan pada bagian pengeditan, awal gambar *audio* asli bagian atas masih memiliki durasi 3:00.193 sedangkan bagian bawah merupakan *audio noise* yang memiliki durasi 0:09.952. Untuk melakukan simulasi ini hanya menggunakan potongan *audio* beberapa detik menyesuaikan durasi *noise* yaitu 0:09.952 yang berisikan kalimat "Proklamasi kami bangsa Indonesia dengan ini" untuk melakukan pemfilteran *audio* dengan *noise*. Setelah pengeditan tersebut selesai *audio* dapat di *export project* pada bagian kanan atas dan hasil *project* tersebut dapat diunduh.



Gambar 4. 3 Convert Audio MP3 ke WAV.

Karena hasil dari unduhan sebelumnya file berbentuk MP3 dan file yang dibutuhkan adalah file dalam bentuk wav, sehingga dilakukan *convert* file dari MP3 ke wav. Kegiatan *convert* dapat dilihat pada Gambar 4.3 yang dilakukan pada web convertio.co.id.

4.2 Penentuan Frekuensi *Cut-off*

Dalam melakukan penentuan frekuensi terdapat beberapa pertimbangan yang sudah dilakukan sebelumnya yaitu dengan mengumpulkan beberapa sumber yang sudah dijelaskan pada pendahuluan yaitu frekuensi *cut-off* yang digunakan untuk memfilter audio *Low Pass Filter* adalah 4 KHz keatas. Karena *Low Pass Filter* merupakan jenis filter yang melewati frekuensi rendah dan sinyal dengan frekuensi tinggi diblokir. Sehingga untuk melakukan perbandingan dari frekuensi *cut-off* simulasi ini menggunakan nilai frekuensi cut-off 4 KHz, 10 KHz, dan 20 KHz. Dengan adanya perbedaan frekuensi *cut-off* tersebut memudahkan dalam menganalisa hasil suara, *respon frekuensi*, dan rata-rata SDG berupa grafik dan nilai rata-rata.

4.3 Simulasi MATLAB

Pada kegiatan simulasi melakukan pemfilteran *audio low pass filter* dengan menggunakan metode *Elliptic*. Gambaran simulasi dapat dilihat pada Gambar 4.4, Gambar 4.5, dan Gambar 4.6.

```
% Define filter specifications
Fs = 44100;          % Sampling frequency
Fp = 1000;           % Passband frequency
Fc = 4000;           % Cutoff frequency
Rp = 1;              % Passband ripple (dB)
Rs = 60;             % Stopband attenuation (dB)

% Calculate filter order and coefficients
Wp = Fp / (Fs / 2);
Wc = Fc / (Fs / 2);
[n, Wn] = ellipord(Wp, Wc, Rp, Rs);
[b, a] = ellip(n, Rp, Rs, Wn, 'low');

% Load audio signal
[y, Fs] = audioread('proklamasi.wav');

% Apply filter to audio signal
y_filtered = filter(b, a, y);

% Save the filtered audio
audiowrite('fc_4000.wav', y_filtered, Fs);
```

Gambar 4. 4 *Code Filter Elliptic Low Pass Filter Audio*.

```
% Plot filter frequency response
figure;
freqz(b, a, [4096], Fs);
title('Elliptic Low-pass Filter Frequency Response');

% Play the original audio
disp('Playing original audio...');
sound(y, Fs);

% Wait for the original audio to finish playing
pause(length(y) / Fs);

% Plot the waveform of the original and filtered audio
time = (0:length(y)-1) / Fs;
figure;
plot(time, y, 'b', 'LineWidth', 1);  % Original audio in blue
hold on;
plot(time, y_filtered, 'r', 'LineWidth', 1);  % Filtered audio in red
hold off;
xlabel('Time (seconds)');
ylabel('Amplitude');
title('Original and Filtered Audio Waveform');
legend('Original Audio', 'Filtered Audio');
grid on;
```

Gambar 4. 5 *Code Filter Elliptic Low Pass Filter Audio.*

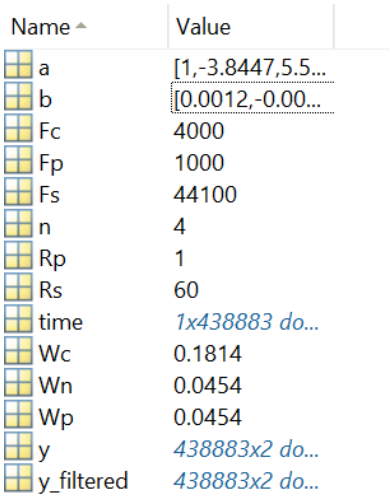
```
% Play the filtered audio
disp('Playing filtered audio...');
sound(y_filtered, Fs);

% Wait for the filtered audio to finish playing
pause(length(y_filtered) / Fs);
```

Gambar 4. 6 *Code Filter Elliptic Low Pass Filter.*

Pada Gambar 4.4, Gambar 4.5, dan Gambar 4.6 merupakan simulasi Matlab untuk melakukan pemfilteran *audio Low Pass Filter* menggunakan metode *Elliptic*. Filter ini menghasilkan sinyal *audio* yang lebih halus dengan membuang frekuensi tinggi yang tidak diinginkan. Fungsi *ellipord* dan *ellip* digunakan untuk menghitung orde dan koefisien filter berdasarkan spesifikasi filter yang diinginkan. Setelah itu, kita menerapkan filter pada sinyal *audio* asli menggunakan fungsi filter. Hasilnya adalah sinyal *audio* yang lebih bersih dan bebas dari frekuensi tinggi yang tidak diinginkan. Sinyal *audio* yang difilter kemudian disimpan dan diplot. Pada kegiatan pemfilteran tersebut juga dapat memutar kembali sinyal *audio* asli yang terdapat noise dan yang sudah difilter menggunakan fungsi *sound*.

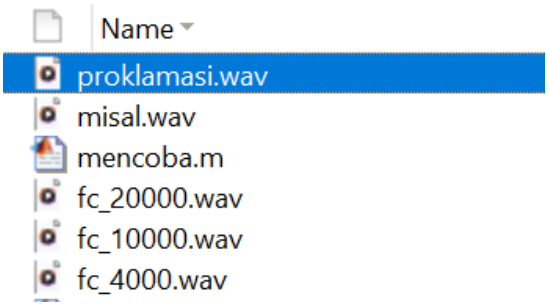
Pada bagian pertama baris 1 sampai 6 merupakan bagian dari kumpulan nilai parameter metode filter *elliptic low pass filter*, dapat dijelaskan bahwa nilai Fs (Frekuensi Sampling) pada umumnya yang digunakan yaitu 44.1 KHz, Fp (Frekuensi *Passband*) nilai parameter ini menyesuaikan hasil pemfilteran jadi semakin tinggi frekuensi *passband* semakin banyak semakin banyak frekuensi yang akan dilewatkan oleh filter, Fc (Frekuensi *Cut-off*) pada bagian ini nilai dapat diubah kapanpun menyeseuaikan hasil pemfilteran untuk rentang nilai frekuensi *cut-off* pada *low pass filter* yaitu 4 KHz sedangan frekuensi *cut-off* yang akan digunakan dalam simulasi ini untuk melakukan perbandingan yaitu 4 KHz, 10 KHz, dan 20 KHz. *Passband Ripple* (Rp) pada *low pass filter* biasanya menggunakan nilai rendah untuk menghindari distorsi yang signifikan pada sinyal *audio* yang difilter.

Bagian kedua dari simulasi tersebut yaitu menghitung filter order dan koefisien filter menggunakan fungsi *ellipord* dan *ellip*. Fungsi-fungsi ini biasanya digunakan untuk merancang filter digital. Nilai filter order ditentukan oleh nilai n, sedangkan koefisien filter adalah nilai b dan a. Fungsi *ellip* digunakan untuk merancang filter *elips*, yang merupakan jenis filter yang memiliki *rolloff* yang lebih curam dan *band* transisi yang lebih tajam dibandingkan dengan jenis filter lainnya. Pada *code* tersebut juga terdapat rumus Wp dan Wc untuk mendapatkan nilai matriks dari [n, Wn] dan [b, a] pada bagian *'low'* dapat diartikan bahwa pemfilteran audio tersebut menggunakan *Low Pass Filter*. Pada Gambar 4.7 dapat dilihat hasil dari perhitungan dari pemfilteran audio seperti nilai orde, Wn, dan Wc.



Gambar 4. 7 Details Hasil Perhitungan.

Pada bagian ketiga yaitu pembacaan audio yang terdapat *noise* yang akan difilter yaitu menggunakan *audio* proklamasi yang sudah ditambah *noise* sebelumnya. Sebelum audio difilter *audio* harus di impor terlebih dahulu pada bagian *current folder* audio tersebut Bernama ‘proklamasi.wav’ sebelum memfilter audio tersebut pastikan file audio tersebut berupa .wav. seperti contoh dapat dilihat pada Gambar 4.8.



Gambar 4. 8 *Current Folder.*

Pada bagian selanjutnya baris ke-17 merupakan penerapan filter ke sinyal audio menggunakan fungsi filter. Fungsi filter digunakan untuk menerapkan filter digital ke sinyal. Dua argumen pertama dari fungsi filter adalah koefisien filter b dan a yang telah dihitung sebelumnya, sedangkan argumen ketiga adalah sinyal masukan y. Keluaran dari fungsi filter adalah sinyal yang telah difilter y_filtered.

Pada baris 20 simulasi kode tersebut merupakan kegiatan penyimpan *audio* yang telah difilter menggunakan fungsi audiowrite. Fungsi *audiowrite* digunakan untuk menulis file *audio*. Argumen pertama dari fungsi *audiowrite* adalah nama file yang akan disimpan pada *audio* tersebut, sedangkan argumen kedua adalah sinyal *audio* yang telah difilter y_filtered sebelumnya, dan argumen ketiga adalah frekuensi sampling Fs. Dalam kasus ini, sinyal audio yang telah difilter disimpan sebagai file WAV dengan nama ’fc_4000.wav’.

Pada kode baris ke-23 Gambar 4.5 merupakan simulasi kode untuk memplot hasil respons frekuensi dari filter *elliptic* yang telah didesain sebelumnya. Fungsi *freqz* digunakan untuk menghitung dan memplot respons frekuensi dari filter digital. Dalam kasus ini, argumen pertama dan kedua dari fungsi *freqz* adalah koefisien filter b dan a yang telah dihitung sebelumnya. Argumen ketiga dari fungsi freqz adalah sebuah matriks [4096] yang menunjukkan jumlah titik-titik frekuensi pada nilai matriks tersebut nilai tersebut dapat dilihat nilai lebih besar, maka plot

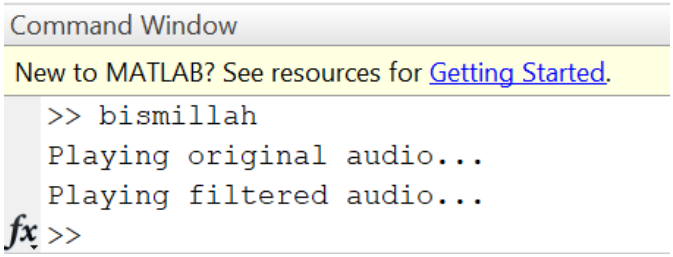
respon frekuensi dari filter akan memiliki resolusi frekuensi yanag lebih baik. Nilai matriks tersebut akan digunakan untuk menghitung respons frekuensi dari filter. Argumen keempat adalah frekuensi sampling Fs. Output dari fungsi *freqz* adalah sebuah plot dari respons frekuensi dari filter. Dalam kasus ini, plot tersebut diberi judul *'Elliptic Low-pass Filter Frequency Response'*.

Pada Gambar 4.5 baris ke-28 *Code* Matlab di atas akan memainkan audio asli yang telah dibaca menggunakan perintah "*audioread*" sebelumnya. Perintah "*sound*" digunakan untuk memainkan audio dengan menghasilkan output suara. Pada kode tersebut, *audio* yang akan dimainkan adalah "y" dengan frekuensi sampel "Fs". Perintah "disp" digunakan untuk menampilkan pesan ke layer. Pada kode tersebut, pesan yang ditampilkan adalah "*Playing original audio…*".

Kode pada baris ke-32 tersebut digunakan untuk membuat jeda pada program agar menunggu *audio* asli selesai dimainkan sebelum melanjutkan eksekusi kode selanjutnya. Jeda waktu dihitung berdasarkan panjang *audio* asli yang disimpan di variabel "y" dan frekuensi sampel "Fs". Perintah "*pause*" digunakan untuk memberikan jeda waktu pada program.

Pada bagian selanjutnya baris ke-35 Gambar 4.5 kode tersebut digunakan untuk membuat grafik *waveform* dari *audio* asli dan *audio* yang telah difilter. Grafik tersebut menunjukkan amplitudo sinyal audio dalam satuan dB pada sumbu y dan waktu dalam satuan detik pada sumbu x. Pada kode tersebut, variabel "*time*" menunjukkan waktu dalam satuan detik. Perintah "plot" digunakan untuk membuat grafik. "*hold on*" dan "*hold off*" digunakan untuk menambahkan beberapa plot ke dalam satu grafik. "xlabel", "ylabel", dan "title" digunakan untuk menambahkan label pada sumbu x, y, dan judul grafik, masing-masing. "legend" digunakan untuk menambahkan legenda pada grafik. "*grid on*" digunakan untuk menampilkan *grid* pada grafik.

Kode pada bagian %*Play* filtered audio tersebut digunakan untuk memainkan audio yang telah difilter. Perintah "*disp*" digunakan untuk menampilkan pesan di command window. "*sound*" digunakan untuk memainkan audio di MATLAB. Audio yang akan diputar disimpan di variabel "y_filtered", dan frekuensi sampel disimpan di variabel "Fs". Dalam menjalankan audio original dan audio yang sudah difilterdapat dilihat pada Gambar 4.9.



Gambar 4. 9 Memainkan *Audio Original* dan *Audio* yang Difilter.

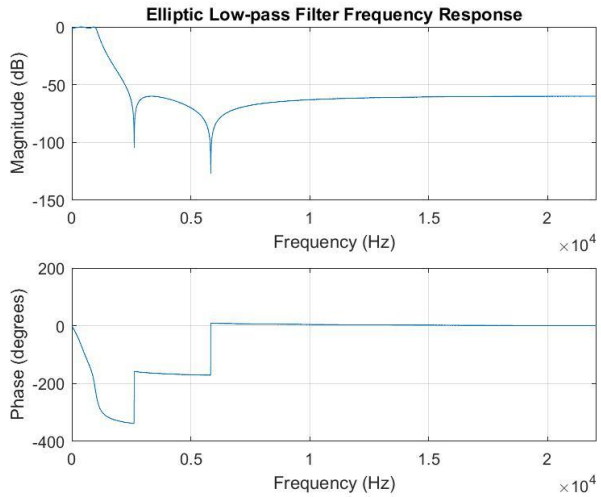
Kode bagian terakhir tersebut digunakan untuk menunggu *audio* yang telah difilter selesai diputar. Perintah "*pause*" akan memberhentikan eksekusi program selama durasi *audio* yang diputar. Durasi *audio* dihitung dengan membagi panjang audio dengan frekuensi sampel.

4.4 Analisis Hasil Respon Frekuensi *Elliptic Low Pass Filter*

Respon frekuensi *elliptic lowpass filter* dapat digambarkan dalam bentuk grafik, yang menunjukkan bagaimana filter merespons sinyal pada berbagai frekuensi. Perbandingan respon frekuensi dengan nilai Fc (Frekuensi *Cut-off*) 4 KHz, 10 KHz, dan 20 KHz sebagai berikut :

4.4.1 Grafik Respon Frekuensi *Elliptic* LPF (Fc 4 KHz)

Pada Gambar 4.10 merupakan hasil grafik respon frekuensi dari pemfilteran *audio Elliptic* menggunakan Low Pass Filter dengan menggunakan frekuensi *cut-off* 4 KHz.



Gambar 4. 10 Grafik Respon Frekuensi (Fc 4 KHz).

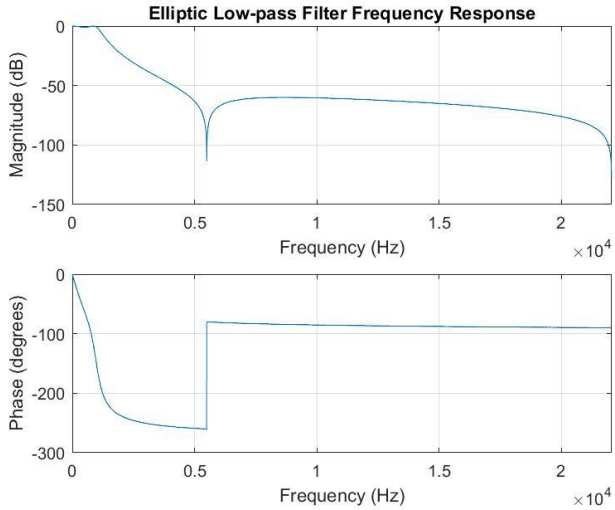
Filter ini digunakan untuk menghilangkan frekuensi tinggi pada sinyal audio. Pada sumbu x grafik menunjukkan frekuensi dalam *Hertz* (Hz), sedangkan pada sumbu y grafik menunjukkan *magnitude* dari sinyal *audio* pada frekuensi tersebut. Dengan menggunakan filter ini, frekuensi tinggi diatas frekuensi 4 KHz pada sinyal *audio* akan dihilangkan sehingga hanya frekuensi rendah yang akan diteruskan. Dengan begitu, kualitas *audio* akan menjadi lebih baik dan lebih jernih. Dapat dilihat dari nilai *magnitude* menunjuk pada 0 desibel (dB) yang berada pada sumbu y dan pada sumbu x menuju ke nilai 2.5 KHz kemudian membentuk *ripple* ke nilai 6 KHz. Sesuai standar spesifikasi filter frekuensi *elliptic low pass* filter identik dengan hasil respon frekuensi yang curam kebawah menghasilkan *audio* yang baik sesuai dengan nilai frekuensi cut-offnya. Nilai frekuensi *cut-off* semakin kecil hasil respon frekuensi semakin curam kebawah, karena dalam simulasi ini nilai frekuensi *cut-off* terkecil adalah 4 KHz jadi hasil *audio* hampir sempurna tidak ada *noise* sesuai dengan hasil *audio original* sebelumnya dan grafik respon frekuensinya curam kebawah. Simulasi pertama ini merupakan hasil grafik yang terbaik karena menggunakan frekuensi *cut-off* terendah yaitu 4 KHz.

Dengan melihat grafik phase pada Gambar 4.10 menggambarkan bagaimana phase sinyal berubah pada berbagai frekuensi ketika melewati filter. Grafik *phase* berguna untuk analisis dan desain filter, terutama jika Anda memerlukan informasi tentang pergeseran waktu sinyal di frekuensi tertentu. Namun, untuk mendapatkan gambaran lengkap tentang respons filter, seringkali lebih baik untuk mempertimbangkan baik grafik *magnitude* maupun grafik fase secara bersamaan. Dapat dilihat dari hasil grafik magnitude dan grafik phase memiliki pergeseran dengan nilai yang sama.

Parameter-parameter yang digunakan dalam simulasi yaitu frekuensi *cut-off* 4 KHz, Fs 44.1 KHz, Rp 1 dB, Rs 60 dB. Sedangkan nilai orde yang digunakan yaitu 4 untuk mendapatkan nilai tersebut persamaan yang digunakan pada Tabel 4.2 n ke-4. Akan tetapi dalam memudahkan penentuan orde dari filter *elliptic* proses tersebut maka menggunakan metode grafik pada Gambar 2.8. Untuk mendapatkan nilai spesifikasi filter digital dalam bentuk rad (radian) menggunakan persamaan Tabel 2.2 pada *low pass filter*, transfer function filter dalam domain *laplace*, sehingga formula yang digunakan yaitu persamaan 2.2.

4.4.2 Grafik Respon Frekuensi Elliptic LPF (Fc 10 KHz)

Pada Gambar 4.11 merupakan hasil grafik respon frekuensi dari pemfilteran audio Elliptic menggunakan Low Pass Filter dengan menggunakan frekuensi cut-off 10 KHz.



Gambar 4. 11 Grafik Respon Frekuensi (Fc 10 KHz).

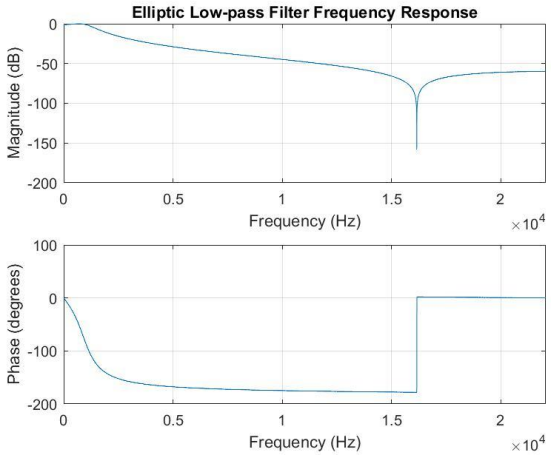
Pada Gambar 4.11 merupakan hasil grafik respon frekuensi dari pemfilteran audio *elliptic* menggunakan *Low Pass Filter* dengan menggunakan frekuensi cut-off 10 KHz. Filter ini digunakan untuk menghilangkan frekuensi tinggi pada sinyal audio. Pada sumbu x grafik menunjukkan frekuensi dalam *Hertz* (Hz), sedangkan pada sumbu y grafik menunjukkan magnitude dari sinyal *audio* pada frekuensi tersebut. Dengan menggunakan filter ini, frekuensi tinggi diatas 10 KHz pada sinyal *audio* akan dihilangkan sehingga hanya frekuensi rendah yang akan diteruskan. Dengan begitu, kualitas audio akan menjadi lebih baik dan lebih jernih. Dapat dilihat dari nilai *magnitude* menunjuk pada 0 desibel (dB) yang berada pada sumbu y dan pada sumbu x menuju ke nilai 6 KHz. Sesuai standar spesifikasi filter frekuensi *elliptic low pass* filter identik dengan hasil respon frekuensi yang curam kebawah menghasilkan *audio* lebih baik sesuai dengan nilai frekuensi cut-offnya. Nilai frekuensi *cut-off* semakin kecil hasil respon frekuensi semakin curam, karena dalam simulasi ini nilai frekuensi *cut-off* yang digunakan adalah 10 KHz jadi hasil *audio* kurang maksimal masih yerdapat sedikit *noise* dan grafik respon frekuensi sedikit landai kurang curam kebawah.

Dengan melihat grafik *phase* pada Gambar 4.11 menggambarkan bagaimana phase sinyal berubah pada berbagai frekuensi ketika melewati filter. Grafik *phase* berguna untuk analisis dan desain filter, terutama jika Anda memerlukan informasi tentang pergeseran waktu sinyal di frekuensi tertentu. Namun, untuk mendapatkan gambaran lengkap tentang respons filter, seringkali lebih baik untuk mempertimbangkan baik grafik *magnitude* maupun grafik fase secara bersamaan. Dapat dilihat dari hasil grafik *magnitude* dan grafik phase memiliki pergeseran dengan nilai yang sama.

Parameter-parameter yang digunakan dalam simulasi yaitu frekuensi *cut-off* 10 KHz, Fs 44.1 KHz, Rp 1 dB, Rs 60 dB. Sedangkan nilai orde yang digunakan yaitu 4 untuk mendapatkan nilai tersebut persamaan yang digunakan pada Tabel 4.2 n ke 4. Akan tetapi dalam memudahkan penentuan orde dari filter *elliptic* proses tersebut maka menggunakan metode grafik pada gambar 2.8. Untuk mendapatkan nilai spesifikasi filter digital dalam bentuk rad (radian) menggunakan persamaan Tabel 2.2 pada *low pass filter*, transfer function filter dalam domain *laplace*, sehingga formula yang digunakan yaitu persamaan 2.2.

4.4.3 Grafik Respon Frekuensi *Elliptic* LPF (Fc 20 KHz)

Sedangkan ada Gambar 4.12 merupakan hasil grafik respon frekuensi dari pemfilteran *audio Elliptic* menggunakan Low Pass Filter dengan menggunakan frekuensi *cut-off* 20 KHz.



Gambar 4. 12 Grafik Respon Frekuensi (Fc 20 KHz)

Filter ini digunakan untuk menghilangkan frekuensi tinggi pada sinyal audio. Pada sumbu x grafik menunjukkan frekuensi dalam *Hertz* (Hz), sedangkan pada

sumbu y grafik menunjukkan *magnitude* dari sinyal audio pada frekuensi tersebut. Dengan menggunakan filter ini, frekuensi tinggi diatas 20 KHz pada sinyal *audio* akan dihilangkan sehingga hanya frekuensi rendah yang akan diteruskan. Dengan begitu, kualitas *audio* akan menjadi lebih baik dan lebih jernih. Dapat dilihat dari nilai *magnitude* menunjuk pada 0 desibel (dB) yang berada pada sumbu y dan pada sumbu x menuju ke nilai 16 KHz. Sesuai standar spesifikasi filter frekuensi *elliptic low pass filter* identik dengan hasil respon frekuensi yang curam kebawah menghasilkan *audio* lebih baik sesuai dengan nilai frekuensi cut-offnya. Nilai frekuensi *cut-off* semakin kecil hasil respon frekuensi semakin curam, karena dalam simulasi ini nilai frekuensi *cut-off* yang digunakan adalah 20 KHz jadi hasil *audio* masih terdapat banyak *noise* karena frekuensi tinggi yang diblokir lebih sedikit dan grafik respon frekuensi landai kurang curam kebawah.

Dengan melihat grafik *phase* pada Gambar 4.12 menggambarkan bagaimana *phase* sinyal berubah pada berbagai frekuensi ketika melewati filter. Grafik *phase* berguna untuk analisis dan desain filter, terutama jika Anda memerlukan informasi tentang pergeseran waktu sinyal di frekuensi tertentu. Namun, untuk mendapatkan gambaran lengkap tentang respons filter, seringkali lebih baik untuk mempertimbangkan baik grafik *magnitude* maupun grafik fase secara bersamaan. Dapat dilihat dari hasil grafik *magnitude* dan grafik *phase* memiliki pergeseran dengan nilai yang sama. Dapat disimpulkan bahwa semakin kecil nilai frekuensi cut-off hasil magnitude respon frekuensi semakin curam dan semaikn besar nilai frekuensi cut-off semakin landai hasil magnitude respon frekuensinya.

Parameter-parameter yang digunakan dalam simulasi yaitu frekuensi *cut-off* 20 KHz, Fs 44.1 KHz, Rp 1 dB, Rs 60 dB. Sedangkan nilai orde yang digunakan yaitu 4 untuk mendapatkan nilai tersebut persamaan yang digunakan pada Tabel 4.2 n ke 4. Akan tetapi dalam memudahkan penentuan orde dari filter *elliptic* proses tersebut maka menggunakan metode grafik pada gambar 2.8. Untuk mendapatkan nilai spesifikasi filter digital dalam bentuk rad (radian) menggunakan persamaan Tabel 2.2 pada *low pass filter*, transfer function filter dalam domain *laplace*, sehingga formula yang digunakan yaitu persamaan 2.2. untuk langkah-langkah dari perhitungan *laplace* tersebut dimulai dari penentuan spesifikasi sampai mendapatkan nilai koefisien A dan B.

4.4.4 Perhitungan Desain Filter

a. Spesifikasi

Tipe Filter: LPF (Low Pass Filter) Elliptic.

Passband Ripple (Rp): 1 dB.

Frekuensi Cut-off: 4 kHz.

Stopband Attenuation (Rs): 60 dB.

Frekuensi Sampling (fs): 44.1 kHz.

Passband Frekuensi: 1 kHz.

b. Sepesifikasi Filter Dalam Radian

Frekuensi *cut-off*, $\omega_c = 2\pi$ (f_c) persamaan dari Tabel 2.2

$\omega_c = 2\pi$ (4 KHz)

= 1 (25132.7412) rad/s

= 25132.7412 rad/s

Frekuensi stopband, ω_{stop} tidak perlu dihitung karena kita tidak mengetahui f_{stop} dan orde filter n sudah diketahui.

c. Orde Filter

Orde filter telah diketahui N= 4. Jadai prototype yang digunakan yaitu orde 4 persamaan pada Tabel 2.3.

$$H(s) = \frac{s^4+0.6336s^2+0.6837}{s^4+1.1351s^3+1.8831s^2+1.1387s+0.7242}$$

d. Tranformasi *prototype* tersebut ke filter LPF dengan $\omega_c = 25132$ rad/s dengan mengganti semua s-nya.

$$\frac{1s}{\omega_c} = \frac{1s}{25132} = 3979 \cdot 10^{-5}s$$

$$H(s) = \frac{s^4+0.6336s^2+0.6837}{s^4+1.1351s^3+1.8831s^2+1.1387s+0.7242}$$

$$= \frac{(3979\cdot10^{-1}s^4+0.6336\cdot15836\cdot10^{-9}\,s^2+0.6837)}{3979\cdot10^{-1}s^4+4520510^{-5}s^3+74881\cdot3979\cdot10^{-11}s^2+45176\cdot10^{-7}s+0.7242}$$

Kemudian hasil tersebut dijadikan kedalam bentuk pecahan *parsial* sehingga akan mendapatkan nilai koefisien dari A dan B yaitu A = [1, -3.844695954290729,

5.565237890123991, -3.594091566669077, 0.873665822646615] sedangkan untuk koefisien dari B = [0.001150964272689, -0.003693533792268, 0.005188695099537, -0.003693533792268, 0.001150964272689].

4.5 Analisis Hasil *Audio Waveform*

Audio waveform adalah grafik yang menunjukkan bagaimana amplitudo sinyal audio berubah seiring waktu. Grafik ini sangat membantu dalam memvisualisasikan sinyal *audio* dan memudahkan pengeditan serta manipulasi sinyal tersebut. Pada sumbu x, waktu ditampilkan dalam satuan detik atau milidetik, sedangkan pada sumbu y, amplitudo sinyal *audio* ditampilkan dalam satuan desibel (dB).

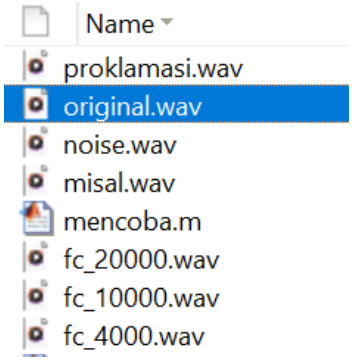
4.5.1 Grafik *Audio* Belum Terdapat *Noise*

Dalam menampilkan grafik *waveform* pada audio dapat menggunakan simulasi pada matlab dengan menggunakan *code* pada Gambar 4.13.

```
% Langkah 1: Baca audio
[audio, fs] = audioread('original.wav');

% Langkah 2: Hitung panjang audio dan buat time vector
n = length(audio);
time = (0:n-1) / fs;

% Langkah 3: Plot grafik audio
figure;
plot(time, audio);
title('Grafik Audio (Mono)');
xlabel('Waktu (detik)');
ylabel('Amplitudo');
grid on;

```

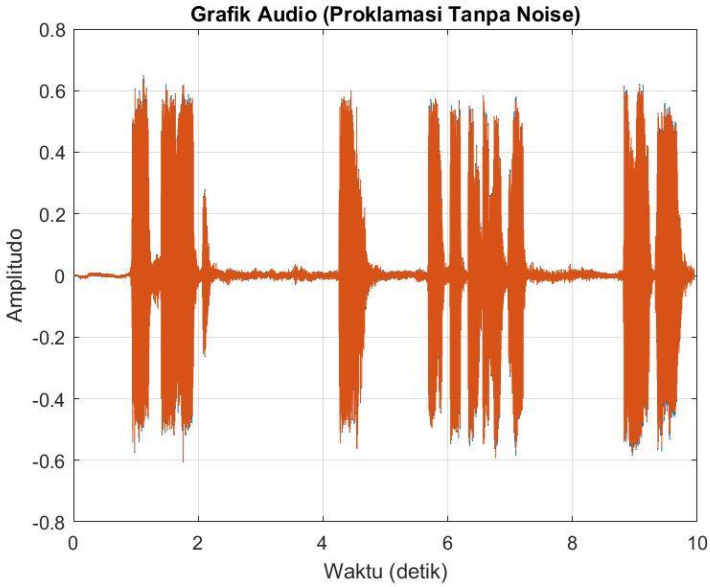
Gambar 4. 13 Code Matlab *Original Audio*

Pada *code* Matlab Gambar 4.13 terdapat tiga langkah dalam melakukan pembacaan grafik *waveform audio original* yang belum terdapat *noise*. Langkah pertama adalah membaca file *audio* dengan format WAV menggunakan perintah "*audioread*" dengan penamaan *audio* ‘original.wav’.



Gambar 4. 14 *Current Folder Audio Original.*

Langkah kedua adalah menghitung panjang audio dan membuat time *vector* menggunakan variabel "n" dan "fs". Langkah ketiga adalah menampilkan grafik *audio* dengan menggunakan perintah "*plot*". Pada grafik tersebut, sumbu x menunjukkan waktu dalam detik, sedangkan sumbu y menunjukkan amplitudo pada *audio* tersebut.



Gambar 4. 15 Grafik *Audio Original.*

Pada Gambar 4.15 merupakan grafik *audio original* yang belum di beri *noise* sama sekali, pada grafik sumbu y tersebut menunjukkan grafik amplitudo dengan nilai 0.6 satuan amplitudo pada grafik *audio* biasanya diukur dalam desibel (dB). Sedangkan pada grafik sumbu y menunjukkan nilai waktu audio dan nilai tersebut berada pada 0:09.952 detik.

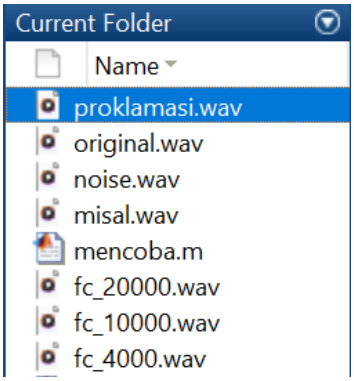
4.5.2 Grafik *Audio* Setelah Terdapat *Noise*

Dalam menampilkan grafik *waveform* pada audio dapat menggunakan simulasi pada matlab dengan menggunakan *code* pada Gambar 4.13.

```
% Langkah 1: Baca audio
[audio, fs] = audioread('proklamasi.wav');

% Langkah 2: Hitung panjang audio dan buat time vector
n = length(audio);
time = (0:n-1) / fs;

% Langkah 3: Plot grafik audio
figure;
plot(time, audio);
title('Grafik Audio (Proklamasi dengan Noise)');
xlabel('Waktu (detik)');
ylabel('Amplitudo');
grid on;
```

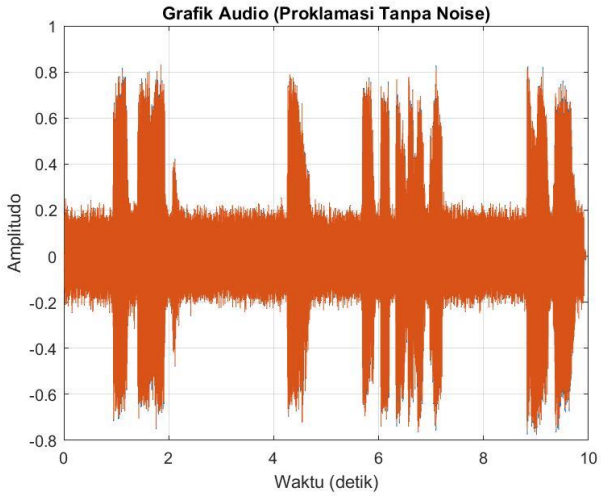
Gambar 4. 16 *Code* Matlab *Audio* yang Terdapat *Noise*

Pada *code* Matlab Gambar 4.16 memiliki Langkah yang sama sseperti pembacaan grafik *audio waveform audio original* sebelumnya yaitu memiliki tiga langkah dalam melakukan pembacaan grafik *waveform audio original* yang sudah ditambahkan *noise*. Langkah pertama adalah membaca file *audio* dengan format WAV menggunakan perintah "*audioread*" dengan penamaan *audio* 'proklamasi.wav'. Dapat dilihat pada Gambar 4.17 *Current Folder Audio Original*.



Gambar 4. 17 *Current Folder Audio Original* yang di Tambah *Noise*

Langkah kedua adalah menghitung panjang *audio* dan membuat *time vector* menggunakan variabel "n" dan "fs". Langkah ketiga adalah menampilkan grafik *audio* dengan menggunakan perintah "*plot*". Pada grafik tersebut, sumbu x menunjukkan waktu dalam detik, sedangkan sumbu y menunjukkan amplitudo pada *audio* tersebut.

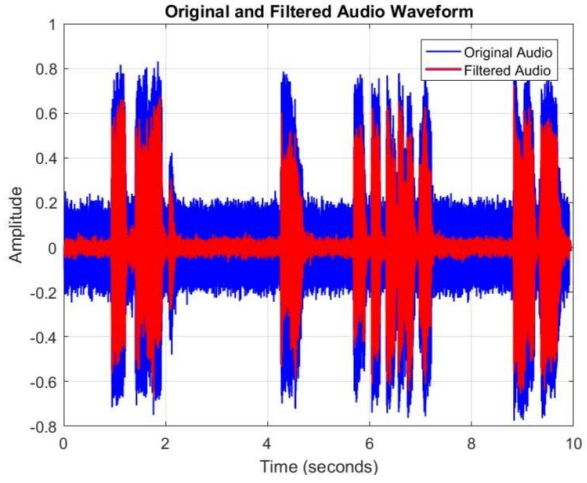


Gambar 4. 18 Grafik *Audio* yang Terdapat *Noise*

Pada Gambar 4.18 merupakan grafik *audio original* yang sudah ditambahkan oleh *noise*, pada grafik sumbu x tersebut menunjukkan grafik amplitudo dengan nilai 0.8 sehingga dengan penambahan *noise* dapat merupah nilai amplitudo dengan satuan desibel (dB). Sedangkan pada grafik sumbu y menunjukkan nilai waktu *audio* dan nilai tersebut berada pada 0:09.952 detik, dapat dilihat untuk waktu tidak mengalami prubahan.

4.5.3 Grafik Filter *Audio Waveform* dengan Fc 4 KHz

Grafik *waveform* pada Gambar 4.19 tersebut digunakan untuk melakukan perbandingan *audio original* yang ditambahkan *noise* dengan *audio* yang diberi frekuensi *cut-off* 4 KHz.

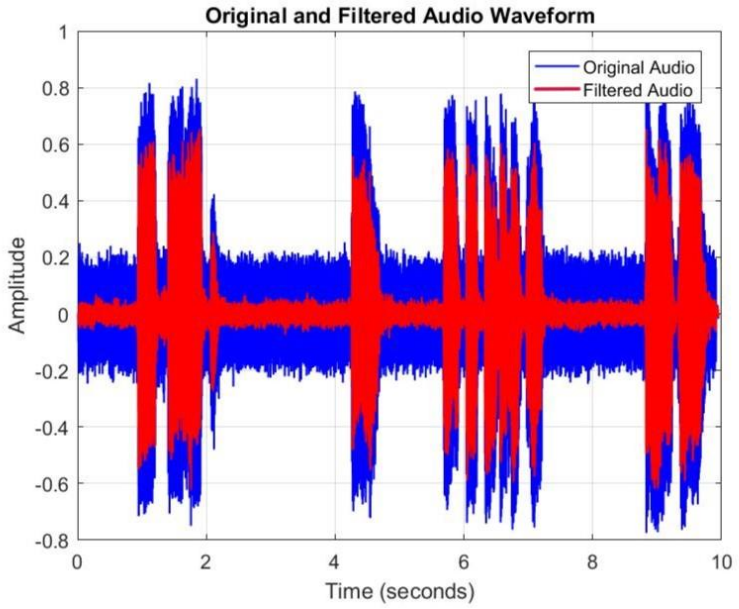


Gambar 4. 19 Grafik *Audio Waveform* Frekuensi *Cut-off* 4 KHz

Perbandingan grafik pada Gambar 4.19 mengalami perubahan sehingga pada simulasi tersebut terjadi kegiatan pemfilteran dari *audio* yang terdapat *noise* sebelumnya. Karena pada *audio* yang terdapat *noise* sebelumnya amplitudo pada sumbu y berada pada nilai 0.8 desibel (dB) yaitu ditunjukkan pada grafik warna biru, sedangkan grafik berwarna merah berada pada nilai rata-rata 0.59 desibel (dB) yang ditunjukkan pada sumbu y, pemfilteran tersebut menggunakan nilai frekuensi *cut-off* 4 KHz. Dapat dikatakan bahwa dengan menggunakan frekuensi *cut-off* ini hasil audio lebih jernih daripada hasil *audio* yang menggunakan frekuensi *cut-off* lebih dari 4 KHz. Untuk grafik sumbu x yaitu menunjukkan nilai dari waktu *audio*, dapat dilihat tidak ada perubahan dari nilai *audio* sebelum dan sesudah difilter. Sehingga dengan adanya perbedaan tersebut membuktikan bahwa audio terfilter dengan baik.

4.5.4 Grafik Filter *Audio Waveform* dengan Fc 10 KHz

Grafik *waveform* pada Gambar 4.20 tersebut digunakan untuk melakukan perbandingan *audio original* yang ditambahkan *noise* dengan *audio* yang diberi frekuensi cut-off 4 KHz.



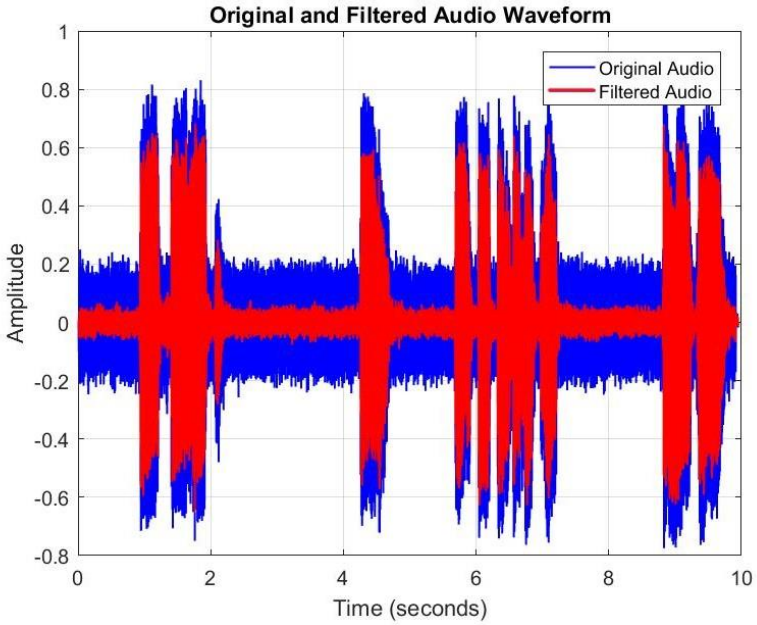
Gambar 4. 20 Grafik *Audio Waveform* Frekuensi *Cut-off* 10 KHz

Perbandingan grafik pada Gambar 4.20 mengalami perubahan sehingga pada simulasi tersebut terjadi kegiatan pemfilteran dari *audio* yang terdapat *noise*

sebelumnya. Karena pada *audio* yang terdapat *noise* sebelumnya amplitudo pada sumbu y berada pada nilai 0.8 desibel (dB) yaitu ditunjukkan pada grafik warna biru, sedangkan grafik berwarna merah berada pada nilai rata-rata 0.6 desibel (dB) yang ditunjukkan pada sumbu y, pemfilteran tersebut menggunakan nilai frekuensi *cut-off* 10 KHz. Dapat dikatakan bahwa dengan menggunakan frekuensi *cut-off* ini hasil *audio* sedang masih ada noise sedikit daripada hasil *audio* yang menggunakan frekuensi *cut-off* 4 KHz. Untuk grafik sumbu x yaitu menunjukkan nilai dari waktu audio, dapat dilihat tidak ada perubahan dari nilai *audio* sebelum dan sesudah difilter. Walaupun pada hasil grafik hanya beda tipis perubahannya dari pemfilteran yang menggunakan frekuensi *cut-off* (Fc) 4 KHz tapi hasil *audio* lebih terlihat.

4.5.5 Grafik Filter *Audio Waveform* dengan Fc 20 KHz

Grafik *waveform* pada Gambar 4.20 tersebut digunakan untuk melakukan perbandingan *audio original* yang ditambahkan noise dengan *audio* yang diberi frekuensi *cut-off* 4 KHz.



Gambar 4. 21 Grafik *Audio Waveform* Frekuensi *Cut-off* 20 KHz

Perbandingan grafik pada Gambar 4.21 mengalami perubahan sehingga pada simulasi tersebut terjadi kegiatan pemfilteran dari *audio* yang terdapat *noise* sebelumnya. Karena pada audio yang terdapat *noise* sebelumnya amplitudo pada

sumbu y berada pada nilai 0.8 desibel (dB) yaitu ditunjukkan pada grafik warna biru, sedangkan grafik berwarna merah berada pada nilai rata-rata 0.62 desibel (dB) yang ditunjukkan pada sumbu y, pemfilteran tersebut menggunakan nilai frekuensi *cut-off* 20 KHz. Dapat dikatakan bahwa dengan menggunakan frekuensi *cut-off* ini hasil *audio* terdapat lebih banyak *noise* daripada hasil *audio* yang menggunakan frekuensi *cut-off* 4 KHz dan 10 KHz. Untuk grafik sumbu x yaitu menunjukkan nilai dari waktu *audio*, dapat dilihat tidak ada perubahan dari nilai *audio* sebelum dan sesudah difilter. Walaupun pada hasil grafik hanya beda tipis perubahannya dari pemfilteran yang menggunakan frekuensi *cut-off* (Fc) 4 KHz dan 10 KHz tapi hasil *audio* lebih terlihat bahwa hasil *audio* yang menggunakan 20 KHz lebih banyak terdengar noisenya.

4.6 Analisis Hasil Data Skala Nilai SDG

Penilaian SDG pada *audio* ini untuk mengevaluasi kualitas *audio* dengan mempertimbangkan parameter yang berkaitan dengan kinerja sinyal, seperti distorsi, kebisingan, dan respons frekuensi. Tujuannya adalah untuk memastikan bahwa kualitas *audio* yang dihasilkan sesuai dengan standar tertentu. Dapat dilihat dari Tabel 4.1 Penilaian *Subjective Differences Grade* (SDG) dari 30 responden. Dari Tabel tersebut akan diambil nilai rata-rata. Rata-rata dari nilai SDG dapat menggunakan persamaan 2.1.

Tabel 4. 1 Penilaian *Subjective Differences Grade* (SDG) Responden

No	Nama	Nilai SDG Fc1 4 KHz	Nilai SDG Fc2 10 KHz	Nilai SDG Fc3 20 KHz
1	Sufiana	4	3	1
2	Syifareona	4	1	2
3	Soulenia R	4	2	1
4	Angger	5	3	3
5	Wildan M	5	5	3
6	Mahewara L	4	3	2
7	Ganif	5	5	3
8	Riyana	5	4	2
9	Rikko	5	4	4

No	Nama	Nilai SDG Fc1 4 KHz	Nilai SDG Fc2 10 KHz	Nilai SDG Fc3 20 KHz
10	Hanuf S	5	4	3
11	Ukhti	5	4	3
12	Nadhia	5	4	3
13	Desi	5	4	4
14	Milla	4	3	3
15	Rania	5	5	5
16	Usi Fajri	3	4	2
17	M Rendi	5	4	3
18	Aditya W	4	3	3
19	Indra S	4	4	3
20	Nur Ayu	5	4	3
21	Indra B	4	3	2
22	Yurisni	4	4	3
23	Vieri Muflih	4	3	2
24	Amelia Ayu	5	4	3
25	Salsabila	5	4	3
26	Sabda	5	4	2
27	Fina Fauniza	5	4	3
28	Nedta Febry	5	4	3
29	Sandry	5	4	3
30	Nauval Hilmi	5	4	3
Jumlah		138	112	83

1. Hasil Nilai Rata-Rata SDG dengan Frekuensi *Cut-Off* 4 KHz

$$\overline{X} = \frac{138}{30} = 4.6$$

Dari perhitungan nilai rata-rata SDG dengan frekuensi *cut-off* 4 KHz adalah 4.6. Rata-rata ini didapatkan dari persamaan 2.1, nilai tersebut menunjukkan hasil penilaian responden terhadap kualitas *audio* yang telah diuji bagian Tabel 4.1 pada

Fc1. Nilai rata-rata merupakan hasil pembagian dari seluruh jumlah nilai skala Fc1 yaitu 138 dengan jumlah seluruh responden yaitu 30 responden. Semakin tinggi nilai rata-rata, semakin baik kualitas *audio* yang diuji. Namun, perlu diingat bahwa nilai rata-rata hanya memberikan gambaran umum tentang kualitas *audio* dan tidak dapat menunjukkan detail tentang parameter-parameter yang dinilai. Dengan menggunakan frekuensi *cut-off* 4 KHz ini nilai rata-rata hampir mendekati sempurna karena sudah melebihi dari baik dengan melihat penilaian skala SDG Tabel 2.1 yang berartikan 4 baik.

2. Hasil Nilai Rata-Rata SDG dengan Frekuensi *Cut-Off* 10 KHz

$$\overline{X} = \frac{112}{30} = 3.7333$$

Selanjutnya pada perhitungan nilai rata-rata SDG dengan frekuensi *cut-off* 10 KHz adalah 3.7333, nilai tersebut didapatkan dengan menggunakan persamaan 2.1. Rata-rata ini menunjukkan hasil penilaian responden terhadap kualitas *audio* yang telah diuji bagian Tabel 4.1 pada Fc2. Nilai rata-rata merupakan hasil pembagian dari seluruh jumlah nilai skala Fc2 yaitu 112 dengan jumlah seluruh responden yaitu 30 responden. Dengan penilaian rata-rata 3.7333 berartikan bahwa hasil *audio* yang telah difilter menggunakan nilai 10 KHz hampir mendekati baik dengan melihat nilai skala SDG Tabel 2.1. Penilaian ini berkurang karena frekuensi *cut-off* yang digunakan cukup besar sehingga frekuensi tinggi yang diblokir tidak terlalu banyak dan mengakibatkan suara *noise* masih sedikit terdengar.

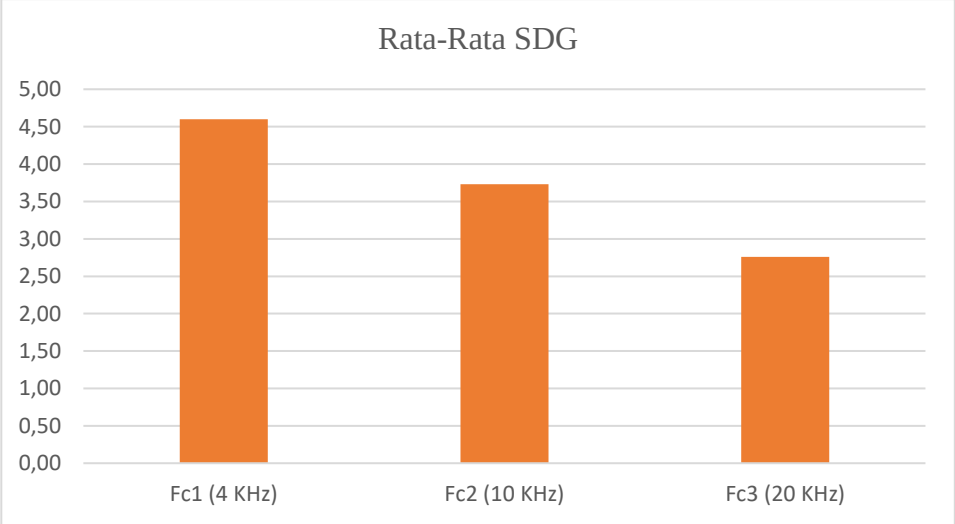
3. Hasil Nilai Rata-Rata SDG dengan Frekuensi *Cut-Off* 20 KHz

$$\overline{X} = \frac{83}{30} = 2.7666$$

Nilai rata-rata SDG yang terakhir menggunakan frekuensi *cut-off* 20 KHz dengan hasil nilai 2.7666, nilai tersebut didapatkan dengan menggunakan persamaan 2.1. Rata-rata ini menunjukkan hasil penilaian responden terhadap kualitas *audio* yang telah diuji bagian Tabel 4.1 pada Fc3. Nilai rata-rata merupakan hasil pembagian dari seluruh jumlah nilai skala Fc3 yaitu 83 dengan jumlah seluruh responden yaitu 30 responden. Dengan penilaian rata-rata 2.7666 berartikan bahwa

hasil *audio* yang telah difilter menggunakan nilai 20 KHz hampir mendekati cukup dengan melihat nilai skala SDG Tabel 2.1. Penilaian ini sangat rendah karena frekuensi *cut-off* yang digunakan sangat besar sehingga frekuensi tinggi yang diblokir sangat sedikit dan mengakibatkan suara *noise* sangat terdengar dan *noise* cukup mengganggu.

4. Hasil Diagram Batang Rata-Rata SDG



Gambar 4. 22 Rata- Rata SDG

Dari Gambar 4.2 dapat dilihat bahwa sumbu y menyatakan hasil dari rata-rata nilai SDG yang didapatkan dari jumlah nilai skala yang telah diambil sebelumnya dengan jumlah dari responden yaitu 30 orang. Untuk sumbu x menunjukkan Fc1 (4 KHz), Fc2 (10 KHz), dan Fc3 (20 KHz) yaitu nilai frekuensi *cut-off* yang digunakan dalam pengambilan data SDG dan simulasi pemfilteran. Dapat dilihat bahwa nilai frekuensi *cut-off* dengan nilai 4 KHz (Fc1 4 KHz), frekuensi *cut-off* dengan nilai 10 KHz (Fc2 10 KHz), dan frekuensi *cut-off* dengan nilai 20 KHz (Fc3 20). Dapat disimpulkan bahwa dari ketiga frekuensi *cut off*, nilai rata rata yang paling banyak pada pemfilteran suara lowpass yaitu saat menggunakan 4 KHz, sedangkan nilai rata-rata yang paling sedikit yaitu pada saat menggunakan frekuensi cut-off 20 KHz. Ini menunjukkan bahwa kualitas suara setelah difilter paling baik menggunakan frekuensi cut-off 4 KHz.