TÌM TẦN SỐ CƠ BẢN CỦA TÍN HIỆU TIẾNG NÓI SỬ DỤNG HÀM TỰ TƯƠNG QUAN, HÀM VI SAI BIÊN ĐỘ TRUNG BÌNH, BIẾN ĐỔI FAST FOURIER TRANSFORM (FFT)

Nguyễn Đình Mẫn, Trần Thế Nam, Võ Trần Anh Khoa

Nhóm 4, lớp HP: 18.89

Điểm	Bảng phân công nhiệm vụ		Chữ ký của SV
	Nguyễn Đình Mẫn (nhóm trưởng)	• Phân công nhiệm vụ, đảm bảo tiến độ của các thành viên trong nhóm • Đọc tài liệu, viết báo cáo các vấn đề về xử lý tín hiệu tín nói, tần số cơ bản của một tín hiệu tuần hoàn(tr.3) • Đọc tài liệu, cài đặt thuật toan(tr.9-10,tr.13,14), viết báo cáo trình bày về thuật toán sử dụng hàm tự tương quan.(tr.4-tr.5) • Viết báo cáo phần kết quả và thực nghiệm, kết luận(tr.25) • Tổng hợp, viết báo cáo hoàn chỉnh	
	Trần Thế Nam	• Đọc tài liệu, cài đặt thuật toán(tr10-11), viết báo cáo trình bày về thuật toán sử dụng thuật toán biến đổi Fast Fourier Transform (FFT)(tr.11-12). • Làm slide, thuyết trình phần cơ sở lý thuyết, sơ đồ khối của thuật toán biến đổi Fast Fourier Transform.(tr.6-9) • Tổng hợp tần số cơ bản tín hiệu thủ công	
	Võ Trần Anh Khoa	• Đọc tài liệu, cài đặt thuật toán, viết báo cáo trình bày về thuật toán hàm vi sai biên độ trung bình (AMDF).(tr5-6) • Thuyết trình phần cơ sở lý thuyết, sơ đồ khối của thuật toán hàm vi sai biên độ trung bình (AMDF)	

Lời cam đoan: Chúng tôi, gồm các sinh viên có chữ ký ở trên, cam đoan rằng báo cáo này là do chúng tôi tự viết dựa trên các tài liệu tham khảo liệt kê ở phần VI của báo cáo. Các số liệu thực nghiệm và mã nguồn chương trình nếu không chỉ dẫn nguồn tham khảo đều do chúng tôi tự làm. Nếu vi phạm thì chúng tôi xin chịu trách nhiệm và tuân theo xử lý của giáo viên hướng dẫn.

TÓM TẮT— Tần số cơ bản của tín hiệu là tham số cần thiết trong việc xử lý tín hiệu âm thanh đây là đại lượng đặc trưng cho tín hiệu tuần hoàn. Bài báo cáo này thực hiện việc tìm tần số cơ bản của tín hiệu tiếng nói trên miền thời gian sử dụng hàm tự tương quan, hàm vi sai biên độ trung bình, và tín hiệu trên miền tần số sử dụng thuật toán biến đổi Fast Fourier Transform với 4 file tín hiệu mẫu được cung cấp. Kết quả thực nghiệm cho ta thấy có thể xác định được tần số cơ bản của tín hiệu tiếng nói trên miền thời gian và miền tần số.

Từ khóa— Tính tần số cơ bản, hàm tự tương quan, hàm vi sai biên độ trung bình, biến đổi Fast Fourier Transform, lọc trung vị.

Mục lục

I. ĐẶT VẤN ĐỀ

Xử lý tín hiệu âm thanh ngày càng được ứng dụng rộng rãi vào cuộc sống.Xử lý tín hiệu tiếng nói có ứng dụng rất nhiều mặt như nhận dạng tiếng nói, người nói, chất lượng giọng nói. Vì vậy việc xử lý tiếng nói đã có vai trò rất quan trọng trong cuộc sống của chúng ta. Để làm được điều này, một trong những cách thường được áp dụng đó là xác định tần số cơ bản.

Tần số cơ bản (F0) của một tín hiệu tuần hoàn bằng nghịch đảo của chu kỳ cơ bản của tín hiệu. Chu kỳ được xác định bằng khoảng thời gian ngắn nhất mà tín hiệu tuần hoàn trên miền thời gian. Nó mang ý nghĩa vật lý đặc trưng cho tin hiệu tuần hoàn. Vì vậy việc xác định tần số cơ bản là rất quan trọng. Trong báo cáo này, chúng tôi sử dụng hàm tự tương quan, hàm vi sai biên độ trung bình, biên đổi Fast Fourier Transform để xác định được tần số cơ bản này.

Báo cáo được chia thành 6 phần:

Phần I: Đặt vấn đề

Phần II: Trình bày cơ sở lý thuyết của xử lý tín hiệu tiếng nói và các thuật toán

Phần III: Trình bày mã chương trình cài đặt các thuật toán

Phần IV: Trình bày kết quả thực nghiệm, đánh giá độ chính xác giữa các thuật toán.

Phần V: Trình bày kết luận, đề xuất các phương hướng phát triển và cải thiện trong tương lai

Phần VI: Tài liệu tham khảo

II. LÝ THUYẾT XỬ LÝ TÍN HIỆU TIẾNG NÓI VÀ CÁC THUẬT TOÁN

A. Tổng quan về xử lý tín hiệu tiếng nói

Xử lý tiếng nói là sự nghiên cứu tiếng nói của con người dưới dạng tín hiệu, và các phương pháp xử lý những tín hiệu này. Tín hiệu tiếng nói thường được thể hiện dưới dạng số, tức là được "số hóa", và do đó, xử lý tiếng nói có thể được coi là giao của "xử lý tín hiệu số" và "xử lý ngôn ngữ tự nhiên".[1]

Tần số cơ bản của một tín hiệu tiếng nói là chính là tốc độ rung của dây thanh[2], được tính bằng nghịc đảo của chu kỳ cơ bản của tín hiệu, chu kỳ cơ bản được xác định bằng khoảng thời gian ngắn nhất mà tín hiệu lặp lại trên miền thời gian.

Trong phạm vi nghiên cứu, chúng tôi tập trung vào việc xác định tần số cơ bản F0 của tín hiệu tiếng nói. Trong báo cáo này chúng tôi xác định tần số cơ bản dựa vào hàm tự tương quan, hàm vi sai biên độ trung bình và biến đổi Fast Fourier Transform. Bên cạnh đó, chúng tôi sử dụng thuật toán lọc trung bị để làm trơn kết quả tần số nhận được.

B. Các thuật toán

1. Hàm tự tương quan

a) Cơ sở lý thuyết

Trong xử lý tín hiệu số nói chung và xử lý tín hiệu tiếng nói nói riêng, hàm tự tương quan dùng để biến đổi tín hiệu tuần hoàn thành một tín hiệu tuần hoàn khác có các điểm cực đại có thể xác định được dễ dàng, nhờ đó ứng dụng để xác định chu kỳ cơ bản T0 và tần số cơ bản F0[3].

Hàm tự tương quan của tín hiệu được xác định bởi công thức sau:

$$r_{xx}(l) = \lim_{N\to\infty} \frac{1}{2N+1} \sum_{n=-N}^{N} x(n)x(n+l) \qquad [4]$$

Trong đó: $r_{xx}(l)$ là giá trị hàm tự tương quan theo độ trễ l

2N+1 là độ dài khung tín hiệu

x(n) là biên độ tín hiệu tại thời điểm n

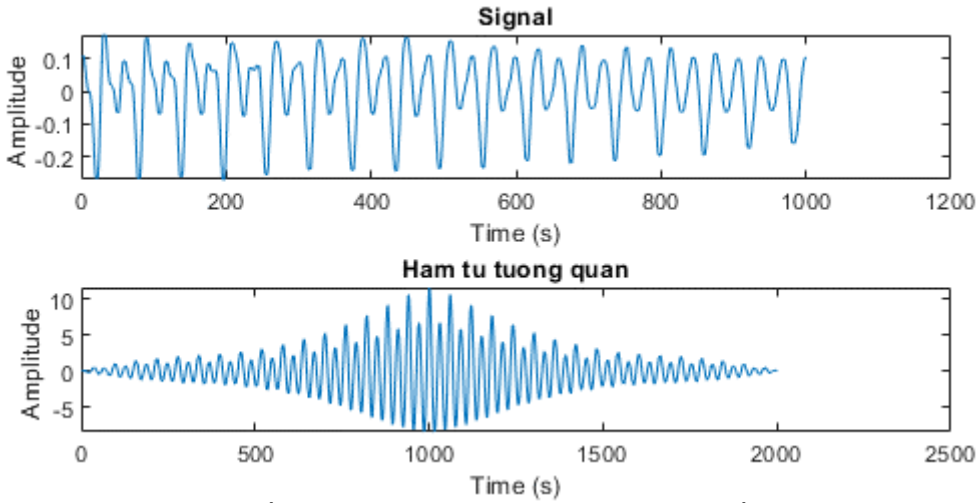
Do tính chất của hầu hết tín hiệu âm thanh là ổn định và biến đổi chậm trong khoảng thời gian ngắn, thông thường, người ta thường sử dụng phương pháp phân tích ngắn hạn. mặt khác, tần số cơ bản là một trong những đặc trưng của tín hiệu âm thah. Vì vậy công thức trên có thể viết thành:

$$r_t(\tau) = \sum_{i=t+1}^{t+W} x_i x_{i+\tau} \qquad [5]$$

Trong đó: x_i là biên độ tín hiệu tại thời điểm i

$r_t(\tau)$ là giá trị hàm tự tương quan theo độ trễ tại khung tín hiệu

W là độ dài khung tín hiệu



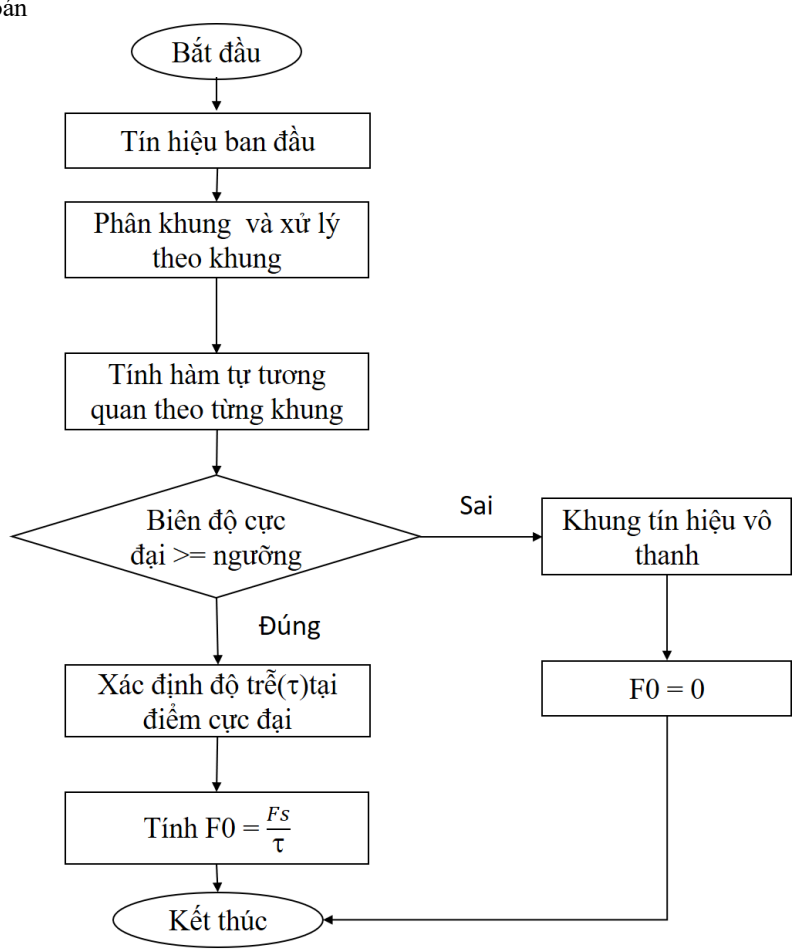
Hình 1. Đồ thị hàm tự tương quan của đoạn tín hiệu tuần hoàn

Từ đồ thị ta thấy, hàm tự tương quan là một hàm chẵn, đạt giá trị cực đại tại thời điểm có độ trễ l = 0, Đại lượng $r_{xx}(0)$ bằng đúng với năng lượng của tín hiệu. Tại các điểm là bội của chu kỳ cơ bản, giá trị hàm tự tương quan sẽ đạt giá trị cực đại. Tính chất này làm cho hàm tự tương quan trở thành cơ sở cho việc tính toán chu kỳ các tín hiệu.

Nếu tín hiệu là tuần hoàn thì hàm tự tương quan sẽ cho ra các đỉnh tại những thời điểm là bội số của chu kỳ tín hiệu. Chọn ra một đỉnh cao nhất ở hàm tự tương quan bằng cách xét hết toàn bộ các giá trị của độ trễ $\tau > 0$, ta gọi đó là τ_{max}. Vì với tín hiệu tiếng nói thì tần số cả lẫn nam và nữ nằm trong khoảng từ 80 Hz đến 400 Hz nên giới hạn của độ trễ sẽ từ $\frac{Fs}{80}$ đến $\frac{Fs}{400}$. Từ đó ta công thức tính tần số cơ bản F0 của khung tín hiệu đang xét:

$$F_0 = \frac{Fs}{\tau_{max}}$$

b) Sơ đồ khối thuật toán



Hình 2. Sơ đồ khối của thuật toán

c) Các tham số quan trọng của thuật tuán

- Độ dài khung tín hiệu: thông thường ít nhất 2 chu kỳ liên tiếp trong một khung tín hiệu là điều kiện để xác định được chu kỳ cơ bản T0, từ đó suy ra tần số cơ bản F0 của tín hiệu
- Ngưỡng xác định hữu thanh/vô thanh: Tham số này ảnh hưởng đến việc xác định âm hữu thành và vô thanh của tín hiệu tiêng nói. Nếu ngưỡng này thấp thì khi tính F0 có thể bị nhầm từ vô thanh thành hữu thanh và ngược lại.

d) Vấn đề và giải pháp khắc phục

Vấn đề: Một số khung có giá trị tần số F0 đột biến do tín hiệu đầu vào có nhiễu nên khi tính hàm tự tương quan trên từng khung làm xảy ra sự đột biến đó.

Giải pháp: Dùng hàm lọc trung bị của Matlab để lọc tín hiệu trước khi dung dữ liệu đó để tính hàm tự tương quan.

2. Hàm vi sai biên độ trung bình (AMDF)

a) Cơ sở lý thuyết

AMDF là một loại khác của hàm tự tương quan. Trong AMDF, nó lấy giá trị tuyệt đối của độ chênh lệch giữa tín hiệu ban đầu và tín hiệu trễ (phép trừ) thay vì lấy tích giữa chúng để giảm tính toán phức tạp, làm cho AMDF phù hợp hơn vào việc tính toán trong thời gian thực [7].

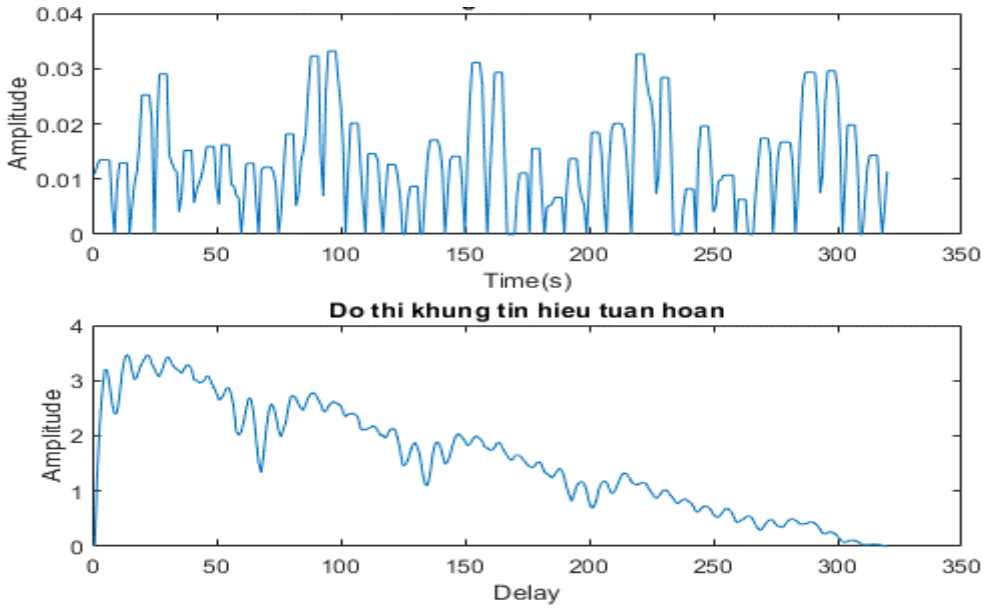
Hàm vi sai biên độ trung bình (AMDF) được xác định bởi công thức sau:

$$D(\tau) = \sum_{n=0}^{N-\tau-1} |x(n) - x(n+\tau)| \qquad [8]$$

Trong đó: $D(\tau)$ là giá trị hàm vi sai biên độ trung bình theo độ trễ τ.

$N - \tau - 1$ là độ dài khung tín hiệu

x(n) là biên độ tín hiệu tại thời điểm n.

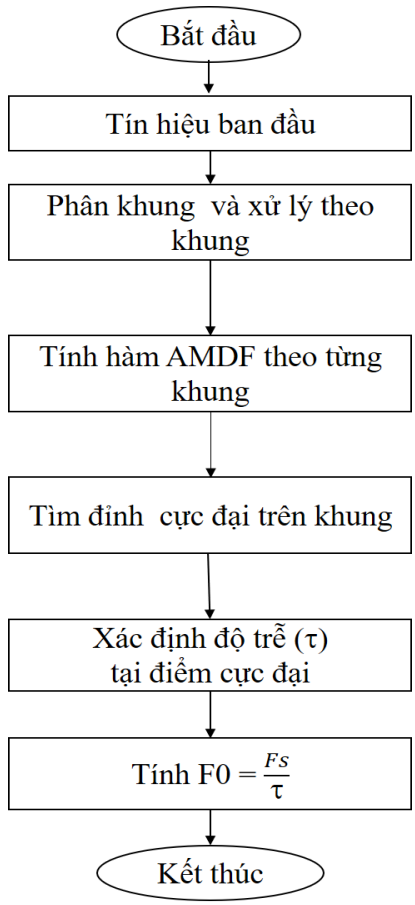


Hình 3. Đồ thị tín hiệu và đồ thị hàm vi sai biên độ của tín hiệu đó

Nếu tín hiệu là tuần hoàn thì hàm vi sai biên độ trung bình sẽ cho ra các đỉnh tại những thời điểm là bội số của chu kỳ cơ bản của tín hiệu. Chọn ra một đỉnh cao nhất ở hàm tự tương quan bằng cách xét hết toàn bộ các giá trị của độ trễ $\tau > 0$, ta gọi đó là τ_{max}. Vì với tín hiệu tiếng nói thì tần số cả lẫn nam và nữ nằm trong khoảng từ 80 Hz đến 400 Hz nên giới hạn của độ trễ sẽ từ $\frac{Fs}{80}$ đến $\frac{Fs}{400}$. Từ đó ta công thức tính tần số cơ bản F0 của khung tín hiệu đang xét:

$$F_0 = \frac{Fs}{\tau_{max}}$$

b) Sơ đồ khối thuật toán



Hình 4. Sơ đồ khối thuật toán

c) Các tham số quan trọng của thuật toán

- Độ dài khung tín hiệu: thông thường ít nhất 2 chu kỳ liên tiếp trong một khung tín hiệu là điều kiện để xác định được chu kỳ cơ bản T0, từ đó suy ra tần số cơ bản F0 của tín hiệu

d) Vấn đề và giải pháp khắc phục

Vấn đề: Thuật toán tìm đỉnh cực đại vẫn chưa tối ưu, sai số tương đối lớn.

Một số khung có giá trị tần số F0 đột biến do tín hiệu đầu vào có nhiễu nên khi tính hàm vi sai biên độ trên từng khung làm xảy ra sự đột biến đó.

Giải pháp: Hiện tại vẫn đưa ra được phương pháp tối ưu để tìm được vị trí biên độ cực đại.

3. Hàm biến đổi Fast Fourier Transform(FFT)

a) Cơ sở lý thuyết

Biến đổi Fourier rời rạc - Discrete Fourier Transform (DFT) là tiến hành biến đổi các tập dữ liệu thời gian rời rạc thành biểu diễn ở các tần số rời rạc. Điều này trái ngược với DTFT sử dụng thời gian rời rạc, nhưng chuyển đổi thành tần số liên tục. Vì thông tin tần số kết quả có bản chất là rời rạc, nên máy tính rất phổ biến sử dụng các phép tính DFT khi cần thông tin tần số.

DFT được định nghĩa bởi công thức như sau:

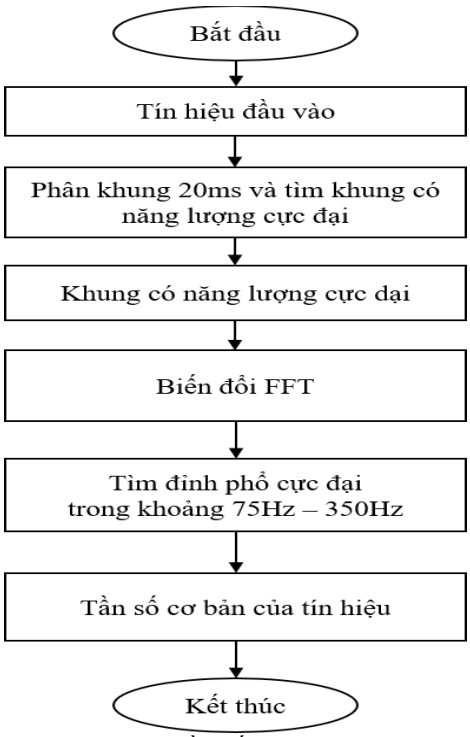
$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-\frac{j2\pi kn}{N}} \qquad [6]$$

Trong đó: X[k] là tín hiệu ở miền tần số
x[n] là tín hiệu ở miền thời gian
N là độ dài của dãy được biến đổi

thực hiên DFT của một tính hiệu N điểm cần $O(N^2)$ phép tính, nhưng biến đổi Fourier nhanh - Fast Fourier Transform (FFT) chỉ cần O(N.logN) phép tính. trong khi FFT cũng tiến hành biến đổi các tập dữ liệu thời gian rời rạc thành biểu diễn tần số rời rạc và trả về cùng kết quả DFT, nhưng có thời gian nhanh hơn. Bởi vì Do đó, FFT là thuật toán tối ưu hơn để giải quyết vấn đề đi tìm tần số cơ bản F0 được đặt ra.

Ta sẽ dùng hàm trong thư viện Matlab là fft() để tính toán FFT, và hàm findpeaks() để tìm các đỉnh phổ của tín hiệu.

b) Sơ đồ khối thuật toán



Hình 5. Sơ đồ khối thuật toán FFT

c) Các tham số quan trọng của thuật toán
- Số điểm tính FFT
- Độ dài khung tín hiệu 20ms
- Ngưỡng xét tần số cơ bản

4. Ảnh hưởng của bộ lọc trung vị, hàm cửa sổ Hamming và số điểm N_FFT

a) **Hàm lọc trung vị**

1) Cơ sở lý thuyết

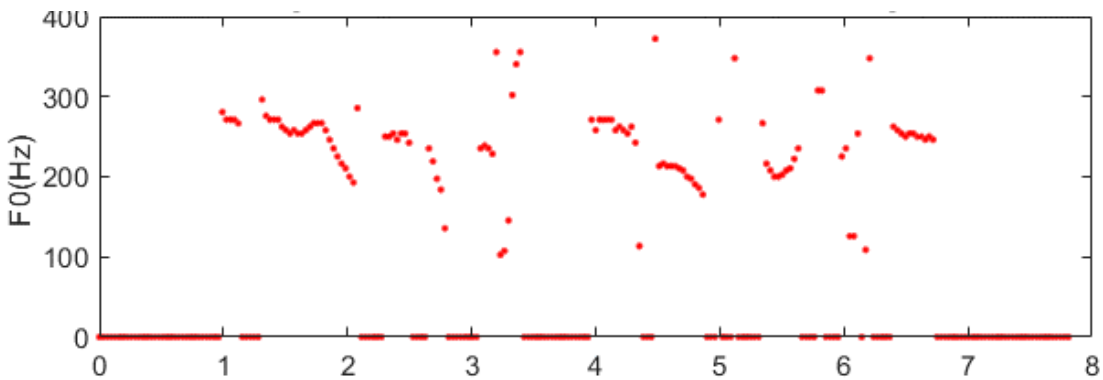
Trong hầu hết các ứng dụng xử lý tín hiệu, làm trơn tuyến tính hầu như được sử dụng để loại bỏ các thành phần nhiễu trong tín hiệu. Tuy nhiên, với một vài ứng dụng xử lý tiếng nói, làm trơn tuyến tính không hoạt động hiệu quả do tính chất của tín hiệu được làm trơn. Một ví dụ là đường F0 xác định từ tín hiệu tiếng nói không những chứa các giá trị thay đổi một cách bất thường (outliers) so với các giá trị lân cận mà còn gián đoạn tại các vùng chuyển tiếp giữa âm vô thanh và âm hữu thanh.

Việc làm trơn trung vị giúp loại bỏ được giá trị nhảy vọt so với các giá trị lân cận mà vẫn bảo toàn các điểm gián đoạn trong tín hiệu.

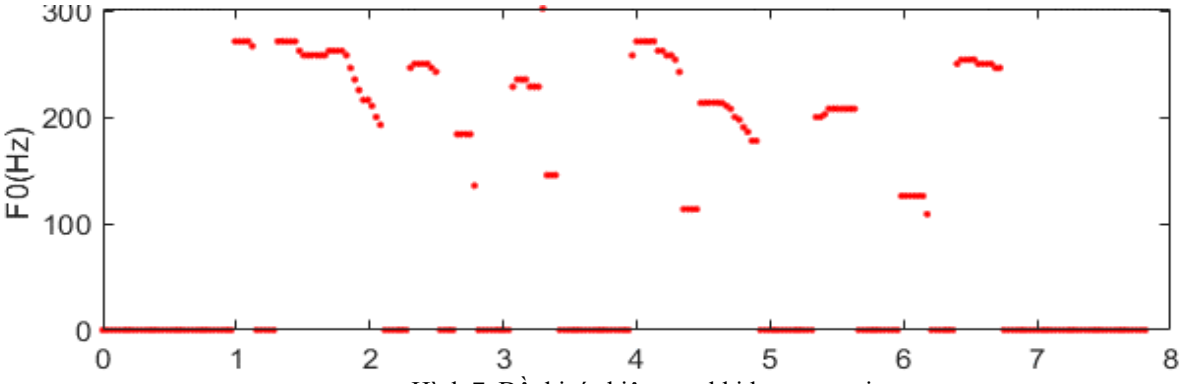
Biến đổi trung vị với độ dài cửa sổ lọc N tuân theo các tính chất: $M_N[\propto x(n) = \propto M_N[x(n)]$

Các giá trị trung vị không bị làm nhòe các điểm gián đoạn nếu không có sự gián đoạn trong các mẫu N/2

Mặc dù lọc trung vị giữ lại được các đoạn sắc nét trong tín hiệu, kỹ thuật này lại không loại bỏ hoàn toàn được các thành phần giống nhiễu của tín hiệu.



Hình 6. Đồ thị tín hiệu trước khi lọc trung vị



Hình 7. Đồ thị tín hiệu sau khi lọc trung vị

2) Tham số của bộ lọc

Trong thuật toán này, độ dài của cửa sổ N là tham số quan trọng nhất. Để lấy chính xác điểm trung vị, N thường là số lẻ. Bên cạnh đó, nếu kích thước N càng lớn, thì giá trị tính toán sẽ được thu hẹp và điểm bị lỗi sẽ được dịch chuyển gần hơn so với các điểm đúng còn lại. Nhưng nếu N quá lớn, có thể bộ lọc trung vị k những thay đổi cả các điểm bất thường, mà còn làm thay đổi các điểm đúng của tín hiệu, gây sai lệch tín hiệu.

b) **Hàm cửa sổ Hamming và số điểm N_FFT**

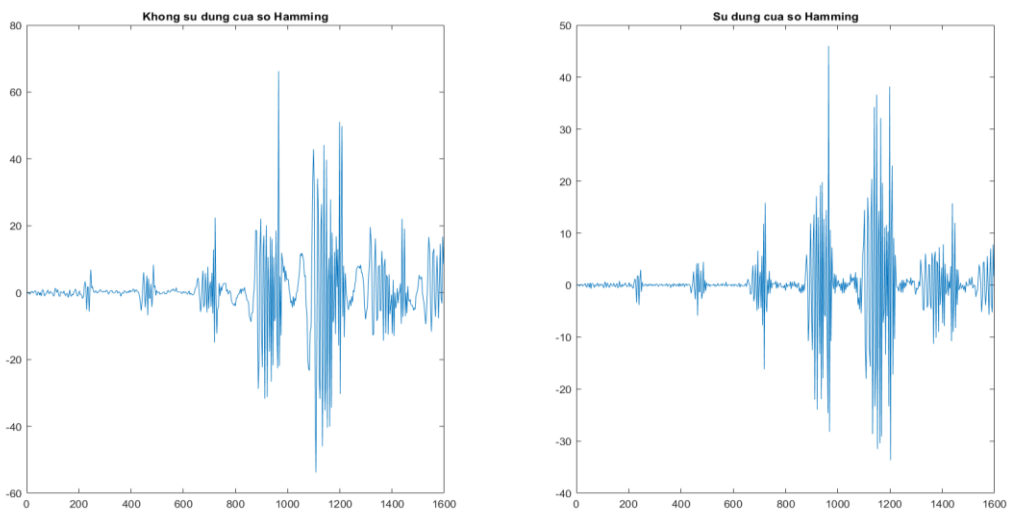
1) Cơ sở lý thuyết

Phép biến đổi rời rạc DFT tác động trên tín hiệu có độ dài hữu hạn, nên cần phải hạn chế độ dài của các tín hiệu để nghiên cứu phổ của các tín hiệu đó. Vì vậy khi phân tích tín hiệu tiếng nói, ta phải xử lý tín hiệu thành từng đoạn hay từng khung có thời gian nhỏ từ 1-30ms.

Hàm cửa sổ có rất nhiều loại, tuy nhiên trong báo cáo này chúng tôi sử dụng hàm cửa sổ Hamming, đây là hàm cửa sổ được sử dụng trong phân tích phổ để giảm hiện tượng rò phổ. Hàm cửa sổ Hamming thường được dùng nhiều hơn trong các tín hiệu có phạm vi rộng[6]

Dạng tổng quát của cửa sổ Hamming:

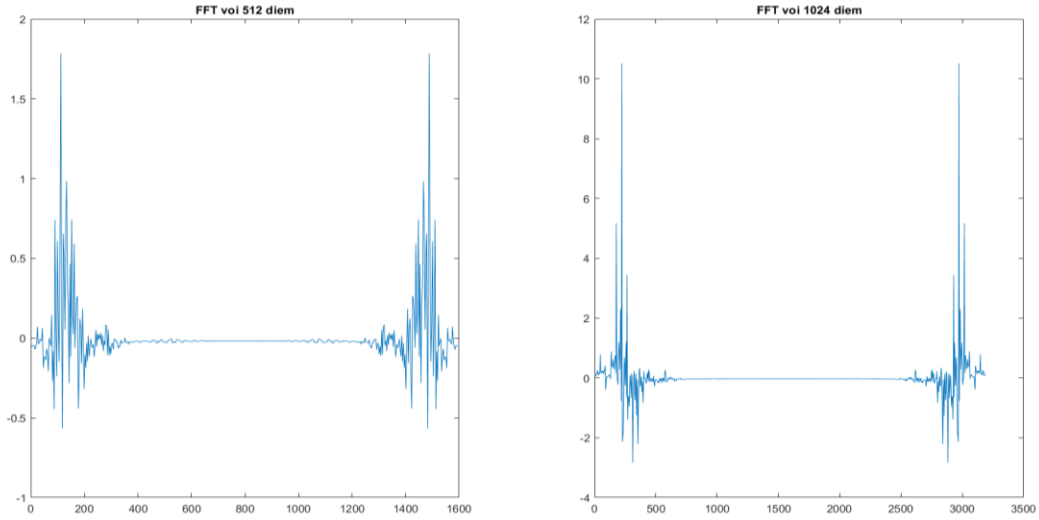
$$W(n) = \begin{cases} 0.54 + 0.46\cos\left(\frac{2\pi n}{N-1}\right) & với \le n \le N-1 \\ 0 & với\ n\ còn\ lại \end{cases}$$



Hình 8. Đồ thị khi sử dụng và không sử dụng cửa sổ hamming

Việc sử dụng cửa sổ Hamming là hợp lí hơn cửa sổ hình chữ nhật vì ở đây cửa sổ Hamming giảm đi hiện tượng rò phổ một cách đáng kể. Việc này giúp ích cho việc xử lí tín hiệu một cách chính xác hơn.

2) Ảnh hưởng của số điểm N_FFT



.Hình 9. Đồ thị thể hiện khi sử dụng N_FFT với 512 điểm và 1024 điểm

Với việc sử dụng số điểm khác nhau cho phân tích FFT trên cùng 1 đoạn tín hiệu, cụ thể là 512 và 1024, ta có thể thấy rằng số lượng điểm phân tích FFT lớn ở một mức độ phù hợp, thì nó làm giảm đáng kể số lượng các đỉnh phổ có biên độ xấp xỉ và nằm gần nhau hơn. Lí do là do cùng 1 độ dài khung cửa sổ nhưng nếu ta tăng N-FFT thì số lần lấy mẫu sẽ tăng lên dễ quan sát hơn, làm cho đồ thị được mượt hơn.

III. MÃ CHƯƠNG TRÌNH CÀI ĐẶT CÁC THUẬT TOÁN

```
clear all
% Load file am thanh voi bien do ghi vao mang a
% Tan so lay mau Fs
% [x,Fs] = audioread('lab_male.wav');
% [x,Fs] = audioread('lab_female.wav');
 [x,Fs] = audioread('studio_male.wav');
% [x,Fs] = audioread('studio_female.wav');

%==========================CHUONG TRINH CHINH=============================
% Do dai moi khung tin hieu
fr_time = 0.02;
% So luong mau tren moi khung tin hieu
fr_len = fr_time*Fs;
% Tong so luong khung hinh
fr_num = floor(length(x)/fr_len);

%-------------------------Thuat toan Tu tuong quan-------------------------

% Chuyen gia tri x ve dang ma tran fr_num hang, fr_len cot
data_fr = convertx(fr_num,fr_len,x);
% Ham tinh gia tri cua so hamming
w = hamming_window(fr_len);
% Ham tim F0
[F0_TTQ, xcorr_fr, des_maxpeak_fr, delay_fr]  = Find_F0(fr_num,fr_len,x,Fs,w);
% Loc trung vi voi N = 7;
F0_TTQ = medfilt1(F0_TTQ,7);
% Ham tinh tan so co ban trung binh cua file
F0_ave_TTQ = funct_F0_ave(F0_TTQ);

figure(1);
subplot(211);
plot(x);
title('Tin hieu dau vao','LineWidth',1);
xlabel('Time (s)');
ylabel('Amplitude ');
```

```
subplot(212);
t = 1:fr_len:fr_len*fr_num;
stem(t,F0_TTQ,'.', 'r','linestyle','none');
title('Duong bieu dien tan so co ban F0, Ham tu tuong quan');
xlabel('Delay');
ylabel('F0(Hz) ');

figure(2);
des = 50;
subplot(211);
plot(data_fr(des,:));
title('Do thi khung tin hieu do dai 20ms','LineWidth',1);
xlabel('Time(s)');
ylabel('Amplitude');

if( F0_TTQ(des) == 0)
    title1 = 'Do thi cua khung tin hieu khong tuan hoan';
else
    title1 =['Do thi ham tu tuong quan cua tin hieu tuan hoan co F0 = ',
num2str(F0_TTQ(des))];
end
subplot(212);
plot(xcorr_fr(des,:));
yline(0.3*max(xcorr_fr(des,:)),"Color","red");
title(title1,'LineWidth',1);
hold on;
if(F0_TTQ(des)~= 0)
    plot(delay_fr(des), des_maxpeak_fr(des),"v");
end
xlabel('Delay');
ylabel('Amplitude');
hold off;


%-------------------------------------------------------------------------------

%----------------------------Thuat toan vi sai bien do-------------------------

% Sao chep data x sang data u de dung thuat toan amdf
u = repelem(x,1);
% Xu ly tin hieu u de thuc hien thuat toan amdf
u =sigprocessing(x);
% Chuyen gia tri x ve dang ma tran fr_num hang, fr_len cot
data_fr = convertx(fr_num,fr_len,u);
% Ham tim F0
[F0_AMDF, amdf_fr] = Find_F0_amdf(fr_num,fr_len,u,Fs);
% Loc trung vi voi N = 7;
F0_AMDF = medfilt1(F0_AMDF,7);
% Ham tinh gia tri trung binh tan so co ban F0
F0_ave_AMDF = funct_F0_ave(F0_AMDF);

figure(3);
subplot(211);
plot(x);
subplot(212);
t = 1:fr_len:fr_len*fr_num;
F0_AMDF = medfilt1(F0_AMDF,5);
stem(t,F0_AMDF,'.', 'r','linestyle','none');
title('Duong bieu dien tan so co ban F0, Ham AMDF');
xlabel('Delay');
ylabel('F0(Hz) ');
```

```
figure(4);
des = 120;
subplot(211);
plot(data_fr(des,:));
title('Do thi khung tin hieu do dai 20ms','LineWidth',1);
xlabel('Time(s)');
ylabel('Amplitude');

subplot(212);
plot(amdf_fr(des,:));
title('Do thi khung tin hieu tuan hoan, ham AMDF','LineWidth',1);
hold on;
xlabel('Delay');
ylabel('Amplitude');
hold off;

figure (5);
des1 = 65;
subplot(211);
plot(data_fr(des1,:));
title('Do thi khung tin hieu do dai 20ms','LineWidth',1);
xlabel('Time(s)');
ylabel('Amplitude');

subplot(212);
plot(amdf_fr(des1,:));
title('Do thi khung tin hieu khong tuan hoan, ham AMDF','LineWidth',1);
hold on;
xlabel('Delay');
ylabel('Amplitude');
hold off;
%-------------------------------------------------------------------------

%----------------------------Thuat toan bien doi FFT-----------------------
figure(6);
% Xuat tin hieu ra o mien thoi gian
t=1/Fs:1/Fs:(length(x)/Fs);     % Truc thoi gian
subplot(211), plot(t,x,'LineWidth',1);
title('Tin hieu ban dau');
xlabel('Thoi gian (s)');
ylabel('Bien do');

% Nhan tin hieu voi cua so Hamming
x = x.*hamming(N);

% Tao vecto khung chua tin hieu co do dai 20ms
khung = zeros(1,fr_len);

N_FFT = 16192;% So luong mau tan so(luy thua cua 2)

% Roi rac hoa truc tan so
k=1:N_FFT;
w=k*Fs/N_FFT;

% Tim khung co nang luong lon nhat, thuc hien tinh toan tren khung do
% PowMax Nang luong cuc dai, Start vi tri bat dau cua khung co PowMax
PowMax = 0; Start = 1;
for i = 1:fr_num

    for j =(i-1)*fr_len +1 : (fr_len*i)
```

```
        khung(j) = x(j);
    end
    Pow = 0;    % Bien tam thoi, luu gia tri nang luong cua khung dang duoc xet
den
    for j = (i-1)*fr_len +1 : (fr_len*i)
        Pow = Pow + khung(j).*2;
    end
    % Tim khung chua nang luong lon nhat cung diem bat dau cua no
    if Pow > PowMax
        PowMax = Pow;
        Start = i;
    End
end

% Signal la tin hieu co nang luong cuc dai, chinh la tieng noi
% Su dung Signal de xac dinh tan so co ban F0
Signal = x((Start-1)*fr_len +1 : (fr_len*Start));

% Bien doi Fast Fourier Transform tren Signal, voi
FFT = abs(fft(Signal, N_FFT));


% Mang magnitude chua gia tri bien do cua Signal da duoc bien doi FFT
% Mang co do dai tu 1-350, moi chi so tuong ung voi gia tri cua tan so
magnitude = (FFT(1:350));

% Gia tri ban dau cua tan so
F0 = 75;

% Gia tri bien do t?i tan so bat dau
max = magnitude(110);

% Tai diem co gia tri bien do cuc dai,
% Xem do chinh la tan so co ban F0 can tim
for i = 110:length(magnitude)
    if max < magnitude(i)
         max = magnitude(i);
         F0 = i;
    end
end

% Ve FFT ra o mien tan so
subplot(212), plot(w(1:N_FFT/2),FFT(1:N_FFT/2));
xlabel('Tan so (Hz)');
ylabel('Bien do');
hold on;
% Ve ra gioi han gia tri tan so tu 75Hz den 350Hz
xline((75), '--r',1), xline((350), '--r',1);
% Danh dau dinh pho ma ta xac nhan do la tan so F0
hold on;
plot(w(F0), FFT(F0), 'x');

% Gia tri cua thuc cua tan so
F0 = F0*Fs/N_FFT;
title(['Khung tin hieu co nang luong cuc dai, Tan so F0 = ',num2str(F0),'
(Hz)']);
% In ket qua F0
fprintf('Tan so co ban F0: %0.2f Hz\n', F0);
%--------------------------------------------------------------------------

%=========================DINH NGHIA HAM=====================================
```

```
% Thuat toan dua du lieu x ve dang ma tran
% Dau vao: gia tri cua tin hieu x
% Dau ra: Ma tran chua gia tri tren tung khung hinh
function [data_fr] = convertx(fr_num,fr_len,x)
    for i = 0:fr_num-2
        temp = x(i*fr_len+1:(i+1)*fr_len); % chuyen gia tri hang ngang sang ma
tran
        data_fr(i+1,:) = temp;
    end
 end

% Ham tim tan so co ban cua tung khung va diem co bien do cuc dai
% fr_num: tong so khung cua tin hieu
% fr_len: tong so tin hieu tren moi khung
% x: tin hieu dau vao sau khi loc trung vi
% Fs: tan so lay mau cua tin hieu
function [F0, xcorr_fr, des_maxpeak_fr, delay_fr] =
Find_F0(fr_num,fr_len,x,Fs,w)
    xcorr_fr = zeros(fr_num,fr_len);
    for k = 1 : fr_num
        fr_index = (k-1)*fr_len; %chi so mang cua tin hieu x tai khung thu t
        r = zeros(1,fr_len);
    for delay = 1 : fr_len
        for j = 1 : fr_len - delay
            r(delay) = r(delay) + x(fr_index + j) * x(fr_index + j + delay -
1)* w(j)^2;
            xcorr_fr(k,delay) = xcorr_fr(k,delay)+ x(fr_index + j) * x(fr_index
+ j + delay -1 )* w(j)^2;
        end                                     % Cong thuc ham tu tuong quan
    end
         if (r(1) < 0.03)                       % Loai bo nhung khung co gia
tri dau tien be hon nguong 0.03
            F = 0;
          else
            F = 0;
            max_peak = 0;
            delay = 0;
            for i = 2 : length(r) - 1
                if (r(i) > r(i -1) && r(i) > r(i+1) && r(i) >= max_peak)   %
                                                Ham tim diem cuc dai
                    max_peak = r(i);                % Lay diem cuc dai
                    delay = i;
                end
            end
            if ( max_peak >= xcorr_fr(1,1)*0.3) % Lay nguong 30% diem cuc dai
                F = Fs/delay;                   % Tinh F0
                if (F <= 80 || F >= 400)        % So sanh dieu kien thoa man F0
                    F = 0;
                end
            end
            delay_fr(k) = delay;                % Luu vi tri delay
            des_maxpeak_fr(k) = max_peak;       % Luu vi tri cac dinh delay
        end
        F0(k) = F;
    end
end
% Gia tri cua so hamming
% Dau vao: do dai khung cua so (length_w)
%-Dau ra: gia tri ham cua so hamming
 function [w]= hamming_window(fr_len)
     w=[];
     for i=1:fr_len
```

```
         w(i) = 0.54 - 0.46*cos(2*pi*(i-1)/(fr_len-1));    % cong thuc hamming
     end
 end

% Ham tim tan so co ban cua tung khung va diem co bien do cuc dai
% fr_num: tong so khung cua tin hieu
% fr_len: tong so tin hieu tren moi khung
% x: tin hieu dau vao sau khi loc trung vi
% Fs: tan so lay mau cua tin hieu
function [F0, amdf_fr] = Find_F0_amdf(fr_num,fr_len,x,Fs)
    amdf_fr = zeros(fr_num,fr_len);
    for k = 1 : fr_num
        fr_index = (k-1)*fr_len; %chi so mang cua tin hieu x tai khung thu k
        for delay = 1 : fr_len
            for j = 1 : fr_len - delay
                amdf_fr(k,delay) = amdf_fr(k,delay)+ abs(x(fr_index + j) -
x(fr_index + j + delay -1 ));
            end
        end
       [pks, y] = findpeaks(-amdf_fr(k,40:190));   % Tim cac dinh sau khi tinh
amdf tren 1 khung
        max_peak = max(pks);                        % Tim dinh lon nhat
        F = 0;
        delay = 0;
        for i = 1: length(pks)
            if(max_peak == pks(i))             % So sanh dinh lon nhat voi mang
dinh de lay vi tri tre
                delay = y(i);                  % vi tri tre
                F = Fs/delay;                  % Tim ta so co ban
            end
        end
        if ( F <= 80 || F >= 400)               % kiem tra tan so co ban thuoc
khoang 80-400Hz
            F = 0;                              % Gan vao mang gia tri F0
        end
        F0(k) = F;
    end
end

% Ham xu ly tin hieu data de dung cho thuat toan amdf
% Dau vao: Mang data x
% Dau ra: Mang data u sau khi xu ly
function [u] =sigprocessing(x)
for i =1:length(x)
    u(i) = abs(x(i));       % Lay tri gia tri cua tin hieu tai mau thu i
    if(u(i) < 0.004)        % Neu mau thu i < 0.004 thi cho gia tri mau = 0
        u(i) = 0;
    end
end
end
function [amdf_fr] = AMDF(fr_num,fr_len,x)
    amdf_fr = zeros(fr_num,fr_len);
    for k = 1 : fr_num
        fr_index = (k-1)*fr_len;
        for delay = 1 : fr_len
            for j = 1 : fr_len - delay
                amdf_fr(k,delay) = amdf_fr(k,delay)+ abs(x(fr_index + j) -
x(fr_index + j + delay -1 ));
            end
        end
    end
end
```

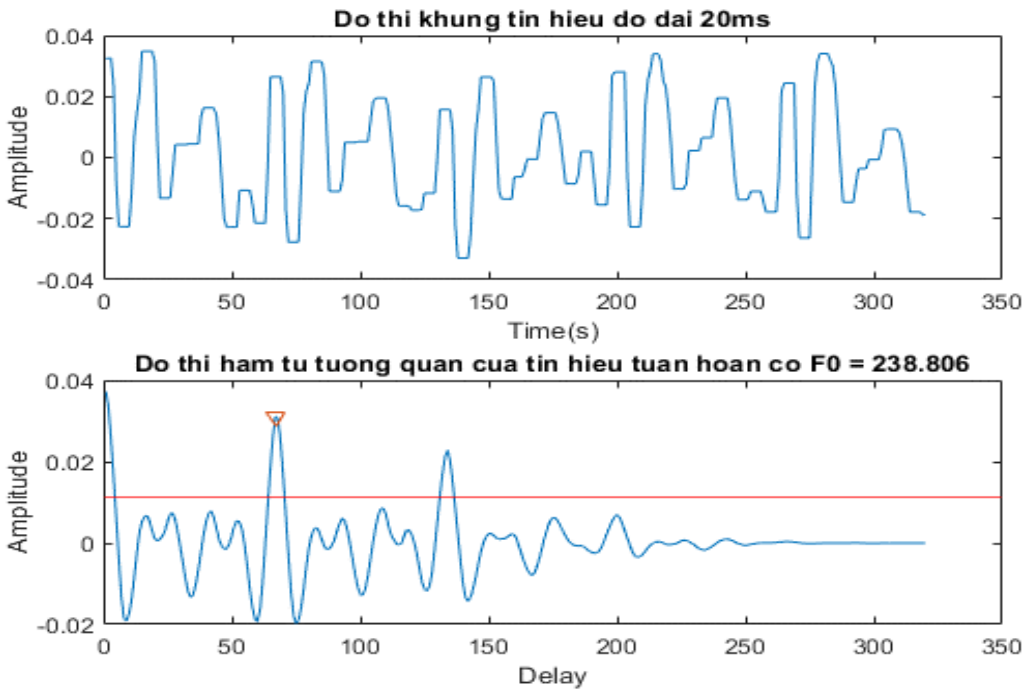
```
% Ham tim gia tri trung binh cua F0
% Dau vao: Mang gia tri cua F0
% Dau ra: Gia tri trung binh F0 cua file tin hieu
function [F0_ave] = funct_F0_ave(F0)
    Dem = 0 ;
    F0_ave = 0 ;
for i = 1:length(F0)
    if F0(i) > 0                  % Kiem tra F0
        F0_ave = F0_ave + F0(i); % Tinh trung binh F0
        Dem = Dem + 1;           % Tang bien dem de lay gia tri chia
    end
end
    F0_ave = F0_ave/Dem;
end
```

IV. KẾT QUẢ THỰC NGHIỆM

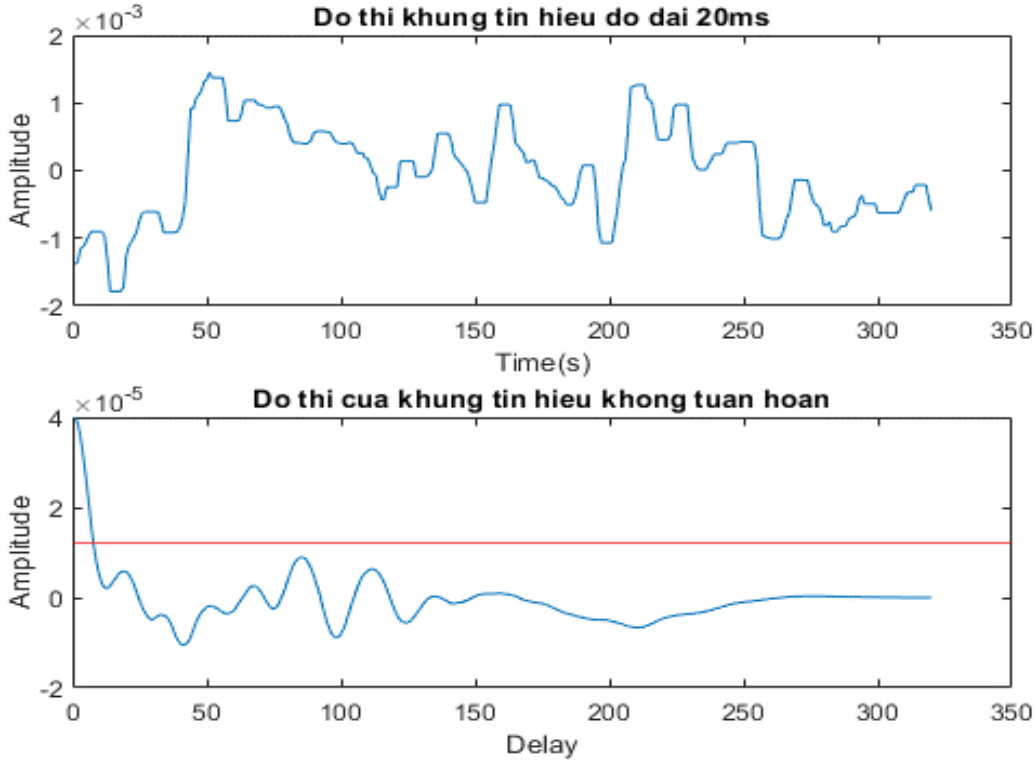
A. Thuật toán Hàm tự tương quan

1. Kết quả định tính

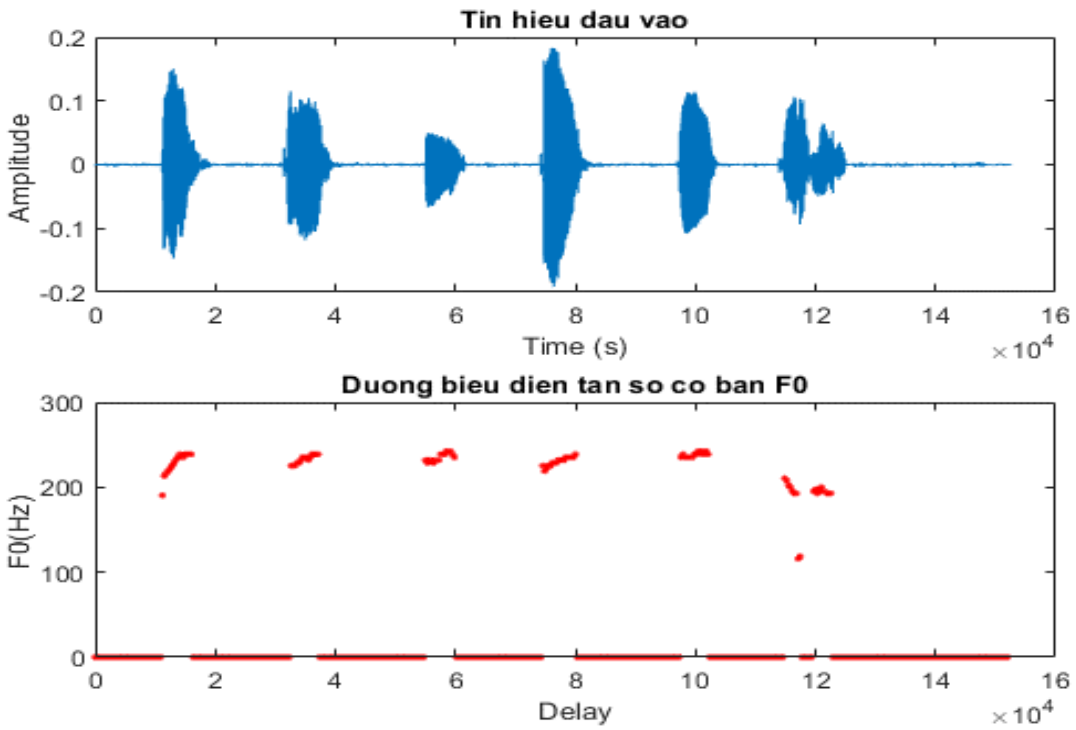
1.1 File lab_female.wav



Hình 10. Đồ thị đoạn tín hiệu âm thanh tuần hoàn và hàm tự tương quan của nó

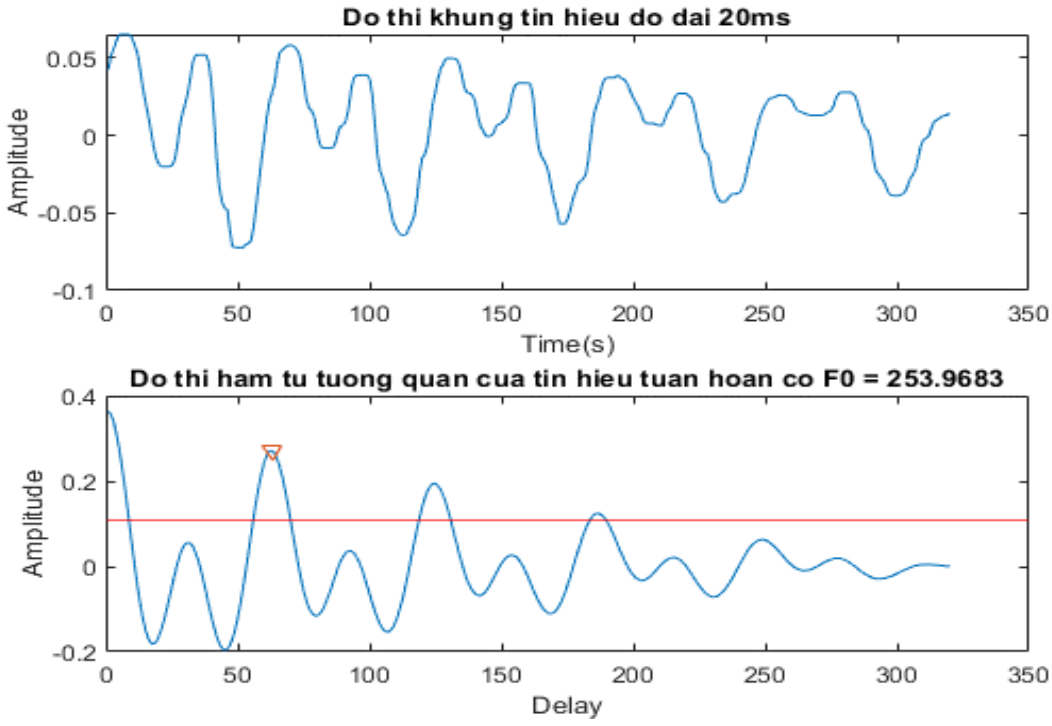


Hinh 11. Đồ thị đoạn tín hiệu âm thanh không tuần hoàn và hàm tương quan của nó

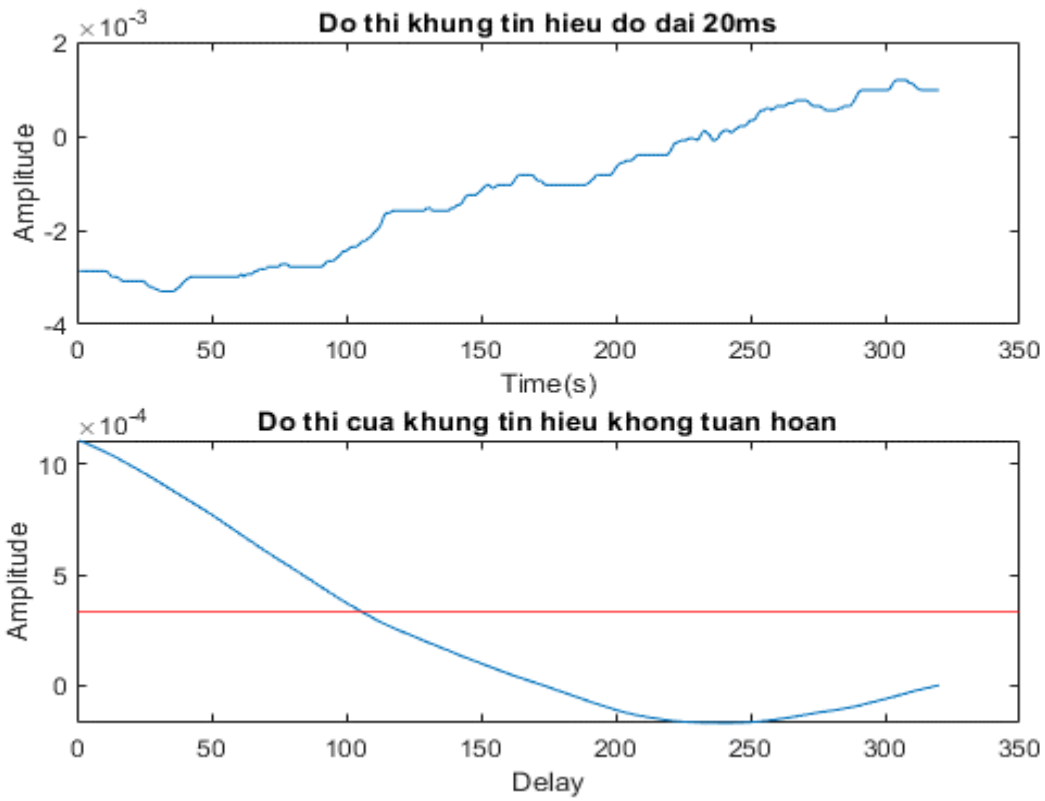


Hình 12. Đồ thị tín hiệu và biểu diễn tần số cơ bản F0 của tín hiệu

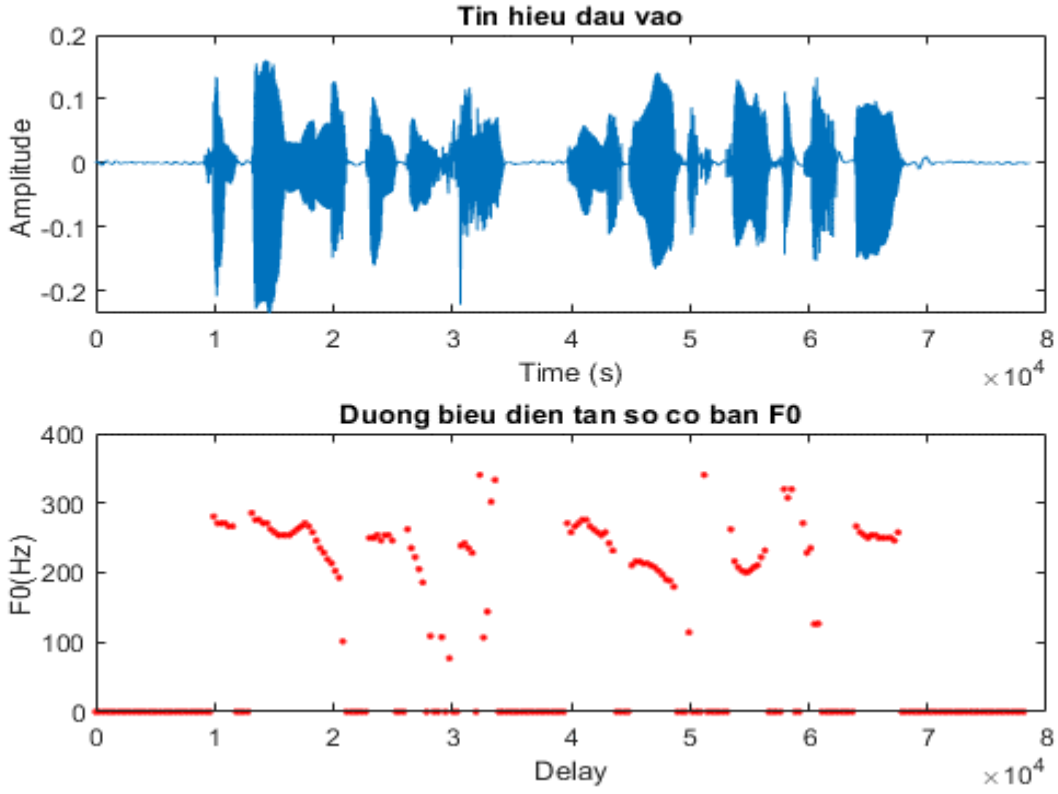
1.2 File studio_female.wav



Hình 13 . Đồ thị đoạn tín hiệu âm thanh tuần hoàn và hàm tự tương quan của nó



Hình 14. Đồ thị đoạn tín hiệu âm thanh không tuần hoàn và hàm tương quan của nó



Hình 15. Đồ thị tín hiệu và biểu diễn tần số cơ bản F0 của tín hiệu

2. Kết quả định lượng

File	F0 Thuật toán(Hz)	F0 thủ công(Hz)	Độ chênh lệch(%)
Lab_male.wav	122.35	117.19	4.40
Lab_female.wav	224.401	202.596	10.76
Studio_male.wav	127.939	121.390	5.40
Studio_female.wav	240.907	230.67	4.44

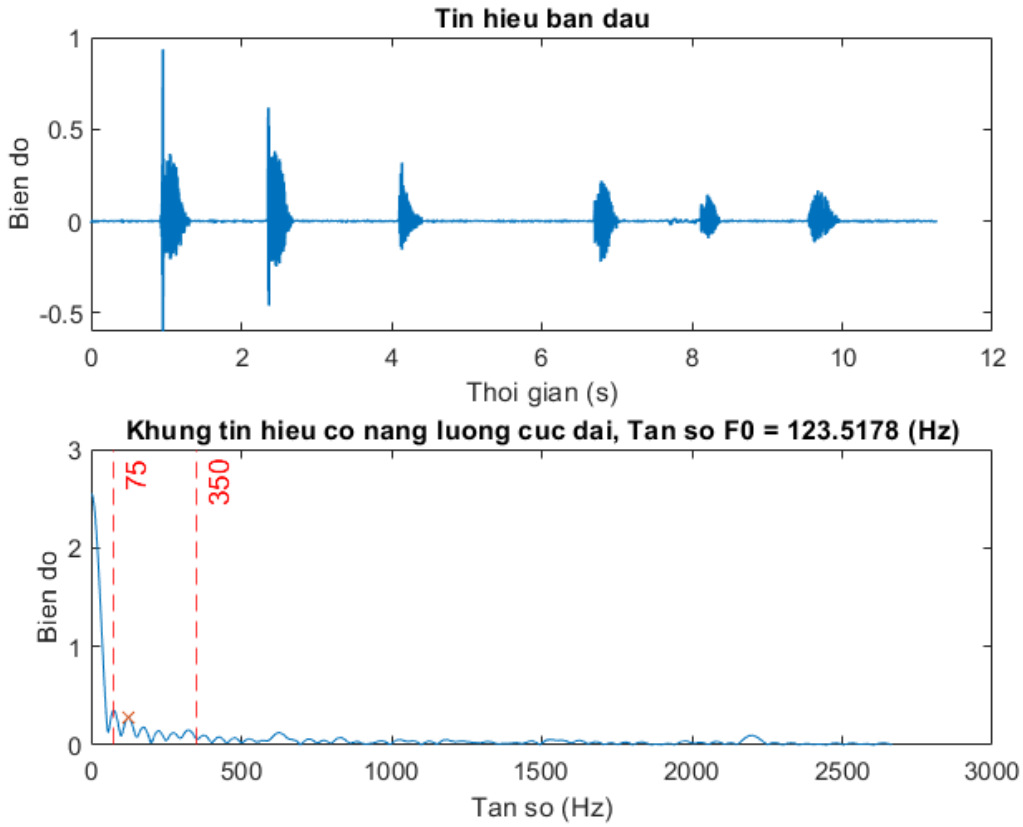
3. Nhận xét

Từ bảng số liệu ta thấy, tần số cơ bả của các file tín hiệu đầu vào so với tần số cơ bản được xác định thủ công bằng phần mềm wavesufer không có sự chênh lệch lớn. Tuy nhiên đối với hai file lab_male.wav và file studio_male.wav có sự chệnh lệch lớn hơn hai file còn lại nhưng không đáng kể. Điều này xảy ra do khoảng lấy tần số từ 80 Hz – 400Hz có sự ảnh hưởng đến thuật toán lấy tần số cơ bản của mỗi khung.
Điều này cho thấy, hàm tự tương quan có thể xác định được khá chính xác tần số cơ bản của file tín hiệu.

B. Thuật toán biến đổi Fast Fourier Tranform (FFT)

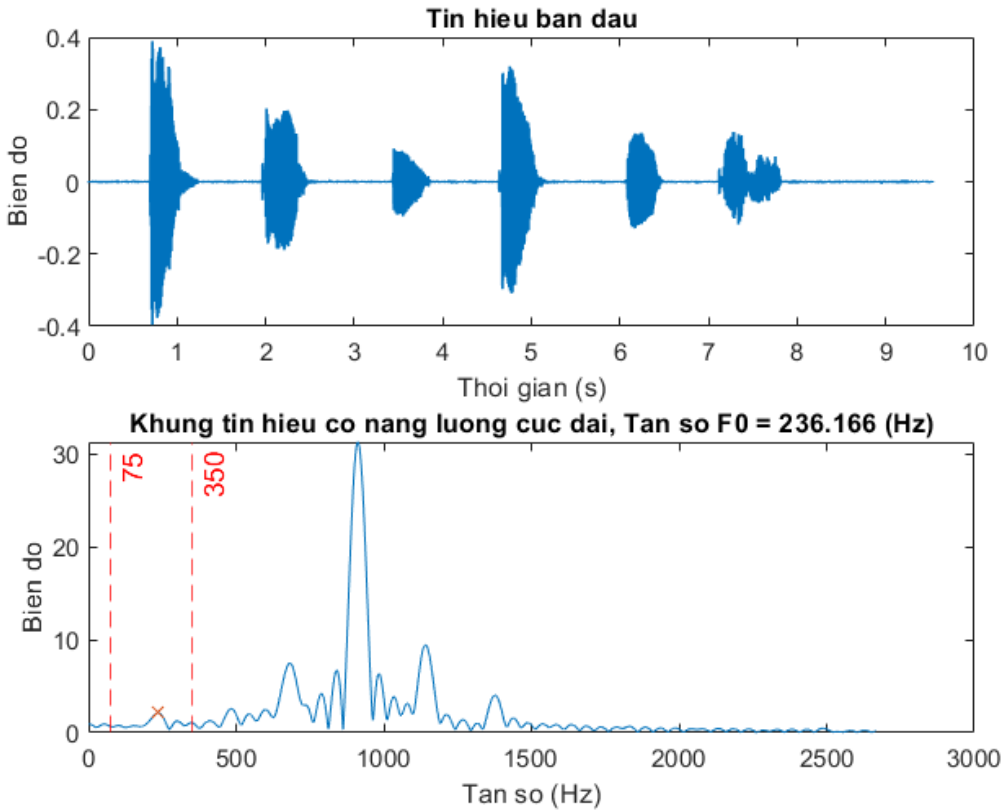
1. Kết quả định tính

1.1 File lab_male.wav



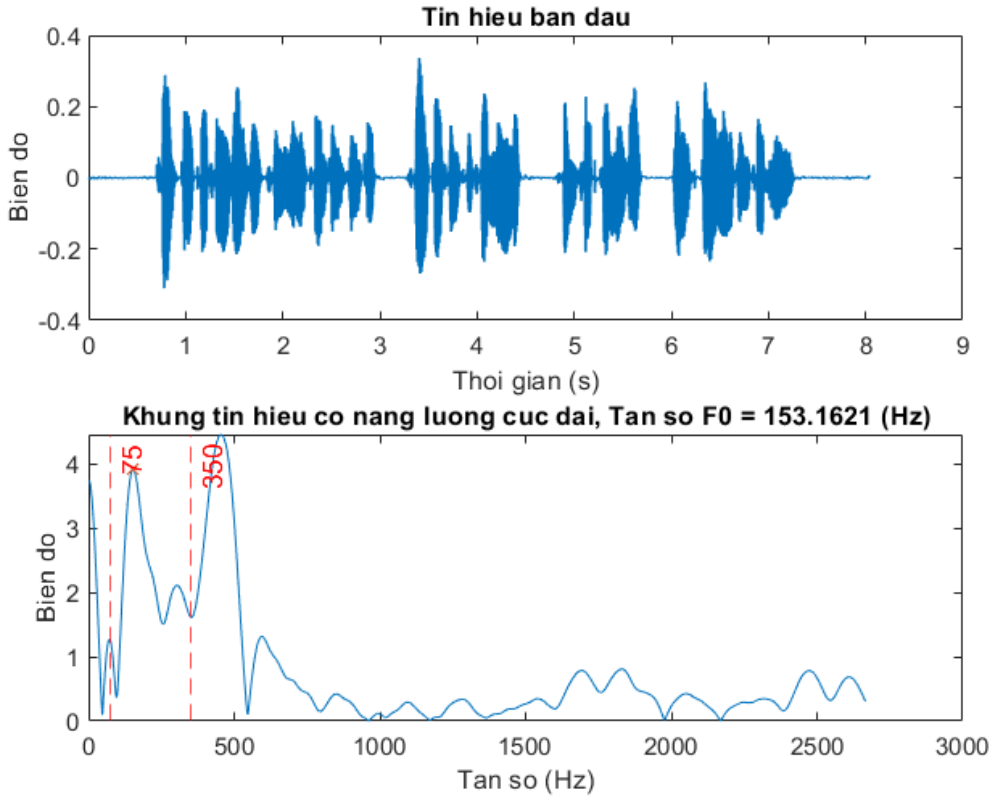
Hình 16. Đồ thị phổ biên độ của tín hiệu và tần số cơ bản

1.2 File lab_female.wav



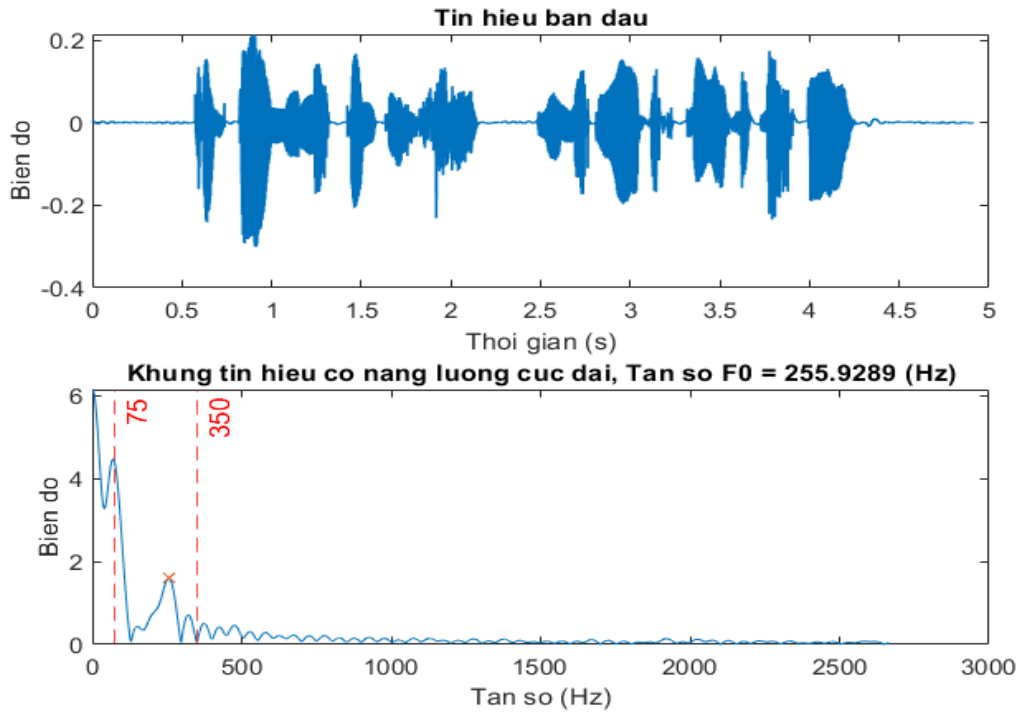
Hình 17. Đồ thị phổ biên độ của tín hiệu và tần số cơ bản

1.3 File studio_male.wav



Hình 18. Đồ thị thể hiện phổ biên độ và tần số cơ bản

1.4 File studio_female.wav



Hinh 19. Đồ thị phổ biên độ của tín hiệu và tần số cơ bản

2. Kết quả định lượng

File	F0 Thuật toán(Hz)	F0 thủ công(Hz)	Độ chênh lệch(%)
Lab_male.wav	123.51	117.19	5.39
Lab_female.wav	236.16	202.596	16.57
Studio_male.wav	153.16	121.390	26.17
Studio_female.wav	255.93	230.67	10.95

3. Nhận xét

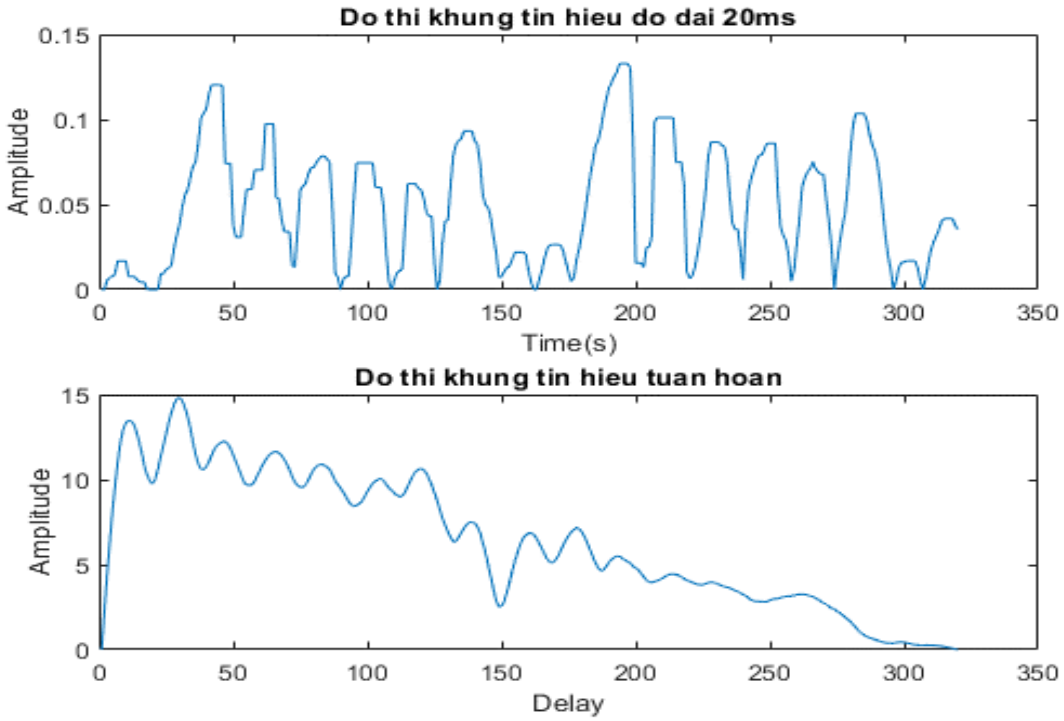
Sự sai khác trong các tần số giữa việc đo thủ công và tính toán qua thuật toán là không quá lớn. Tuy nhiên, đối với file âm thanh studio_male, sai lệch ở đây lại rất cao, gần 27%. Sai số trên có thể là do thuật toán còn khá đơn giản, thiếu sự đánh giá tổng thể trên toàn bộ tín hiệu mà chỉ dựa vào khung tín hiệu (20ms) có năng lượng lớn nhất, điều này chỉ đánh giá được tại cục bộ điểm ta xét, thiếu độ khách quan, chính xác.

Ngoài ra khi khảo sát, tồn tại nhiều biên độ lớn trong khoảng tần số giọng nói của con người ta xét (75Hz – 350Hz) mà có đỉnh phổ không thuộc khoảng tần số trên.

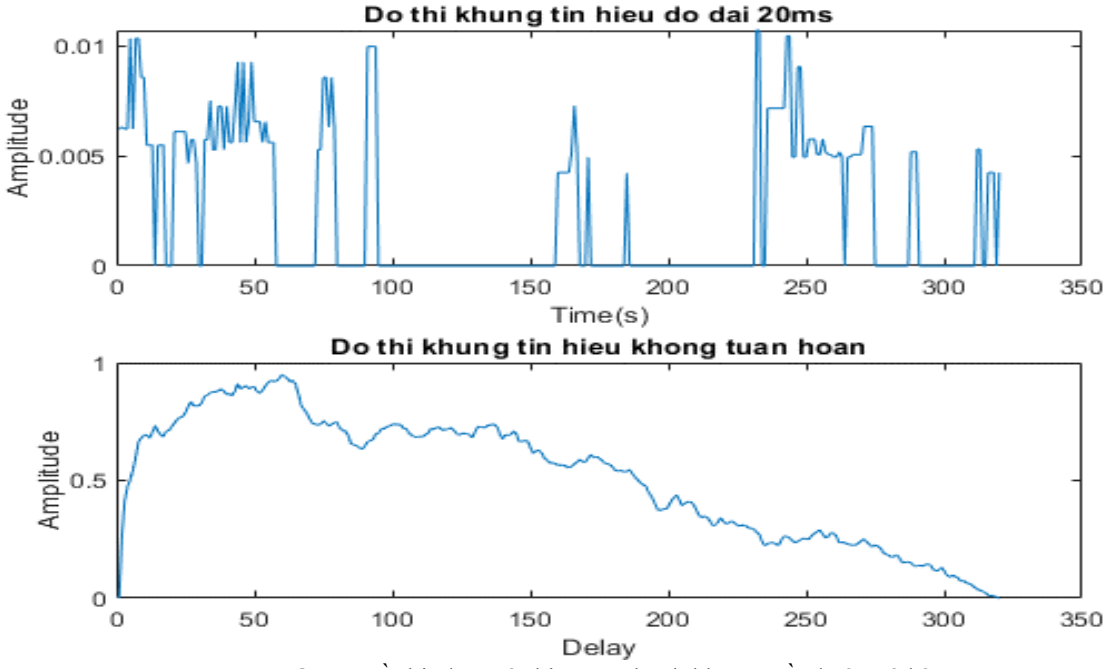
C. Thuật toán Vi sai biên độ trung bình (AMDF)

1. Kết quả định tính

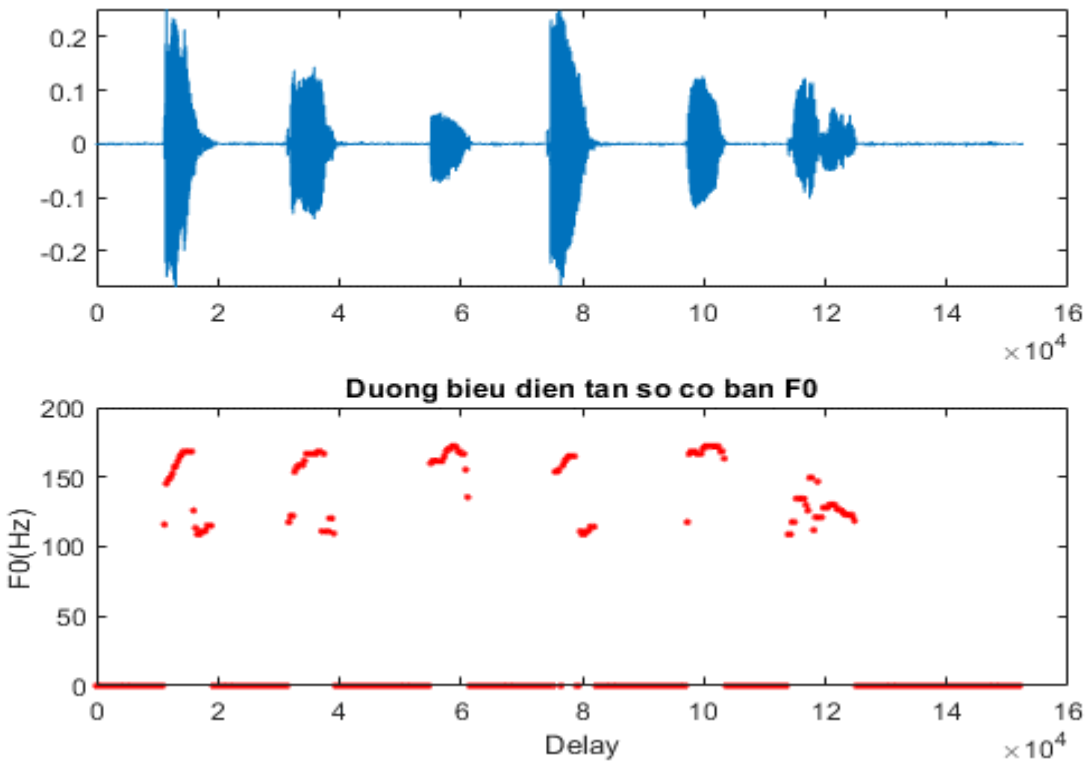
1.1 File Lab_female.wav



Hình 20. Đồ thị đoạn tín hiệu âm thanh tuần hoàn và hàm AMDF

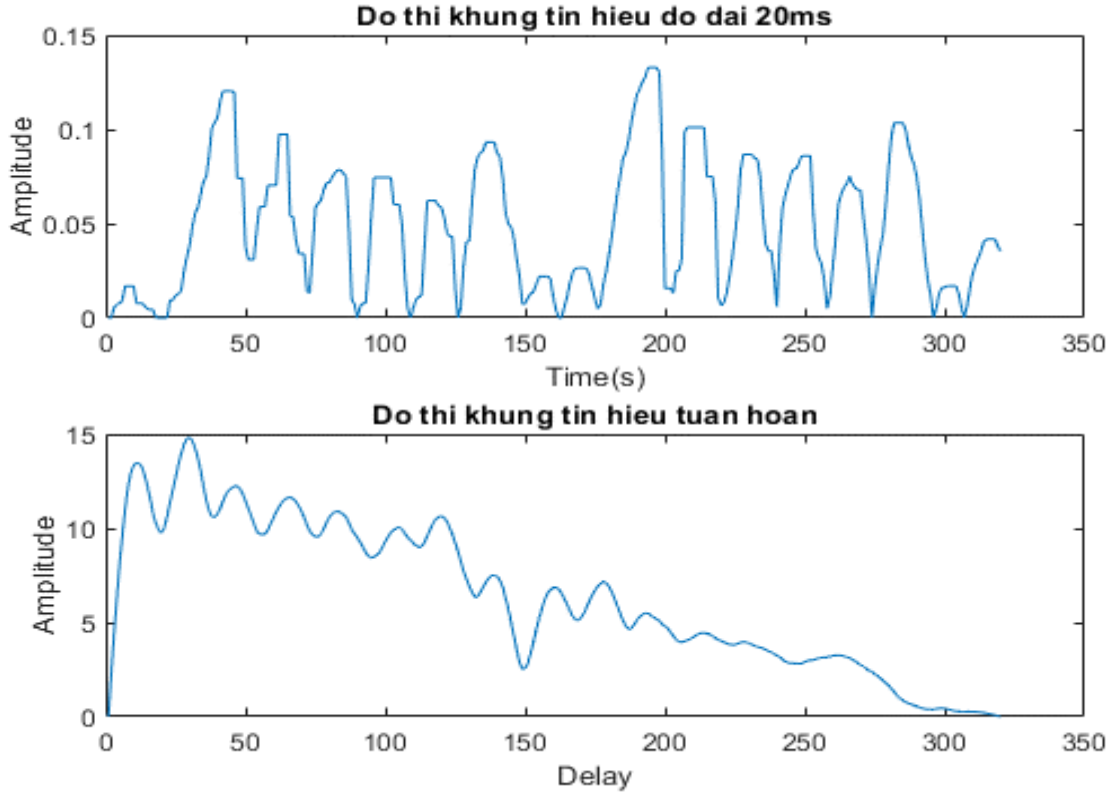


Hinh 21. Đồ thị đoạn tín hiệu âm thanh không tuần hoàn và hàm AMDF

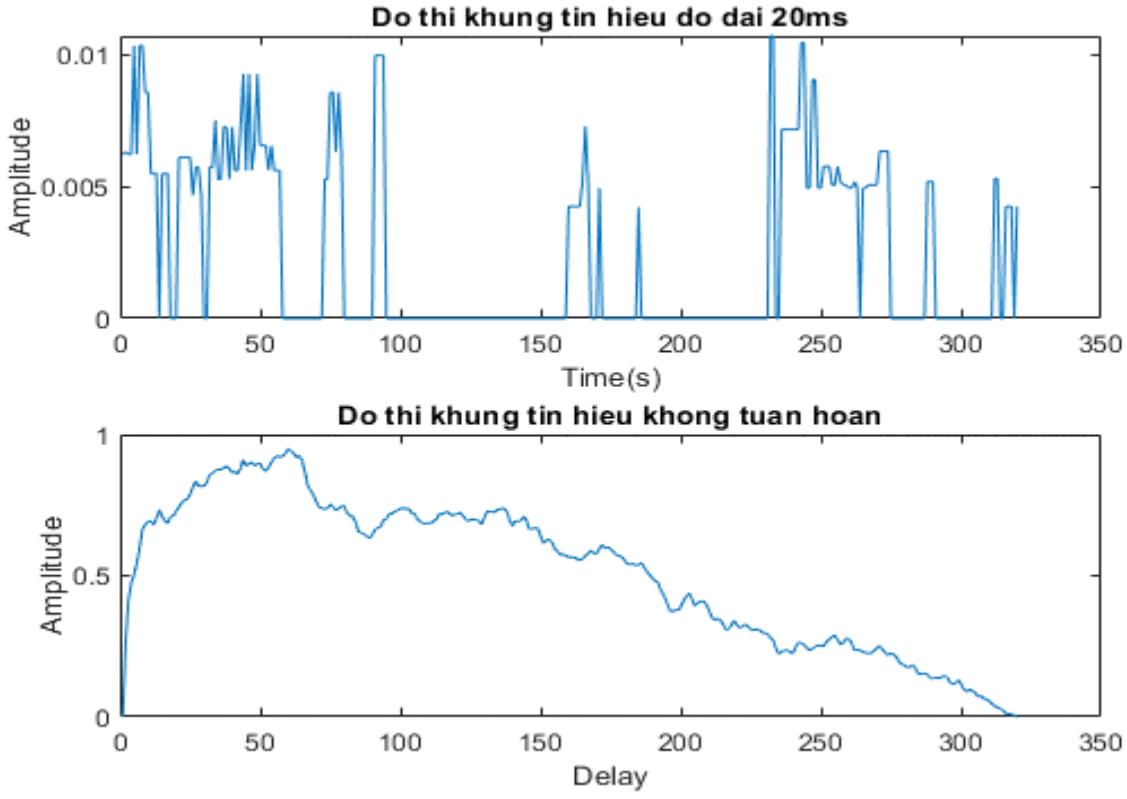


Hình 21 . Đồ thị phổ biên độ của tín hiệu và tần số cơ bản

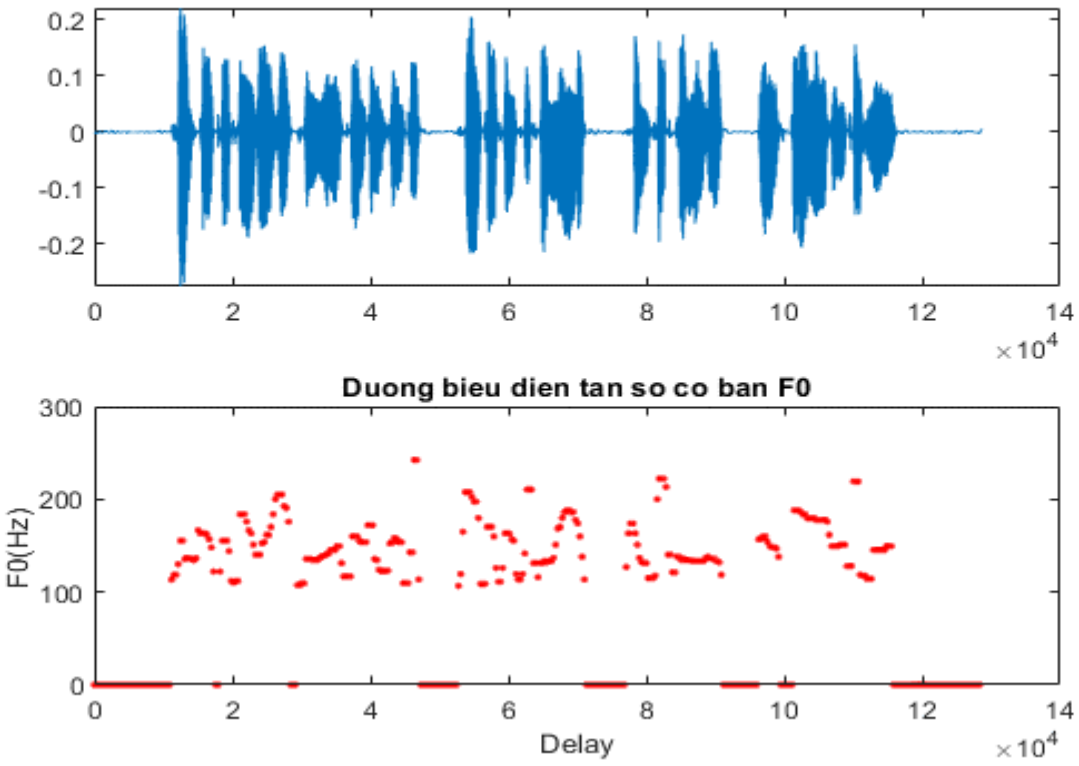
1.2 File studio_male.wav



Hinh 23. Đồ thị đoạn tín hiệu âm thanh tuần hoàn và hàm AMDF



Hinh 24. Đồ thị đoạn tín hiệu âm thanh không tuần hoàn và hàm AMDF



Hình 25. Đồ thị tín hiệu và biểu diễn tần số cơ bản F0 của tín hiệu

2. Kết quả định lượng

File	F0 Thuật toán(Hz)	F0 thủ công(Hz)	Độ chênh lệch(%)
Lab_male.wav	125.63	117.19	7.2
Lab_female.wav	149.28	202.60	26.32
Studio_male.wav	144.41	121.39	19.01
Studio_female.wav	134.06	230.67	41.88

3. Nhận xét

- Từ bảng số liệu cho thấy giá trị F0 thuật toán tính được sai số rất lớn, điều này do thuật toán tìm đỉnh vẫn chưa tối ưu, nhiều khung không lấy được F0 hoặc xác định sai tần số F0.
- Hầu hết các file sai số từ 25% so với F0 được xác định thủ công. Sai số lớn nhất là ở file studio_female.wav. Do tín hiệu khi chia khung có nhiều chu kỳ trên mỗi khung, vì vậy thuật toán xác định điểm cực đại đã xảy ra lỗi gây nên sai số lớn này.

D. Kết quả và nhận xét các thuật toán

1. Kết quả của các thuật toán

File	F0 thủ công(Hz)	F0 Thuật toán(Hz) Tự tương quan	Độ chênh lệch (Hz)	F0 Thuật toán(Hz)-FFT	Độ chênh lệch (Hz)	F0 Thuật toán(Hz)-AMDF	Độ chênh lệch (Hz)
Lab_male.wav	117.19	122.35	4.4	123.51	5.39	125.63	7.2
Lab_female.wav	202.596	224.401	10.76	236.16	16.57	149.28	26.32
Studio_male.wav	121.39	127.939	5.4	153.16	26.17	144.41	19.01
Studio_female.wav	230.67	240.907	4.44	255.93	10.95	134.06	41.88

2. So sánh và nhận xét giữa các thuật toán

- Hầu hết giá trị F0 tính được từ thuật toán tự tương quan cho kết quả tốt nhất. Sai số nằm trong khoảng từ 5-10%. Tuy nhiên kết quả này có thể chấp nhận được.
- Thuật toán AMDF cho độ chính xác F0 rất thấp. Sai số từ 30-40%. Vì vậy kết quả này không thể tin tưởng được.
- Thuật toán FFT cho kết quả trung bình. Tuy nhiên vẫn sai số tương đối lớn từ 5-25%. Kết quả này vẫn chưa thể tin tưởng được.
- Từ các nhận xét trên cho thấy thuật toán tư tương quan cho kết quả tốt và tối ưu nhất Với giá trị tính được có thể tin tưởng để thực hiện những thuật toán khác trong việc xử lý tín hiệu tiếng nói.

V. KẾT LUẬN

A. Kết quả đạt được

Báo cáo này thực hiện việc xác định tần số cơ bản F0. Được thực hiện dựa vào ba thuật toán hàm tự tương quan, hàm vi sai biên độ trung bình và hàm biến đổi fast fourier transform.

Từ kết quả qua quá trình thực nghiệm trên 4 file tín hiệu cho trước, đã cho thấy thuật toán hàm tự tương quan hoạt động khá hiệu quả so với hai thuật totoasnMDF và FFT. Trong phạm vi nghiên cứu, kết quả của hai thuật toán AMDF và FFT không thể tin tưởng để thực hiện các thuật toán xử lý tiếng nói. Bên cạnh đó, việc thuật toán hoạt động có ổn định hay không vẫn phụ thuộc vào môi trường thu âm, đối với môi trường có quá nhiều tạp âm thì kết quả vẫn còn sai số, ảnh hưởng đến kết quả của thuật toán.

B. Phương hướng phát triển

Trong tương lai, để xử lý các vấn đề xảy ra đối với thuật toán này, chúng tôi sẽ tìm hiểu thêm về các thuật toán xác định tần số F0 khác, cải thiện lại các thuật toán AMDF và FFT để cho kết quả tốt hơn và cải thiện hiệu suất của thuật thuật toán. Cuối cùng, những kết quả đạt được này sẽ ứng dụng trong việc phân tích và xử lý nhận dạng giọng nói.

VI. TÀI LIỆU THAM KHẢO

[1] Speech segementation, from https://en.wikipedia.org/wiki/Speech_segmentation
[2] Tần số âm cơ bả from: https://vi.wikipedia.org/wiki/Tần_số_âm_cơ_bản
[3] John G. Proakis and Dimitris G. Manolakis, Digital Signal Processing: Principles, Algorithms & Applications, Prentice Hall, 1995.
[4] Lawrence R. Rabiner and Ronal W.Schafer , Digital Processing Of Speech Signals, Prentice Hall, no.4 pp.141-142. 1978
[5] Alain de Cheveigne, Hideki Kawahara, "YIN, a fundamental frequency estimator for speech and music", Journal of the Acoustical Society of America, vol. 111, no. 4, pp. 1917-1930, 2002.
[6] Discrete Fourier Transform: https://vi.wikipedia.org/wiki/Biến_đổi_Fourier_nhanh
[7] The International Arab Journal of Information Technology, Vol. 8, No. 2, April 2011.
[8] CS425 Audio and Speech Processing_Hodgkinson, no.3 pp.57-58, 2012.