TÌM TẦN SỐ CƠ BẢN CỦA TÍN HIỆU TRÊN MIỀN THỜI GIAN DÙNG HÀM TỰ TƯƠNG QUAN VÀ TRÊN MIỀN TẦN SỐ DÙNG PHÉP BIẾN ĐỔI FOURIER NHANH, KẾT HỢP LỌC TRUNG VỊ.

Nguyễn Hữu Hoàng Hưng, Nguyễn Trần Hậu, Đặng Xuân Lộc, Nguyễn Thái Minh

Nhóm 2, lớp HP: 1022470.1810.16.15

Điểm	Bảng phân chia nhiệm vụ		Chữ ký sinh viên
	Nguyễn Trần Hậu	Đọc tài liệu, cài đặt thuật toán, viết báo cáo và thuyết trình về tính tần số cơ bản trên miền thời gian dùng hàm tự tương quan.	
	Đặng Xuân Lộc	Đọc tài liệu, cài đặt thuật toán, viết báo cáo và thuyết trình về lọc trung vị, tổng hợp slide.	
	Nguyễn Thái Minh	Đọc tài liệu hàm cửa sổ và khảo sát số điểm fft, tổng hợp và viết báo cáo hoàn chỉnh.	
	Nguyễn Hữu Hoàng Hưng (nhóm trưởng)	Phân công nhiệm vụ và đảm bảo tiến độ của mỗi thành viên. Đọc tài liệu, cài đặt thuật toán, viết báo cáo và thuyết trình về tính tần số cơ bản trên miền tần số. Viết báo cáo kết quả thực nghiệm.	

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 ghi rõ trong phần VII. 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— Tìm tần số cơ bản của tín hiệu là bài toán cần thiết trong xử lý tín hiệu âm thanh, đặc biệt là tín hiệu tiếng nói. Bài thực hành này thực hiện việc xác định tần số cơ bản của tín hiệu tiếng nói trên miền thời gian dùng hàm tự tương quan và trên miền tần số dùng phép biến đổi Fourier nhanh. Để làm trơn kết quả tần số cơ bản tìm được, lọc trung vị được sử dụng. Các thử nghiệm với tín hiệu của 5 nguyên âm (/a/, /e/, /i/, /o/ và /u/) cho thấy sai số trung bình của thuật toán hàm tự tương quan và phép biến đổi Fourier lần lượt là 8,1365Hz và 5,9316Hz. Kết quả thực nghiệm cũng cho thấy ta 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, biến đổi Fourier nhanh, lọc trung vị.

MỤC LỤC

I. ĐẶT VẤN ĐỀ

Xử lý tiếng nói từ khi xuất hiện đã có một vai trò rất quan trọng trong cuộc sống của chúng ta. Cùng với sự phát triển của khoa học kỹ thuật, nhu cầu xử lý tiếng nói của con người ngày càng tăng cao. Xử lý tiếng nói có ứng dụng về nhiều mặt, về cơ bản có ứng dụng như nhận dạng tiếng nói, người nói, tăng chất lượng giọng nói và tổng hợp tiếng nói. Để làm được điều đó, việc xác định tần số cơ bản là rất quan trọng. Có nhiều phương pháp khác nhau như AMDF, LPC, xử lý đồng hình, tự tương quan, phép biến đổi Fourier nhanh, … để xác định được tần số cơ bản.

Tần số cơ bản (còn gọi là F_0 hoặc cao độ) của một tín hiệu tuần hoàn bằng nghịch đảo của chu kỳ cơ bản của tín hiệu đó. Chu kỳ cơ bản là khoảng thời gian nhỏ nhất mà tín hiệu tuần hoàn trên miền thời gian. Tần số cơ bản mang thông tin có ý nghĩa vật lý đặc trưng cho tín hiệu tuần hoàn nên việc xác định nó là rất quan trọng trong xử lý tín hiệu số nói chung và tín hiệu giọng nói nói riêng. Trong bài báo cáo này, hàm tự tương quan được sử dụng để tính tần số cơ bản trên miền thời gian và sử dụng phép biến đổi Fourier nhanh (FFT) để phân tích phổ và tính tần số cơ bản trên miền tần số. Lọc trung vị (median smoothing) được sử dụng để làm trơn kết quả F_0 thu được.

Bài báo cáo có bố cục như sau. Phần II trình bày về cơ sở lý thuyết của các thuật toán và vấn đề liên quan đến việc tính tần số cơ bản trên miền tần số, trên miền thời gian, lọc trung vị và hàm cửa sổ. Phần III trình bày mã nguồn cài đặt các thuật toán. Phần IV trình bày kết quả thực nghiệm mô tả dữ liệu dùng để đánh giá độ chính xác của thuật toán, đưa ra các đánh giá định tính và định lượng. Phần V trình bày kết luận.

II. CƠ SỞ LÝ THUYẾT XỬ LÝ TÍN HIỆU TIẾNG NÓI VÀ CÁC THUẬT TOÁN TÍNH TẦN SỐ CƠ BẢN

A. Vấn đề cần giải quyết

Xử lý tín hiệu số đang bùng nổ nhanh chóng trong ngành công nghiệp điện tử và viễn thông hiện nay bởi nhiều lợi thế hơn so với xử lí tín hiệu liên tục. Xử lý tín hiệu số có nhiều ứng dụng đa dạng, đặc biệt trong việc xử lí tiếng nói hay xử lí âm thanh. Trong quá trình xử lí tín hiệu tiếng nói, việc xác định tần số cơ bản của tín hiệu là vấn đề cần thiết trong bất cứ bài toán nào. Vậy tần số cơ bản là gì? Tần số cơ bản của một tín hiệu tiếng nói chính là tốc độ rung của dây thanh [1], được tính bằng nghịch 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 [2].

Trong xử lí tín hiệu số, có rất nhiều phương pháp để xác định tần số cơ bản của tín hiệu tiếng nói. Trong bài báo cáo này, chúng ta sẽ xác định tần số cơ bản trên miền thời gian sử dụng hàm tự tương quan và trên miền tần số thông qua phép biến đổi Fourier nhanh. Để làm trơn các kết quả tần số cơ bản nhận được, lọc trung vị được sử dụng.

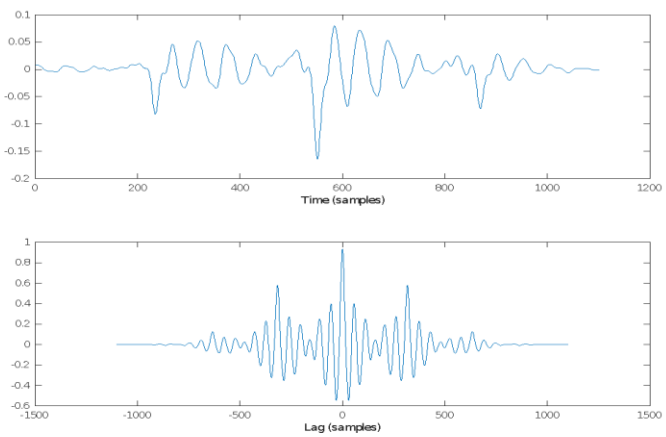
B. Hàm tự tương quan

1. Cơ sở lý thuyết

Hàm tự tương quan là công cụ được sử dụng phổ biến để xác định chu kỳ cơ bản của tín hiệu tiếng nói (có thể lẫn nhiễu) và nó cũng là cơ sở cho nhiều phương pháp phân tích phổ khác. Với tín hiệu tuần hoàn, ta có định nghĩa hàm tự tương quan:

$$r_{xx}(l) = \frac{1}{N}\sum_{n=0}^{N-1}(x(n)x(n-l))$$

Với N là độ rộng của cửa sổ, l là độ trễ được tính tại thời điểm n.



Hình 1. Tín hiệu và hàm tự tương quan của nó.

Hàm tự tương quan là hàm chẵn, đạt giá trị cực đại tại l = 0. Đại lượng $r_{xx}(0)$ chính bằng năng lượng của tín hiệu. Tầm quan trọng hàm tự tương quan nằm ở việc hàm sẽ đạt các giá trị cực đại tương ứng tại các điểm là bội của

chu kỳ cơ bản của tín hiệu. Khi đó các tần số cơ bản là tần số xuất hiện của các cực đại đó. Tính chất này khiến hàm tự tương quan trở thành cơ sở cho việc tính toán chu kỳ của tất cả các loại tín hiệu, bao gồm cả tín hiệu tiếng nói.

2. Sơ đồ khối

Sơ đồ khối thuật toán hàm tự tương quan được trình bày trên Hình 2.

Hình 2. Sơ đồ khối hàm tự tương quan.

3. Các tham số quan trọng của thuật toán

Công thức tính tần số cơ bản của tín hiệu: $F_0 = F_s / L$, trong đó F_0 là tần số cơ bản, F_s là tần số lấy mẫu và L là độ trễ mà tại đó hàm tự tương quan đạt cực đại.

Vì tần số lấy mẫu F_s đã biết, nên để tính tần số cơ bản, ta phải đi tìm độ trễ L.

Vì tín hiệu tiếng nói gần như tuần hoàn, nên hàm tự tương quan sẽ cho ra các cực đại tại những thời điểm là bội số của chu kỳ tín hiệu[2]. Từ đồ thị hàm tự tương quan, ta sẽ xác định các cực đại và độ trễ tương ứng tại các cực đại đó, từ độ trễ ta sẽ xác định được tần số cơ bản của tín hiệu.

Do tín hiệu ban đầu được chia nhỏ thành từng khung qua phép lấy cửa sổ, đồng thời do tín hiệu tiếng nói chưa tuần hoàn hoàn hảo, nên ứng với mỗi khung tín hiệu sẽ có 1 giá trị F_0, các giá trị này xấp xỉ nhau, bằng việc chia nhỏ tín hiệu và sử dụng hàm tự tương quan lên mỗi khung tín hiệu, kết quả thu được là mảng 1 chiều có số phần tử bằng số khung tín hiệu, mỗi phần tử là 1 tần số cơ bản tương ứng với 1 khung tín hiệu.

4. Vấn đề và giải pháp khắc phục

a) Phép lấy cửa sổ

Do đặc tính của mỗi hàm cửa sổ, nên trong quá trình thực hiện việc lấy cửa sổ sẽ phát sinh hiện tượng gọi là rò phổ. Để xử lí vấn đề này, ta sẽ sử dụng cửa sổ Hamming cho phép lấy cửa sổ nhằm hạn chế tối đa khả năng rò phổ so với cửa sổ chữ nhật (rectangular) sẽ được phân tích ở Phần E.

b) Số lượng cực đại của khung tín hiệu

Vì khung tín hiệu có chiều dài hữu hạn nên năng lượng cũng hữu hạn, điều này dẫn đến biên độ các đỉnh trong đồ thị tương quan sẽ giảm dần về 2 bên, với trục đối xứng là điểm chính giữa đồ thị. Điều này khiến cho việc xác định cực đại sẽ khó khăn, do đó ta chỉ lấy 3 cực đại có biên độ lớn nhất, qua đó việc tính toán tần số cơ bản sẽ bớt phức tạp hơn.

C. Phép biến đổi Fourier

1. Cơ sở lý thuyết

a) Phép biến đổi Fourier rời rạc (Discrete Fourier Transform - DFT)

Trong toán học, phép biến đổi Fourier rời rạc, đôi khi còn được gọi là biến đổi Fourier hữu hạn, là một phép biến đổi trong giải tích Fourier cho các tín hiệu thời gian rời rạc. Đầu vào của biến đổi này là một chuỗi hữu hạn các số phức hoặc các số thực. Đặc biệt, biến đổi này được sử dụng rộng rãi trong xử lý tín hiệu và các ngành liên quan đến phân tích tần số của một tín hiệu. Biến đổi này được tính nhanh bởi thuật toán biến đổi Fourier nhanh (FFT)[3].

Biến đổi Fourier rời rạc cho phép tính phiên bản tần số rời rạc của biến đổi Fourier của tín hiệu rời rạc (Discrete-time Fourier Transform – DTFT).

Công thức DFT N-điểm

$$X[k] = X(e^{j\omega}) = \sum_{n=0}^{N-1} x(n)e^{-\frac{jk2\pi n}{N}} \text{ (với } \omega = \frac{k2\pi}{N} \text{ và } 0 \le k \le N-1) \text{ (*)}$$

X[k] đại diện cho biên độ và pha ở các bước sóng khác nhau của tín hiệu vào x[n]. Phép biến đổi DFT tính các giá trị X[k] từ các giá trị x[n].

Một số tính chất của phép biến đổi Fourier rời rạc bao gồm tính tuần hoàn, tuyến tính và tính chập.

X[k] tuần hoàn với chu kỳ N, nghĩa là X[k+N] = X[k] $\forall k$ [10].

Tính chất tuyến tính $\begin{cases} x1(n) \overset{DFT}{\longleftrightarrow} X1[k] \\ x2(n) \overset{DFT}{\longleftrightarrow} X2[k] \end{cases}$ => $a_1x_1(n) + a_2x_2(n) \overset{DFT}{\longleftrightarrow} a_1X_1[k] + a_2X_2[k]$ [10].

Tính chập $\begin{cases} x1(n) \overset{DFT}{\longleftrightarrow} X1[k] \\ x2(n) \overset{DFT}{\longleftrightarrow} X2[k] \end{cases}$ => $x(n) = x_1(n) \otimes x_2(n) \overset{DFT}{\longleftrightarrow} X[k] = X_1[k].X_2[k]$ [10].

b) Phép biến đổi Fourier nhanh (Fast Fourier Transform – FFT)

Một biến đổi Fourier nhanh (FFT) là một thuật toán hiệu quả để tính biến đổi Fourier rời rạc (DFT) và biến đổi ngược. Có nhiều thuật toán FFT khác nhau sử dụng kiến thức từ nhiều mảng khác nhau của toán học [4].

Thuật toán FFT tính nhanh DFT N-điểm của tín hiệu rời rạc x[n] với $N = 2^m \ge \text{length}(x[n])$.

Vì thuật toán FFT chỉ áp dụng cho trường hợp $N = 2^m$ nên tổng (*) có thể phân tích thành hai tổng như sau:

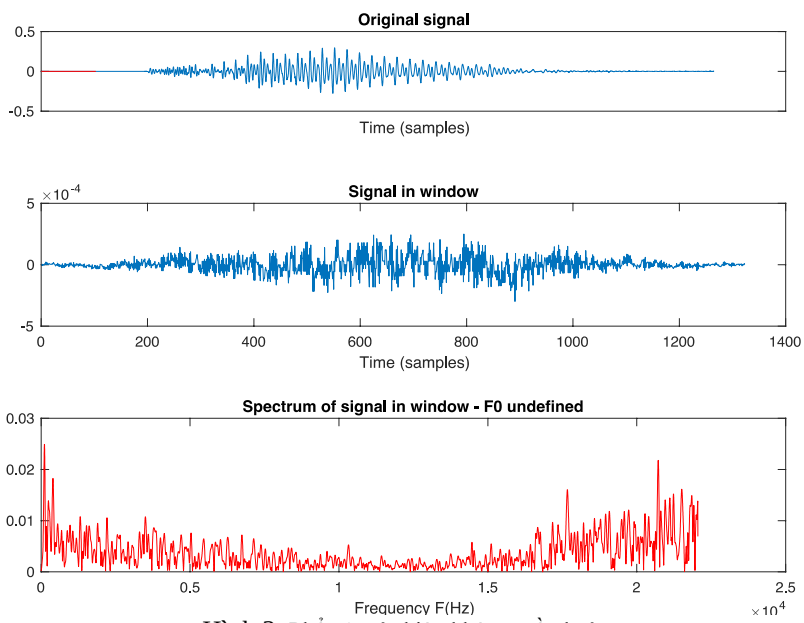
$$X[k] = \sum_{n\ lẻ} x(n)e^{-\frac{jk2\pi n}{N}} + \sum_{n\ chẵn} x(n)e^{-\frac{jk2\pi n}{N}} = \sum_{m=0}^{\frac{N}{2}-1} x(2m)e^{-\frac{jk4\pi m}{N}} + \sum_{m=0}^{\frac{N}{2}-1} x(2m+1)e^{-\frac{jk4\pi m}{N}} e^{-\frac{jk2\pi}{N}} = X_e[k] + e^{-\frac{jk2\pi}{N}} X_o[k]$$

$X_e[k]$ và $X_o[k]$ lần lượt là biến đổi Fourier của hai dãy {x[2m] | m = 0, 1, 2, …, N/2 -1 } và {x[2m+1] | m = 0, 1, 2, …, N/2 -1 }. Có nghĩa là mỗi một $X_e[k]$ và $X_o[k]$ được phân tích thành tổng của hai phép biến đổi Fourier rời rạc của N/2 điểm. Tiếp tục quá trình trên cho đến khi ta được phép biến đổi Fourier của 2 điểm. Ngoài ra, do tính tuần hoàn của chu kỳ N/2 nên chỉ cần tính $X_e[k]$ và $X_o[k]$ với $N/2 \le k \le N-1$ và $0 \le k \le N/2 - 1$.

Bằng thuật toán FFT, cần $\frac{N}{2}\log_2 N$ phép nhân phức thay cho $(N-1)^2$ phép nhân phức và $N\log_2 N$ phép cộng phức thay vì $N(N-1)$. Do đó, tính trực tiếp từ định nghĩa DFT (*) đòi hỏi $O(N^2)$ phép tính, FFT sẽ giúp tính cùng kết quả đó trong O(N*logN) phép tính [4].

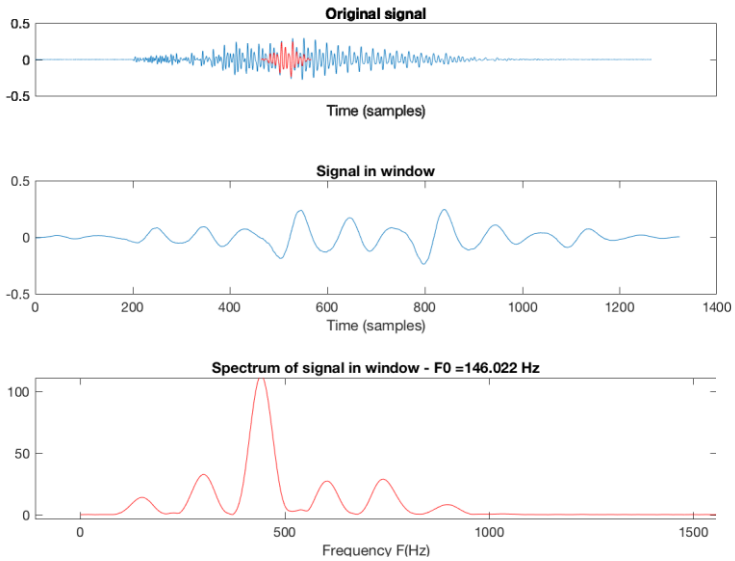
Trong bài báo cáo này, sử dụng hàm fft() trong thư viện Matlab để phân tích.

Dưới đây là hình ảnh của phổ thu được trên tín hiệu nguyên âm /u/ bằng hàm FFT 32768 điểm



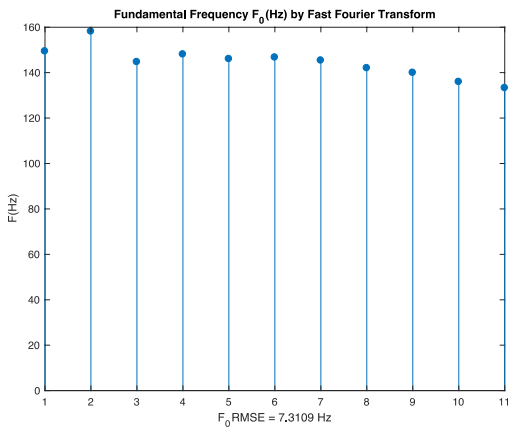
Hình 3. Phổ của tín hiệu không tuần hoàn.

Hình 3 mô tả phổ của tín hiệu không tuần hoàn (tín hiệu nhiễu) trong file thu âm của nguyên âm /u/.



Hình 4. Phổ của tín hiệu tuần hoàn.

Hình 4 mô tả phổ của tín hiệu tuần hoàn (tín hiệu âm thanh) trong file thu âm của nguyên âm /u/.

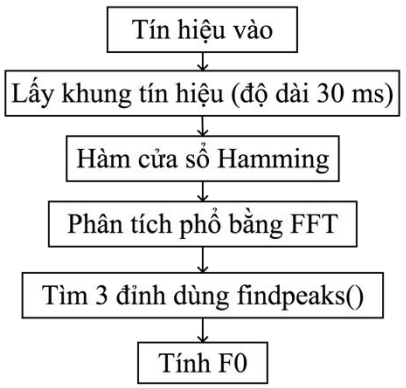


Hình 5. Kết quả tính tần số cơ bản của tín hiệu nguyên âm /u/ trên miền tần số.

Hình 5 mô tả các kết quả F0 tính được trong file thu âm của nguyên âm /u/ thông qua mỗi cửa sổ.

2. Sơ đồ khối

Sơ đồ khối thuật toán tìm tần số cơ bản F_0 trên miền tần số được trình bày trên Hình 6.



Hình 6. Sơ đồ khối tìm tần số cơ bản F_0 trên miền tần số.

3. Các tham số quan trọng của thuật toán

Hàm fft() trong thư viện Matlab:

Cú pháp Y = fft(X,N) với X là tín hiệu vào (trên miền thời gian rời rạc) và N là số điểm trong phép biến đổi DFT-N điểm.

Nếu length(X) > N thì X được thêm vào các giá trị 0 cho tới khi length(X) = N để thực hiện phép biến đổi.

Nếu length(X) < N thì X được cắt ngắn sao cho length(X) = N.

Kết quả trả về (Y) là mảng chứa các số phức biểu diễn phổ (spectrum) của x[n].

4. Vấn đề và giải pháp khắc phục

Một vấn đề phát sinh rất phổ biến đó là khó khăn trong việc xác định được các khung lựa chọn là khung của âm thanh hay khung của nhiễu. Giải pháp đề ra là dựa trên biên độ của tín hiệu ta có thể phân biệt được đâu là nhiễu (tín hiệu không tuần hoàn) và đâu là âm thanh (tín hiệu tuần hoàn). Như trên hình 3 và hình 4, ta có thể thấy rằng đối với tín hiệu nhiễu thì biên độ thường thấp hơn nhiều so với tín hiệu âm thanh nên ta có thể nhận biết bằng cách giới hạn biên độ.

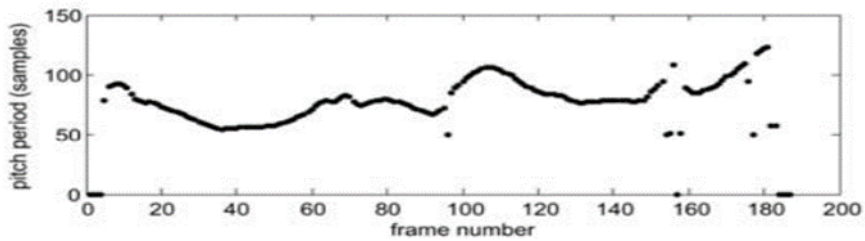
Trong việc tìm phổ của tín hiệu liên tục cũng như tín hiệu rời rạc, tất cả các giá trị của tín hiệu trên miền thời gian đều rất cần thiết. Tuy nhiên, trên thực tế, chúng ta chỉ có thể tính toán và quan sát trong khoảng thời gian giới hạn. Do vậy, phổ tín hiệu chỉ có thể được tính xấp xỉ từ lượng dữ liệu giới hạn thông qua cửa sổ. Từ đó, ta sẽ bị hạn chế khả năng phân biệt hai thành phần tần số được tách ra nhỏ hơn tần số cơ bản. Phổ trong cửa sổ sẽ trải dài ra khắp miền tần số nên năng lượng của tín hiệu gốc tập trung tại một tần số sẽ bị trải ra khắp miền tần số bởi hàm cửa sổ. Hiện tượng này được gọi là rò phổ, đây là hiện tượng rất phổ biến khi sử dụng hàm cửa sổ [6]. Giải pháp được đề ra để giảm rò phổ là nên lựa chọn những hàm cửa sổ có "sidelobes" nhỏ hơn hàm cửa sổ chữ nhật trên miền tần số. Trong bài báo cáo này, hàm cửa sổ Hamming, được trình bày ở phần E, được sử dụng.

Vì có nhiều cửa sổ khác nhau nên ta có nhiều kết quả F_0 khác nhau trên cùng một tín hiệu. Các kết quả F_0 hầu hết đều xấp xỉ nhau. Tuy nhiên, trên thực tế, do tín hiệu âm thanh của chúng ta không tuần hoàn hoàn hảo nên sẽ có một số khung thu được kết quả F_0 chênh lệch khá nhiều so với các kết quả còn lại. Để giải quyết vấn đề này, lọc trung vị, được trình bày ở phần tiếp theo, được sử dụng để làm mịn các kết quả tần số cơ bản trên tín hiệu.

D. Lọc trung vị

1. Cơ sở lý thuyết

Trong hầu hết trường hợp, một bộ lọc tuyến tính nói chung là loại bỏ các thành phần tạp âm ra khỏi tín hiệu. Tuy nhiên, trong một số trường hợp thì các bộ lọc tuyến tính không hoàn toàn phù hợp bởi loại tín hiệu cần được làm mịn. Một ví dụ là đường viền ở hình dưới đây, có những thành phần lỗi rõ ràng và phải được đưa trở lại phù hợp với phần còn lại của dữ liệu. Một bộ lọc tuyến tính lowpass không chỉ thất bại trong việc mang các điểm sai lệch trở lại dòng mà còn bóp méo đường viền tại quá trình chuyển đổi giữa giọng nói hữu thanh và vô thanh (giai đoạn 0). [7]



Hình 7. Ví dụ về một tín hiệu trước khi dùng lọc trung vị.[11]

Đối với các trường hợp như vậy, một số thuật toán lọc tuyến tính có thể duy trì các tín hiệu gián đoạn nhưng tính chất lớn lọc ra không tồn tại, một bộ lọc phi tuyến sử dụng sự kết hợp giữa trung vị chạy (running median) và lọc tuyến tính được đưa ra để thỏa mãn tính chất mong muốn. [7]

Một tín hiệu giọng nói x(n) sẽ có công thức: x(n) = S[x(n)] + R[x(n)].

Với S(x) là phần lọc (phần mịn), R(x) là phần thô (tạp âm) của tín hiệu x.

Một phép biến đổi phi tuyến có khả năng tách S[x(n)] ra khỏi R[x(n)] chính là trung vị chạy của tín hiệu x(n).

Đầu ra của trung vị chạy lọc, ML[x(n)] đơn giản là trung vị của L số, x(n), x(n-1), …, x(n-L+1).

Biến đổi trung vị với độ dài L tuân theo các tính chất mong muốn của bộ lọc: ML(ax[n]) = a. ML(x[n]). Các trung vị không bị nhòe gián đoạn (smeared out) nếu không có sự gián đoạn trong các mẫu $\frac{L}{2}$.

$$ML(ax1[n] + bx2[n]) \neq a.ML(x1[n]) + b.ML(x2[n]).$$

Các bộ lọc trung vị thường bảo toàn các gián đoạn sắc nét của tín hiệu, nhưng không đảm bảo việc lọc các thành phần tiếng ồn (tạp âm) một cách đầy đủ.ML(ax[n]) = a.ML(x[n]).

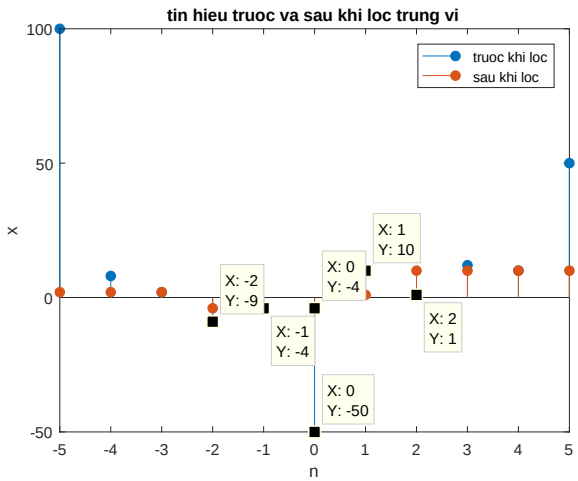
Số trung vị là số nằm chính giữa trong một tập hợp dãy số (n phần tử) đã được sắp xếp theo thứ tự tăng dần. [7]

Nếu số phần tử của dãy số là lẻ (n = 2k+1 phần tử), thì số trung vị sẽ là số ở vị trí thứ n + 1. Ví dụ trong dãy số 1, 2, 6, 7, 8, 16, 18 thì số trung vị sẽ là 7. [7]

Nếu số phần tử của dãy là số chẵn (n = 2k phần tử), thì số trung vị sẽ là trung bình cộng của phần tử thứ n và thứ n+1. Ví dụ dãy số 1, 3, 4, 6, 7, 9, 15, 16 thì số trung vị sẽ là trung bình cộng của 6 và 7 (bằng 6.5). [7]

2. Các tham số quan trọng của lọc trung vị (Median Smoothing)

Trong matlab, để lọc trung vị của tín hiệu cần lọc, ta dùng hàm MedSmoothing (Hàm tự viết) để lọc trung vị các tín hiệu một chiều. Cú pháp của hàm là MedSmoothing(x,N). Với x là tín hiệu đầu vào sau khi tính F_0 bằng hàm tự tương quan (đã được rời rạc hóa) hoặc trên miền tần số, N là số chiều (bậc) của tín hiệu hay chiều dài hàm cửa sổ. Để lấy chính xác điểm trung vị, N thường là số lẻ.



Hình 8. Tín hiệu trước và sau khi đi qua lọc trung vị n = 0.

Xét tín hiệu ban đầu tại n = 0, giá trị của x sẽ là -50. Sau khi lọc tín hiệu x với bậc là 5 (xét các giá trị n = -2, -1, 0, 1, 2 với các giá trị là -9 , -4, -50, 10, 1) thì giá trị của x tại n = 0 là -4. Như vậy tín hiệu điểm bất thường tại n = 0 đã được loại bỏ.và thay vào đó là điểm thấp hơn rất nhiều.

Từ ví dụ trên, có thể thấy rằng tác dụng của lọc trung vị là lọc những điểm có biên độ bất thường của tín hiệu, hay nói cách khác là làm mịn tín hiệu. Bậc N của bộ lọc cần phải được xem xét để tín hiệu sau khi lọc không được còn những điểm bất thường hoặc quá trơn.

3. Khảo sát lọc trung vị trên các kết quả thực nghiệm của tần số cơ bản

Xét các kết quả tần số cơ bản của từng nguyên âm cơ bản /a/, /e/, /i/, /o/ và /u/ trước khi lọc (sau khi được tính bằng hàm tự tương quan) và sau khi lọc trung vị với các bậc khác nhau.

Bảng 1. Bảng số liệu so sánh F_0 (được tính trên miền thời gian) của các nguyên âm trước và sau khi lọc trung vị.

Tín hiệu	Trước khi lọc	Lọc với N = 3	Lọc với N = 5	Lọc với N = 7
/a/	F_0 mean = 137,0845 Hz F_0 RMSE = 6,1495 Hz	F_0 mean = 137,4475 Hz F_0 RMSE = 5,3810 Hz	F_0 mean = 136,4816 Hz F_0 RMSE = 5,4512 Hz	F_0 mean = 136,0219 Hz F_0 RMSE = 5,0767 Hz
/e/	F_0 mean = 139,2630 Hz F_0 RMSE = 9,9666 Hz	F_0 mean = 139,2078 Hz F_0 RMSE =9,9397 Hz	F_0 mean = 138,4337 Hz F_0 RMSE = 9,6954 Hz	F_0 mean = 138,3254 Hz F_0 RMSE = 9,6934 Hz
/i/	F_0 mean = 135,8407 Hz F_0 RMSE = 9,2633 Hz	F_0 mean = 136,0345 Hz F_0 RMSE = 6,4899 Hz	F_0 mean = 136,1413 Hz F_0 RMSE = 6,2293 Hz	F_0 mean = 134,3279 Hz F_0 RMSE = 8,8427 Hz
/o/	F_0 mean = 137,4709 Hz F_0 RMSE = 7,6768 Hz	F_0 mean = 137,5110 Hz F_0 RMSE = 3,9514 Hz	F_0 mean = 136,3764 Hz F_0 RMSE = 5,3537 Hz	F_0 mean = 135,5654 Hz F_0 RMSE = 6,9525 Hz

/u/	F_0 mean = 138,0304 Hz F_0 RMSE = 15,7358 Hz	F_0 mean = 136,5642 Hz F_0 RMSE = 16,6882 Hz	F_0 mean = 137,8570 Hz F_0 RMSE = 15,7298 Hz	F_0 mean = 137,8562 Hz F_0 RMSE = 15,7298 Hz

Xét các kết quả tần số cơ bản của từng nguyên âm cơ bản /a/, /e/, /i/, /o/ và /u/ trước khi lọc (sau khi được tính trên miền tần số bằng phép biến đổi Fourier nhanh) và sau khi lọc trung vị với các bậc khác nhau.

Bảng 2. Bảng số liệu so sánh F_0 (được tính trên miền tần số) của các nguyên âm trước và sau khi lọc trung vị.

Tín hiệu	Trước khi lọc	Lọc với N = 3	Lọc với N = 5	Lọc với N = 7
/a/	F_0 mean = 136,6013 Hz F_0 RMSE = 7,7731 Hz	F_0 mean = 135,7841 Hz F_0 RMSE = 6,4626 Hz	F_0 mean = 135,4958 Hz F_0 RMSE = 6,1802 Hz	F_0 mean = 135,1112 Hz F_0 RMSE = 5,8212 Hz
/e/	F_0 mean = 140,6999 Hz F_0 RMSE = 7,4707 Hz	F_0 mean = 140,6999 Hz F_0 RMSE = 7,4707 Hz	F_0 mean = 140,2105 Hz F_0 RMSE = 7,3312 Hz	F_0 mean = 140,2105 Hz F_0 RMSE = 7,3312 Hz
/i/	F_0 mean = 144,6762 Hz F_0 RMSE = 19,6454 Hz	F_0 mean = 137,8723 Hz F_0 RMSE = 6,3922 Hz	F_0 mean = 136,6480 Hz F_0 RMSE = 5,7125 Hz	F_0 mean = 136,1526 Hz F_0 RMSE = 5,0896 Hz
/o/	F_0 mean = 137,6587 Hz F_0 RMSE = 6,3942 Hz	F_0 mean = 137,5626 Hz F_0 RMSE = 4,8207 Hz	F_0 mean = 136,6493 Hz F_0 RMSE = 5,2043 Hz	F_0 mean = 136,2167 Hz F_0 RMSE = 5,0958 Hz
/u/	F_0 mean = 144,4927 Hz F_0 RMSE = 7,3097 Hz	F_0 mean = 143.8198 F_0 RMSE = 6,5979 Hz	F_0 mean = 143,2692 Hz F_0 RMSE = 6,6419 Hz	F_0 mean = 142,9022 Hz F_0 RMSE = 6,6978 Hz

Nếu bậc N của lọc trung vị thấp (N = 3) thì tín hiệu vẫn không thay đổi nhiều, vẫn còn nhiều điểm bất thường ở các giá trị n. Tín hiệu đã được lọc mịn hơn tại bậc N = 5 và bậc N = 7. Với bậc N = 5 thì các tín hiệu vẫn còn một vài giá trị bất thường nhưng các giá trị n khác của tín hiệu cũng giữ được độ chính xác. Khi bậc N = 7 thì hầu như các giá trị bất thường không còn, nhưng có một vài giá trị n đã bị thay thế (mất) thông tin. Giá trị N càng lớn thì tín hiệu càng mịn, tuy nhiên tín hiệu cũng sẽ càng mất tín hiệu quan trọng. Vì vậy, giá trị N nên là 5 hoặc 7 để tín hiệu có thể vừa loại bỏ được những giá trị bất thường của tín hiệu, vừa giữ được các thông tin quan trọng.

Sai số toàn phương trung bình của các tín hiệu sau khi lọc cũng là tiêu chí quan trọng để xác định bậc nào nên được sử dụng để lọc. Bậc nào có lệch chuẩn càng thấp thì càng nên dùng bậc đó.

Với các nguyên âm đã được xét như trên , có thể rút ra được như sau:

a) Đối với tần số cơ bản được tính trên miền thời gian

Nên chọn N = 7 ứng với tín hiệu /a/

Nên chọn N = 7 ứng với tín hiệu /e/

Nên chọn N = 5 ứng với tín hiệu /i/

Nên chọn N = 3 ứng với tín hiệu /o/

Nên chọn N = 7 ứng với tín hiệu /u/

b) Đối với tần số cơ bản được tính trên miền tần số

Nên chọn N = 7 ứng với tín hiệu /a/

Nên chọn N = 7 ứng với tín hiệu /e/

Nên chọn N = 7 ứng với tín hiệu /i/

Nên chọn N = 3 ứng với tín hiệu /o/

Nên chọn N = 3 ứng với tín hiệu /u/

E. Khảo sát ảnh hưởng của việc dùng hàm cửa sổ và số điểm tính FFT

1. Vấn đề cần giải quyết

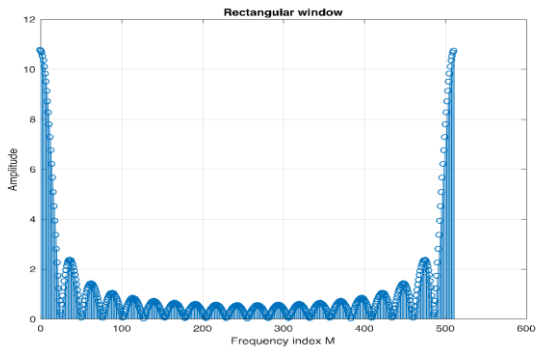
Các đặc điểm của tín hiệu tiếng nói luôn biến đổi theo thời gian. Chẳng hạn như cao độ (pitch), mức độ kích thích giữa âm vô thanh, âm hữu thanh và khoảng lặng, các biên độ khác nhau của tín hiệu, hay sự biến thiên tần số cơ

bản trong miền tín hiệu tuần hoàn. Chính vì các đặc điểm trên luôn luôn biến đổi nên giả định trong hầu hết các phương pháp xử lí tín hiệu tiếng nói là các thành phần trong tín hiệu tiếng nói thay đổi không đáng kể hoặc cố định so với thời gian. Để áp dụng giả định trên, trong lý thuyết phân tích, các phân tích phải được tiến hành trong một khoảng thời gian giới hạn. Vì vậy nên khi đi phân tích tín hiệu tiếng nói có khoảng thời gian tương đối dài, phải áp dụng các phương pháp xử lí ngắn hạn nhằm chia tín hiệu thành từng đoạn (segment) hay từng khung nhỏ (frame) trong một khoảng thời gian tương đối nhỏ (thường khoảng từ 10-30ms). Các đoạn này thường được lấy chồng lên nhau nhằm tránh việc mất mát tín hiệu.

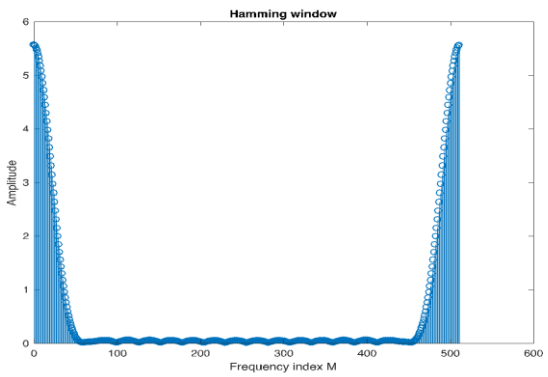
Trong hầu hết các phương pháp xử lí ngắn hạn, việc chia tín hiệu thành các đoạn hoặc khung được thực hiện bằng cách nhân tín hiệu tiếng nói với 1 hàm cửa sổ có chiều dài hữu hạn tương ứng với chiều dài của khung tín hiệu mong muốn.

Hàm cửa sổ trong xử lí tín hiệu số là một hàm toán học có giá trị 0 tại những điểm không thuộc khoảng giới hạn (chiều dài cửa sổ). Sau khi nhân tín hiệu với một hàm cửa sổ (phép lấy cửa sổ) ngoại trừ những vị trí thuộc chiều dài cửa sổ, tất cả các vị trí không thuộc chiều dài cửa sổ đều có giá trị bằng 0. Tuy nhiên, việc lấy cửa sổ tín hiệu không chỉ nhằm phân đoạn tín hiệu, mà mục đích chính của việc này là làm thon tín hiệu, nhằm cải thiện các đặc điểm của tín hiệu trên miền tần số[8].

Hàm cửa sổ có nhiều loại, phụ thuộc vào mục đích sử dụng khi phân tích tín hiệu. Có nhiều hàm cửa sổ được sử dụng trong việc phân tích tín hiệu như hàm cửa sổ chữ nhật (Rectangular window), hàm cửa sổ tam giác (Triangle window), hàm cửa sổ Hanning/ Hamming, ... Tuy nhiên, trong bài báo cáo này, chúng ta sẽ cùng tìm hiểu khảo sát ảnh hưởng dùng hàm cửa sổ chữ nhật hoặc cửa sổ Hamming/Hanning, và số điểm tính FFT khi phân tích phổ của tín hiệu hình sin ngẫu nhiên với FFT points = 256, 512, 1024.



Hình 9. Tín hiệu hình sin ngẫu nhiên khi nhân với mẫu cửa sổ hình chữ nhật qua phép phân tích FFT.



Hình 10. Tín hiệu hình sin ngẫu nhiên khi nhân với mẫu cửa sổ hamming qua phép phân tích FFT.

2. Cơ sở lý thuyết

a) Chức năng của hàm cửa sổ

Chức năng của hàm cửa sổ là hàm toán học có giá trị bằng không khi nằm ngoài khung cửa sổ được chọn. Về mặt toán học, khi một hàm khác dạng sóng hoặc chuỗi dữ liệu được "nhân" bởi hàm cửa sổ, sẽ có giá trị bằng nếu nằm ngoại khoảng thời gian được chọn của hàm cửa sổ.

Các chức năng khác của hàm cửa sổ được sử dụng trong phân tích quang phổ, các bộ lọc đáp ứng xung hữu hạn, cũng như thiết kế dầm và ăng-ten. [8]

b) Hàm cửa sổ hình chữ nhật

Cửa sổ hình chữ nhật (thỉnh thoảng được gọi là cửa sổ ô tô hoặc cửa sổ Dirichlet) là cửa sổ đơn giản nhất trong các loại cửa sổ, thay thế tất cả các giá trị N của chuỗi dữ liệu bằng 0 (với N là chiều dài mẫu của một hàm thời gian rời

rạc), làm cho nó xuất hiện như thể dạng sóng với w(n) =1. Một cửa sổ hình chữ nhật không thay đổi phân đoạn tín hiệu. Nó làm cho tín hiệu trong khung tín hiệu trong hàm cửa sổ giữ nguyên, bằng không khi nằm ngoài khung tín hiệu được chọn. [8]

Hàm cửa sổ chữ nhật : $h(n) = \begin{cases} 1 \ \ nếu\ 0 \le n \le N-1 \\ 0 \ \ trường\ hợp\ còn\ lại \end{cases}$

c) Hàm cửa sổ Hamming

Hàm cửa sổ Hamming là hàm cửa sổ đặc biệt được sử dụng trong việc phân tích phổ để giảm hiện tượng rò phổ. Hàm cửa sổ Hamming với mức độ của hiện tượng rò phổ thấp sẽ cần thiết để phát hiện tín hiệu có vùng quang phổ rộng.

Hàm cửa sổ Hamming: $h(n) = \begin{cases} 0{,}54 - 0{,}46cos(\frac{2\pi n}{N-1}) \ \ nếu\ 0 \le n \le N-1 \\ 0 \ \ trường\ hợp\ còn\ lại \end{cases}$

Do mức độ của hiện tượng rò phổ, hàm cửa sổ hình chữ nhật thường không được sử dụng trong tín hiệu có quang phổ phạm vi tín hiệu rộng. Còn hàm cửa sổ Hamming có mức độ của hiện tượng rò phổ thấp nên thường được sử dụng trong các tín hiệu có phạm vi quang phổ rộng. Nhưng trong tín hiệu có phạm vi quang phổ hẹp, hàm cửa sổ hình nhật sẽ tối ưu hơn so với hàm cửa sổ Hamming vì nó không làm mất một số tín hiệu quan trọng.

3. Các tham số quan trọng của thuật toán

Hàm cửa sổ hình chữ nhật trong thư viện Matlab có cú pháp w = rectwin(L) trả về một cửa sổ hình chữ nhật có chiều dài L trong vectơ cột w, với L là số điểm để tạo cửa sổ hình chữ nhật.

Tương tự, hàm cửa sổ Hamming trong thư viện Matlab cú pháp w = hamming (L) trả về một cửa sổ Hamming với L là số điểm đối xứng

III. CÀI ĐẶT THUẬT TOÁN

A. Hàm tự tương quan để tính tần số cơ bản trên miền thời gian

```
function [arr_F0] = AutoCorr(inp,fs)
% The function is used for finding fundamental frequency of a signal in the time domain
% inp is the input signal
% fs is sampling frequency
% The output is arr_F0, which is the array containing calculated F0
frame_len = round(0.03*fs);     % Length of each frame 30ms
half = round(frame_len/2);      % For overlapping frames
i = 1;                          % Index of number of elements in the array of F0
h = hamming(frame_len);         % Create a impulse response of the hamming window
% Examine every frame of the signal
for k = 1 : length(inp)/half -1;                    % (length(ip)/half -1) = number of frames
   width = (k-1)*half + 1:(frame_len + (k-1)*half);   % Start index to end index
   frame = h.*inp(width);                           % Multiplied by the hamming window
   % Use xcorr function to determine the lag of the frame
   [rxx , lag] = xcorr(frame, frame);               % Autocorrelation function on the frame
   rxx = rxx(frame_len:end);                        % Fold the signal
   lag = lag(frame_len:end);
   % Calculate the fundamental frequency F0
   if findpeaks(inp(width),'NPeaks',1,'SortStr','descend') > 0.04  % Remove noise by limiting estimate min
amplitude = 0.04
      [y_value,y_peak] = findpeaks(frame,'MinPeakDistance', 285); % Limit min distance between peaks to
increase accuracy
      lag_a = fs/(y_peak(3) - y_peak(2));     % Find 2 lags by using 3 peaks from the correlation graph
      lag_b = fs/(y_peak(2) - y_peak(1));
      if lag_a<400 && lag_a>80              % Min_f=80Hz and max_f=400Hz
         if lag_b<400 && lag_b>80
            arr_F0(i) = (lag_a+lag_b)/2;
            i = i + 1;
         end
      end
   end
end
```

B. Phép biến đổi Fourier nhanh để tính tần số cơ bản trên miền tần số

```
function [yyy_F0] = mientanso(sig,fs)
% The function is used for finding fundamental frequency of a signal in spectrum
% sig is the input signal
% fs is sampling frequency
% The output is yyy_F0, which is the array containing calculated F0 in each windown
N = 32768;                  %N-point fft
frame_len = round(0.03*fs);     %Length of frame (30ms)
half = round(frame_len/2);      %For overlapping frame
h = hamming(frame_len);         %Hamming window function
i=1;                        %Index of element in yyy_F0
for k = 1 : length(sig)/half -1 %Vong lap cac frame
   range = (k-1)*half + 1:(frame_len + (k-1)*half);    %Index of each element in window
   frame = h.*sig(range);                          %Value of each element in window
   %use FFT function (in Matlab) to analyze the spectrum of the frame
   P2 = abs(fft(frame,N));         %The two-sided spectrum P2
   P1 = P2(1:length(P2)/2+1);      %The single-sided spectrum P1
   P1(2:end-1) = 2*P1(2:end-1);
   freq=linspace(0,fs/2,length(P1));   %Spectrum of signal
   if findpeaks(frame,'NPeaks',1,'SortStr','descend') > 0.03   %Remove the noise by limiting the amplitude of
signal in frame
      [y_value,y_peak] = findpeaks(P1,freq,'MinPeakHeight',2);%Find peaks (with the certain minimun
Height to remove the noise in special cases) in the spectrum of signal
      z_a = y_peak(3) - y_peak(2);    %Find 2 F0 by using 3 first peaks in spectrum
      z_b = y_peak(2) - y_peak(1);
      if z_a<400 && z_a>80             %Since 80Hz<F0<400Hz, the results will be denied if one of them is
outside (80Hz,400Hz)
         if z_b<400 && z_b>80
            yyy_F0(i) = (z_a+z_b)/2;%Final result is the mean of 2 F0
            i=i+1;               %Put the result into the yyy_F0 and increase the index.
         end
      end
   end
end
```

C. Lọc trung vị để làm trơn kết quả thu được

```
function y = MedSmoothing(x,N) %y tin hieu sau khi loc, x la tin hieu truoc khi loc, N la bac can loc
   y = x; %khoi tao gia tri tin hieu y
   temp = 1:N; % tao truc thoi gian tin hieu cua so co chieu dai N
   for k = 1:length(x) % k chay tu 1 den chieu dai cua tin hieu x
      for i = 1:N %i chay tu 1 den N
         if(k < ceil(N/2))
            %xet gia tri tin hieu cua so temp ben trai tin hieu x
            if(i < ceil(N/2)-k+1) % i < phan tu giua cua temp
               temp(i) = 0;
            else
               temp(i) = x(k+i-ceil(N/2)); % i > phan tu giua cua temp
            end
         else
            %xet gia tri tin hieu cua so ben phai tin hieu temp
           if(k > length(x) - ceil(N/2) + 1)
              %i < phan tu giua cua temp
               if(i < length(x) + ceil(N/2) - k + 1)
                  temp(i) = x(k-ceil(N/2) + i);
               else
                  temp(i) = 0; %i > phan tu giua cua temp
               end
            else
              %xet tin hieu temp chay o giua x.
              temp(i) = x(k-ceil(N/2)+i);
```

```
            end
          end
        end
        %ham median giup tim phan tu trung vi cua day
        y(k) = median(temp);
    end
end
```

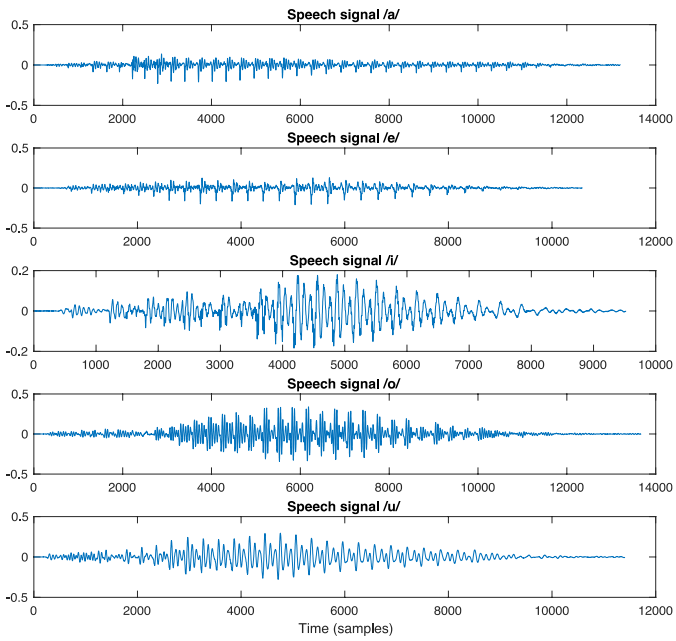
D. Chương trình demo

```
[sig,fs] = audioread('u.wav');              %Fetch the signal
Autocorr_F0 = AutoCorr(sig,fs);             %Find F0 in time domain by using AutoCorrelation Function
Autocorr_F0 = MedSmoothing(Autocorr_F0,7);  %Filter the result by with dimension N=7
fft_F0 = mientanso(sig,fs);                 %Find F0 in frequency domain by using Fast Fourier Transform
fft_F0 = MedSmoothing(fft_F0,7);            %Filter the result by Median Smoothing with dimension N=7
figure(1);
subplot(2,1,1)
stem(Autocorr_F0,'filled');
title('Fundamental Frequency F_0(Hz) by Autocorrelation Function');
ylabel ('F(Hz)');
subplot(2,1,2)
stem(fft_F0,'filled');
title('Fundamental Frequency F_0(Hz) by Fast Fourier Transform');
ylabel ('F(Hz)');
```

IV. KẾT QUẢ THỰC NGHIỆM

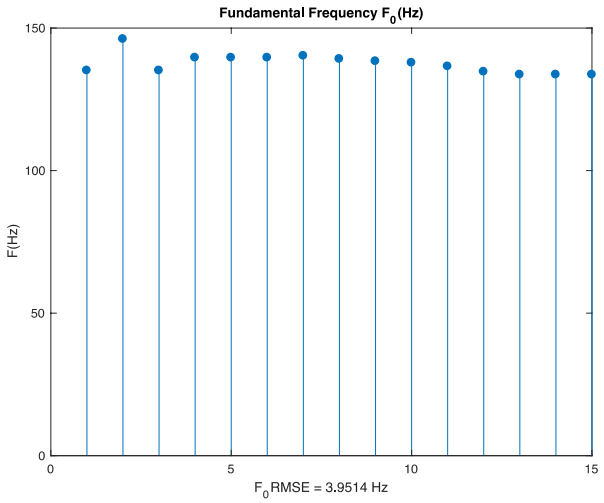
A. Dữ liệu mẫu

Dữ liệu dùng để đánh giá thuật toán là tín hiệu của 5 nguyên âm /a/, /e/, /i/, /o/ và /u/ với tần số lấy mẫu là 44100Hz có biểu diễn đồ thị như hình bên dưới.



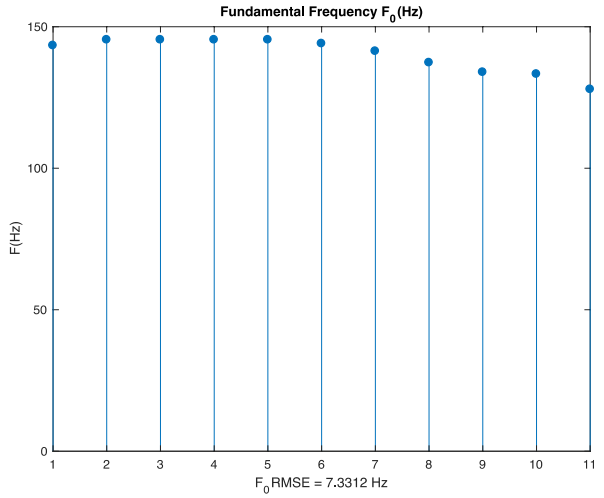
Hình 11. Tín hiệu đầu vào là 5 nguyên âm.

B. Kết quả định tính



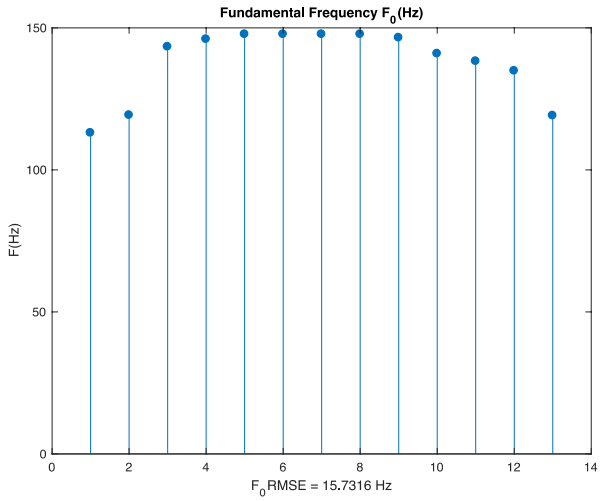
Hình 12. Trường hợp tín hiệu cho kết quả tính F0 chính xác nhất.

Hình 12 mô tả ví dụ về trường hợp cho kết quả tính F0 chính xác nhất sau khi đi qua lọc trung vị của tín hiệu nguyên âm /o/, sai số RMSE là nhỏ nhất.



Hình 13. Trường hợp tín hiệu cho kết quả tính F0 trung bình.

Hình 13 mô tả ví dụ về trường hợp cho kết quả tính F0 có độ chính xác trung bình của tín hiệu nguyên âm /e/, sai số RMSE là trung bình.

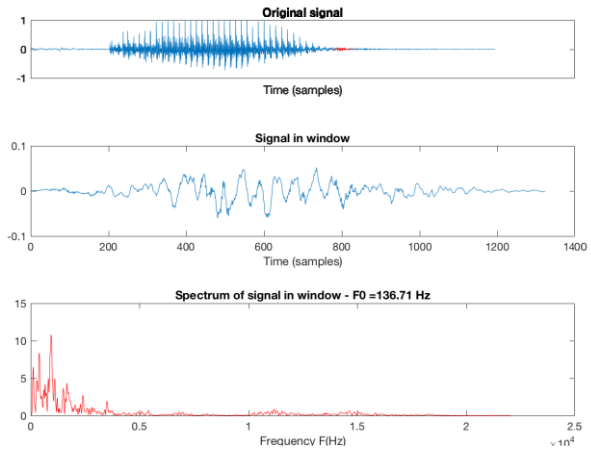


Hình 14. Trường hợp tín hiệu cho kết quả tính F0 kém chính xác nhất.

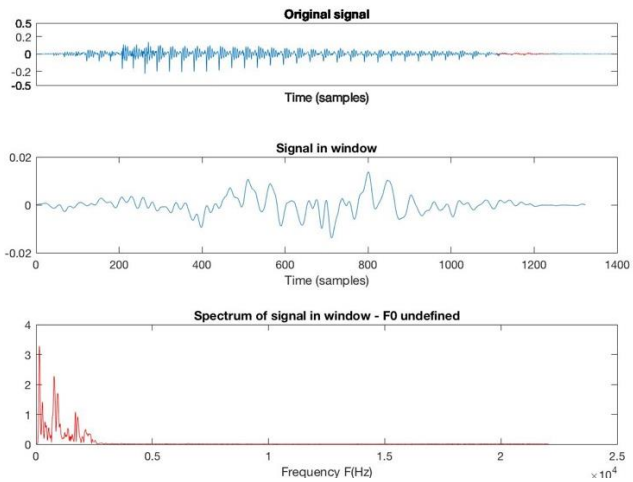
Hình 14 mô tả ví dụ về trường hợp cho kết quả tính F0 kém chính xác nhất sau khi đi qua lọc trung vị của tín hiệu nguyên âm /u/, sai số RMSE là lớn nhất.

C. Kết quả định lượng

Yếu tố đầu tiên ảnh hưởng đến độ chính xác của thuật toán đó là tín hiệu vào, nếu tín hiệu vào được thu âm trong môi trường yên tĩnh, ít nhiễu, thì độ chính xác của thuật toán sẽ cao hơn. Vì nếu nhiễu đáng kể thì ta sẽ khó phân biệt được tín hiệu âm thanh và nhiễu thông qua biên độ. Dưới đây là ví dụ minh hoạ về cửa sổ gần cuối của tín hiệu có nhiễu đáng kể và nhiễu không đáng kể.



Hình 15. Trường hợp kết quả tính F0 của tín hiệu nguyên âm /a/ có nhiễu đáng kể.



Hình 16. Trường hợp kết quả tính F0 của tín hiệu nguyên âm /a/ có nhiễu không đáng kể.

Thông qua hình 15 và hình 16 ta có thể thấy là, nếu nhiễu là đáng kể thì biên độ của nhiễu sẽ rất lớn và gần bằng biên độ của âm thanh của tín hiệu khi không có nhiễu. Do vậy trong trường hợp này ta sẽ gặp khó khăn để loại bỏ trường hợp cửa sổ là nhiễu.

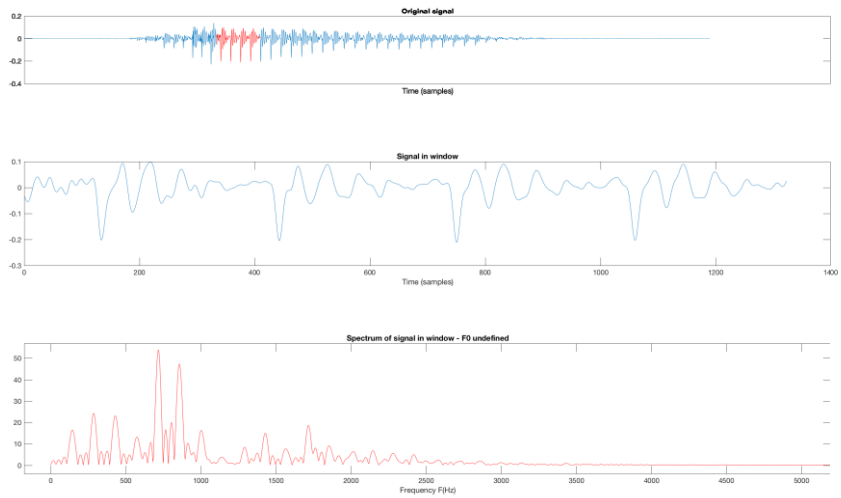
Bên cạnh đó, việc chọn bậc lọc trung vị cũng làm ảnh hưởng đến kết quả của chuỗi F0 thu được. Nếu ta chọn bậc thích hợp thì kết quả F0 sẽ mịn hơn và chính xác hơn. Tuy nhiên, nếu chọn bậc không phù hợp thì sẽ làm giảm độ chính xác của tín hiệu

Bảng 3. Lọc trung vị với các bậc khác nhau.

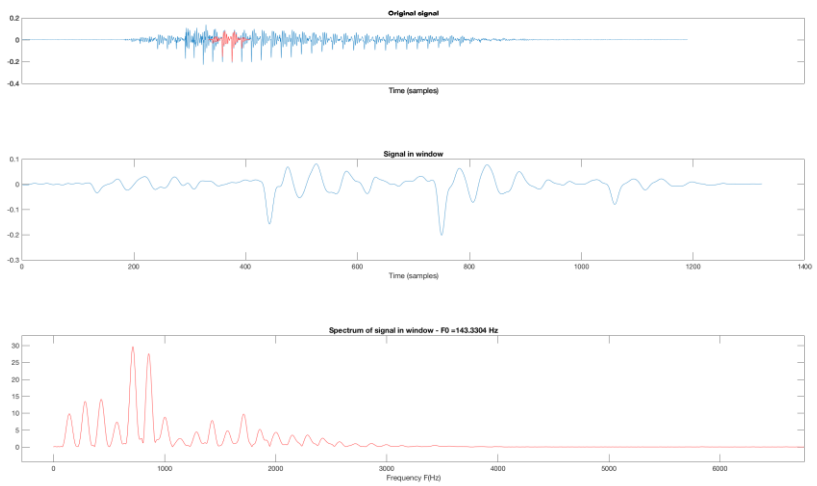
Tín hiệu	Trước khi lọc	Lọc với N = 3	Lọc với N = 5	Lọc với N = 7
/a/	F_0 mean = 136,6013 Hz F_0 RMSE = 7,7731 Hz	F_0 mean = 135,7841 Hz F_0 RMSE = 6,4626 Hz	F_0 mean = 135,4958 Hz F_0 RMSE = 6,1802 Hz	F_0 mean = 135,1112 Hz F_0 RMSE = 5,8212 Hz
/e/	F_0 mean = 140,6999 Hz F_0 RMSE = 7,4707 Hz	F_0 mean = 140,6999 Hz F_0 RMSE = 7,4707 Hz	F_0 mean = 140,2105 Hz F_0 RMSE = 7,3312 Hz	F_0 mean = 140,2105 Hz F_0 RMSE = 7,3312 Hz
/o/	F_0 mean = 137,4709 Hz F_0 RMSE = 7,6768 Hz	F_0 mean = 137,5110 Hz F_0 RMSE = 3,9514 Hz	F_0 mean = 136,3764 Hz F_0 RMSE = 5,3537 Hz	F_0 mean = 135,5654 Hz F_0 RMSE = 6,9525 Hz

Dựa vào bảng 3, ta có thể thấy khi lọc chuỗi F0 của tín hiệu /a/ và /i/ với bậc N = 7 thì kết quả sẽ mịn hơn, giảm độ sai số giữa các kết quả. Tuy nhiên, đối với tín hiệu /o/, nếu ta chọn lọc trung vị bậc N = 7 thì độ chính xác của kết quả sẽ không tăng lên nhiều so với khi không dùng lọc trung vị. Từ đó ta có thể thấy rằng lọc trung vị cũng ảnh hưởng phần nào đến độ sai số giữa các kết quả F0 tính trong từng khung cửa sổ.

Việc chọn hàm cửa sổ cũng rất quan trọng để quyết định độ chính xác của thuật toán. Nếu ta dùng hàm cửa sổ chữ nhật thì sẽ dễ xảy ra hiện tượng rò phổ, nếu ta dùng hàm cửa sổ Hamming thì sẽ khắc phục được phần nào hiện tượng rò phổ.



Hình 17. Chọn cửa sổ chữ nhật.



Hình 18. Chọn cửa sổ Hamming.

Dựa vào hình 17 và hình 18, ta có thể thấy rằng khi chọn hàm cửa sổ chữ nhật, xảy ra hiện tượng rò phổ nên ta khó thể nào phân biệt được đâu là đỉnh của tần số là bội của tần số cơ bản. Khi chọn hàm cửa sổ là Hamming thì vấn đề được giải quyết hiệu quả.

Đối với việc tính F_0 trên miền tần số, một trong những yếu tố quan trọng quyết định độ chính xác của thuật toán đó là việc lựa chọn số điểm DFT. Nếu ta chọn số điểm nhỏ thì độ chia nhỏ nhất (ĐCNN) của các tần số trên phổ của tín hiệu sẽ lớn làm cho phổ của tín hiệu không rõ dẫn đến việc xác định tần số cơ bản cũng không chính xác cao. Ngược lại, nếu chọn số điểm FFT lớn thì độ chia nhỏ nhất của tần số trên phổ của tín hiệu sẽ nhỏ làm cho phổ tín hiệu rõ hơn, dẫn đến việc xác định tần số cơ bản chính xác hơn. Tuy nhiên nếu chọn số điểm FFT quá lớn thì sẽ tốn nhiều tài nguyên của máy tính (thời gian và bộ nhớ). Nên việc lựa chọn số điểm FFT thích hợp cũng là một trong những yếu tố quyết quan trọng quyết định độ chính xác của thuật toán tính F_0 trên miền tần số.

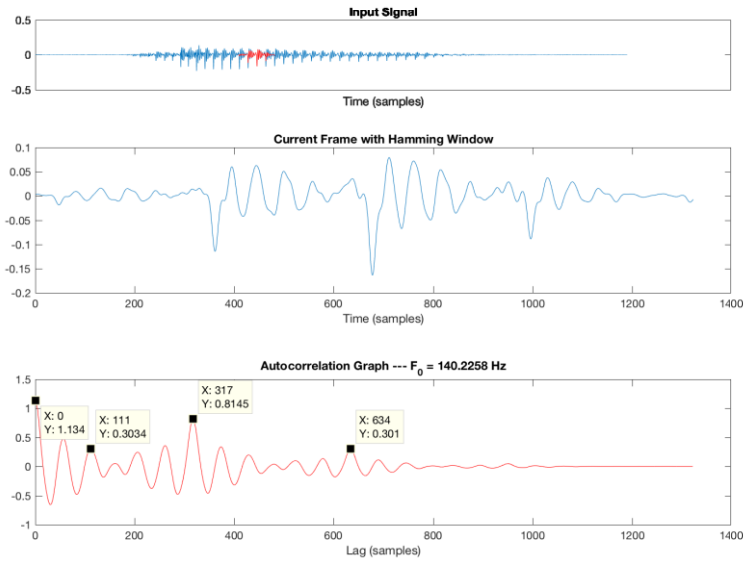
Nếu lấy số điểm FFT là 2^{11} = 2048 thì ĐCNN của tần số là 44100 Hz/2048 = 21,5332 Hz.

Nếu lấy số điểm FFT là 2^{15} = 32768 thì ĐCNN của tần số là 44100 Hz/32768 = 1,3458 Hz.

Nếu lấy số điểm FFT là 2^{17} = 131077 thì ĐCNN của tần số là 44100 Hz/131077 = 0,3364 Hz.

Từ đó ta thấy số điểm lấy FFT phù hợp là 2^{15} = 32768, tức là độ chia nhỏ nhất trên miền tần số là 1,3458 Hz, đủ nhỏ để phổ của tín hiệu được thể hiện rõ và thuận tiện cho việc tính tần số cơ bản mà không tốn quá nhiều tài nguyên của máy tính.

Đối với việc tính F0 trên miền thời gian, một trong những yếu tố quan trọng quyết định độ chính xác của thuật toán đó là khoảng cách tối thiểu giữa hai cực đại. Trong nhiều trường hợp, càng ra xa trung tâm thì các cực đại có biên độ giống nhau nên để tìm ra chính xác cực đại là bội số của chu kỳ cơ bản thì phải giới hạn khoảng cách tối thiểu giữa hai cực đại.



Hình 19. Đồ thị tương quan của một khung tín hiệu nguyên âm /a/.

Hình 19 cho ta thấy rằng, nếu không giới hạn khoảng cách thì 3 cực đại đầu tiên của đồ thị tương quan sẽ không ứng với các cực đại là bội số của chu kỳ cơ bản. Do vậy việc tìm khoảng cách tối thiểu giữa các cực đại là rất quan trọng.

Sau khi đo thủ công (lấy trung bình 5 lần đo trên mỗi tín hiệu nguyên âm) ở bài báo cáo trước, kết hợp với việc sử dụng thuật toán để tính tần số cơ bản tự động ta được bảng kết quả dưới đây.

Bảng 4. Kết quả F_0 đo được bằng các cách khác nhau.

Tín hiệu	Tần số cơ bản F_0 (Hz)		
	Đo thủ công	Hàm tự tương quan	Phép biến đổi Fourier nhanh
/a/	138,1492	136,0219	135,1112
/e/	144,5861	138,3254	140,2105
/i/	140,1293	136,1413	136,1526
/o/	139,6827	137,5110	136,2167
/u/	147,9374	137,8562	142,9022
Trung bình	142,0969	137,1712	138,1186

Dựa vào bảng 4, ta có thể thấy kết quả trung bình tần số cơ bản của 5 nguyên âm khi tính bằng hàm tự tương quan và phép biến đổi Fourier nhanh là gần bằng nhau. Do đó, ta có thể kết luận là tần số cơ bản của tín hiệu âm thanh có thể tính được trên miền thời gian dùng hàm tự tương quan và trên miền tần số dùng phép biến đổi Fourier nhanh. Tuy nhiên, để biết được thuật toán nào chính xác hơn thì ta cần phải thực nghiệm trên nhiều loại tín hiệu hơn và giá trị chuẩn chính xác hơn.

Sai số toàn phương (RMSE) của thuật toán có công thức sau:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y - Y_i)^2}$$

Với Y là giá trị F_0 tính được bằng cách đo thủ công, Y_i là giá trị F_0 tính được trong mỗi frame.

Bảng 5. Kết quả sai số toàn phương của thuật toán trên các tín hiệu.

Tín hiệu	Sai số toàn phương (RMSE) (Hz)	
	Hàm tự tương quan	Phép biến đổi Fourier nhanh
/a/	5,0766	5,8212
/e/	9,6934	7,3312
/i/	6,2293	5,0869
/o/	3,9514	4,8207
/u/	15,7316	6,5979
Trung bình	8,1365	5,9316

Dựa vào bảng 5, ta thấy độ sai số toàn phương của hàm tự tương quan lớn hơn của phép biến đổi Fourier nhanh. Tuy nhiên vì giá trị chuẩn được tính bằng thủ công nên độ chính xác không tuyệt đối nên không thể kết luận độ chính xác của mỗi thuật toán thông qua sai số toàn phương (RMSE). Nhìn chung, mỗi thuật toán tính tần số cơ bản trên miền thời gian và trên miền tần số sẽ có những ưu điểm và nhược điểm khác nhau.

Dùng hàm tự tương quan để tính tần số cơ bản trên miền thời gian có thể áp dụng được cho nhiều loại tín hiệu, kể cả tín hiệu có chất lượng thấp hoặc có nhiễu đáng kể. Bên cạnh đó, hàm tự tương quan còn là cơ sở cho các phương pháp phân tích phổ và việc cài đặt hàm tự tương quan khá đơn giản giúp tiết kiệm tài nguyên máy tính. Tuy nhiên, hàm tự tương quan có một số bất lợi đáng kể là khó điều chỉnh các tham số của thuật toán (như khoảng cách tối thiểu của hai cực đại) một cách chính xác. Khoảng cách giữa các cực đại ảnh hưởng bởi thành tố của âm tiết, do đó tính chu kỳ của tín hiệu cũng bị ảnh hưởng, dẫn đến sai sót trong tính toán.

Việc tính tần số cơ bản trong trên miền tần số bằng phép FFT thì sẽ chính xác hơn với trường hợp lấy điểm DFT lớn, đồng nghĩa với việc các đỉnh trên phổ dày đặc hơn cũng như ít rò phổ hơn, nhưng lại tốn tài nguyên máy tính nhiều hơn. Ngược lại, nếu lấy điểm DFT nhỏ thì độ chính xác giảm đi đáng kể do mật độ của các đỉnh thưa thớt [9].

V. KẾT LUẬN

Bài báo này thực hiện việc cài đặt thuật toán tìm tần số cơ bản của tín hiệu tiếng nói trên miền thời gian dùng hàm tự tương quan và trên miền tần số dùng phép biến đổi Fourier nhanh trong Matlab. Để làm trơn các tần số cơ bản được xác định, lọc trung vị được sử dụng. Các thử nghiệm với tín hiệu của 5 nguyên âm (/a/, /e/, /i/, /o/ và /u/) cho thấy sai số toàn phương trung bình của thuật toán dùng hàm tự tương quan có và không có lọc trung vị lần lượt là 9,7584Hz và 8,1365Hz. Đối với thuật toán dùng phép biến đổi nhanh Fourier thì sai số toàn phương trung bình khi có và không có lọc trung vị lần lượt là 9,7186Hz và 5,9316Hz. Sự chênh lệch giữa các kết quả tính tần số cơ bản của mỗi nguyên âm càng cao thì việc sử dụng lọc trung vị là cần thiết để giảm sai số.

Trong tương lai chúng em sẽ thử nghiệm các thuật toán tính tần số cơ bản tiên tiến hơn để cải thiện độ chính xác của việc tính toán tần số cơ bản trên tín hiệu tiếng nói của con người. Bên cạnh đó, chúng em sẽ thực nghiệm các thuật toán trên tín hiệu vào là một câu nói dài, thay vì chỉ là một nguyên âm ngắn.

VI. NHỮNG ĐIỀU ĐÃ HỌC ĐƯỢC

Thông qua việc thực hiện bài báo cáo này, sinh viên đã học thêm được nhiều bài học cũng như tích lũy thêm được kinh nghiệm về cách trình bày một bài báo cáo. Cải thiện khả năng sử dụng tiếng Anh thông qua việc tìm tư liệu trên Internet cũng như việc đọc các tài liệu tham khảo mà giảng viên đã giao. Biết cách phân công công việc một cách hợp lý. Tăng cường được khả năng giao tiếp trong nhóm. Hiểu rõ được phần việc mỗi cá nhân làm và luôn cải thiện liên tục để nâng cao chất lượng bài báo cáo. Tích lũy thêm nhiều kiến thức bổ ích trong lĩnh vực xử lí tín hiệu, cụ thể là xử lí tín hiệu tiếng nói.

VII. TÀI LIỆU THAM KHẢO

[1] Tần số âm cơ bản. Retrieved from link: https://vi.wikipedia.org/wiki/Tần_số_âm_cơ_bản

[2] Nguyen Binh Thien, Ninh Khanh Duy, " Cải thiện thuật toán tự tương quan tìm cao độ của tín hiệu đàn ghi-ta trên vi xử lý ARM Cortex-M4 ", trường Đại học Bách Khoa, Đại học Đà Nẵng.

[3] Biến đổi Fourier rời rạc. Retrieved from link: https://vi.wikipedia.org/wiki/Biến_đổi_Fourier_rời_rạc

[4] Biến đổi Fourier nhanh. Retrieved from link: https://en.wikipedia.org/wiki/Biến_đổi_Fourier_nhanh

[5] J. Fessler, "The Discrete Fourier Transform", May 27, 2004. Retrieved from link: https://web.eecs.umich.edu/~fessler/course/451/l/pdf/c5.pdf

[6] Tom Bäckström, "Fundamental Frequency Modelling and Estimation Speech Processing", Aalto University, October 2015. Retrieved from link: https://mycourses.aalto.fi/pluginfile.php/146206/mod_resource/content/1/slides_05_f0_estimation.pdf.

[7] Lawrence R. Rabiner, Ronald W. Schafe, " Prentice Hall - Digital Processing Of Speech Signals 1978 ", pp. 158-159.

[8] Window function. Retrieved from link: https://en.wikipedia.org/wiki/Window_function

[9] Fast Fourier Transform and autocorrelation function for the analysis of complex mass spectra. Retrieved from link: https://www.sciencedirect.com/science/article/pii/S1387380613000195?fbclid=IwAR0YQrZQyMRl0921bxip9eqJlwhJzOzMoCUmvzCFz9DJeHHseWbAX6rdvEQ

[10] Phép biến đổi FFT. Retrieved from link http://www.stu.edu.vn/uploads/documents/310810-112932.pdf

[11] Median Smoothing and Speech Processing. Retrieved from link: https://www.ece.ucsb.edu/Faculty/Rabiner/ece259/digital%20speech%20processing%20course/lectures_new/Lecture_algorithms_fall_2010_6tp.pdf?fbclid=IwAR1jEOx8Q64AiHUE3BKx5Gnxi2jnhaMVTGk_uwTq_DvEiSSjX91qKKXp2Dc