

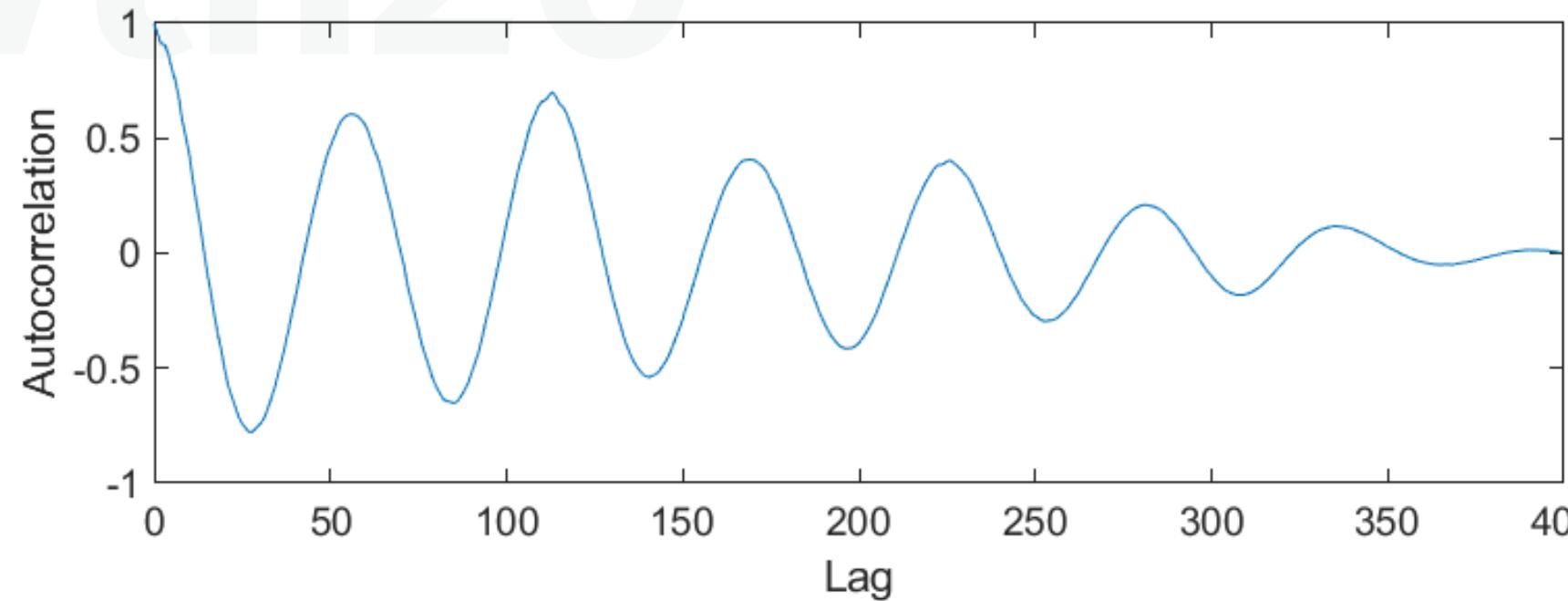
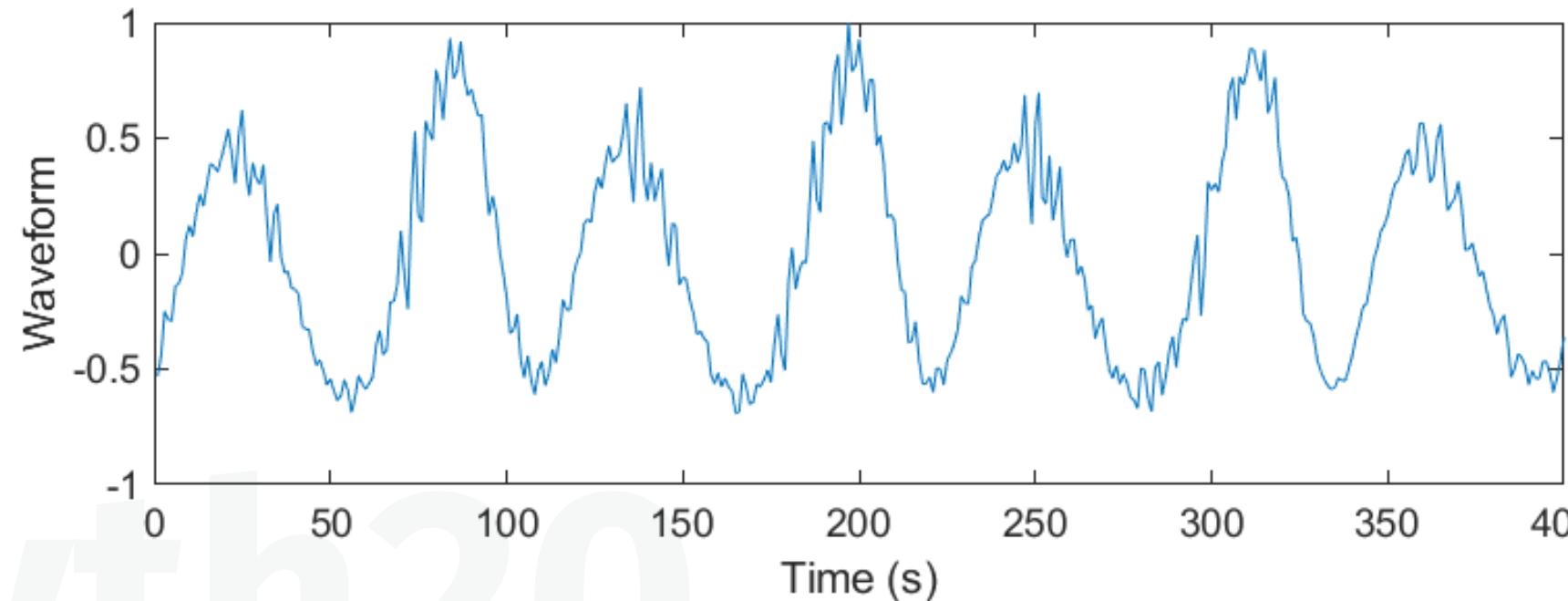


TRƯỜNG ĐẠI HỌC BÁCH KHOA - ĐẠI HỌC ĐÀ NẴNG  
BÁO CÁO MÔN XỬ LÝ TÍN HIỆU SỐ

TÍNH TẦN SỐ CƠ BẢN CỦA TÍN  
HIỆU ÂM THANH BẰNG HÀM TỰ  
TƯƠNG QUAN

GIÁO VIÊN HƯỚNG DẪN : TS. NINH KHÁNH DUY  
SINH VIÊN THỰC HIỆN : HUỲNH VĂN THẠNH  
MSSV : 123190109  
LỚP : 19PFIEV3

# 1. Cơ sở lý thuyết.



Kết quả ACF tại 1 frame.

$$xx[n] = \sum_{m=-\infty}^{\infty} x[m]x[m+n].$$

n: lag/shift  
m: sample index

Autocorrelation Function

$$xx[n] = \sum_{m=0}^{N-1-n} x[m]x[m+n], \quad n \in [0, N[.$$

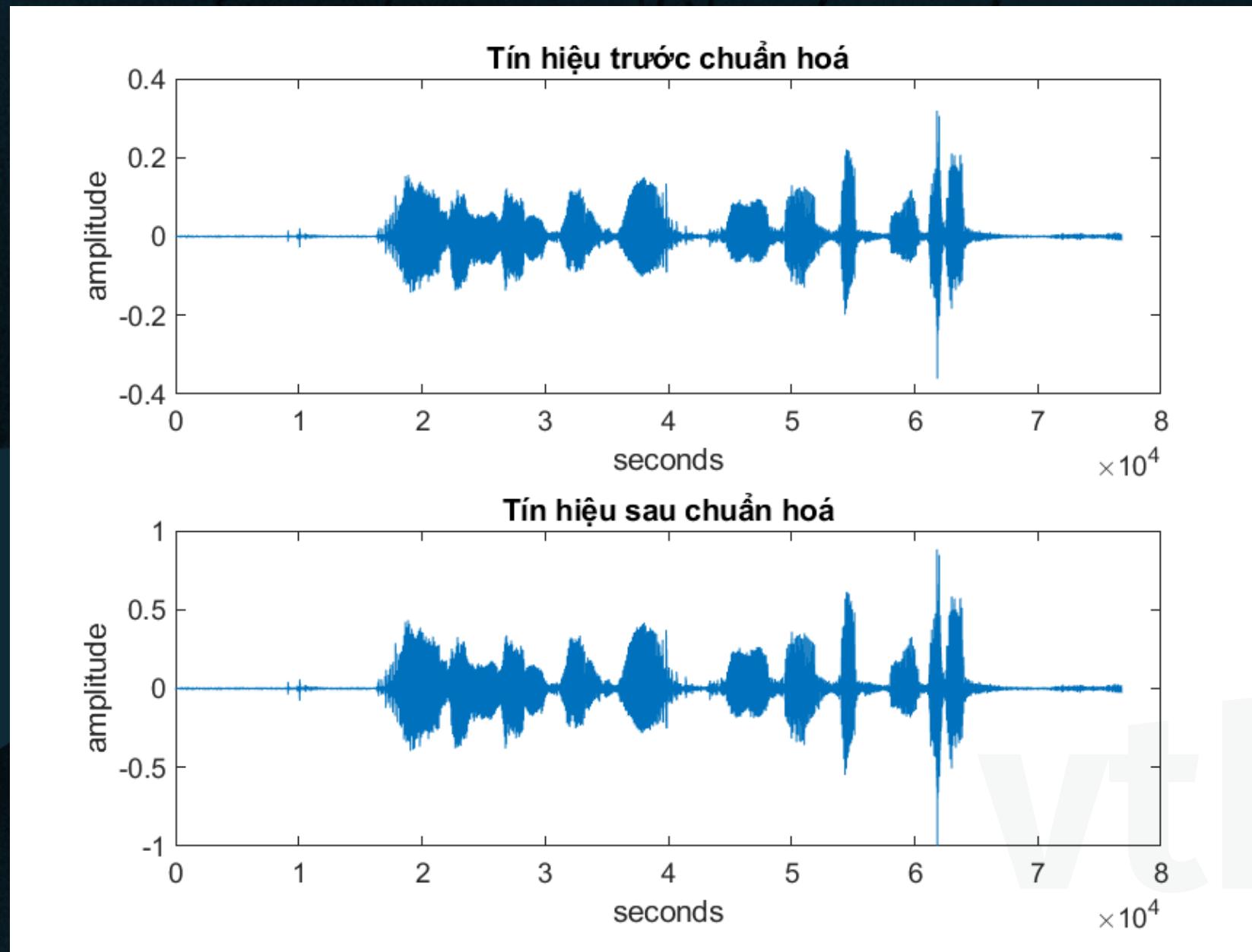
n: lag (samples)  
m: sample index  
N: frame length (samples)

Short-time Autocorrelation Function

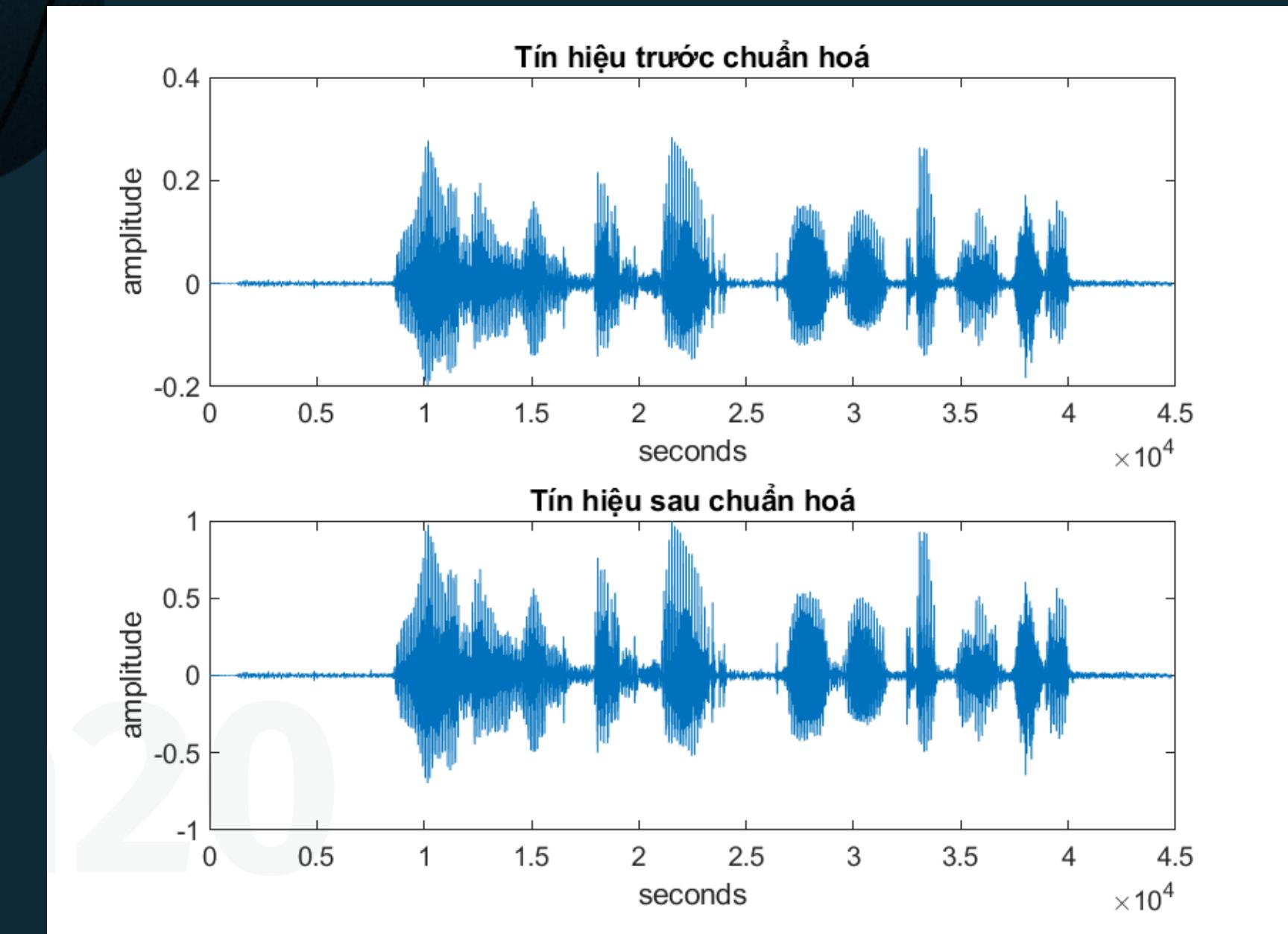
## 2. Sơ đồ thuật toán



### 3. Tín hiệu sau khi Chuẩn hoá

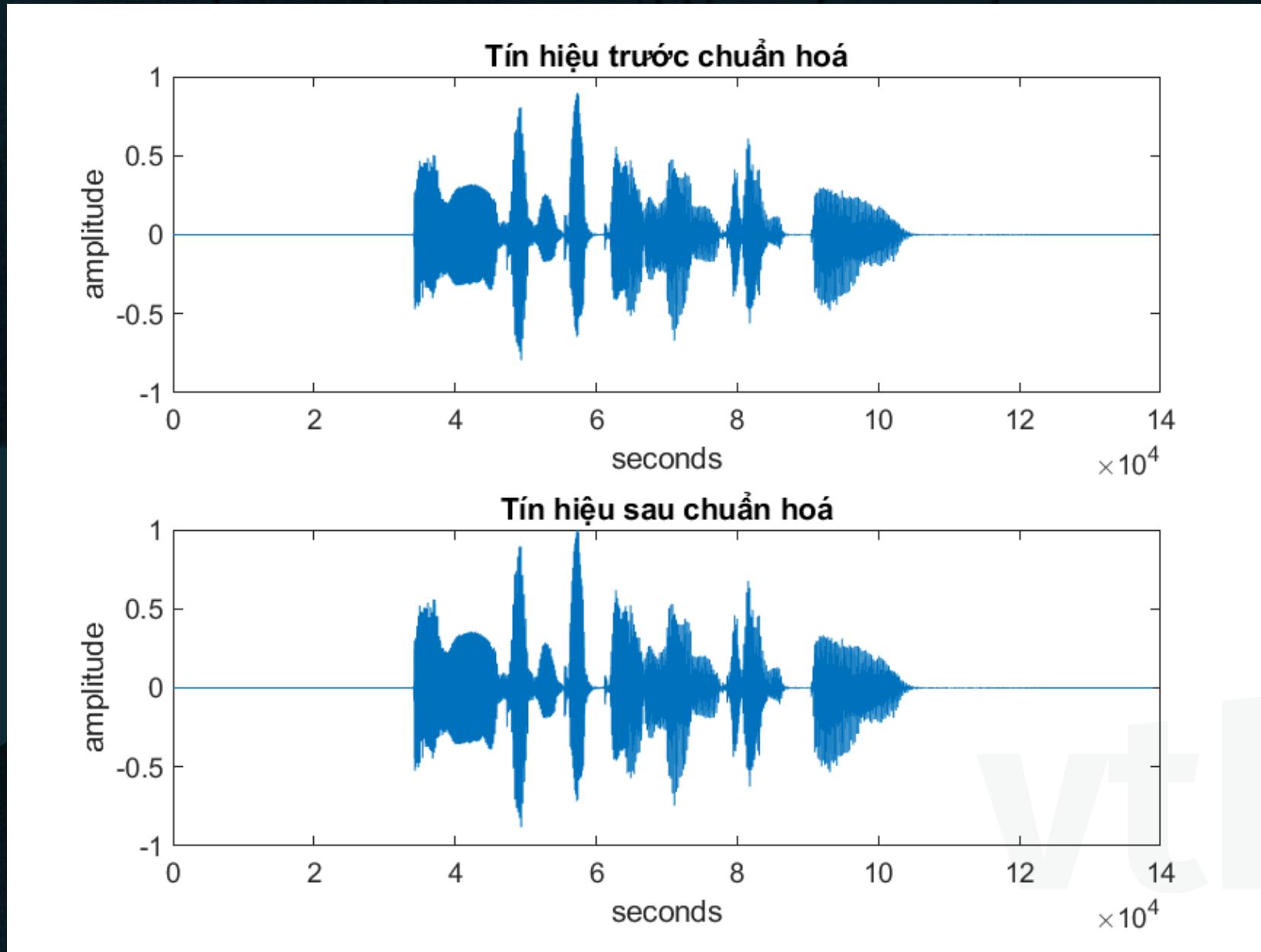


File phone F2

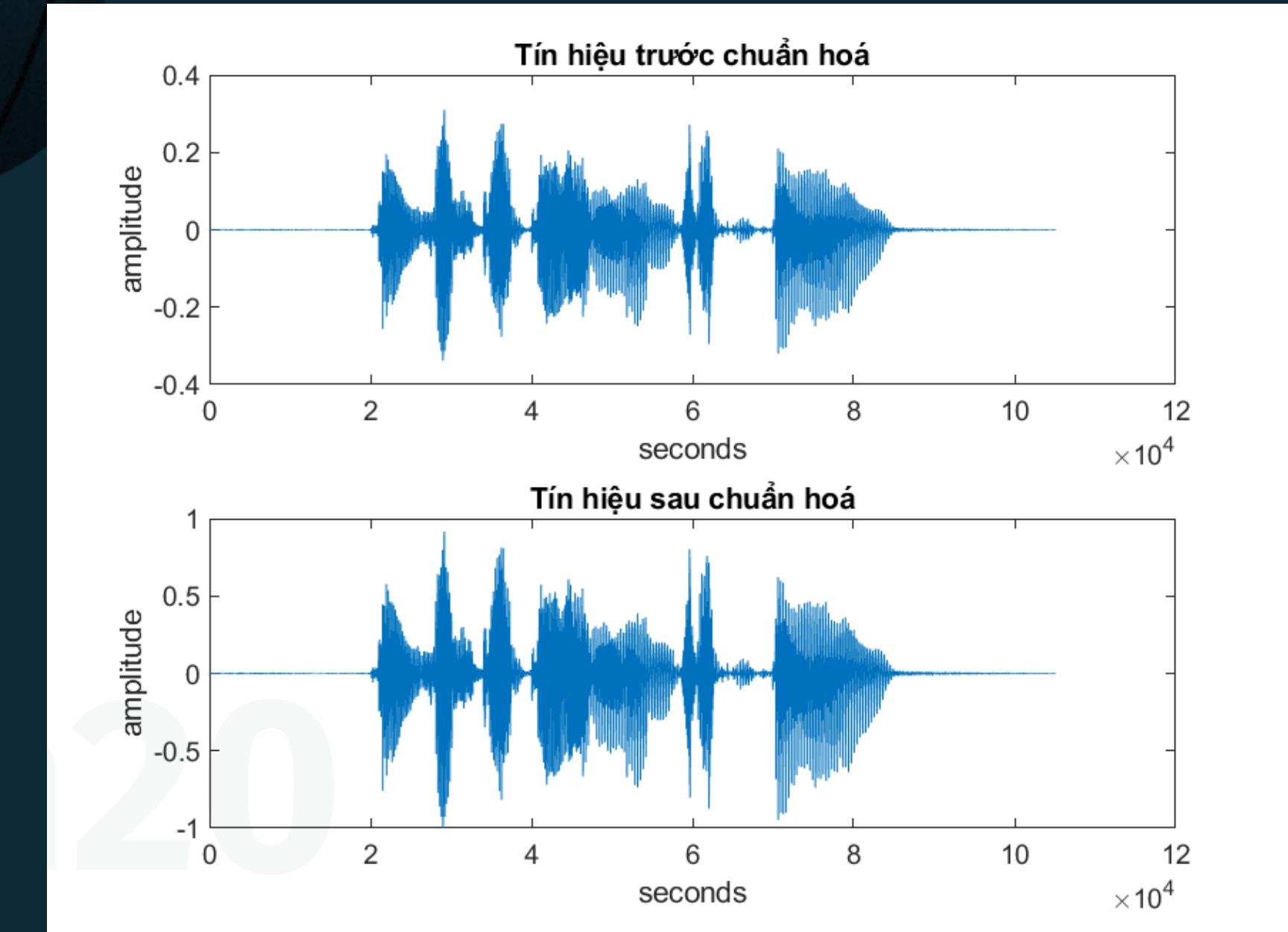


File phone M2

## 2. Tín hiệu sau khi Chuẩn hoá



File studio F2



File studio M2

## 4. Chia khung

- Độ dài khung là 0.025s.

Kết quả:

- phone\_F2: 192 khung, 400 mẫu/khung.
- phone\_M2: 112 khung, 400 mẫu/khung.
- studio\_F2: 125 khung, 1103 mẫu/khung.
- studio\_M2: 95 khung, 1103 mẫu/khung.

The screenshot shows a MATLAB code editor window titled 'ChiaKhung.m'. The code is a function that takes three inputs: 'data', 'fs', and 'time\_frame'. It calculates the number of frames 'n' as the round value of the product of time\_frame and fs. It then determines the total length 'L' of the input data. The number of frames 'N\_frame' is calculated as the floor value of L divided by n. A for loop iterates from 1 to N\_frame, assigning the corresponding segment of the input data to the 'frames' matrix. The loop concludes with an end statement. The entire code is contained within a function definition.

```
function [frames] = ChiaKhung(data, fs, time_frame)
    n = round(time_frame * fs);
    L = length(data);
    N_frame = floor(L/n);

    for i = 1 : N_frame
        frames(i,:) = data((i-1)*n+1 : i*n);
    end
end
```

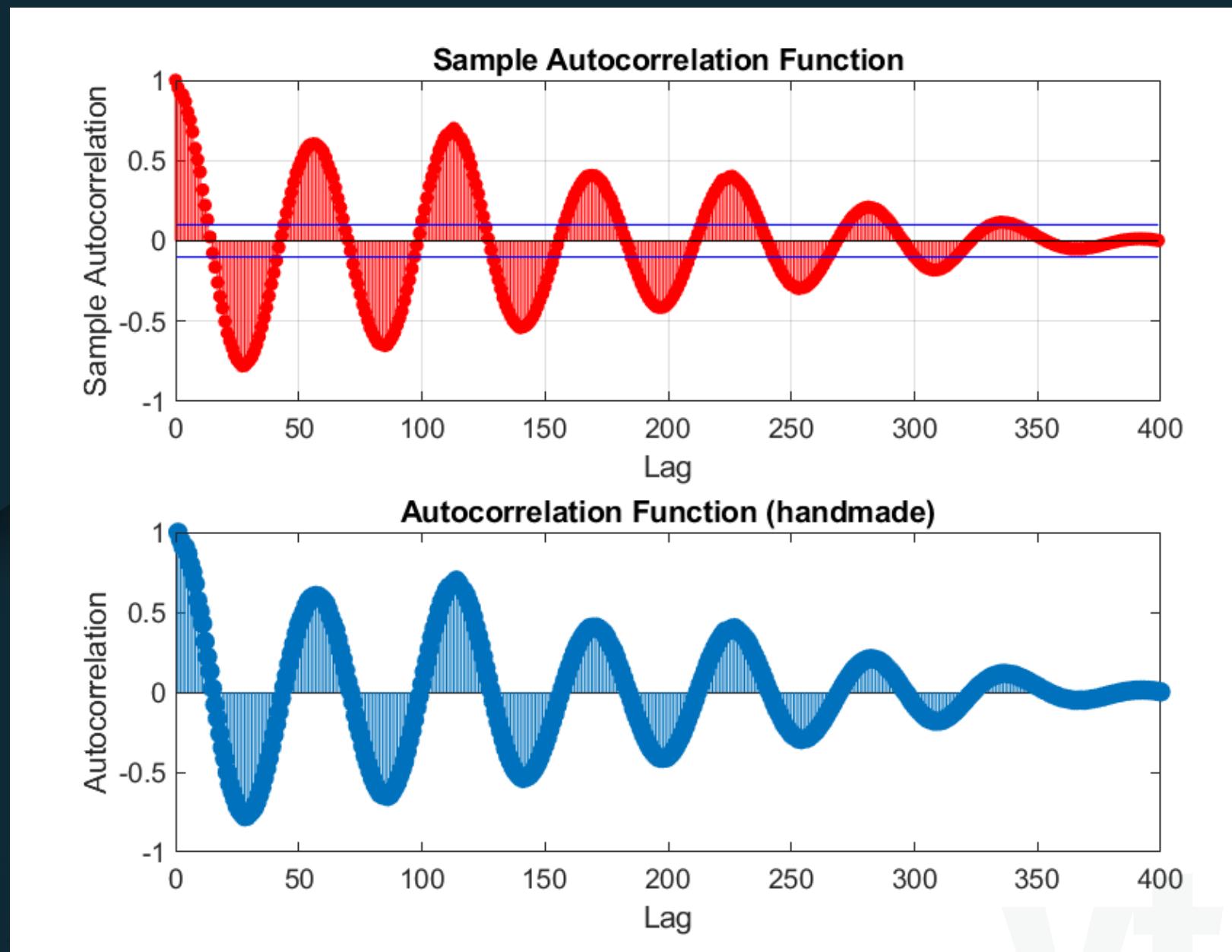
Hàm chia khung.

# 5. Hàm autoCorrelation.

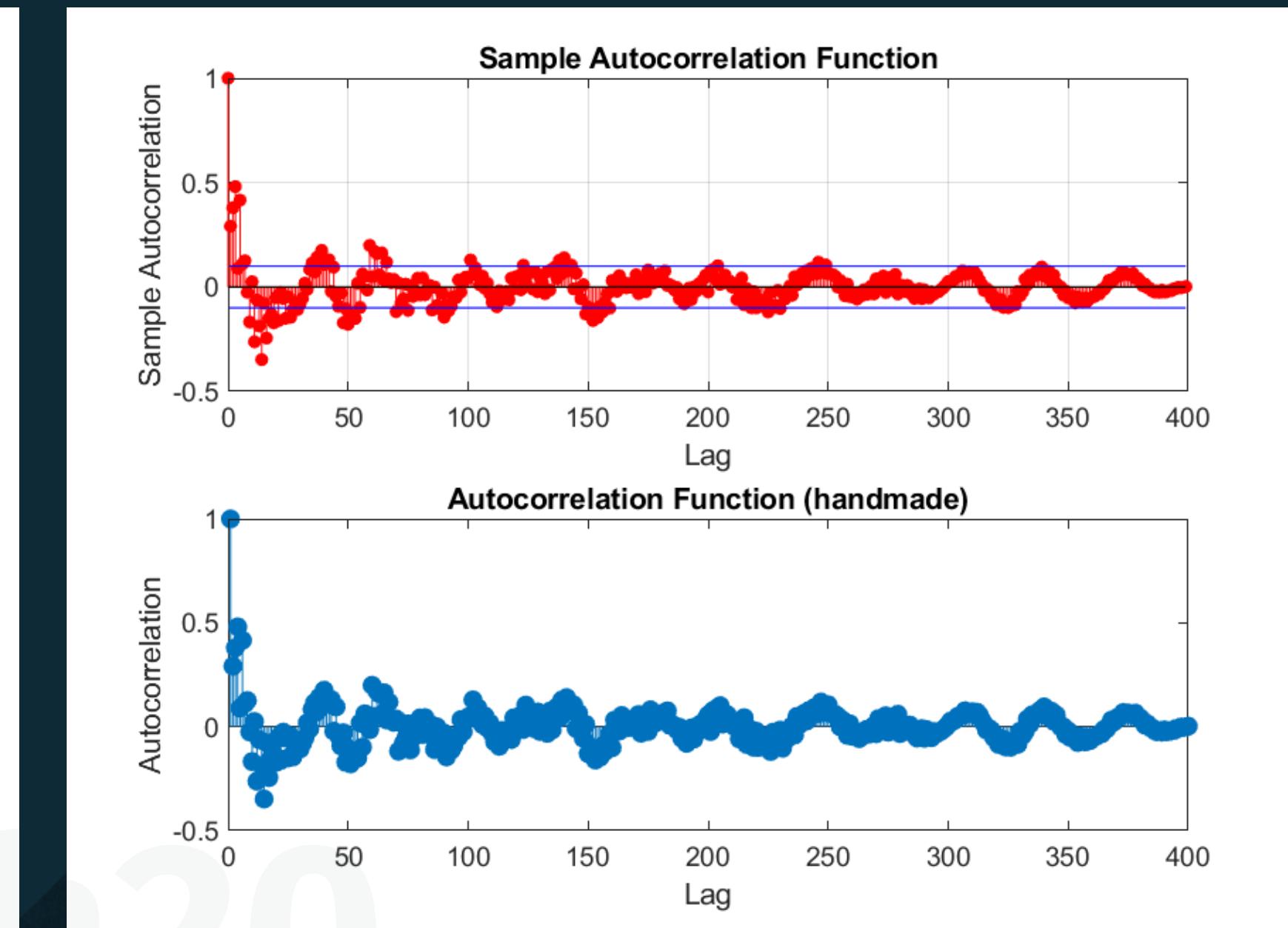
```
1 function [acf, lags] = autoCorrelation(data)
2 % Hàm autoCorrelation nhận vào dữ liệu 1 khung tín hiệu
3 % Trả về acf: dữ liệu sau khi dùng ACF và lags: độ trễ của tín hiệu.
4
5 N = length(data); % Độ dài khung tín hiệu đầu vào (sample).
6 lags = 0:N-1; % Mảng độ trễ của tín hiệu.
7
8 for i = 0: N-1
9     xx=0;
10    for j = 1:N-i
11        s=data(j)*data(j+i);
12        xx=xx+s;
13    end
14    acf(i+1) = xx; % Phần tử thứ i+1 của acf sẽ bằng giá trị của xx vừa tìm được.
15 end
16 % ===== vẽ kết quả trung gian =====
17 acf = ChuanHoa(acf); % Chuẩn hoá giá trị của các phần tử acf nằm trong [-1,1]
18 stem(acf, 'filled');
19 title('Autocorrelation Function (handmade)');
20 xlabel('Lag (s)');
21 ylabel('Autocorrelation');
22 end
```

Hàm autoCorrelation (handmade)

## 6. So sánh hàm autoCorelation tự làm với autocorr của Matlab.

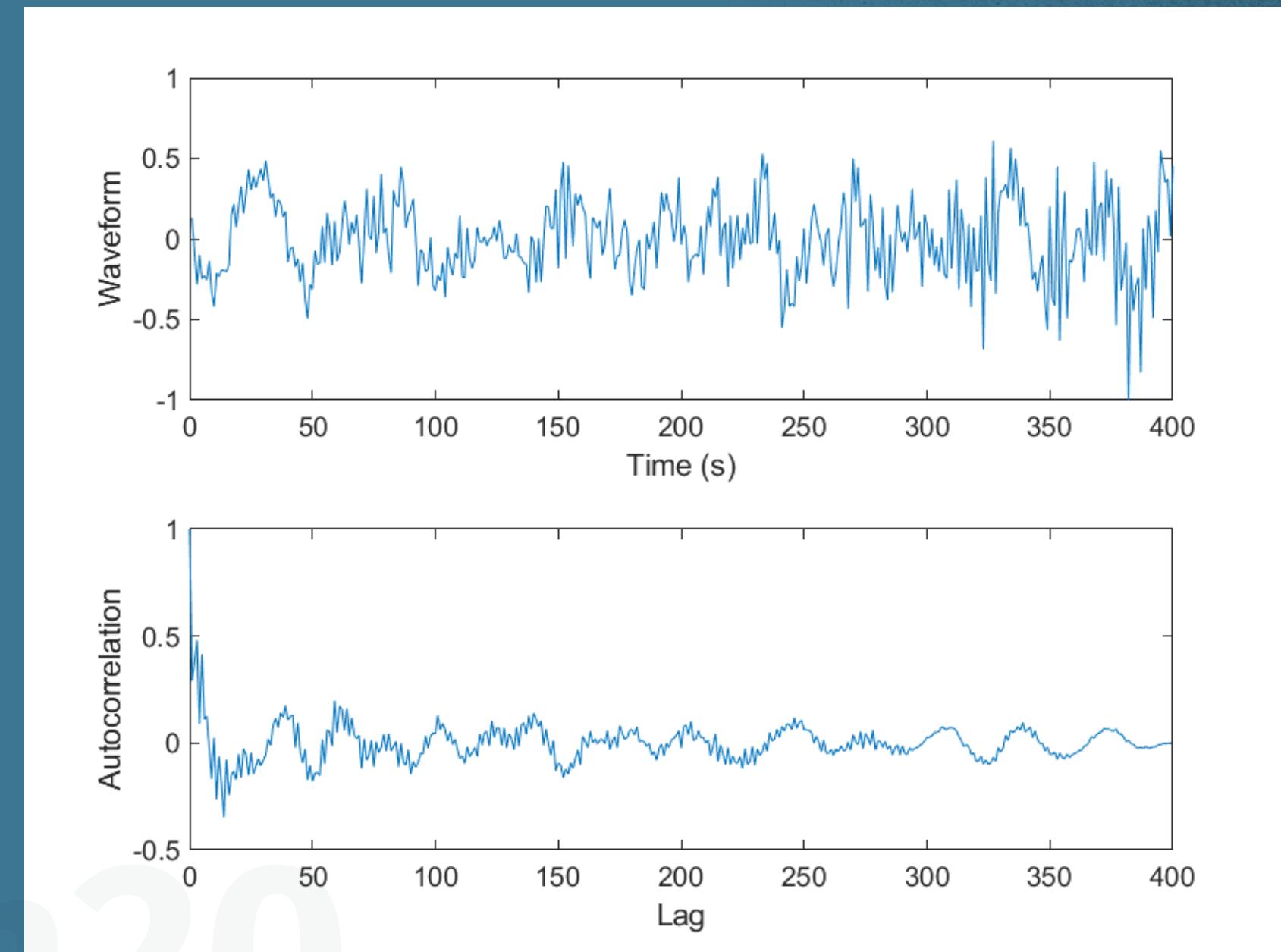
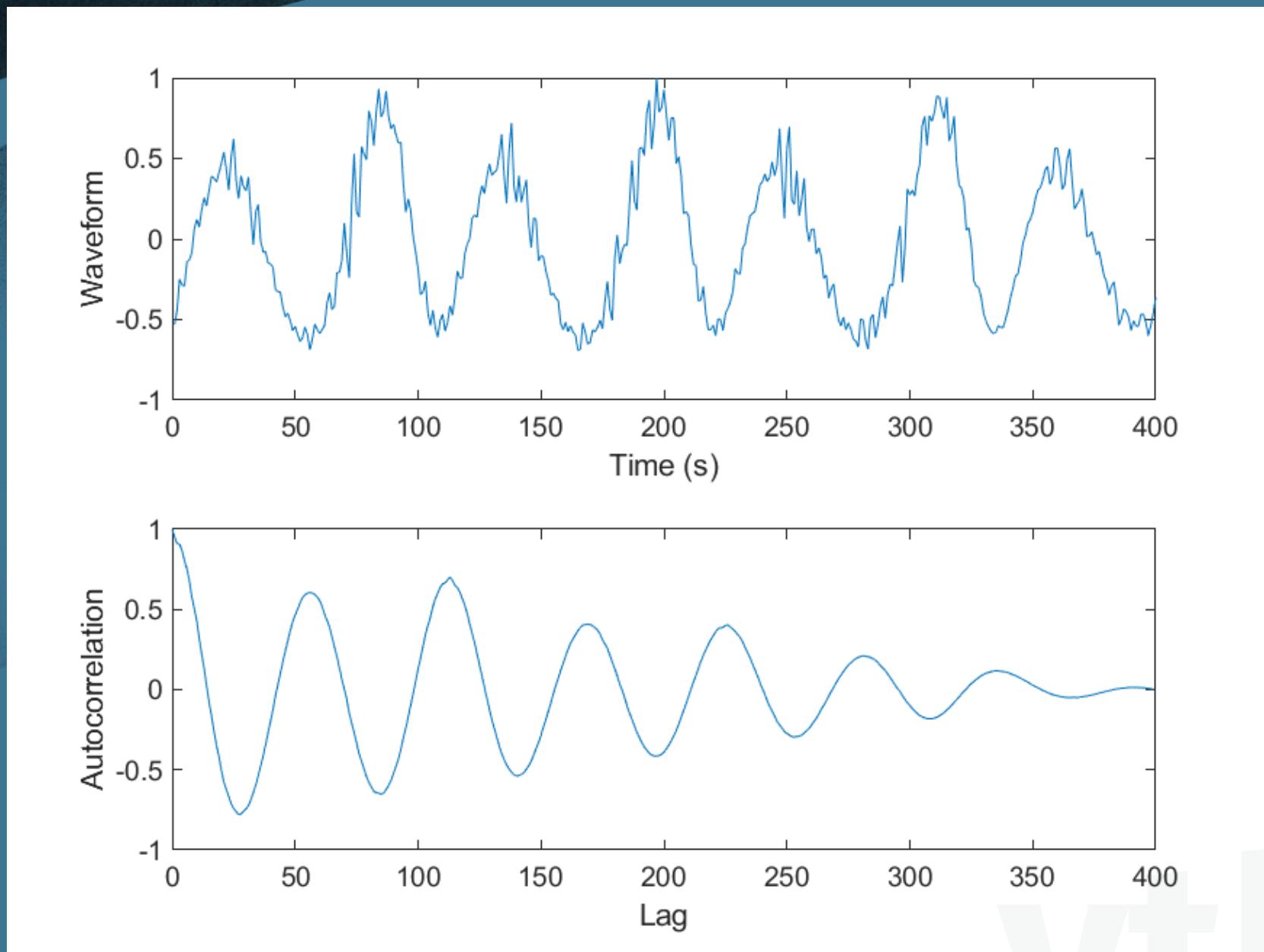


Khung voiced trong Phone\_F2



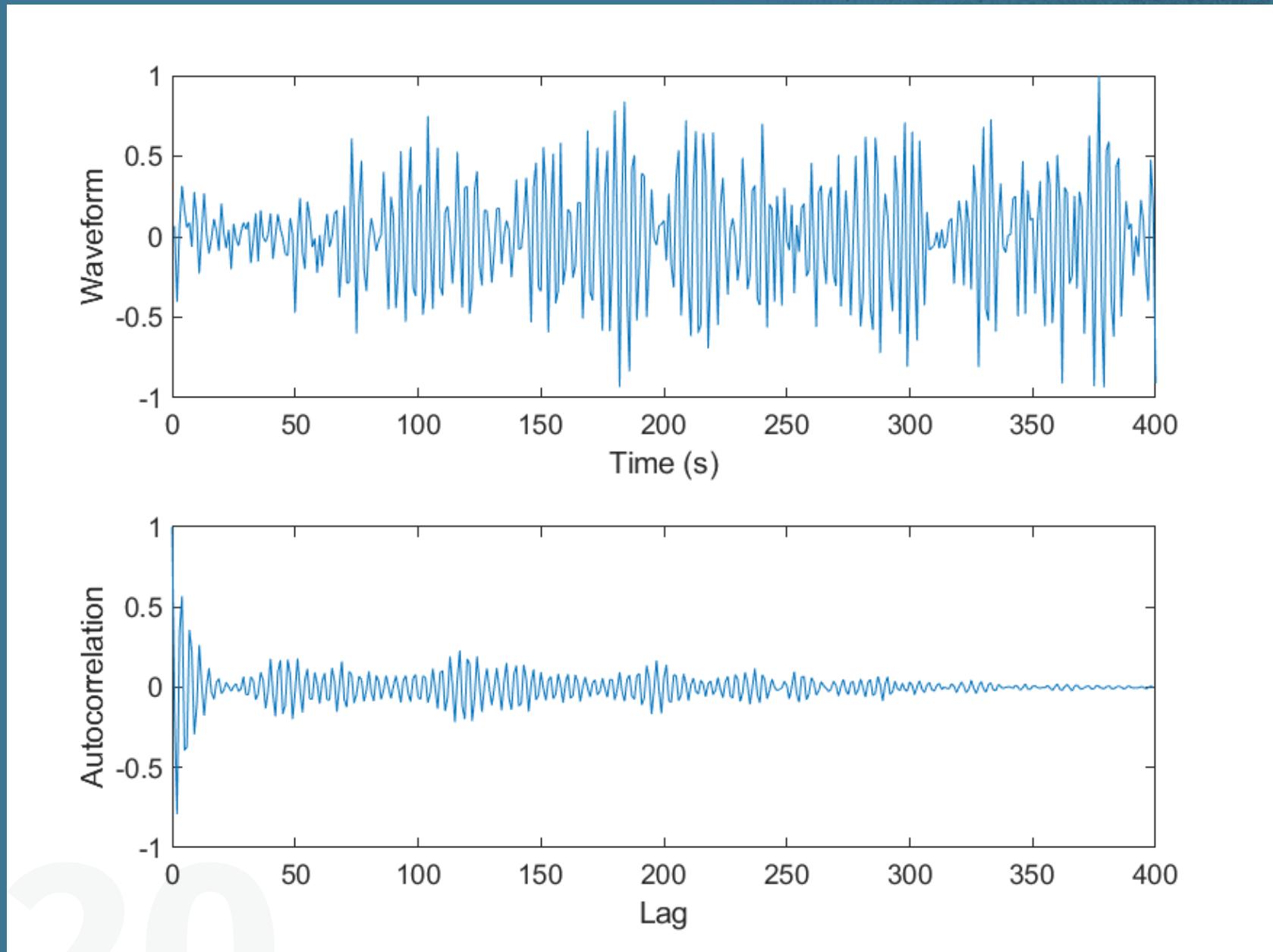
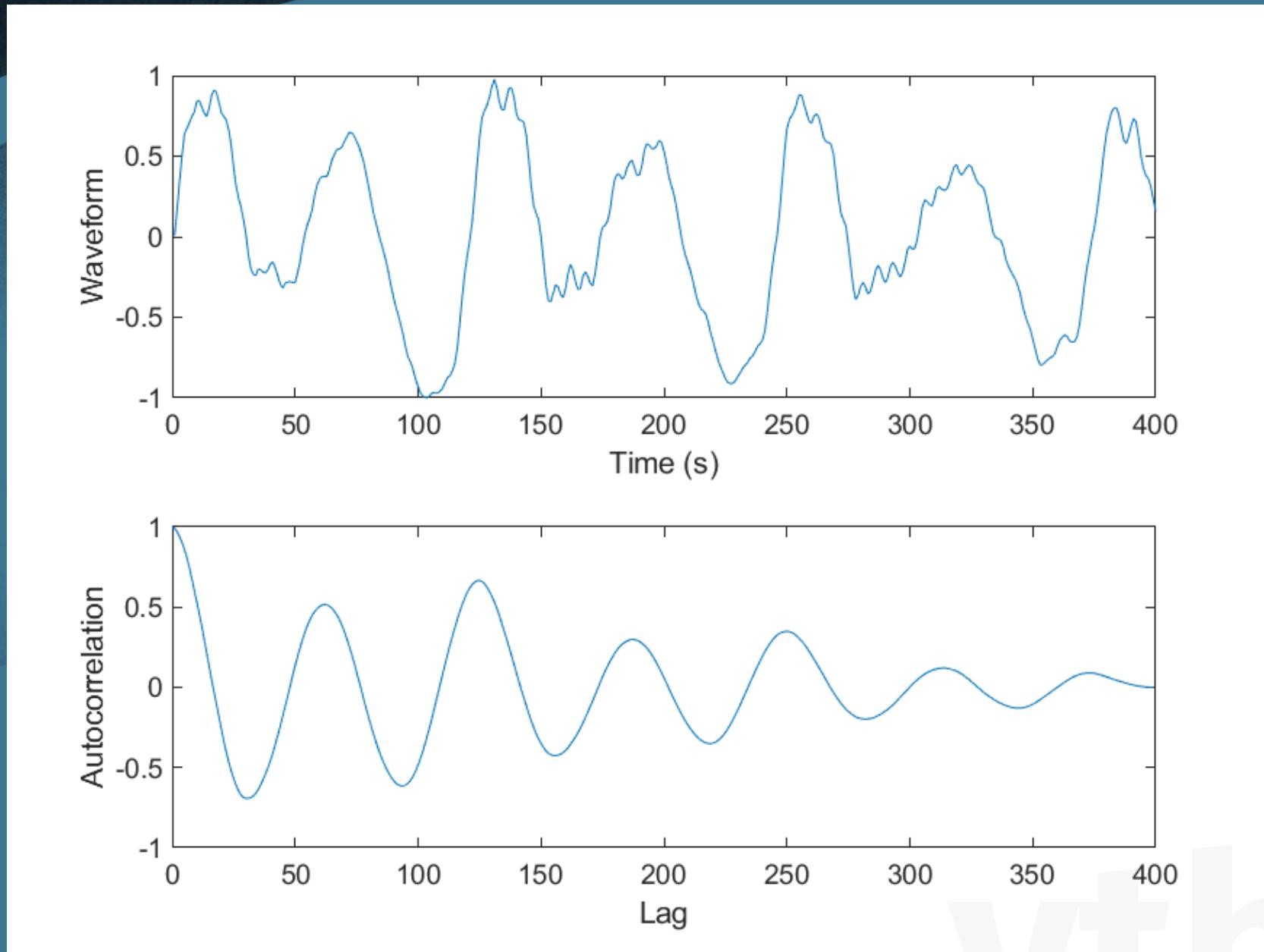
Khung unvoice trong Phone\_F2

## 7. Kết quả trung gian với hàm autoCorrelation.



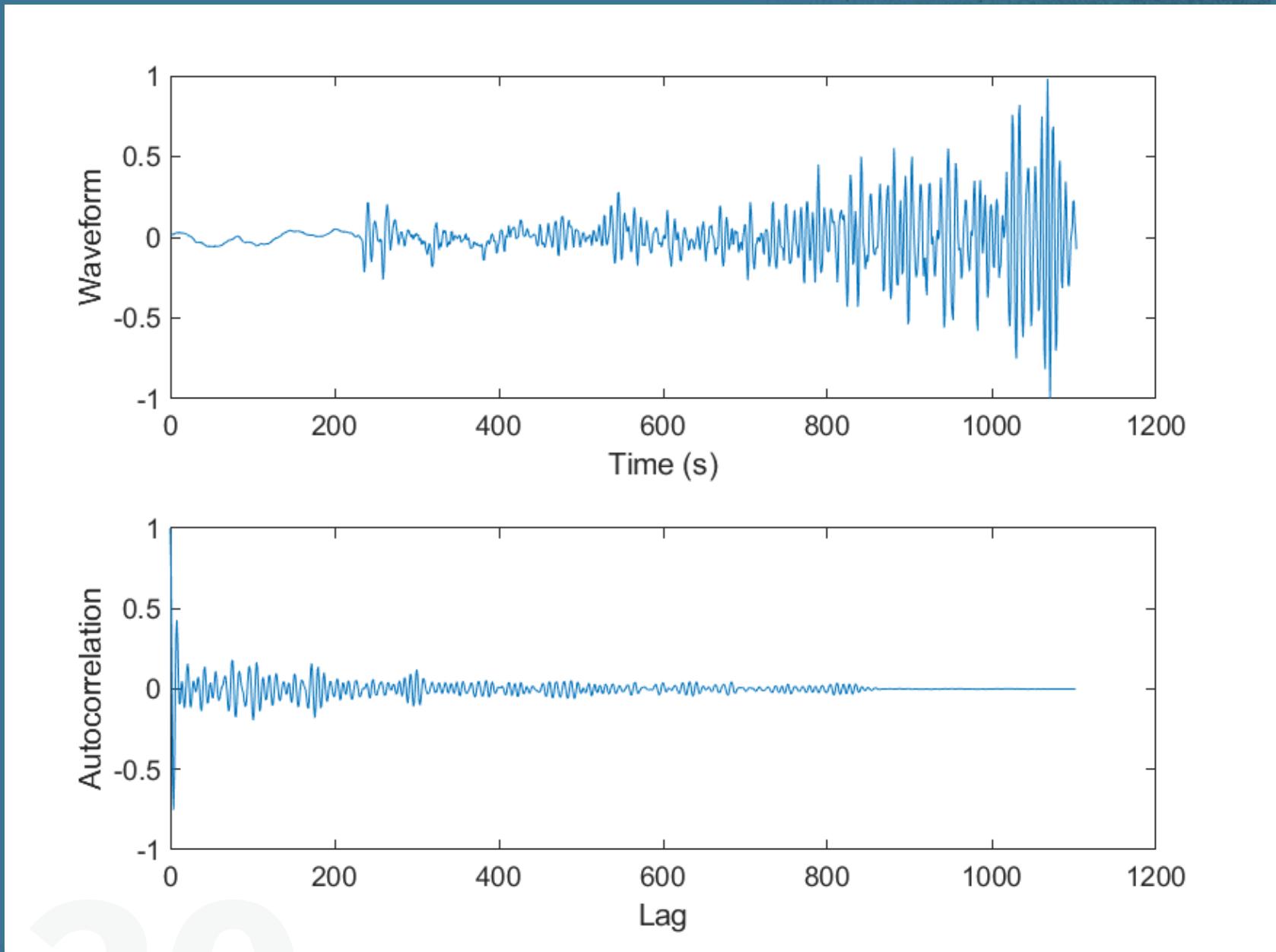
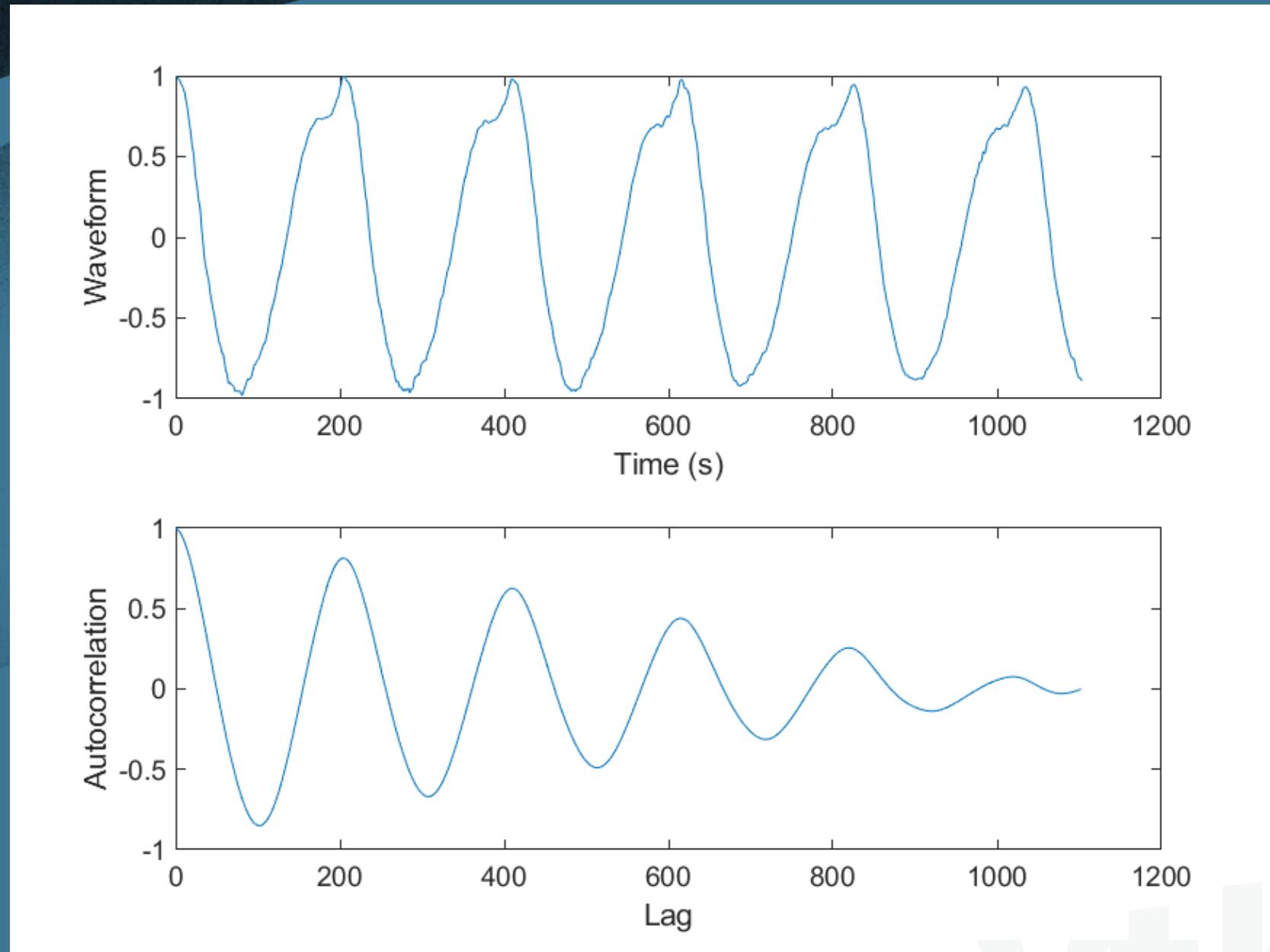
Phone F2

## 7. Kết quả trung gian với hàm autoCorrelation.

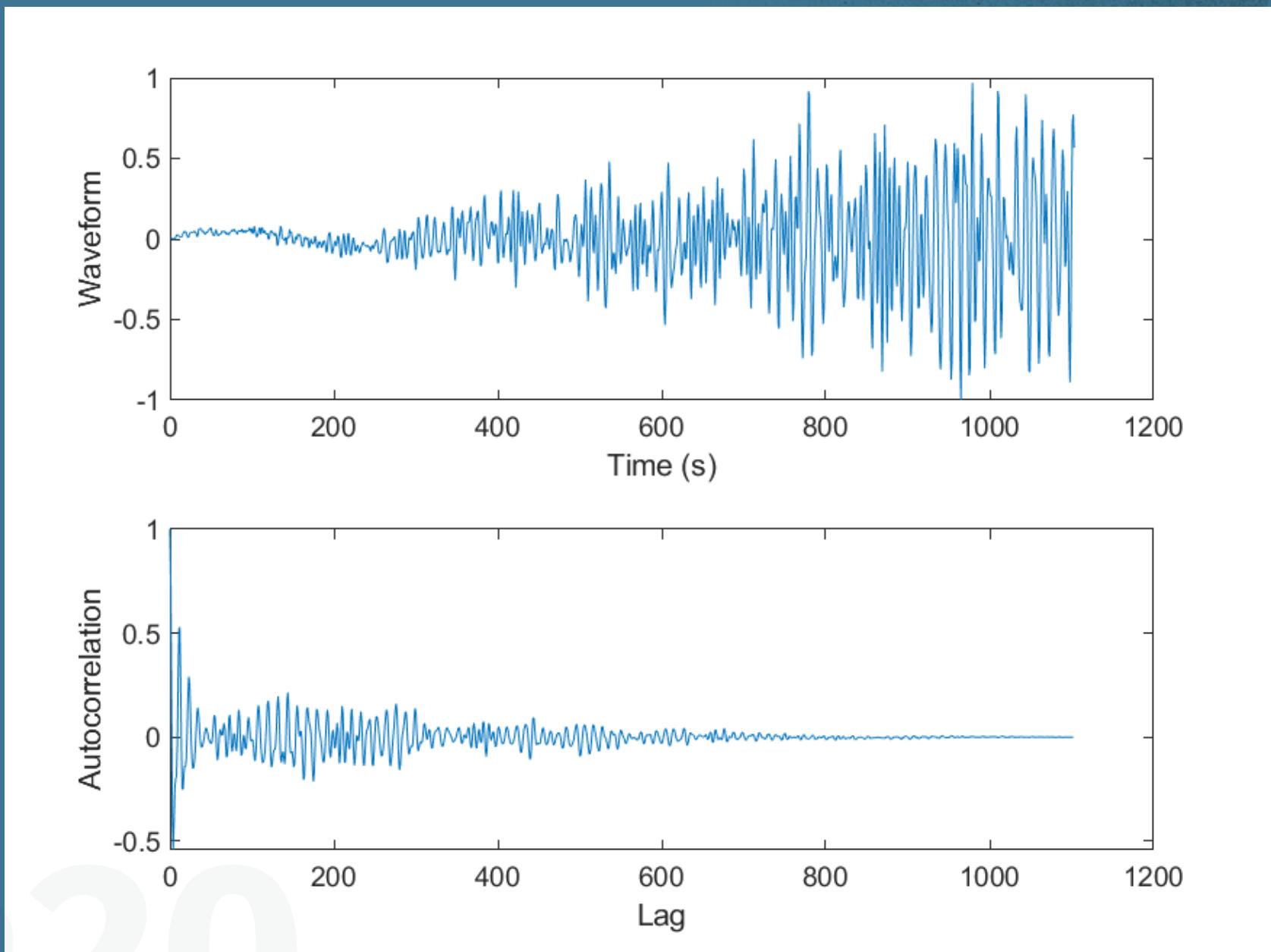
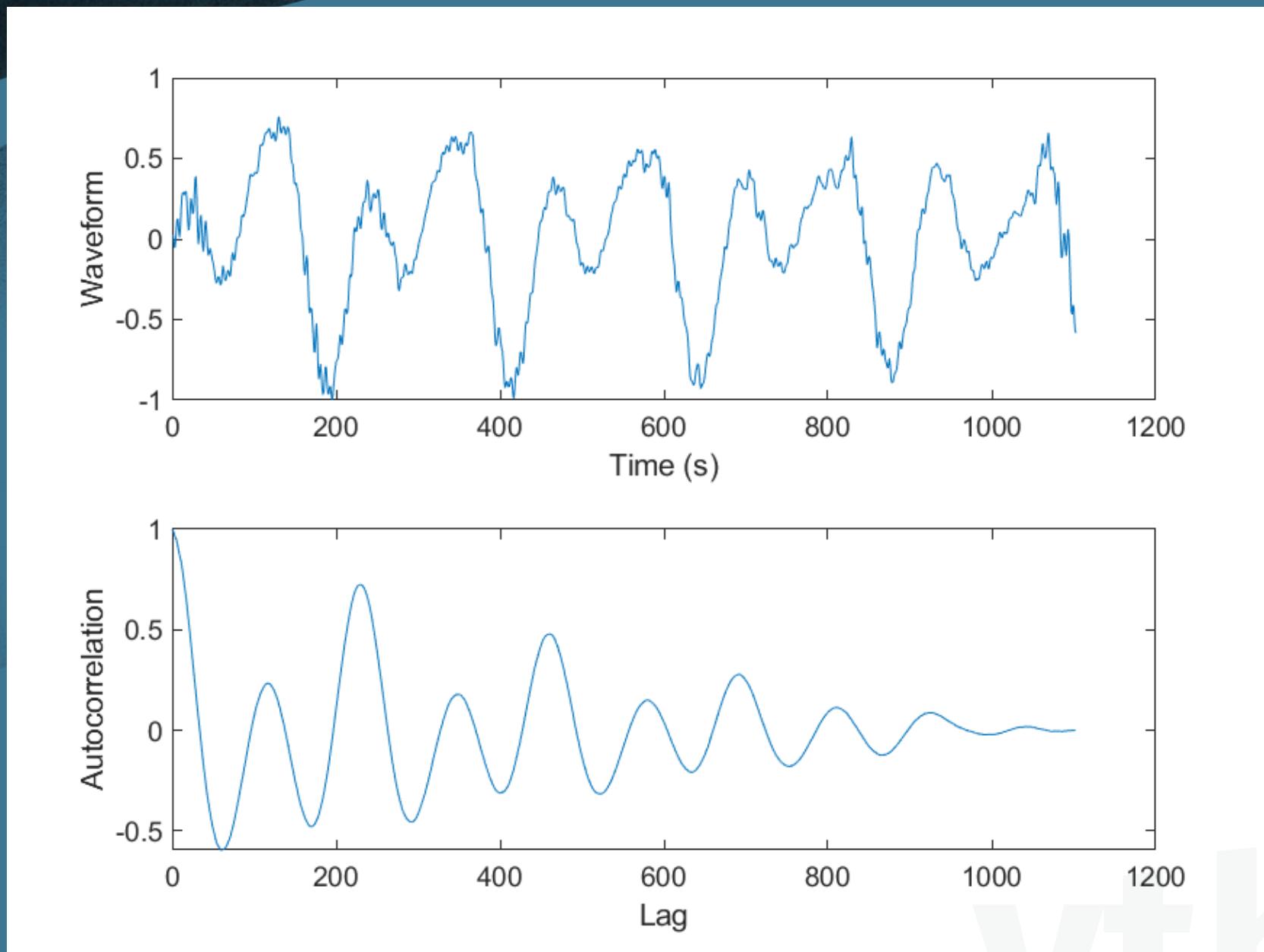


Phone M2

## 7. Kết quả trung gian với hàm autoCorrelation.



## 7. Kết quả trung gian với hàm autoCorrelation.



## 8. Tìm đỉnh cục bộ.

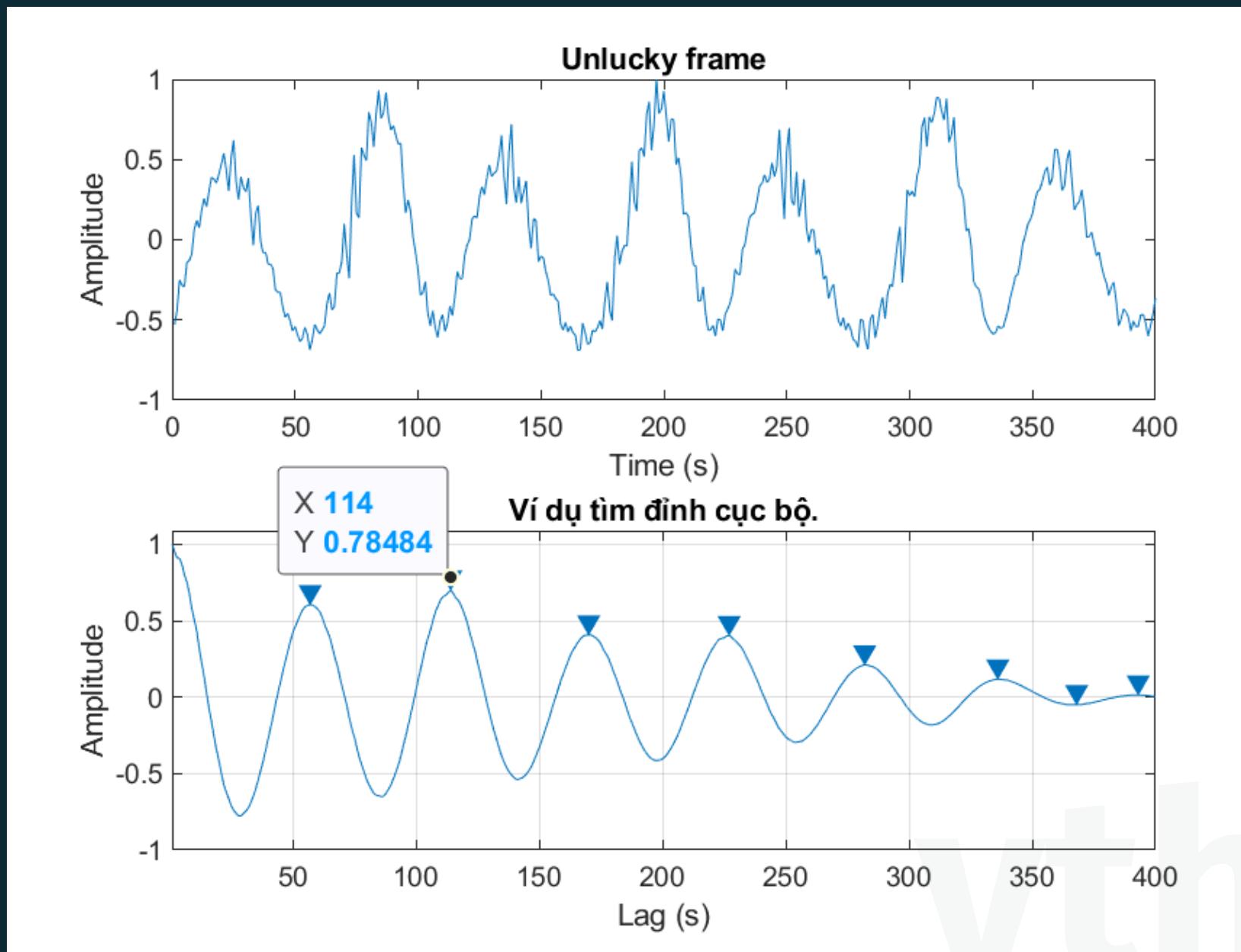


```
1 function [value_Peak, id_Peak] = TimDinhCucBo(ac)
2 % Hàm TimDinhCucBo nhận vào acf của 1 frame,
3 % trả ra giá trị của cực đại lớn nhất và vị trí của cực đại đó trong mảng.
4 L = length(ac); % L chứa độ dài của ac.
5 value_Peak = 0; % Khởi tạo giá trị cho biến value_Peak.
6 id_Peak = 0; % Khởi tạo giá trị cho biến id_Peak.
7 for i=2:L-1
8     if ac(i-1) < ac(i) && ac(i) > ac(i+1)
9         if ac(i) > value_Peak
10            value_Peak = ac(i);
11            id_Peak = i;
12        end
13    end
14 end
15 end
```

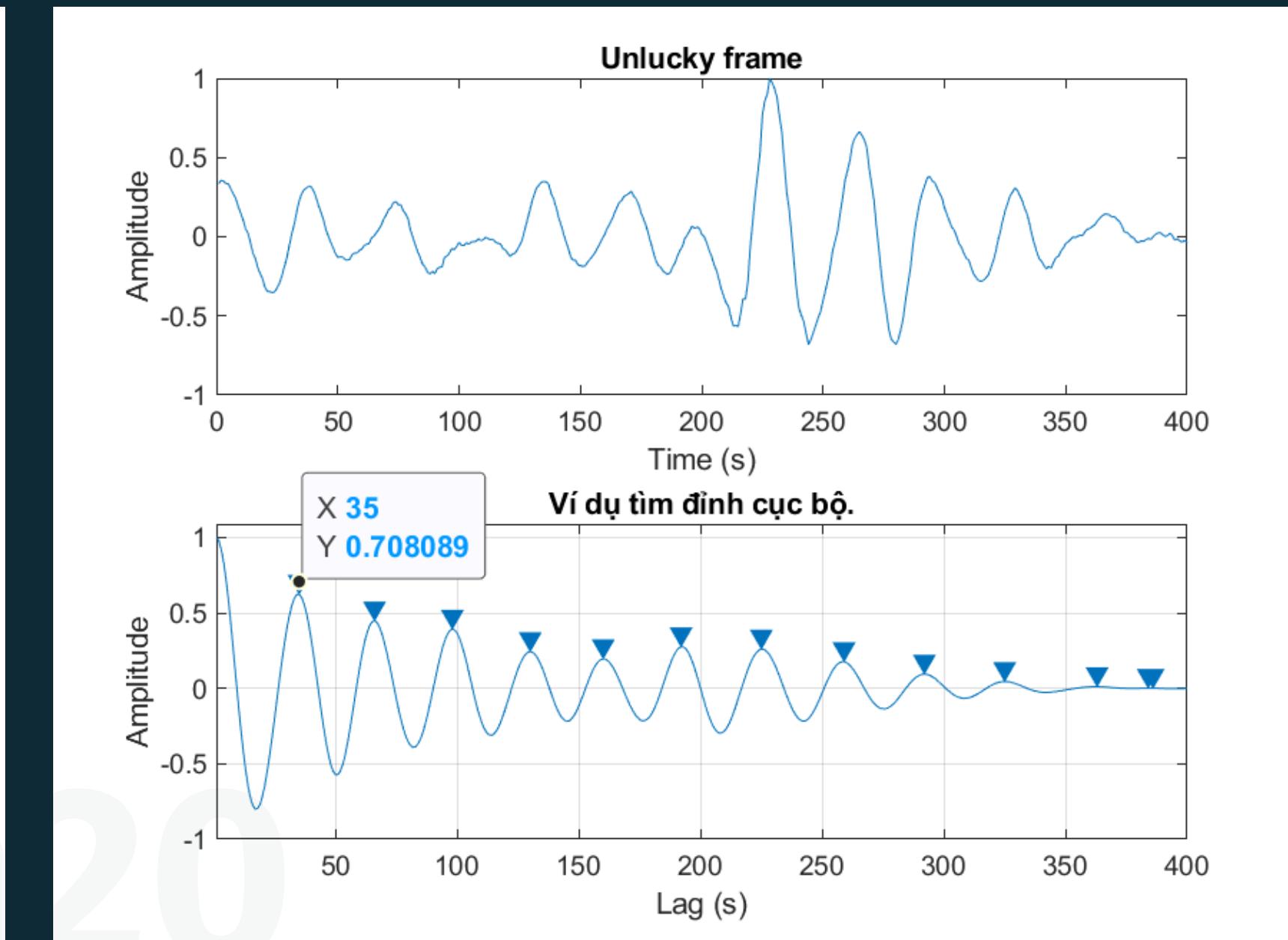
% Kiểm tra  $ac(i)$  có phải là cực trị hay không.  
% Kiểm tra  $ac(i)$  có lớn hơn  $maxValue$  hay  
% Nếu đúng thì gán  $ac(i)$  cho  $maxValue$   
% và gán  $i$  cho  $id$ .

Hàm tìm đỉnh cục bộ.

## 8. Tìm đỉnh cục bộ.

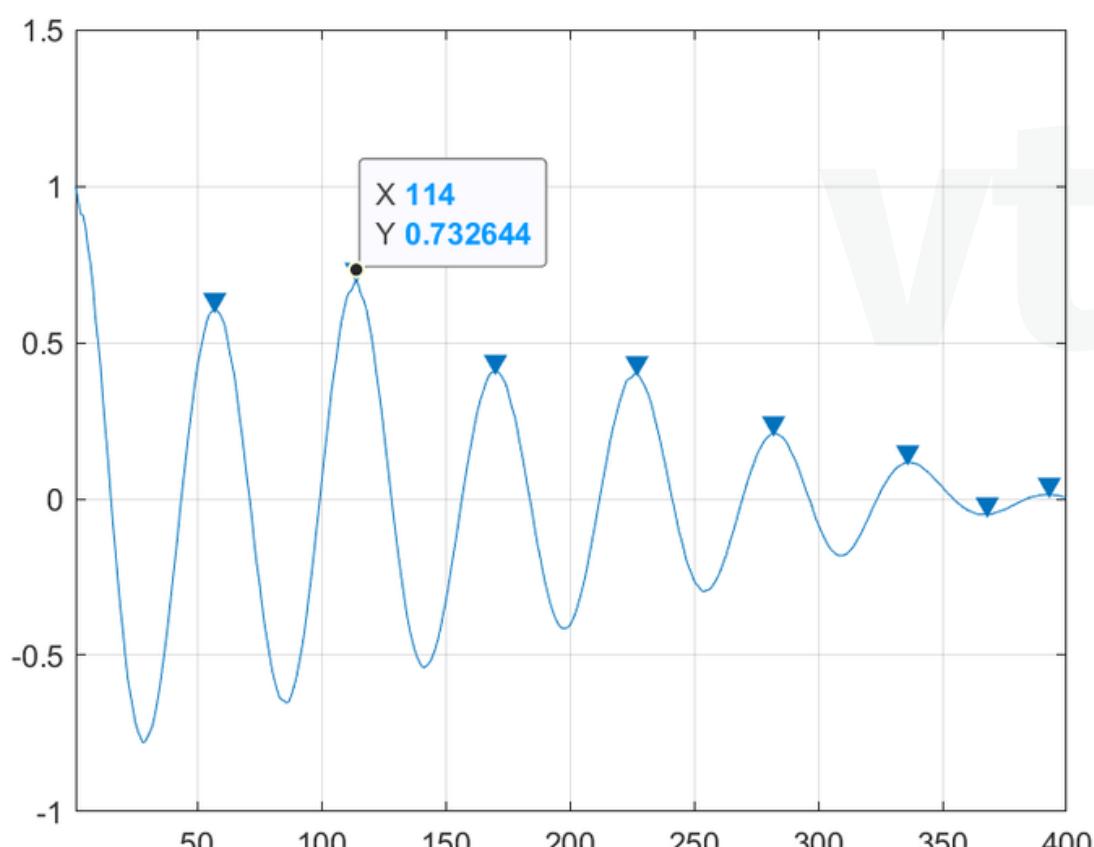


Lucky frame.



Unlucky frame.

## 9. Tìm T0 và F0.



```
1 function value_F0 = TimF0(id_Peak, fs, time_frame)
2 % Hàm TimF0 nhận vào id_Peak: vị trí cực đại có giá trị lớn nhất
3 % fs: tần số của tín hiệu và time của 1 frame để tính ra số mẫu.
4 % Trả ra giá trị F0 tìm được.
5 n = round(time_frame * fs); % Số mẫu của 1 frame
6 T0 = id_Peak*time_frame/n; % Tính chu kỳ T0
7 value_F0 = 1/T0; % F0 sẽ bằng 1/T0
8 end
```

Ví dụ kết quả trả về Hàm TimDinhCucBo

Hàm tìm F0.

- Mỗi frame 0.025s sẽ có 400 mẫu => 114 mẫu xấp xỉ với 0.007s.
- $F0 = 1/0.007$ , xấp xỉ 142.86 hz.

## 10. Tìm ngưỡng.

- Tìm ngưỡng dựa trên giá trị trung bình các cực đại cực cục bộ của tín hiệu đã được lọc ra các khoảng silence.

Kết quả:

- File Phone\_F2: 0.028
- File Phone\_M2: 0.047
- File studio\_F2: 0.034
- File studio\_M2: 0.036

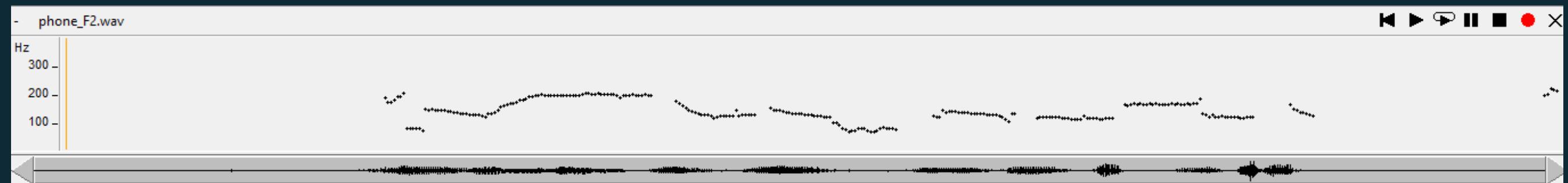
```
1 function Threshold = findThreshold(data)
2 % Hàm findThreshold nhận vào data không chứa silence
3 % Trả ra ngưỡng.
4 [pk, ~] = findpeaks(data);
5 Threshold = round(mean(pk), 3);
6 end
```

Hàm findThreshold.

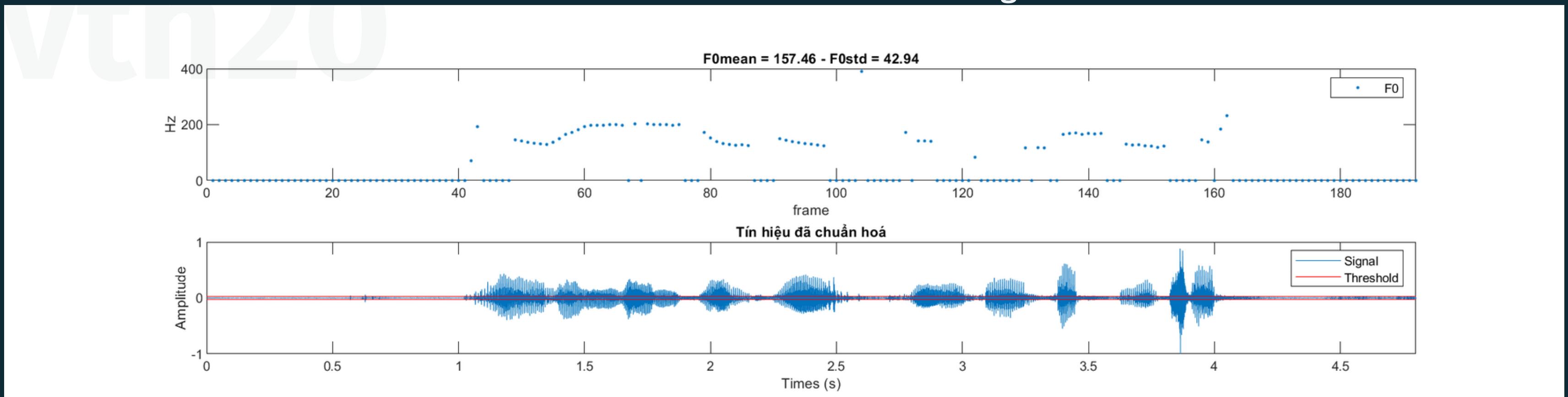
## 11. So sánh kết quả.

F0mean: 145

F0std: 33.7



Pitch Contour trong wavesurfer.



F0mean: 157.46

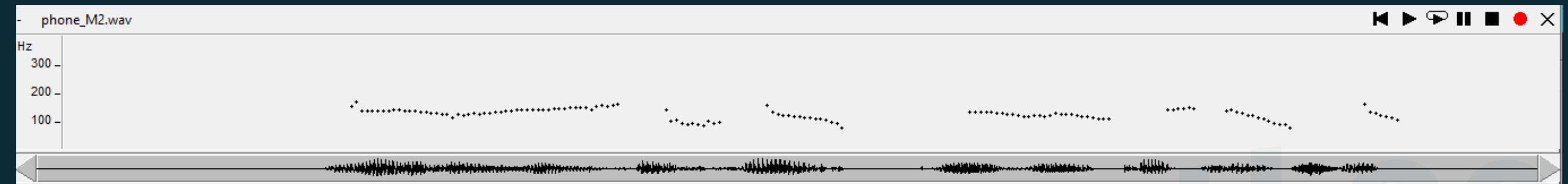
F0std: 42.94

Kết quả file Phone\_F2

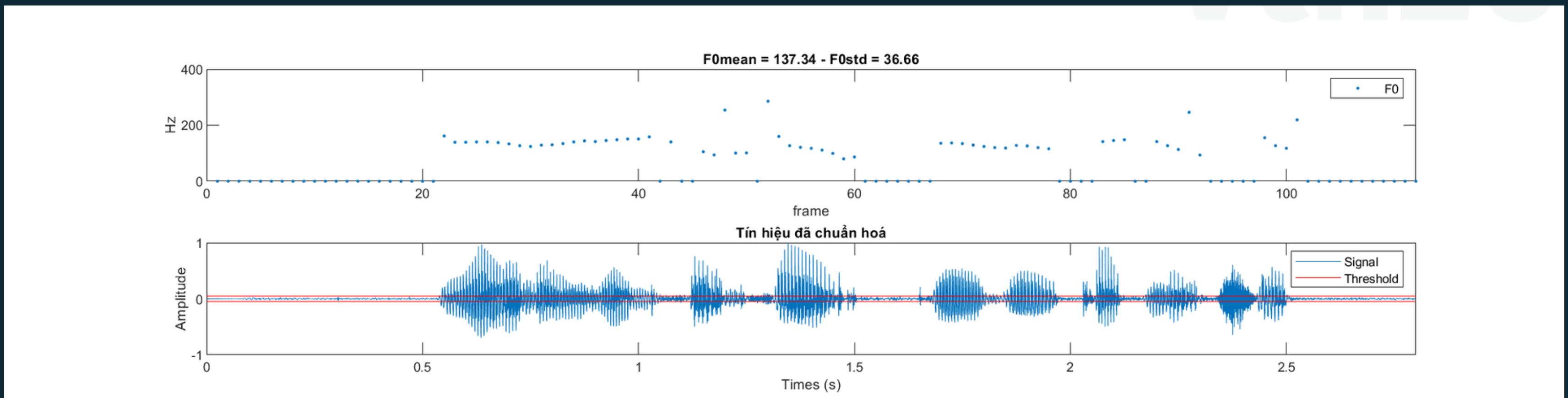
## 11. So sánh kết quả.

F0mean: 129

F0std: 18.6



Pitch Contour trong wavesurfer.



F0mean: 137.34

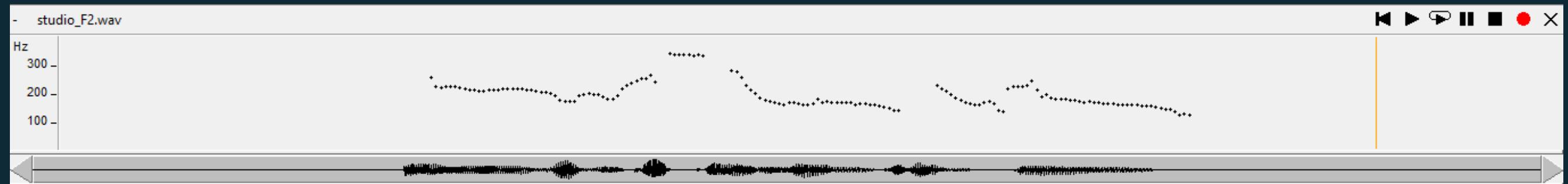
F0std: 36.66

Kết quả file Phone\_M2

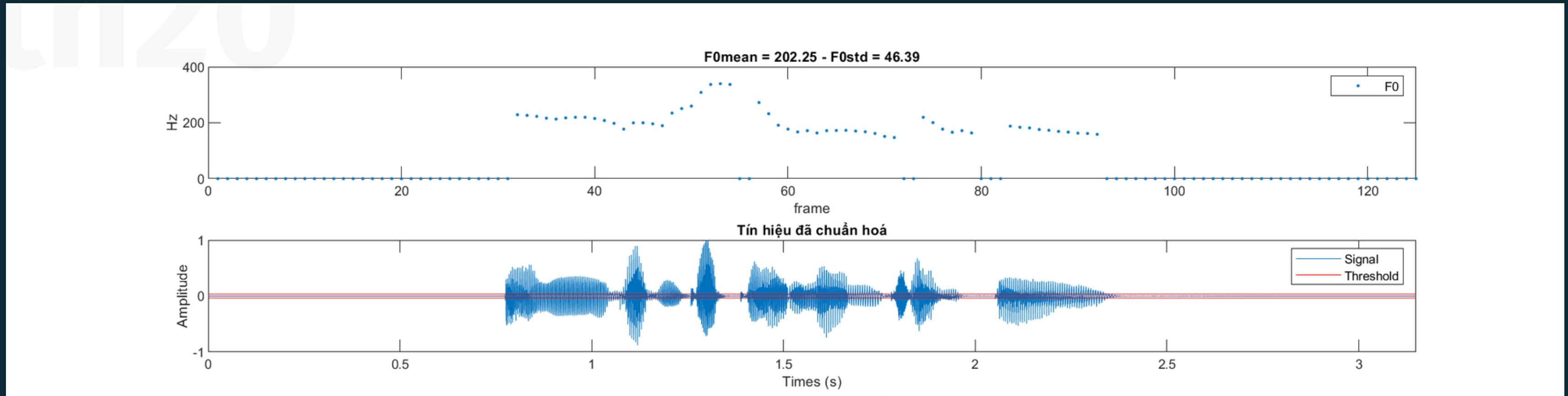
# 11. So sánh kết quả.

F0mean: 200

F0std: 46.1



Pitch Contour trong wavesurfer.



F0mean: 202.25

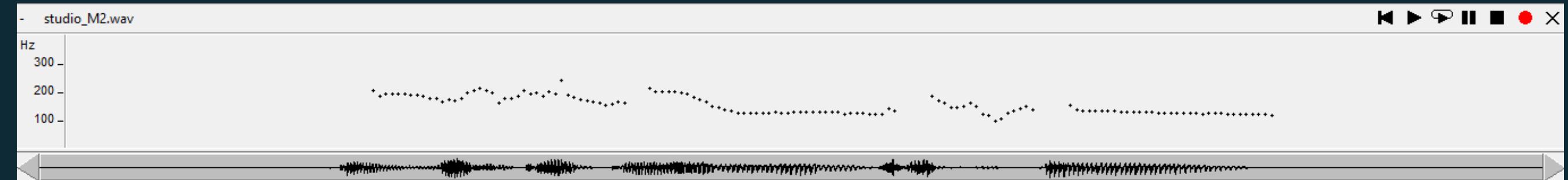
F0std: 46.39

Kết quả file Studio\_F2

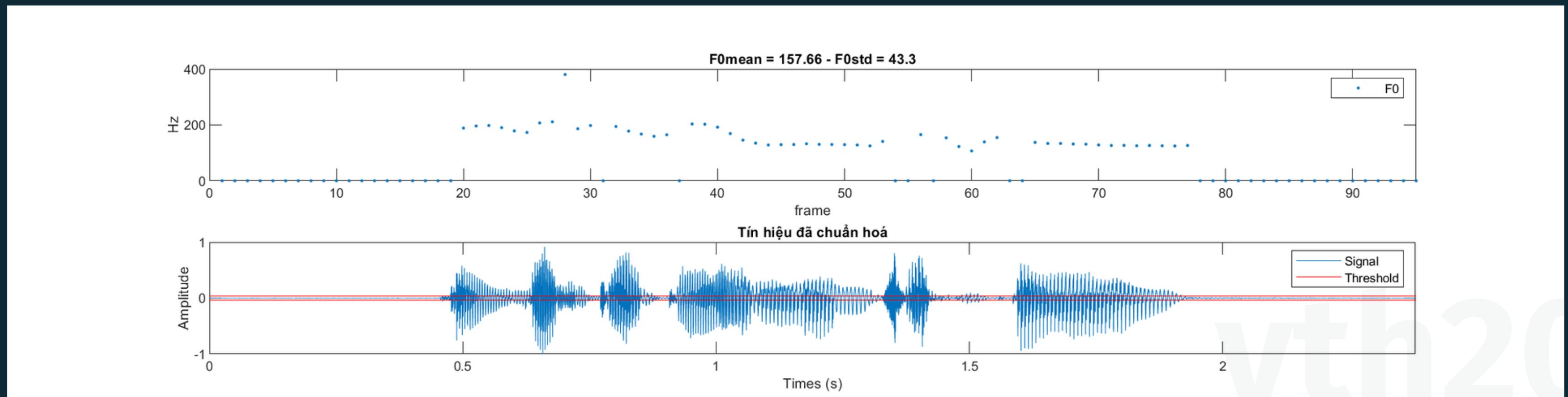
## 11. So sánh kết quả.

F0mean: 155

F0std: 30.8



Pitch Contour trong wavesurfer.



F0mean: 157.66

F0std: 43.3

Kết quả file Studio\_M2

## 12. Bảng thống kê kết quả.

File	F0mean-student	F0std-student	F0mean-teacher	F0std-teacher	Độ lệch F0mean	Độ lệch F0std
Phone F2	157.46	42.94	145	33.7	12.46	9.24
Phone M2	137.34	33.66	129	18.6	8.34	15.05
Studio F2	202.25	46.39	200	46.1	2.25	0.29
Studio M2	157.66	43.3	155	30.8	2.66	12.5
Trung bình					6.42	9.27

## 13. Tài liệu tham khảo.

[I]: Bài giảng môn Xử lý tín hiệu số - Ts. Ninh Khánh Duy.

[II]: Xác định tần số cơ bản của tín hiệu tiếng nói dùng hàm tự tương quan - Luận văn cao học Trần Văn Tâm.

[III]: A method for silence removal and segmentation of speech signals, implemented in Matlab - Theodoros Giannakopoulos.

[IV]: Youtube: Speech and Audio Signal Processing using Matlab - JCBRO Labs.

[V]: CS425 Audio and Speech Processing - Hodgkinson 2012

[...]:

The background features a dark teal color with abstract white shapes. On the left, there are two large, semi-transparent circles: one is light gray and the other is a darker shade of teal. Overlaid on these circles is a black grid pattern consisting of intersecting horizontal and vertical lines.

THANK YOU!

16/11/2021.