מעבדה בעיבוד אותות פזיולוגיים
הנדסה ביורפואית

מגישים:

דן טורצקי
סול אמארה

תאריך:

10.1.2023

תוכן עניינים:

רקע תאורטי

[1]

מערכת הדיבור הינה מערכת מורכבת שמוצאה הינם גלי קול. רבים מאיברי מערכת הדיבור הינם איברים המשמשים את מערכת הנשימה. עובדה זו אינה מפתיעה נוכח העובדה ששתי מערכות אלה מתבססות על הכנסת אוויר מבחוץ והוצאתו מהגוף החוצה. האיברים העיקריים שמעורבים בתהליך הפקת צליל דיבור הינם הריאות, בית הקול (Larynx), לוע (pharynx), חלל הפה והאף. תהליך יצירת הקול הוא כדלהלן – הריאות יוצרות לחץ הגורם לאוויר לצאת מהריאות. אוויר זה עובר דרך מיתרי הקול שהינן רצועות גמישות הנמצאות בתוך בית הקול. המעבר של האוויר דרך מיתרי הקול מרטיט אותן, והתדירות בהן מיתרי הקול רוטטים היא ה – pitch, התדר האופייני של הצליל היוצא. אם מיתרי הקול מופרדים ומוחזקים במקום הצליל שיצא הינו א-קולי, כזה שאין לו תדר בסיסי (למשל הפונמה /f/). משם האוויר ממשיך דרך הלוע, המהווה כתיבת תהודה המספקת הגבר, אך גם תמסורת המוסיפה מאפיינים נוספים לצליל מעבר ל – pitch הבסיסי. משם האוויר ממשיך לחלל הפה ויוצא החוצה. חלל הפה גם לו מאפייני הגברה ותמסורת כשל הלוע. בשונה מהלוע, בחלל הפה חלק מהמאפיינים נקבעים באופן רצוני\לא רצוני ע"י איברים בהם יש לנו שליטה, למשל מיקום הלשון, השפתיים, השיניים כולם משנים את תמסורת האות, כלומר את אופי הצליל המופק.

[2]

אות דיבור אנושי נמצא בטווח התדרים $20Hz - 20kHz$. כאשר אנו מקליטים אות, עלינו לדגום בהתאם לתחום התדרים בוא נמצא המידע בוא אנו מעוניינים ובהתאם למגבלות הטכניות. ככל שנדגום בקצב מהיר יותר, כך יהיה בידינו יותר מידע, אך דבר זה דורש דוגם יותר מהיר ויותר זיכרון לשמור את המידע, כלומר לדגום מהר יותר מגדיל עלויות. כמו כן, אם אין לנו צורך במידע שהנמצא בתדרים הגבוהים מתדר מסוים, אין סיבה לדגום תדרים אלה. קצב הקלטה נפוץ כיום הינו $44.1kHz$. כיוון שתדר דגימה זה גדול מפעמיים התדר המקסימלי שאנו מפיקים בדיבור, איכות הקלטה זו טובה מאוד ומתאימה לשימושים יומיומיים של האזנה להקלטה. אם זאת, לא תמיד יהיה לנו צורך בהקלטה בקצב כה גבוה. למשל, התדר הבסיסי בוא אנו מדברים הינו תדר הנמוך כמעט תמיד מ - $400Hz$. אם כך, כדי לזהות בהקלטה את התדר הבסיסי מספיק שנדגום ב - $800Hz$ (בהנחה כי העברנו את האות ב – Lowpass מתאים). בצורה זו נחסוך את כמות המידע שנצטרך לשמור על מנת לזהות את הדרוש לנו. דוגמא נוספת הינה קווי טלפון. בקווי טלפון, מתאמים טכניים/כלכליים האות מועבר בתדירות של עד $4kHz$, כלומר נדגם ב - $8kHz$. איכות זו, כידוע לנו, פחותה מאיכות הצליל בתקשורת רגילה, כלומר יש פגיעה בחלק מהמידע של אות הדיבור הפוגע באיכות השמע. אם זאת, נוכל להשתמש בהקלטה מסוג זה לזיהוי התדר הבסיסי של האות.

כפי שציינו, אות הדיבור מורכב מתחום תדרים די רחב. על אף שרוב האנרגיה נמצאת במאות הבודדות הראשונות של תחום התדרים, האופי השלם המרכיב את אות הדיבור מורכב מהטווח הרחב יותר. הקלטה בתדר נמוך יכולה לגרום לעיוות של האות. אומנם ה – pitch נמצא בתדרים נמוכים, אך בתדרים גבוהים יותר יש מידע של עיצורים ותנועות החשובים להבנת המילים. מעבר לכך, דגימה בתדר נמוך יכולה כמובן לגרום לאליאסינג במידה ולא משתמשים ב – AAF.

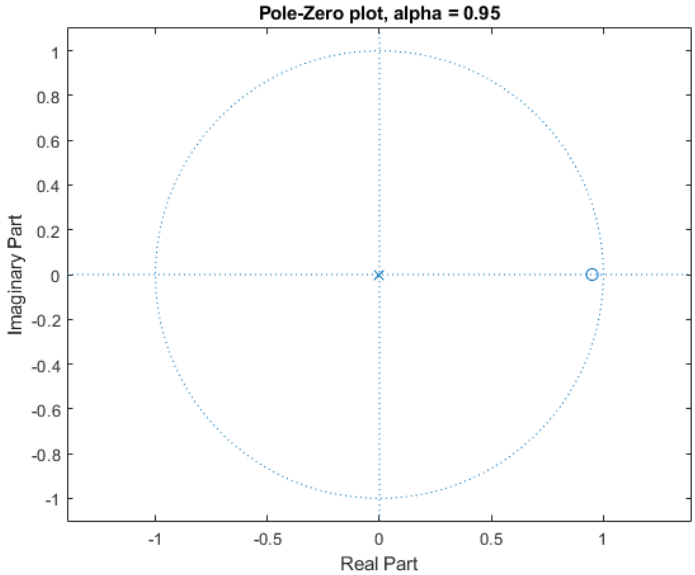
ניסוי 1: עיבוד מקדים לאות דיבור

1.1

- הורדת DC – לאות דיבור אין DC, לכן אם הוא קיים באות הוא תוצר של רעש כלשהו שנכנס למדידה ולכן עלינו להסיר אותו. כמו כן, הורדת ה – DC מקלה על חישוב פיצ'רים מסוימים.
- מסנן מדגיש – באופן טבעי, באות הדיבור, ככל שהתדר עולה האמפ' שלו קטנה. כיוון שבתדרים גבוהים מוחזק מידע רב של הקול, הגברת התדרים הגבוהים משפרת את איכות האות. הביטוי המתמטי של מסנן זה הינו:

$$H(z) = 1 - \alpha z^{-1}, \quad \alpha \approx 1$$

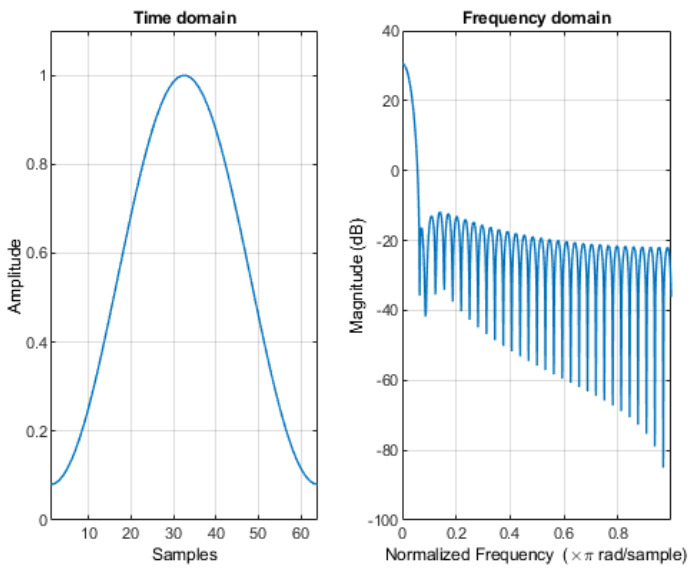
מפת הקטבים ואפסים של מסנן זה עבור בחירת $\alpha = 0.95$ הינה:



איור 1: מפת קטבים ואפסים של מסנן מדגיש עם $\alpha = 0.95$

כפי שניתן לראות, ישנו אפס קרוב ל - $z = 1$, כלומר המסנן משמש כמעביר גבוהים.

- הקטעה למסגרות – חלוקת האות לחלונות קצרים עם חפיפות לשם עיבוד חלקי האות השונים במקטעים קצרים. אורך חלון טיפוסי הינו $\sim 20 - 40msec$, עם חפיפה של $\sim 30 - 75\%$. לרוב אורך החלון הינו קבוע, דבר המאפשר מימוש פשוט של ההקטעה. שיטה זו אינה מתייחסת לתחולה התדרית של האות ולכן ישנן שיטות סגמנטציה אחרות בהן מחלקים את האות למסגרות לא שוות באורכן, כאשר החיתוך נעשה תחת תנאים מסוימים, למשל האופי ספקטרלי של האות לאורך הזמן.
- הכפלה בחלון – חלק מרכזי באנליזה של האות הינו בתדר. אם כך, אנו מבצעים DFT על המסגרות הסופיות בזמן. פעולת ה – DFT מניחה מחזוריות של האות, ואם יש אי רציפות של האות המחזורי (כלומר תחילת וסוף האות עליו מבצעים DFT עם הפרש אמפ' גדול) דבר זה גורם לאפקטים של מריחה וזליגה. כדי להפחית תופעות אלה מכפילים את הפריימים בפונקציות חלון המפחיתות את האמפ' בקצוות האות כדי. פונקציית חלון בשימוש נפוץ הינה חלון Hamming:



איור 2: חלון Hamming באורך 64 דגימות

כפי שניתן לראות, החלון בזמן מפחית את האמפ' בקצוות החלון. בתדר, בפועל מבוצעת קונבולוציה של האות המקורי עם התמרת החלון. ניתן לראות כי ישנו פיק צר באפס והנחתה בתדרים גבוהים יותר, מה שמפחית במידה מסוימת את המריחה והזליגה בהתאמה.

.1.2

פונקציית ה – Pre-processing:

```
function
[ProcessedSig,FramedSig]=PreProcess(Signal,Fs,alpha,WindowLength,Overlap)

%% This function performs standart pre-processing of a speech signal

%% inputs:
% Signal - the speech signal we preform the pre-processing on
% Fs - sampling rate - Hz
% alpha - factor for pre-emphasis filter
% WindowLength - length of segmentation windows (seconds)
% Overlap - precentage of overlap between adjacent frames

%% Outputs:
% ProcessedSig - the pre-processed signal. A vector with the length of the
% original signal, before the windowing phase of the pre-processing.
% FramedSig - a matrix containing the pre-processed signal, where each raw
% is a frame

%%

% DC removal
Signal_noDC = Signal - mean(Signal);

% pre-emphasis filter:
```

```
ProcessedSig = filter([1 -alpha],[1 0],Signal_noDC);

% framing and multiplying in a Hamming window using enframe function
L = WindowLength*Fs;  %window lenth in samples
Inc = L*((100-Overlap)/100);  % incramente between samples
FramedSig = enframe(ProcessedSig,hamming(L),Inc);  %frame matrix
```

ניסוי 2: סגמנטציה אוטומטית

.2.1

שיטת סגמנטציה זו עוסקת בחילוק האות לסגמנטים אשר לכל סגמנט יש אופי תדרי שונה. חילוק האות למקטעים אלה יכול לאפשר זיהוי של אירועים משמעותיים באות. בקונטקסט של אות שמע נרצה להשתמש בשיטה זו להפרדה בין מקטעים קוליים\לא קוליים – זיהוי פונמות ומקטעי שקט. גישת הסגמנטציה הינה להשתמש בשני חלונות – חלון רפרנס וחלון מבחן. חלונות אלה הינם מקטעים מתוך האות. חלון הרפרנס נבחר כחלון ראשוני אותו נבחן אל מול חלון המבחן.

ההשוואה תעשה ע״י מדד כלשהו להבדל בין החלונות, המסומן במקרים רבים כ - $\Delta(n)$. כאשר מדד השונות יעבור סף מסוים למשך כמות דגימות מספיק גדולה אותה נגדיר (כי לעיתים יהיה שינוי רגעי גדול שנובע מהפרעה כלשהי ולא מייצג שינוי אמיתי של אופי האות לאורך זמן), נוכל לומר כי מצאנו סגמנטים השונים אחד מהשני. במצב זה חלון המקטע בין תחילת חלון הרפרנס עד תחילת חלון הטסט יוגדר כסגמנט, חלון הטסט יהפוך להיות חלון הרפרנס החדש, ונקבע חלון טסט חדש אותו נבחן אל מול חלון הרפרנס החדש. בצורה זו נבצע חלוקה של האות לסגמנטים.
ישנן שתי גישות מקובלות לבחירת החלונות. גישה אחת הינה חלון רפרנס קבוע וחלון טסט קבוע. היתרון של שיטה זו הינה שהיא פשוטה יותר ליישום וששיטה זו פחות רגישה לרעשים. הגישה השנייה הינה גישה בה חלון הרפרנס גדל עד לחיתוך הסגמנט. מצד אחד, כיוון שהחלון הרפרנס גדל, הוא מכיל יותר דגימות המכילות יותר מידע וניתוח ספקטרלי הופך לאפקטיבי יותר. מצד שני, שיטה זו יותר מורכבת ליישום והינה יותר רגישה לרעשים מפאת הגדלת כמות הדגימות.

מדד שעושה שכל להשתמש בו הוא הפרש האנרגיות של חלון הרפרנס וחלון הטסט. עבור אותות סטציונרים, ניתן לשערך את ספקטרום האות כך:

$$S_x\left(e^{j\omega},n\right)=\sum_{k=-\infty}^{\infty} r_x(k,n)e^{-j\omega k}$$

כאשר $r_x(k,n)$ הינה פונקציית האוטו-קורלציה של האות ביחס לדגימה ה $-n$ית. נוסחה זו נקראת פריודוגרמה רצה. את פונקציית האוטו-קורלציה ביחס לדגימה ה $-n$ית משערכים ע״י:

$$\hat{r}_x(k,n)=\begin{cases}\sum_{l=0}^{N-1-k} x(l+n+k)x(l+n), & k=0,\dots,N-1\\ 0, & k=N,N+1\dots\end{cases}$$

לבסוף, המדד לשגיאה הספטקרלית מחושב ע״י הפרשי האנגריות:

$$\Delta(n)=\frac{1}{2\pi}\int_{-\pi}^{\pi}\left(S_x\left(e^{j\omega},n\right)-S_x\left(e^{j\omega},0\right)\right)^2 d\omega$$

בעיה בשימוש במדד זה הינה שכאשר ישנה גדילה באנרגיה בפקטור k מדד השגיאה יהיה יחסי ל - $(k-1)^2$, וכאשר תהיה ירידה באנרגיה בפקטור k השגיאה תהיה יחסית ל - $\left(\frac{1}{k}-1\right)^2$, כלומר אין סימטריה במדד השגיאה ביחס להאם האנרגיה גדלה או קטנה באותו פקטור, דבר שהינו בעייתי. כדי לתקן בעיה זו נציג מדד שגיאה אלטרנטיבי:

$$\Delta_1(n)=\frac{\frac{1}{2\pi}\int_{-\pi}^{\pi}\left(S_x\left(e^{j\omega},n\right)-S_x\left(e^{j\omega},0\right)\right)^2 d\omega}{\frac{1}{4\pi^2}\int_{-\pi}^{\pi}S_x(e^{j\omega},n)d\omega\int_{-\pi}^{\pi}S_x(e^{j\omega},0)d\omega}$$

נוסחה מנורמלת זו מבטיחה כי הפעם השגיאה סימטרית ביחס לגדילה\הקטנה של האנרגיה.
באמצעות נוסחת פרסבל והשיערוכים שהצגנו מקודם, השערוך של מדד זה מתוך הדגימות נתון ע"י:

$$\Delta_1(n) = \frac{\sum_{k=-\infty}^{\infty}\left(r_x(k,n) - r_x(k,0)\right)^2}{r_x(0,n)r_x(0,0)}$$

כאשר $r_x(k,n)$ משוערך ע"י $\hat{r}_x(k,n)$ כפי שהוגדר מקודם.

.2.2

פונקציית זיהוי סגמנטי הדיבור:

```
function Idx = FindWordIdx(FramedSig,Fs,WindowLength,Overlap)
%% This function finds areas of the speech signal where there is an actual speech (a
word or a part of a word)

%% inputs:
% FramedSig - A matrix where each raw contains a segment of the signal
% Fs - sampling rate, Hz
% WindowLength - length of each segment (seconds)
% Overlap - precentage of overlap between adjacent frames [0-100]

%% outputs:
% Idx - a MX2 matrix where each raw contains start and end indicies of
% a segment where a word is detected

%%
th = 0.01; %threshold for word recognition
num_frames = size(FramedSig,1); % # of frames
L = WindowLength*Fs; % window lenth in samples
Inc = L*((100-Overlap)/100); % incramente between samples

Idx = zeros(2,num_frames); % intialization of Idx
j=1; % index to iterate on Idx raws

% iterating on frames, if a frame's energy goes other the threshold we will
% find the indicies of its start and end in the not segmented pre-processed
% signal
for i = 1:num_frames % iterating on segments
   Power = sum(FramedSig(i,:).^2); % the segment's power
   if Power > th % if we go above the threshold
      Idx(j,:) = [1+Inc*(i-1), L+Inc*(i-1)];
   end
end
Idx(Idx == 0) = []; %deleting empty raws
end
```

2.3. הפונקציה שכתבנו לביצוע הסגמנטציה:

```
function [seg_ind,delta]=segmentation(signal,winlen,eta,dt,Fs,Idx)

%% This function implements segmentation of speech signal

%% inputs:
% signal - the signal we want to segment
% winlen - length of teset and reference windows (seconds)
% eta - threshold for spectral error measure (Delta1 measure)
% dt - minimum time above threshold 'eta' (seconds)
% Fs - sampling rate (Hz)
% Idx - start & end indicies of the word

%% outputs:
% seg_ind - index of the begining of each segment
% delta - spectral error measure (delta1)

%%

% Deleting the quite parts of the signal:
% removing overlaping indicies from Idx
Idx_new = zeros(size(Idx));
Idx_new(1,:) = Idx(1,:);
j =1; %index of iterations
for i=2:length(Idx)
   % if there is overlap we will combine the two overlapping raws into one
   if Idx(i,1)<=Idx(i-1,2)
      Idx_new(j,2) = Idx(i,2);
   else
      j = j + 1;
      Idx_new(j,:) = Idx(i,:);
   end
end
Idx_new(Idx_new == 0) = []; %deleting raws of zero


% the number of samples in each speech frame
num_frm_smp = (Idx_new(:,2)-Idx_new(:,1))+1;
% length of all speech frames combined
len_loud_signal = sum(num_frm_smp);
% loud_signal will contain only the speech parts of the original signal
loud_signal = zeros(1,len_loud_signal);
j=1;

for i=1:(size(Idx_new,1))
   loud_signal(j:(j+num_frm_smp(i)-1)) = signal(Idx_new(i,1):Idx_new(i,2));
   j = j + num_frm_smp(i) ;
end

% now we implament segmentation on loud_signal using the two functions
% defined at the bottom of this funciton
```

```
L = winlen*Fs; % test and ref windows length in smaples
dt_len = ceil(dt*Fs); %length in samples of dt
count = 0; %counter for dt
seg_ind(1)=1;
j=2; % index to iterate on seg_ind
n1 = 0; % index for reference window
delta = zeros(1,length(loud_signal)-L); %initialization of delta
for n=1:length(delta)
   try
      delta(n) = delta1(loud_signal,n,n1,L); %spectral error measurement
      if delta(n) > eta                  % if error > eta
         count = count + 1;              % counting how many times error > eta
         if count>= dt_len               % if we passed the minimum time of being over eta
we mark a segment
            seg_ind(j) = n;              % seg_ind
            n1 = n;
            j = j+1;
         end
      else
         count = 0;
      end
   catch
      warning(['problem in ', num2str(n), 'iteration'])
   end
end
delta = [delta zeros(1,length(loud_signal)-length(delta))];
end
```

בתוך פונקציה זו השתמשנו בפונקציות נוספות שכתבנו למימוש הפונקציה. פונקציה לחישוב השגיאה הספקטרלית:

```
function err = delta1(signal,n,n1,L)

% This function caluclates delta_1 function as defined in q 2.1
% ------------------------------------------------------------
% inputs:
% signal - signal we use to calculate the spectral error
% n - offset of test window (in samples)
% n1 - offset of reference window (in samples)
% L - length of test and reference windows (in smaples)
% ------------------------------------------------------------
% outputs:
% err - spectral error measurement delta_1(n)
% ------------------------------------------------------------
%%

nume = 0; % initialization of the error numerator of the equation
for k = 0:(L-1)
   nume = nume + (autocor(signal,k,n,L)-autocor(signal,k,n1,L))^2;
end
```

```
err = nume/(autocor(signal,0,n,L)*autocor(signal,0,n1,L));
```

בתוך פונקציה זו השתמשנו בפונקציה נוספת שכתבנו עבור שיערוך אוטו-קורלציה ממדגם סופי:

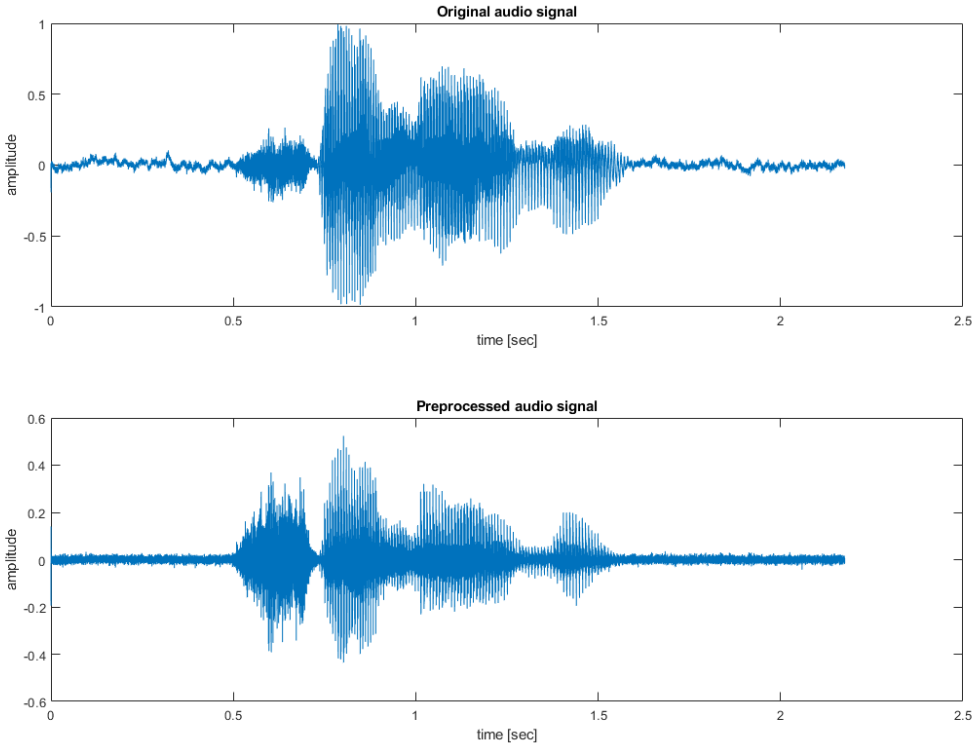
```
function out = autocor(signal,k,n,L)

% This function estimates autocorrelation from specific part of a signal
% inputs:
% signal - signal to autocorelate on
% L - window length we use to estimate the autocrelation
% k - autocorelation is between part of the signal around n and k off-set of the signal
% n - offsetting the window we use for estimation
% ouputs:
% out - the estimation of the autocorrelation
%%

if n<0 || k<0 || k>L-1
  disp(['either n = ', num2str(n) ' or k = ', num2str(k), ' are not possible choices for these
variables'])
  return;
end
out = 1/L*sum(signal((n+k+1):(n+L)).*signal((n+1):(L-k+n)));
```

.2.4

פונמה הינה היחידה הקטנה ביותר של צליל שיכולה להפריד בין שתי מילים שונות. מילה מורכבת לכל הפחות מהברה אחת והברה מורכבת לכל הפחות מפונמה אחת. ניתן להבדיל בין שני סוגים של פונמות. סוג אחד הינו פונמות קוליות, המאופיינות ע"י תדר בסיסי הנוצר מהרטטת מיתרי הקול בזמן שהאוויר היוצא מהריאות עובר דרכן. המודל המתמטי של הפונמות הינו של רכבת הלמים שתדירותה היא של התדר הבסיסי העוברת דרך מסנן לינארי, כאשר לאופי המסנן יש קשר לתצורה של חלל הפה והלוע (המבנה האנטומי הייחודי של הדובר וכן שינויים שהדובר משנה מבחירתו כמו מיקום הלשון, היצרות השפתיים ועוד). כמו כן, פונמות קוליות מאופיינות ע"י אנרגיה גבוהה ו $ZCR-$ (קצב חציית האפס) נמוך יחסית לפונמות א-קוליות. כאשר מיתרי הקול מופרדים ומוחזקים במקום, כך שבעת מעבר של אוויר בבית הקול האוויר אינו נחסם אך גם לא מרטיט את המיתרים נוצרת פונמה א-קולית. המודל המתמטי של פונמה א-קולית הינו של מעבר רעש לבן במסנן לינארי, כאשר המסנן הלינארי מאופיין כפי שתיארנו קודם ע"י הלוע וחלל הפה. בפונמות א-קוליות מרבית האנרגיה נמצאת בתדרים גבוהים והאנרגיה הכוללת נמוכה ביחס לפונמות הקוליות. כמו כן, ה $ZCR-$ שלהן גבוה יחסית לפונמות קוליות.

כעת, נבדוק את הפונקציה שכתבנו בסעיף 2.3 על ההקלטה $'shalom\_exaple.wav'$

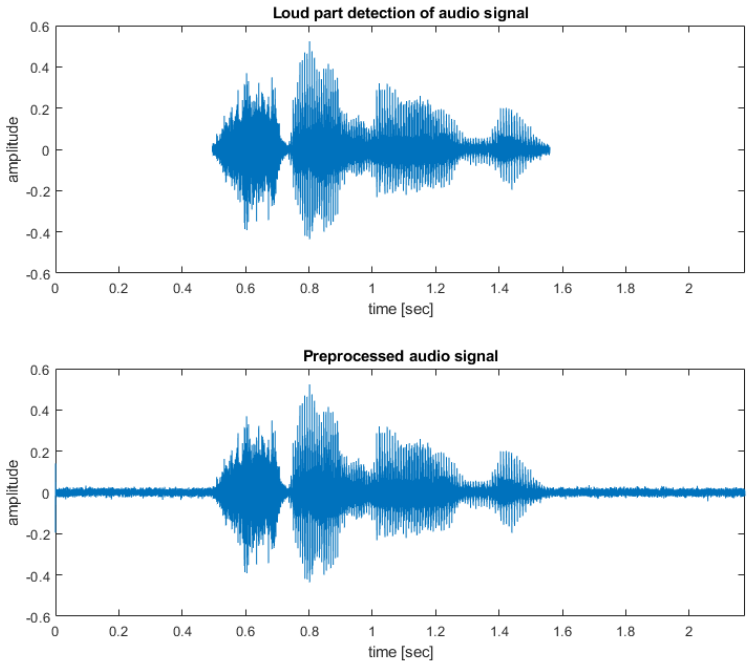
תחילה נציג את האות לפני ואחרי *Pre-processing*:



איור 3 : גרף אמפ' האות לדגומא 'shalom_example' כפונקציה של הזמן לפני ואחרי עיבוד מקדים

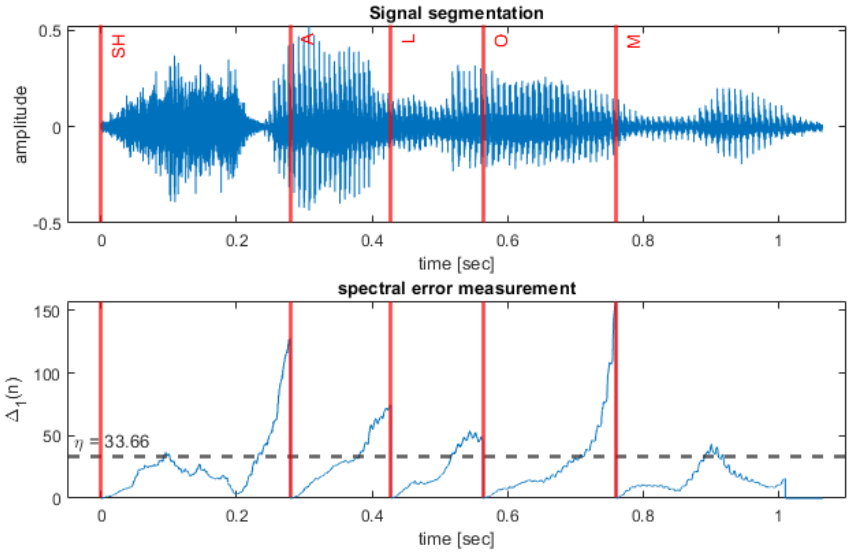
ניתן לראות כי האות שהתקבל נקי יותר ונוח יותר לניתוח.

כעת, נראה את האות לאחר הסרת המקטעים השקטים באמצעות הפונקציה FindWordIdx:



איור 4 : אות השמע לפני ואחרי הסרת המקטעים השקטים

ניתן לראות כי המקטעים השקטים הוסרו בהצלחה.
לבסוף, נבצע סגמנטציה על האות לאחר עיבוד מקדים והסרת החלקים השקטים:



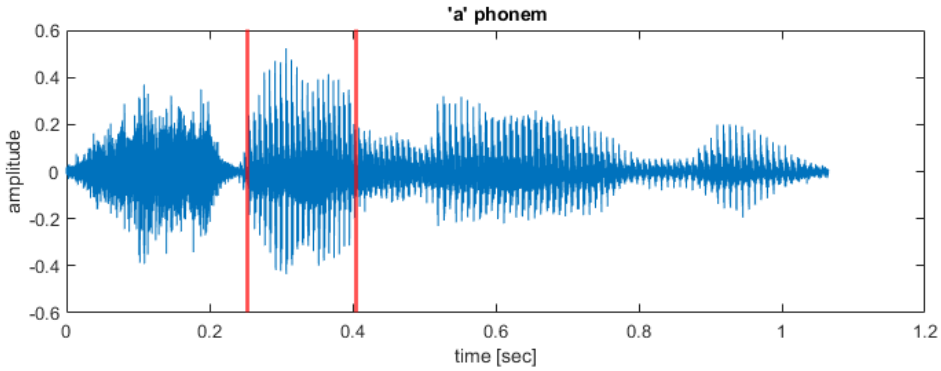
איור 5: סגמנטציית האות. למעלה – אות השמע עם חלוקת הסגמנטים מסומנת, למטה – מדד השגיאה כפונקציה של זמן

ניתן לראות כי ישנו דילאי קטן בזיהוי הסגמנטים פרט לפונמה – M שזהותה מעט מוקדם מידי, אך ככל הסגמנטציה חילקה את האות למס' הפונמות הדרוש במיקום נכונים יחסית.

ניסוי 3: שערוך ספקטרלי ומציאת פורמנטות

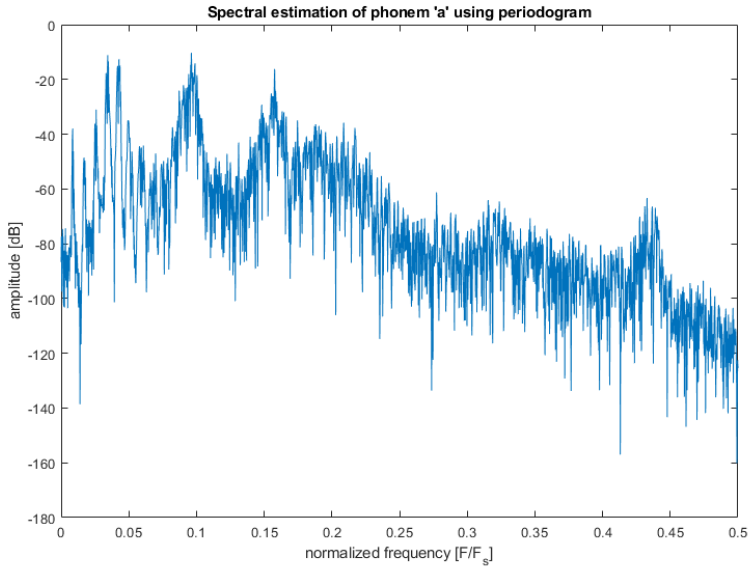
3.1

המקטע של פונמה 'a' שחתכנו מתוך ההקלטה 'shalom':



איור 6: הפונמה 'a' מתוך ההקלטה 'shalom'

שערוך ספקטרלי של הפונמה מתוך פריודוגרמה:



איור 7: שיערוך ספקטרלי של הפונמה 'a' באמצעות פריודוגרמה

ניתן לראות כי השערוך שהתקבל מורעש, אך ניתן לזהות תדרים דומיננטיים ומבנה כללי של הספקטרום.

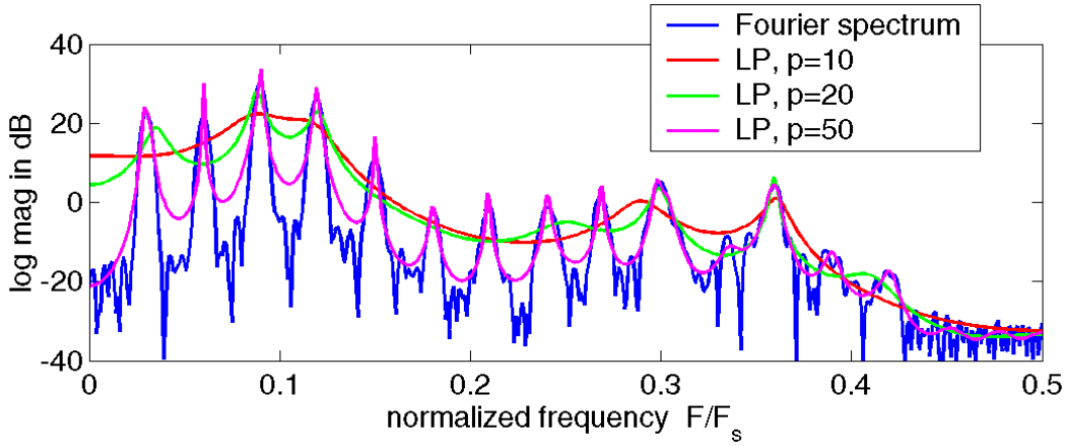
3.2

$LPC - Linear\ Predictive\ Coding$ הינו מודל של שערוך ספקטרום מתוך קומבינציה לינארית של דגימות קודמות. המשוואה המייצגת את המודל הינה:

$$x(n) = \sum_{k=1}^{p} a_k x(n-k) + e(n)$$

כאשר $x(n)$ הינה הדגימה הנוכחית, $\{a_k\ , k = 1, \dots, p\}$ הינם המקדמים אותם אנו מחפשים (p הינו סדר המודל) ו - $e(n)$ הנה שגיאת החיזוי. את המקדמים מוצאים ע"י מינימיזציה של ה - MSE בין האות לחיזוי.

דוגמה לשימוש במודל זה על אות דיבור:

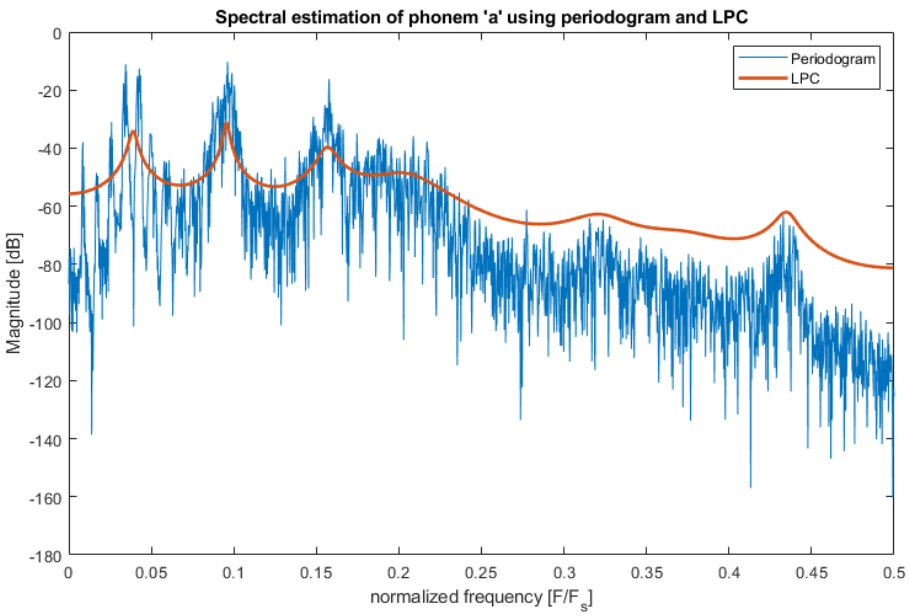


איור 8: ספקטרום של אות דיבור ושערוך באמצעות LPC מסדרים שונים

ניתן לראות כי ככל שנשתמש ביותר דגימות לשערוך נקבל תוצאות הקרובות יותר לספקטרום האמיתי, אך הגדלת סדר המודל דורשת יותר שימוש ביותר דגימות המגדיל את עלות החישוב. כלל אצבע לבחירת סדר המודל עבור אותות דיבור הינו:

Fs/1000 קטבים עבור מערכת הקול (לוע, חלל הפה...) ועוד 6-8 קטבים עבור אנרגיית קרינה (האנרגיה המוחזרת למערכת הקול בתדרי הפורמנטות) ו – glottal pulse.

3.3.



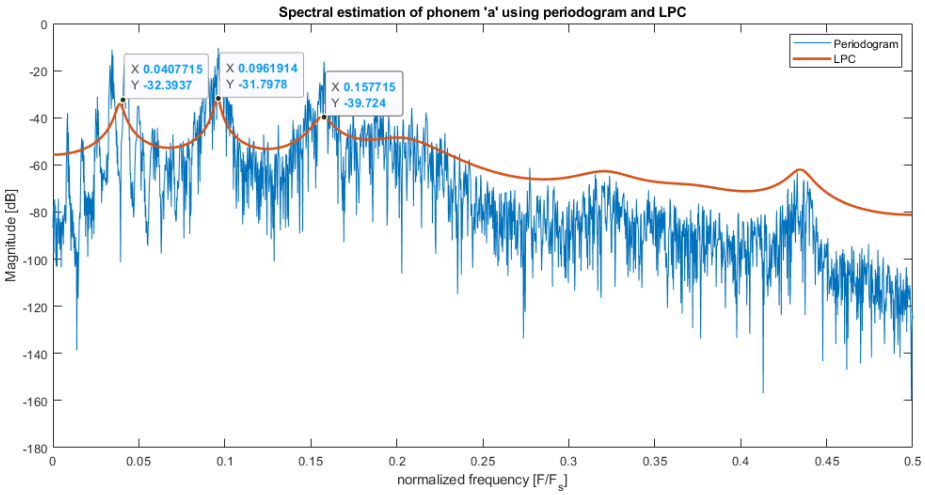
איור 9 : שערוך הספקטרום של פונמה 'a' באמצעות פריודוגרמה ו – LPC

ניתן לראות כי הפריודוגרמה נותנת שערוך יותר מורעש של הספקטרום, אך מכילה יותר מידע היות והיא משוערכת מכל האות. השערוך באמצעות LPC חלק יותר אך פחות מדויק. ניתן לראות כי ה – LPC נותן לנו מיקומם של פיקים בספקטרום, המייצגים פורמנטות. מודל ה – LPC מאפשר שערוך באמצעות הרבה פחות חישובים משמעותית מחישוב ע"י פריודוגרמה, וכתלות במידע לו אנו זקוקים שימוש ב – LPC יכול להיות עדיף בהינתן הפשטות של המודל. לעומת פריודוגרמה, שהינה נוסחה סגורה, במודל ה – LPC ניתן לבחור את סדר המודל כך שנוכל לחלץ את המאפיינים של הספקטרום בהם אנו מעוניינים. ככל שנגדיל את סדר המודל כך הוא יכיל יותר מידע על הספקטרום, אך יהיה יותר יקר חישובית ויהיה רגיש יותר לרעשים.

3.4

פורמנטות הינן תדרים של אות הדיבור שלהם חלקי מרכזי באופי הצליל. הפורמנטות יופיעו בפיקים של הספקטרום. בעוד שהתדר הבסיסי F_0 הינו תדר ה – pitch, תדרי הפורמנטות, ובעיקר שלושת תדרי הפורמנטות הראשונים - F_1, F_2, F_3 מאפיינים במידה רבה את מאפייני התנועה (vowel) שהינן פונמות קוליות, ולרוב יספיקו כדי להבדיל בין תנועות שונות.

מהתבוננות באיור 9 וסימון שלושת הפיקים הראשונים :

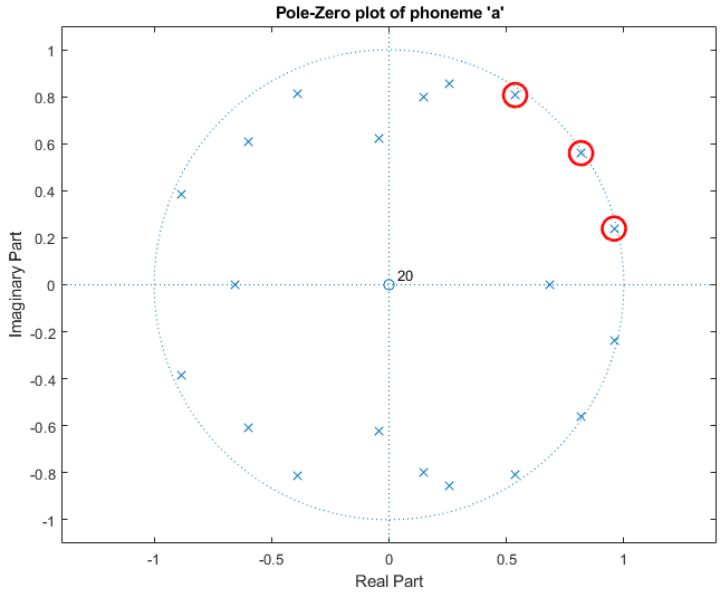


איור 10: שערוך הספקטרום של פונמה 'a' באמצעות פריודוגרמה ו – LPC עם קואורדינטות שלושת הפיקים הראשונים מסומנות

ניתן לראות כי שלושת הפורמנטות הראשונות מתקבלות בתדרים המנורמלים - $0.041, 0.096, 0.158$ וכדי להמיר בחזרה לתדירות נכפיל בתדר הדגימה $16kHz$ ונקבל:

$$F_1 = 656Hz, \qquad F_2 = 1536Hz \ , \ F_3 = 2528Hz$$

3.5



איור 11: מפת קטבים ואפסים של פונמה 'a'

שלושת הקטבים המסומנים הינם הקטבים המייצגים את הפרמנטות. מסימטריות סביב הציר הממשי נתייחס בהסבר רק לחלק החיובי של הציר המדומה. z=1 מייצג את התדר אפס, וכאשר מתקדמים נגד כיוון השעון על מעגל היחידה, כל קוטב (ביחס לזווית שלו עם החלק החיובי של הציר הממשי) הוא בעל תדר גדול יותר עד שמגיעים לזווית של π, המתאימה לתדר המקסימלי (שהינו חצי תדר הדגימה). כיוון ששלושת הפורמנטות הראשונות שייכות לשלושת הקטבים בעלי התדר הקטן ביותר נבחר את שלושת הקטבים בעלי הזווית (ביחס לציר הממשי) הקטנה ביותר שאינה אפס ממש.

את התדרים נחשב כך:

$$F = \frac{\theta_p}{\pi} \cdot F_{max} = \frac{\theta_p}{\pi} \cdot \frac{F_s}{2}$$

תדרי הפורמנטות שהתקבלו הינם:

$$F_1 = 622.7Hz \ , \qquad F_2 = 1529Hz, \qquad F_3 = 2505.1Hz$$

נשווה לתוצאות שקיבלנו מהתבוננות בפיקים של פונקציית התמסורת ולספרות [3]:

טבלה 1: השוואה בין מדידות פורמנטות הפונמה 'a' לספרות

	$F_1\ [Hz]$	$F_2\ [Hz]$	$F_3\ [Hz]$
חילוץ מפונקציית התמסורת	656	1536	2528
קטבי מודל LPC	622.7	1529	2505.1
ספרות	710	1100	2540

הערכים שמצאנו בשתי השיטות דומים, הרי שהם נגזרים מאותו מידע (מאותם קטבים) והם דומים עד כדי היכולת שלנו לסמן את הפיק במדויק. בהשוואה לערכי הספרות, נראה כי יש שגיאות מסוימות. עבור הפורמנטה הראשונה השגיאה ביחס לספרות היא של כ – 10%, עבור הפורמנטה השנייה השגיאה של כ – 39%, ועבור השלישית של כ – 0.5%. אומנם, שגיאות לא קטנות, אך ישנה וריאביליות בערכי הפורמנטות בין אנשים שונים, וכן המקור ממנו לקחנו את ערכים אלה הוא אמריקאי כך שייתכן שמבטא הדובר משנה במידה מסוימת את תדרי הפורמנטה אפילו אם מדובר באותה תנועה.

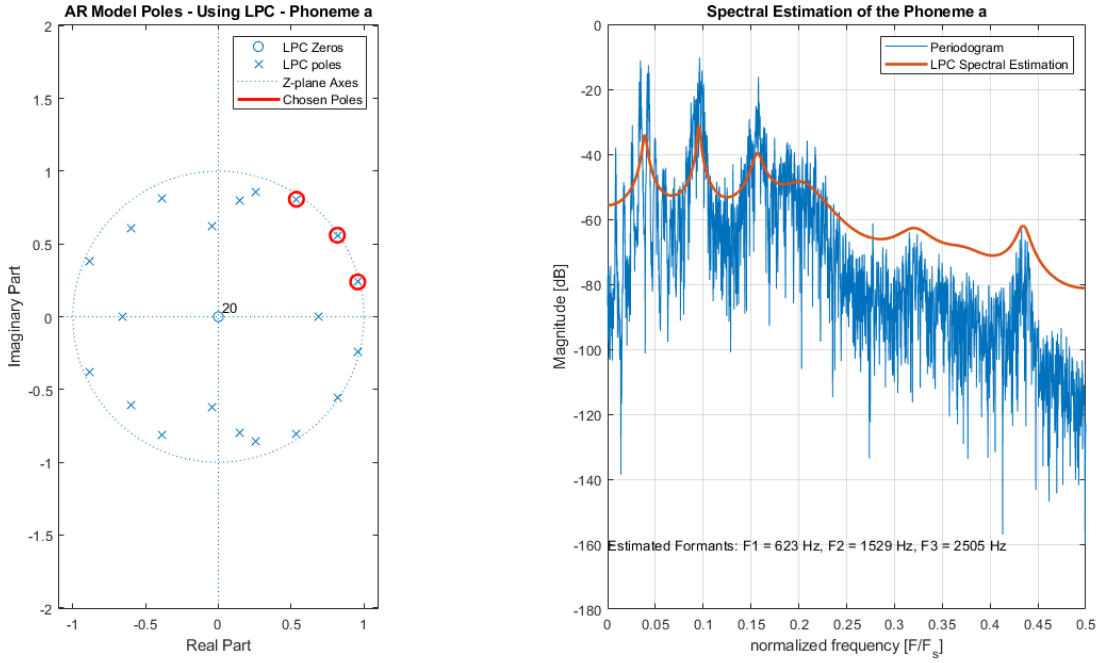
3.6

הפונקציה לשערוך שלושת הפורמנטות הראשונות:

```
function [h1,h2]=estimatePhonemeFormants(PhonemeSig,Fs,phonemeName)
% This function find the first 3 forments of a phoneme, plots a spectral
% estimation using periodogram and LPC, and a Pole-zero map of the LPC
% model
% ----------------------------------------------------
% inputs:
% PhonemeSig - the segment of the phoneme
% Fs - sampling frequency [Hz]
% phonemeName - string of the phoneme's name
% ----------------------------------------------------
% outputs:
% h1,h2 - handles to both graphs described above
% ----------------------------------------------------
%%
[Pxx,w] = periodogram(PhonemeSig); %periodogram
p = Fs/1000 + 4; % LPC model oreder
[a,g] = lpc(PhonemeSig,p); % a = model coeeficients, g = gain
[H,w_LPC] = freqz(g,a,1024); % spectrum of the LPC model

% finding the first 3 formants frequencies from the LPC model
poles = roots(a);
```

```
% thetas = sorted list of the positive angles of the poles
[thetas,ind] = unique(abs(angle(poles)));
thetas(thetas == 0) = []; %if we have pole on the real axis
if length(thetas)<length(ind)
   ind(1) = [];
end
% the 3 smallest thetas represent the 3 Formants
formants = thetas(1:3)/pi*(Fs/2);

% right plot, Spectrum. handle = h1
figure, h1 = subplot(122); plot(w/(2*pi),db(Pxx));
xlabel 'normalized frequency [F/F_s]' , ylabel 'Magnitude [dB]'
title('Spectral estimation of phonem ''a'' using periodogram and LPC')
hold on; plot(w_LPC/(2*pi),db(H),'LineWidth',2)
grid on
legend('Periodogram','LPC Spectral Estimation')
text(0,-160,['Estimated Formants: F1 = ', num2str(round(formants(1))) , ...
    ' Hz, F2 = ', num2str(round(formants(2))), ' Hz, F3 = ', num2str(round(formants(3))), '
Hz']);
title(['Spectral Estimation of the Phoneme ', phonemeName])

hold off

% left plot, Pole map, handle = h2
h2 = subplot(121); [hz,hp,ht] = zplane(g,a);
% drawing circles on the formants
circ = viscircles([real(poles(ind(1:3))), imag((poles(ind(1:3))))] ,0.05,'Color','r');
legend([hz,hp,ht(1),circ],{'LPC Zeros','LPC poles','Z-plane Axes','Chosen Poles'})
title(['AR Model Poles - Using LPC - Phoneme ', phonemeName])
end
```

מימוש הפונקציה על הפונמה *'a'* מתוך ההקלטה לדוגמא:



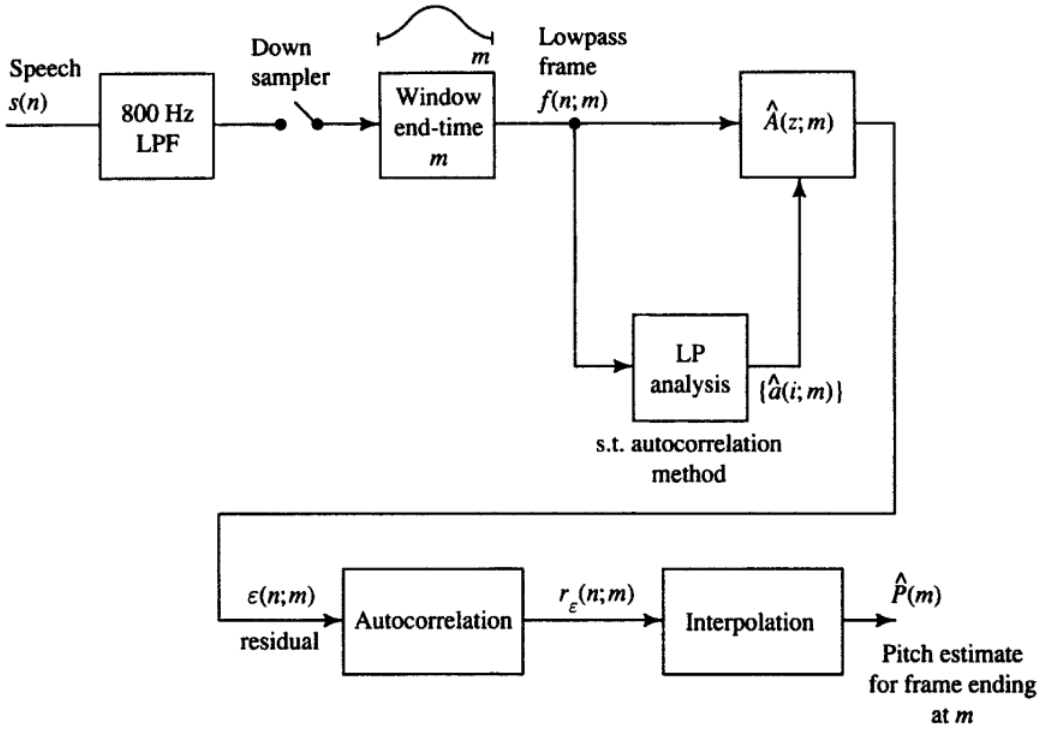
איור 12: מימוש הפונקציה לשערוך פרמנטות על הפונמה 'a'

כפי שניתן לראות, התוצאות תואמות את תוצאות הסעיפים הקודמים.

3.7

Pitch הינו תדר, המסומן לעיתים ב - F_0, הינו התדר הבסיסי של הפונמה והוא התדר בו מיתרי הקול מתנודדים בפונמות קוליות. תדר זה מאפיין האם הצליל ישמע "גבוה" או "נמוך". תחום התדרים של ה - *pitch* באות דיבור ימצא לרוב בין $\sim 50 - 300Hz$, כלומר בטווח התדרים הנמוכים של אות הדיבור.

SIFT – simple inverse filter tracking הינו אלגוריתם לשערוך ה – *Pitch*:



איור 13: דיאגרמה של אלגוריתם SIFT *[4]*

אות דיבור מוקלט לרוב בתדר דגימה של לפחות $8kHz$. כיוון שאנו מעוניינים לשערך $Pitch$ שנמצא לרוב בתדרים הקטנים מ - $400Hz$, אין לנו צורך במידע הנמצא בתדרים גבוהים יותר ועל כן תחילה מעבירים את האות ב - $800Hz\ LPF$, ולאחר מכן ב - $downsampler$. בצורה זו ניתן גם להימנע מטעויות חישוב הקשורות למידע בתדרים גבוהים יותר וגם להפחית את כוח החישוב הדרוש לאלגוריתם. לאחר מכן מבצעים הקטעה לחלונות כי ה $-Pitch$ יכול להשתנות בין פונמות שונות. בתהליך ההקטעה צריך לבחור חלון מספיק גדול כך שיכיל לפחות שני מחזורים של ה- $Pitch$ כיוון שאנחנו מחשבים את ה $-Pitch$ לפי אוטו-קורלציה, כאשר זמן המחזור של ה $-Pitch$ יוגדר ע"י הפיק המשמעותי השני באוטו-קורלציה (לפי סף מסוים). עבור כל חלון, משערכים את מסנן מערכת הקול באמצעות LPC ומכפילים את האות בתמסורת ההופכית. בצורה זו מבטלים את התמסורת של מעבר אות הקול במערכת הקול, ונותרים עם אות שרוב אופיו הוא של התדר האופייני. על אות זה מבצעים אוטו-קורלציה, ומחפשים פיק משמעותי העובר סף מסוים בטווח זמנים המתאים לזמן מחזור של $pitch$ של אות דיבור. [4]

ניסוי 4: מיצויי מאפיינים

4.1

אנרגיה עבור מסגרות אות דיבור מחושבת באמצעות הנוסחה:

$$E_n = \frac{\sum_{m=-\infty}^{\infty}(x[m]\cdot w[m-n])^2}{\sum_{m=-\infty}^{\infty} w^2[m-n]}$$

כאשר w מייצגת את החלון שבו נשתמש, ויש נרמול באנרגיית החלון. פונמות קוליות מתאפיינות באמפליטודה גבוהה ביחס לפונמות א-קוליות ולכן נצפה שעבורן נקבל אנרגיה גבוהה ועבור א-קוליות אנרגיה נמוכה. כאשר נרצה לבצע הפרדה בין סוגי פונמות אלה, ניתן למצוא סף של האנרגיה שייתן את ההפרדה המיטבית. [5]

4.2

הפונקציה שעושה את החישוב עבור האנרגיה של החלון הינה:

```
function EnergySignal=calcNRG(framedSignal)
%input: framedSignal – a matrix of the framed signal, after
preprocessing
%every row is one frame

%output: EnergySignal – a column vector of the energy values
of the signal
num_row=size(framedSignal,2);
window=hamming(L);

num_row=size(framedSignal,1);
for i=1:num_row
    frame=framedSignal(i,:);
    EnergySignal(i)=sum((frame.*window).^2)/sum(window.^2);
end

end
```

4.3

קצב חציית אפס מחושב באמצעות הנוסחה:

$$Z_n = \sum_{m=n}^{n+N-2} \frac{|sign(x[m]) - sign(x[m+1])|}{2} = \frac{1}{N}\sum(x[m]\cdot x[m+1] < 0)$$

כלומר, ספירה של כמות הפעמים בהם יש החלפת סימן. ככל שהתדר גדל כך יהיו יותר חציות אפס ולכן נצפה לקבל ערך גבוהה יותר עבור פונמות א-קוליות.[5]

4.4

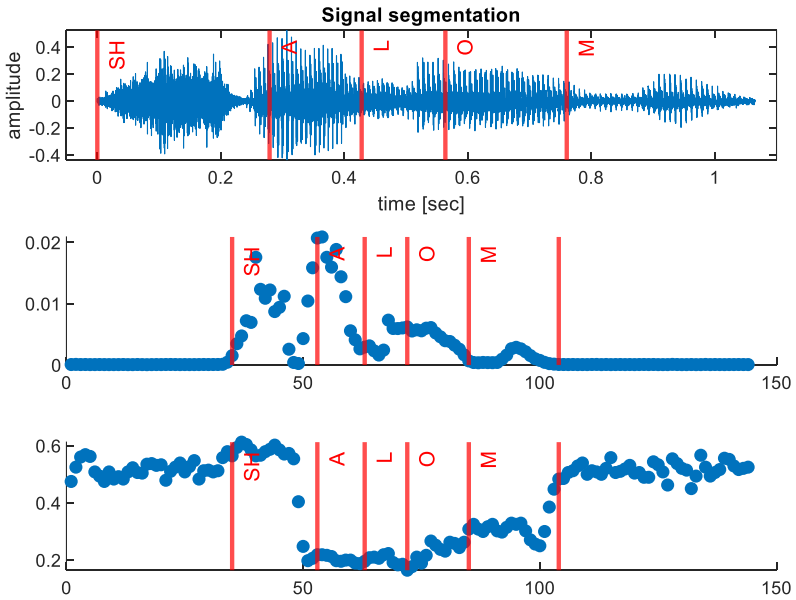
הפונקציה שעושה את החישוב עבור קצב חציית האפס הינה:

```
function ZeroCrossingSignal=calcZCR(framedSignal)
%input: framedSignal - a matrix of the framed signal, after preprocessing
%output: ZeroCrossingSignal - a column vector of the zero-crossing values
of the signal

num_row=size(framedSignal,1);
for i=1:num_row
    frame=framedSignal(i,:);
    frame=frame-mean(frame);
    N=size(frame,1);
    ZeroCrossingSignal(i) = 0;
    for j = 2:N
        if frame(j-1) * frame(j) < 0 % if the sign change it's mean that
this is a zero corssing
            ZeroCrossingSignal(i) = ZeroCrossingSignal(i) + 1;
        end
    end
     ZeroCrossingSignal(i)= ZeroCrossingSignal(i)/N;
end

end
```

תחילה הורדנו את הממוצע מהפריים ולאחר מכן עבור כל פריים ביצענו סכימה של כמות הפעמים בהם יש שינוי סימן. לבסוף, חילקנו בגודל הפריים כפי שמופיע בנוסחה.

4.5

ערכי האנרגיה וקצב חציית אפס שקיבלנו לכל פריים:



איור 14: ערכי האנרגיה וקצב חציית אפס

ציפינו לקבל אנרגיה גבוהה עבור פונמות קוליות ונמוכה עבור פונמות א-קוליות. מהאיור ניתן לראות כי עבור מקטעי שקט האנרגיה שווה לאפס כצפוי. עם זאת, עבור SH , פונמה א-קולית, היינו מצפים לראות אנרגיה נמוכה יותר מהפונמות האחרות דבר שקשה להבחין בו מהאיור. נשער שסתירה זו נובעת מכך שסימוני המעבר בין הפונמות אינם מדויקים כפי שניתן לראות מהגרף הראשון באיור זה. למשל, עבור 'M' ניתן לראות כי המקטע מכיל מקטעי שקט ולכן גם בחישוב האנרגיה יש מקטעים בהם האנרגיה אפס אך הם לא שייכים לפונמה זו.

ציפינו לקבל קצב ZCR גבוה עבור פונמות א-קוליות ונמוך עבור פונמות קוליות. ראשית ניתן לראות כי קצב חציית האפס עבור SH (א-קולית), גבוה מאוד ביחס לשער הפונמות שהן קוליות כפי שציפינו.

4.6

הפונקציה שבנינו:

```
function [FeatsVector,Feat_title]=FeatExt(Phoneme,Fs,framedPhoneme)
% input:
% Phoneme – one phoneme (after pre-processing)
% Fs – sampling frequency
% framedPhoneme – the phoneme after framing

%output:
% FeatsVector – 1X24 vector of features of the analyzed phoneme
% Feat_title – 1X24 cell array of the names of the calculated features

Energy=mean(calcNRG(framedPhoneme)); % calculate the energy to each frame
separately and than calculate the mean
ZeroCrossing=mean(calcZCR(framedPhoneme)); % calculate the ZCR to each
frame separately and than calculate the mean
% pitch:
[fpitch,~]=sift(Phoneme,Fs); % if it's unvoiced, f=0 (and voice=0)

% lpc:
[lpc_coeff,~] = lpc(Phoneme,17);
% create the label string:
lpc_coeff_labels = cell(1,18);
for i = 1:18
    lpc_coeff_labels{i} = strcat('lpc coefficient ',num2str(i));
end

% formants:
p = Fs/1000 + 4; % LPC model oreder
Lags=2^10;
[Freq,~]= formants(Phoneme, p,[], Fs, Lags);


FeatsVector=[Energy;ZeroCrossing;fpitch;lpc_coeff';Freq(1);Freq(2);Freq(3)]
;
Feat_title=['mean Energy';'mean ZCR';'Pitch';lpc_coeff_labels';'Formant
1';'Formant 2';'Formant 3'];
end
```

לאחר מימוש הפונקציה על כל אחת מהפורמנטות קיבלנו את הערכים הבאים:

טבלה 2 : חישוב הערכים לכל פונמה

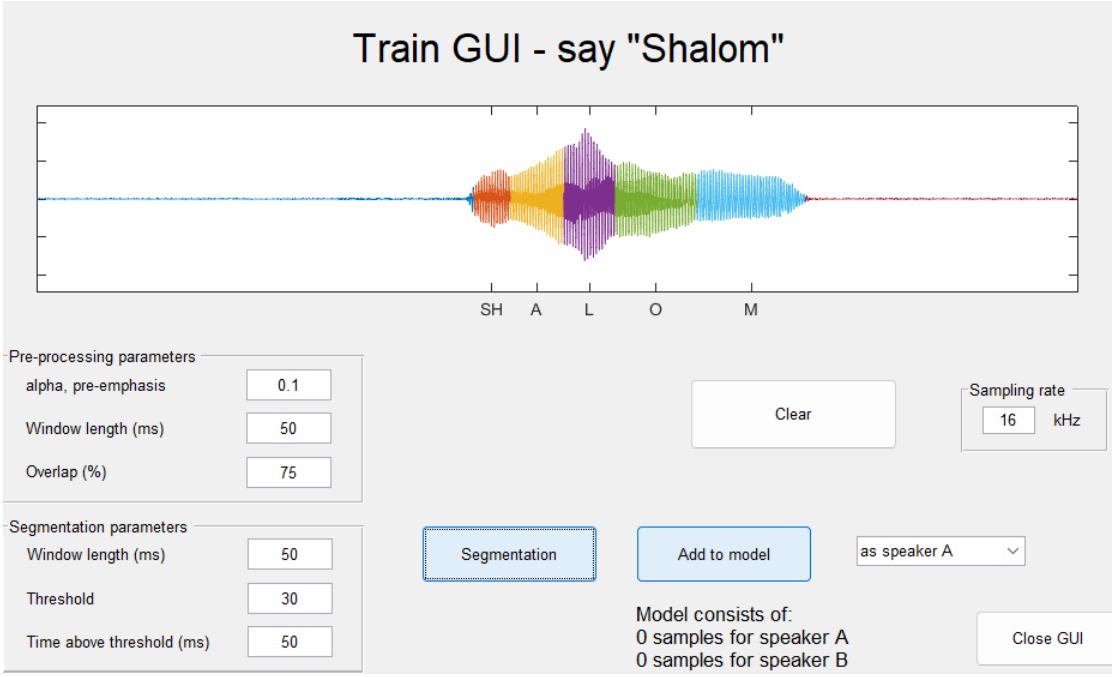
	1 Feat_title	2 SH_FeatsVector	3 A_FeatsVector	4 L_FeatsVector	5 O_FeatsVector	6 M_FeatsVector
1	'mean Ener...	0.0085	0.0123	0.0044	0.0044	0.0010
2	'mean ZCR'	0.4957	0.2030	0.1987	0.2344	0.3239
3	'Pitch'	139.1304	134.4538	124.0310	116.7883	115.9420
4	'lpc coeffici...	1	1	1	1	1
5	'lpc coeffici...	-0.2406	-1.6362	-1.2942	-1.2168	-0.9552
6	'lpc coeffici...	0.9601	1.6013	1.0317	0.7380	0.5435
7	'lpc coeffici...	-0.7986	-0.9254	-0.4629	-0.2370	-0.2556
8	'lpc coeffici...	0.7637	0.6319	0.3188	0.0659	0.1615
9	'lpc coeffici...	-0.7958	-0.7265	-0.6097	-0.5760	-0.4685
10	'lpc coeffici...	0.6689	0.8777	0.5649	0.4954	0.4728
11	'lpc coeffici...	-0.4149	-0.7311	-0.2726	-0.0665	-0.1183
12	'lpc coeffici...	0.5534	0.8133	0.2379	0.3377	0.3324
13	'lpc coeffici...	-0.3787	-0.9757	-0.5026	-0.3388	-0.3911
14	'lpc coeffici...	0.2941	1.0975	0.6398	0.5172	0.1567
15	'lpc coeffici...	-0.2524	-0.8875	-0.5498	-0.3021	-0.2042
16	'lpc coeffici...	0.2330	0.6223	0.2844	0.0491	0.1874
17	'lpc coeffici...	-0.0739	-0.4095	-0.1837	-0.2039	-0.1615
18	'lpc coeffici...	0.3426	0.4543	0.1627	0.1404	0.1604
19	'lpc coeffici...	0.1749	-0.2051	0.0360	-0.1054	0.0546
20	'lpc coeffici...	0.2917	0.2252	0.1061	0.0818	0.0754
21	'lpc coeffici...	0.1391	-0.0296	0.0845	0.1907	-0.0680
22	'Formant 1'	579.2564	626.2231	454.0117	516.6341	594.9119
23	'Formant 2'	1.5656e+03	1.5186e+03	1.4560e+03	1.0802e+03	1.4403e+03
24	'Formant 3'	2.6928e+03	2.5205e+03	2.4736e+03	2.4736e+03	3.2720e+03

מטבלה זו ניתן לראות את מימוש הפונקציה לכל אחת מהפונמות.

הינו מצפים שהאנרגיה הממוצעת עבור SH תהיה נמוכה משמעותית מהאחרות כיוון שהיא פונמה א-קולית והאחרות קוליות. עם זאת, לא קיבלנו אנרגיה נמוכה כרצוי. נשער שהשגיאה נובעת מאי אידאליות הסגמנטציה עליה הסברנו בסעיפים קודמים וניתן לראות שהסגמנט הראשון מכיל מקטע קטן קולי מתוך הפונמה 'A'. כמו כן, ציפינו לא לקבל פיטץ' עבור פונמה זו (בפונקציה מוגדר כערך אפס) אך נשער שהתוצאה שקיבלנו נובעת מסיבה זו גם. בנוסף, מהטבלה ניתן לראות כי ערך הZCR הגבוה ביותר עבור הפונמה הא-קולית ונמוך עבור הקוליות. למרות זאת, ערך הZCR עבור הפונמה 'M' גבוה יותר מהאחרות. נשער כי בשל סגמנטציה לא אידאלית של פונמה זו המכילה מקטעי שקט בהם ערך הZCR גבוה קיבלנו עליה בערך מעבר למצופה.

ניסוי 5: בניית המודל, יצירת בסיס נתונים

בניסוי זה ראשית שמרנו את כל הפונקציות שבנינו בתיקיית Train. לאחר מכן, ביצענו הקלטה באמצעות guiו ובחרנו את הפרמטרים שיתנו את הסגמנטציה הטובה ביותר.



איור 15: דוגמה לסגמנטציה שביצענו והפרמטרים שנבחרו

מאיור זה ניתן לראות את תוצאת הסגמנטציה עבור הקלטה לדוגמא של דובר A. ניתן לראות כי בוצע זיהוי של כל מאפייני הסיגנל, אם כי הסיגמנטציה איננה אידאלית. שינינו את הפרמטרים בתוכנית לקבלת הסגמנטציה הרצויה ובחרנו את אלה שנתנו את התוצאה הטובה ביותר במרבית ההקלטות. לאחר שבחרנו את הפרמטרים ביצענו 10 הקלטות לכל אחד מהדוברים. את הפרמטרים הנבחרים ניתן לראות באיור זה גם.

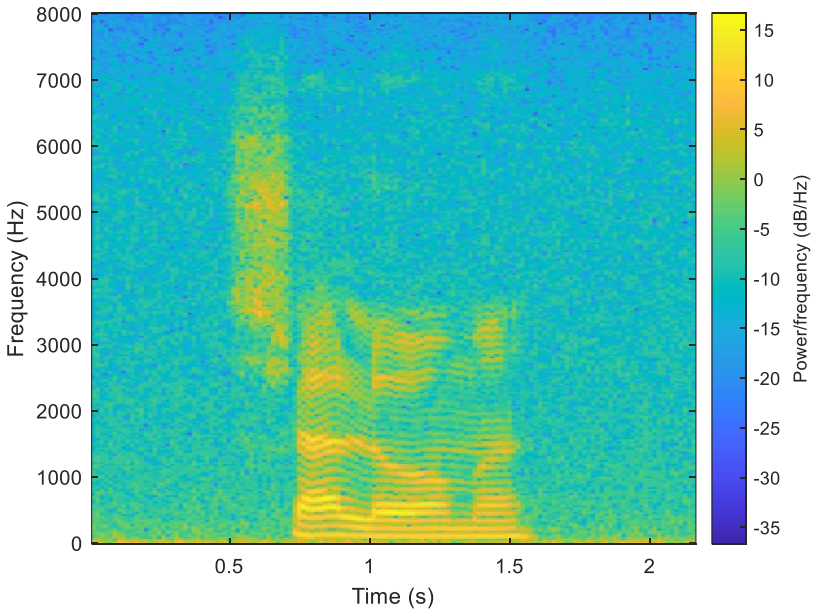
ניסוי 6: ניתוח STFT

6.1

שיטת STFT הספקטוגרמה, הינה דרך בה ניתן להציג הן את השינויים התדיריים והן את השינויים הזמניים. היחוד של שיטה זו נובע מהיכולת לא רק לראות מהם התדרים הדומיננטים באות אלא גם היא מאפשרת לראות מתי תדרים אלו התרחשו. אותות דיבור הינם אותות בהם יש חשיבות רבה לתחום הזמן כיוון שכל פונמה בעלת תדר שונה ושינוי בין הפונמות מתרחש בקצב מהיר. בשל כך, בעזרת שיטת הSTFT ניתן להבחין מתי התרחשו השינויים ולאפיין כל מקטע זמן בנפרד לפי תדירותו.
בהצגת הספקטוגרמה יש מספר מאפיינים המשפיעים על הרזולוציה.
גודל החלון אותו נבחר- ככל שנקטין את גודל החלון הרזולוציה בזמן תעלה אבל הרזולוציה התדרית תפגע.
חפיפה בין החלונות- הוספת חפיפה שומרת על הרזולוציה התדרית ומשפרת את הרזולוציה הזמנית.

6.2

כפי שהסברנו, פונמות קוליות הינן פונמות אותן ממדלים לרכבת הלמים שעוברת דרך מסנן לינארי עם תדר בסיס. פונמות א-קוליות הינן ממודלות כרעש לבן דרך מסנן לינארי.
במילה shalom ישנן פונמה א-קולית (sh) ו4 פונמות קוליות (a,l,o,m).



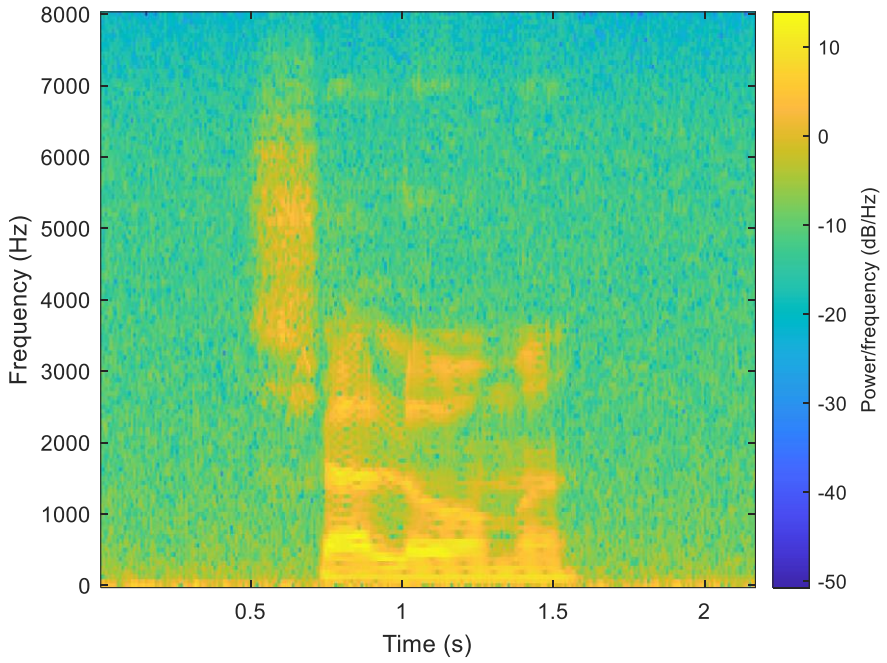
איור 16 : ספקטוגרמה של המילה 'shalom'

מאיור זה ניתן לראות את ההבדלים בין פונמות קוליות לא-קוליות. בעוד בפונמות קוליות ניתן לראות את רכבת ההלמים (זמנים 0.7-1.65 שניות) בפונמות א-קוליות (0.5-0.6 שניות) אין מרכיבי תדר אלה כצפוי. כמו כן, ניתן לראות את המעבר בין פונמה אחת לשנייה בתחום הזמן מתוך איור זה, ואת מרכיבי התדר של כל פונמה בנפרד. איור זה ממחיש את השימוש של STFT עבור אותות דיבור ואת חשיבותו שכן שימוש בשיטות אחרות לא מאפשר לבצע הפרדה זו בין מרכיבי הדיבור אלה נותן את מרכיבי התדר הכללים שנמצאים באות ללא התייחסות לזמן.

באיור זה בחרנו להשתמש בחלון באורך של 500 דגימות וחפיפה של 50% מהחלון (250). השימוש בגודל חלון זה מאפשר שיפור ברזולוציה התדרית כך שרואים את מרכיבי התדרים בצורה ברורה

ואת רכבת ההלמים. השימוש בחפיפה של 50% מאפשר שיפור ברזולוציה הזמנית כך שניתן לראות את ההבדלים בין מרכיבי האות כלומר את המעבר בין הפונמות.

כעת נקטין את גודל החלון ל250 ונשמור את החפיפה ב50% ונציג את הספקטוגרמה שהתקבלה:



איור 17: ספקטוגרמה של המילה 'shalom'

מאיור זה ניתן לראות כי קיבלנו פגיעה ברזולוציה התדרית כיוון שכעת לא ניתן לראות את רכבת ההלמים באופן ברור ולא את התדרים המרכיבים אותה. תופעה זו נובעת מהקטנת גודל החלון שפוגעת ברזולוציה התדרית כיוון שכל חלון מכיל פחות דגימות והתוצאה הינה מריחה של התדר.

ניסוי 7: סיווג דובר באמצעות המערכת

7.5

בזיהוי הדובר האלגוריתם צדק 5\5 עבור דובר B ו0\5 עבור דובר A. כלומר, בכל פעם שעשינו את הבדיקות ההסתברות להיות בהתפלגות של דובר B גבוהה יותר מההסתברות להיות בהתפלגות של דובר A. כיוון שהמסווג לדוברים משתמש בפיצ׳רים לכל אחת מהפונמות, ייתכן כי העובדה שהפונמות לא מופרדות בצורה מספיק טובה, הובילה לאיכות נמוכה של פיצ׳רים כך שהסיווג לא בוצע בצורה טובה. ניתן לשפר את המודל ע״י הגדלת כמות הדגימות, או שיפור הסגמנטציה.

7.6

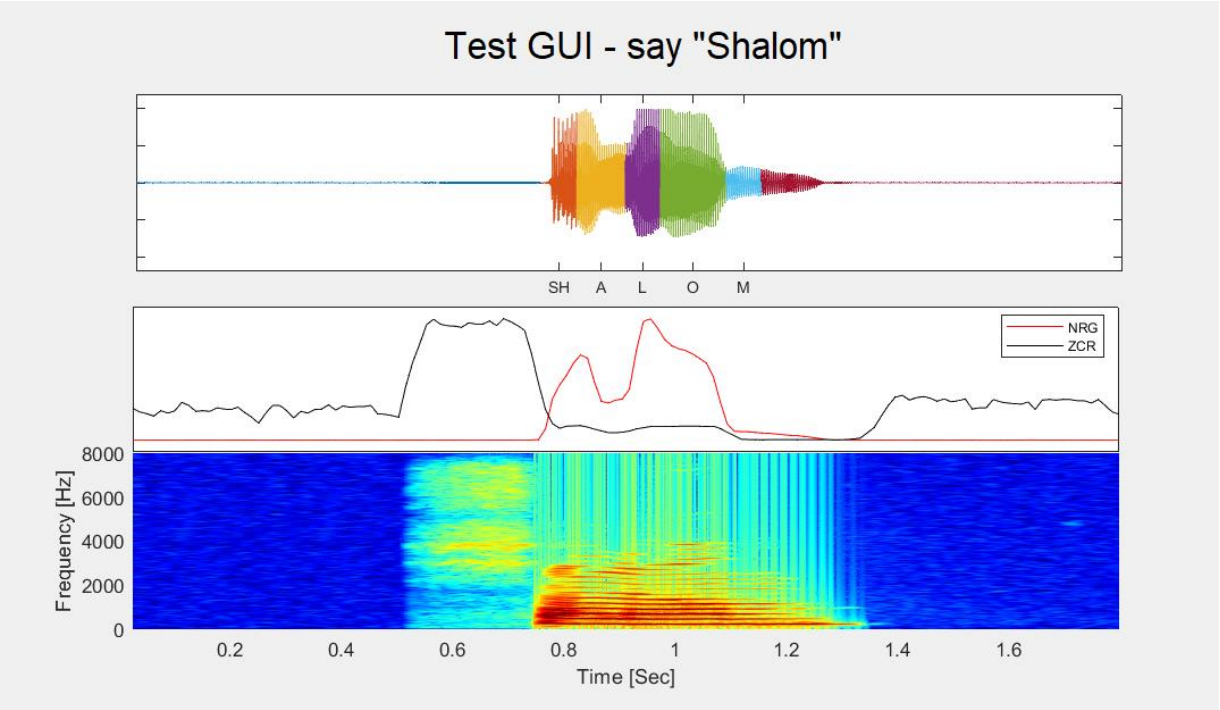
הפונקציה classifySpeaker משתמש בהקלטות מתוך המודל ששמרנו בניסוי 5. עבור כל דובר, היא מחשבת את הממוצע וסטיית התקן של כל אחד מהפיטצ׳רים לכל פונמה, ומחשבת את פונקציית צפיפות ההסתברות בהתאם לערכים אלה. לאחר מכן, עבור ההקלטה הנוכחית היא מחשבת מה ההסתברות שהיא מכל אחת מההתפלגויות, והקבוצה בה קיבלנו את ההסתברות הגבוה יותר נבחרת להיות הזיהוי של הפונקציה. היא פועלת בדומה למודל naïve bayes gaussian שמשתמש באלגוריתם זה.

הגדרת פונקציית צפיפות ההסתברות:

$$p(x\,|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

כאשר μ מייצג את הממוצע וσ את השונות.

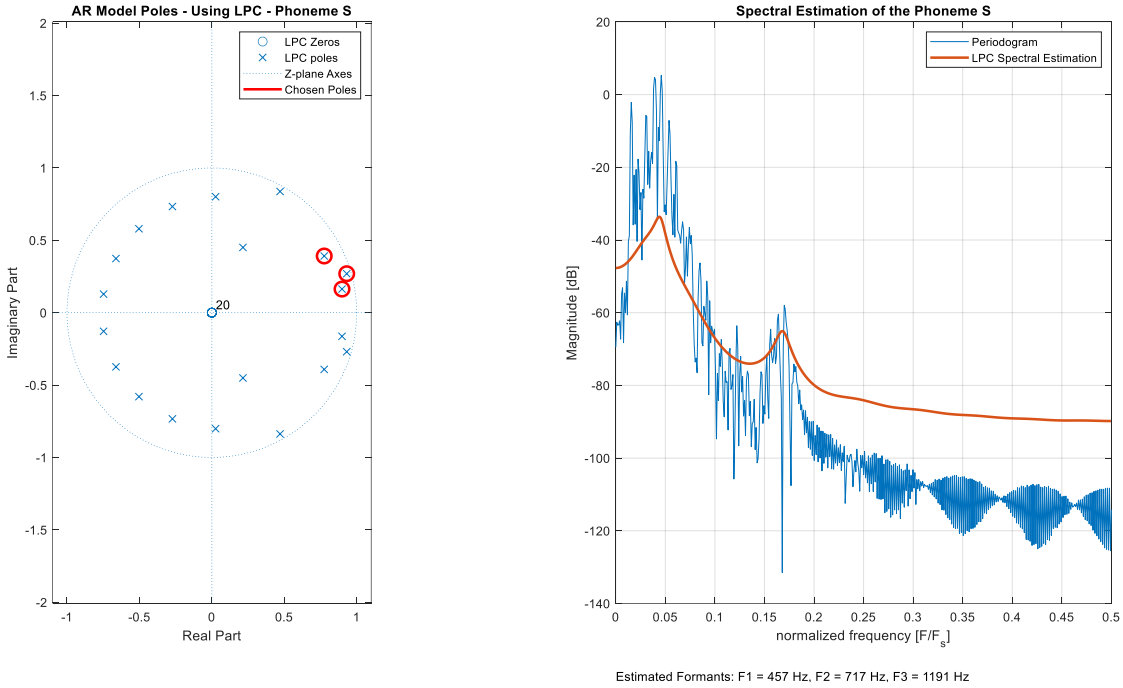
7.7 נראה כי הפונקציות מתממשקות כראוי לGUI:



איור 18: דוגמא עבור גרף האנרגיה והZCR והספקטוגרמה

מאיור זה ניתן לראות דוגמה לחישוב האנרגיה והZCR עבור הקלטת המילה שלום. כפי שהסברנו קודם, נצפה לאנרגיה גבוהה עבור פונמות קוליות ונמוכה עבור פונמות א-קוליות (SH). בנוסף,

נצפה לקבל קצב ZCR גבוה עבור פונמות א-קוליות ונמוך עבור פונמות קוליות. מהגרף ניתן לראות כי עבור ZCR גבוה האנרגיה נמוכה, שם נשער שהתרחשה הקלטת הSH. בהשוואה לספקטוגרמה ניתן לראות כי השערתינו נכונה שכן אין מרכיבי רכבת הלמים בטווח זמנים זה (סביב 0.6 שניות), כלומר זהו מרכיב הSH בסיגנל. לאחר מכן, ניתן לראות בפסטוגרמה את מרכיבי רכבת ההלמים לכל אחת מהפונמות הקוליות. בהשוואה לגרף העליון בזמנים אלה, האנרגיה גבוהה והZCR נמוך כרצוי.



איור 19 : דוגמה עבור SH

מאיור זה ניתן לראות דוגמה לגרפים עבור פונמת הSH כאשר מתוכם ניתן להסיק שחישוב הפיטצ'רים עובד כראוי.

7.8

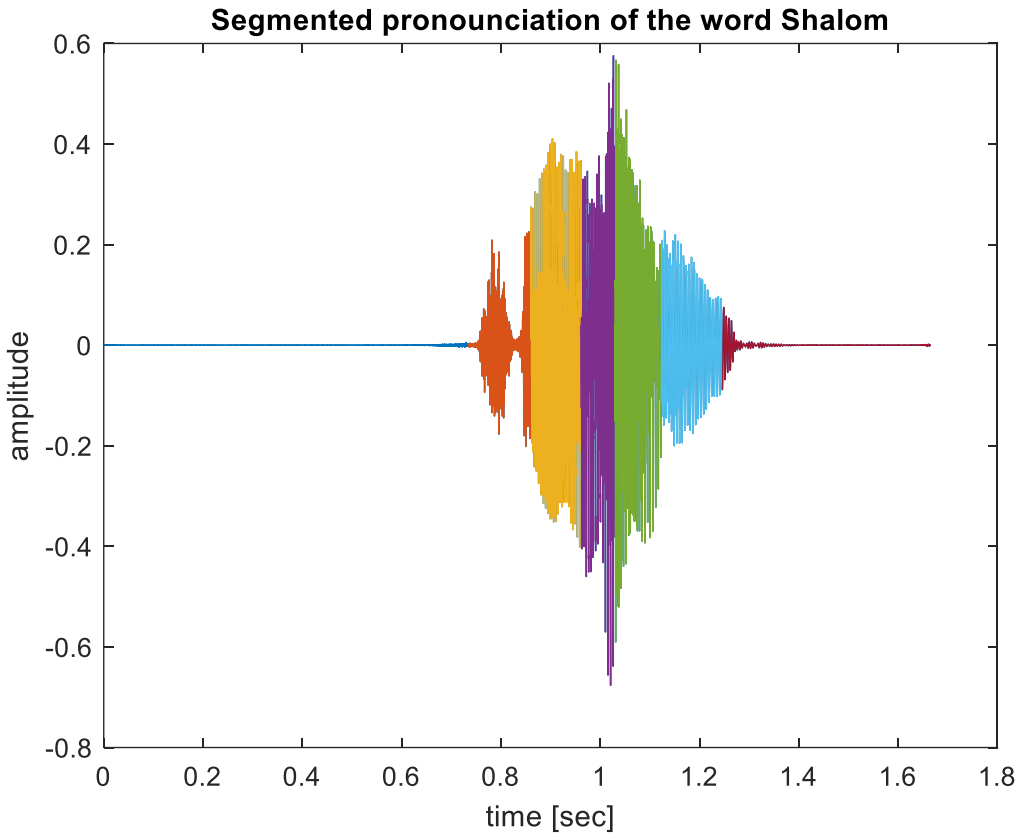
הפונקציה שמבצעת את שמירת הגרפים :

```
function
exportGraphs(folder_name,Signal,Fs,phon,seg_ind,STFTwinLength,STFToverlap,ST
FTnfft,STFTcmin,STFTcmax,NRG,ZCR,Flag);
%input:
%folder_name – full address of the folder to save the files in
% Signal – the recorded signal
% Fs – sampling frequency
% phon – an array of chars, representing the phonemes
% seg_ind – result of the segmentation process
% STFTwinLength – window length for the STFT (in samples)
% STFToverlap – overlap length for the STFT (in samples)
% STFTnfft – nfft length (in samples)
% STFTcmin – minimum value for color axis scaling
% STFTcmax – maximum value for color axis scaling
% NRG – energy signal
% ZCR – ZCR signal
% Flag – indicates which graphs to generate.
```

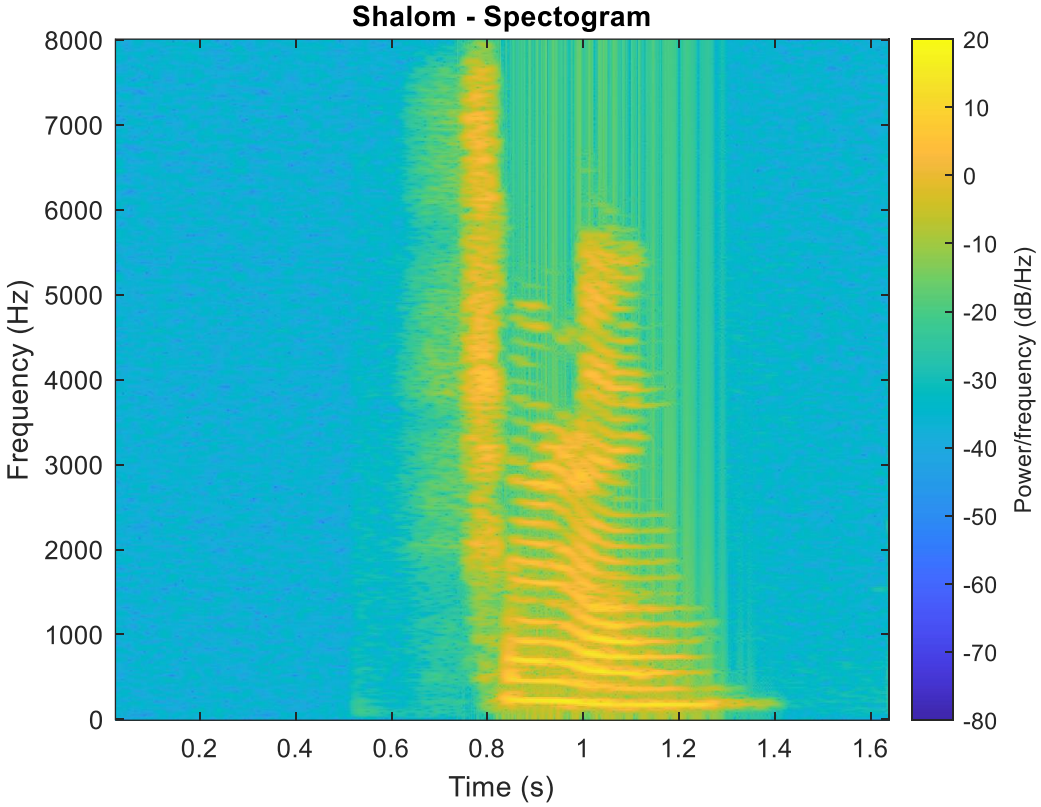
```
%output:
%save 13 graphs to the folder

%1.
% The temporal signal graph after preprocessing and segmentation
t = 0:1/Fs:(length(Signal)-1)/Fs; %time vector for loud_signal
colors=["#0072BD","#D95319","#EDB120","#7E2F8E","#77AC30","#4DBEEE","#A21
42F"];
h=figure;plot(t,Signal);hold on;
for i=1:length(seg_ind)-1
   plot(t(seg_ind(i):seg_ind(i+1)),Signal(seg_ind(i):seg_ind(i+1)),'Color',colors(i));
end
hold off
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Segmented pronounciation of the word
'Shalom''

saveas(h,fullfile(folder_name, 'signal_segmentation.jpg'));
saveas(h,fullfile(folder_name, 'signal_segmentation.fig'));


%2.
%spectogram
[S,F,T] = spectrogram(Signal,hamming(STFTwinLength),STFToverlap,STFTnfft,Fs);

h=figure;imagesc(T,F,10*log10(abs(S)));
axis xy; xlabel('Time (s)');ylabel('Frequency (Hz)'); title('Shalom - Spectogram')
c = colorbar; c.Label.String = 'Power/frequency (dB/Hz)'; clim([STFTcmin
STFTcmax]);
saveas(h,fullfile(folder_name, 'STFT.jpg'));
saveas(h,fullfile(folder_name, 'STFT.fig'));


if Flag~=0 %if flag is equal to zero- save only the first two graphs.
   %3.
   % Energy & ZCR
   h3=figure;plot(1:length(NRG),NRG,'Color','r');hold on;
   plot(1:length(ZCR),ZCR,'Color','b'); legend('NRG','ZCR');
   xlabel('Frame #'); ylabel('Arbitary amplitude'); title('Energy and Zero-crossing
measures of the gramed signal');
   saveas(h3,fullfile(folder_name, 'NRG_ZCR.jpg'));
   saveas(h3,fullfile(folder_name, 'NRG_ZCR.fig'));

   %4. for each Phoneme:
   for i=1:length(phon)-1
   [h1,h2]=estimatePhonemeFormants(Signal(seg_ind(i):seg_ind(i+1)),Fs,phon(i));
   saveas(h1,[folder_name '/' phon(i) 'Spectrum.jpg']);
   saveas(h1,[folder_name '/' phon(i) 'Spectrum.fig']);
   saveas(h2,[folder_name '/' phon(i) 'Polemap.jpg']);
   saveas(h2,[folder_name '/' phon(i) 'Polemap.fig']);
   end
```

```
end
```

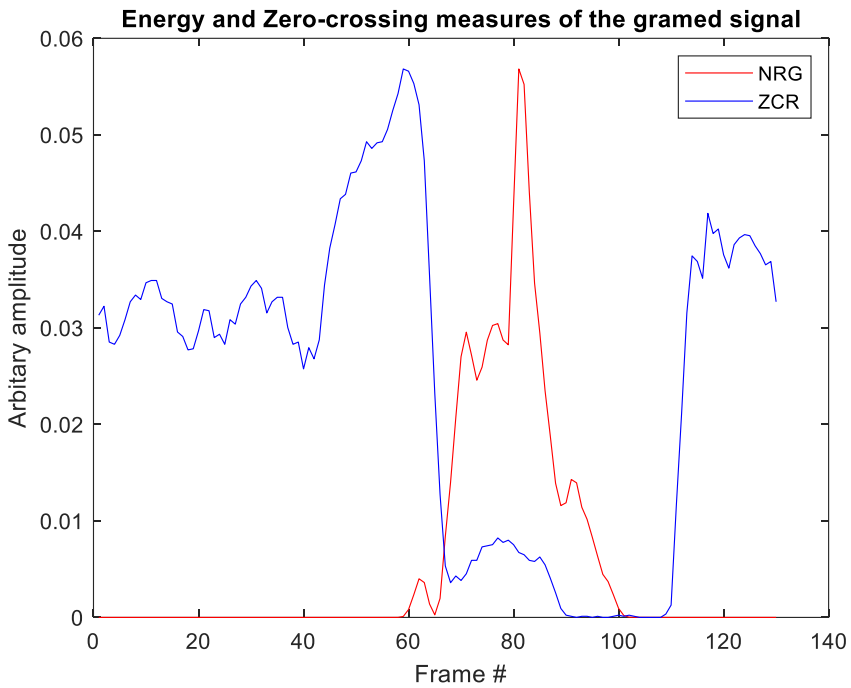
הגרפים שנשמרו בתיקייה:



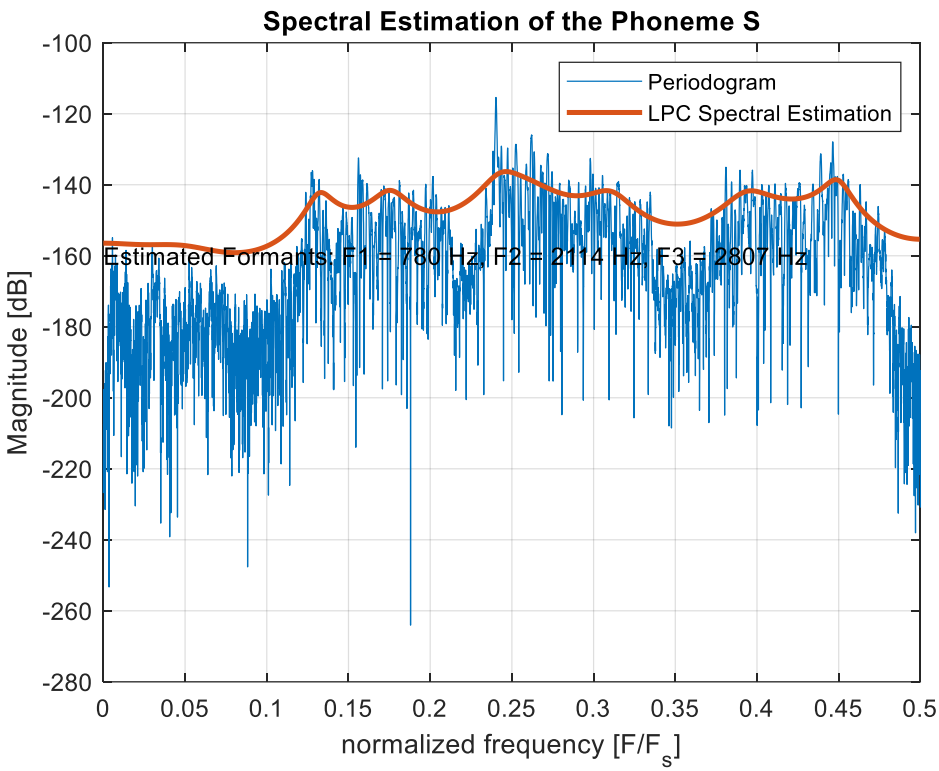
איור 20: הסיגנל הזמני והסטגמנטציה שבוצעה



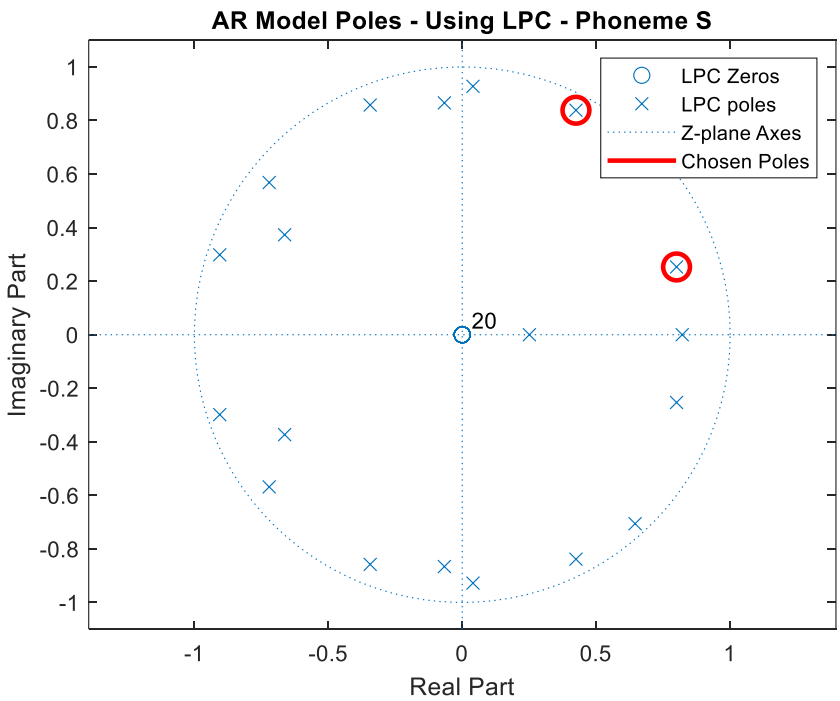
איור 21 : הספקטוגרמה של האות



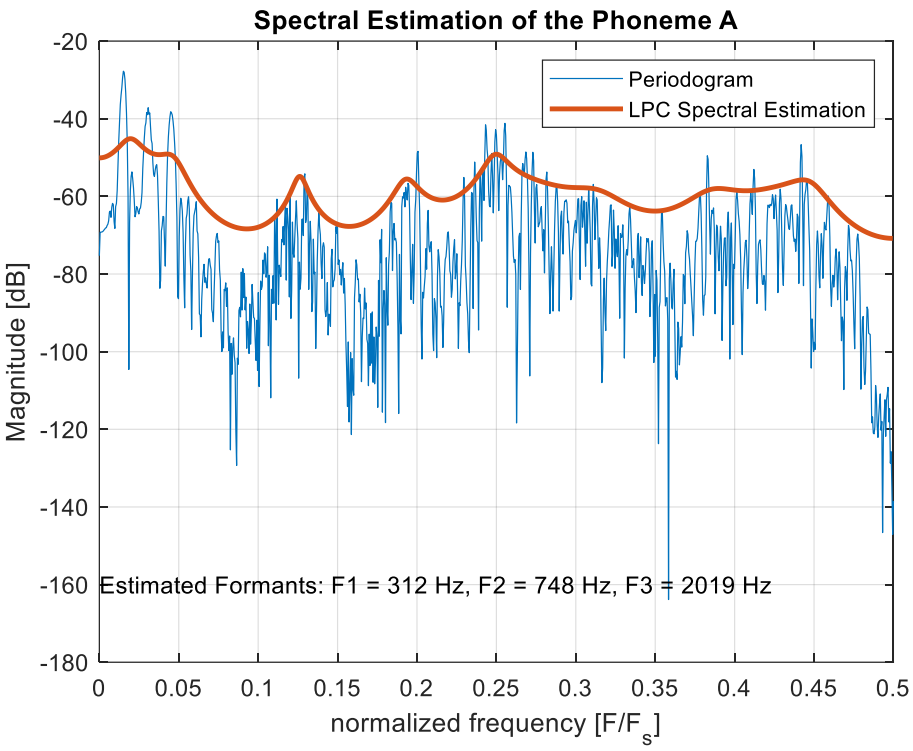
איור 22 : גרף האנרגיה וקצב חציית האפס



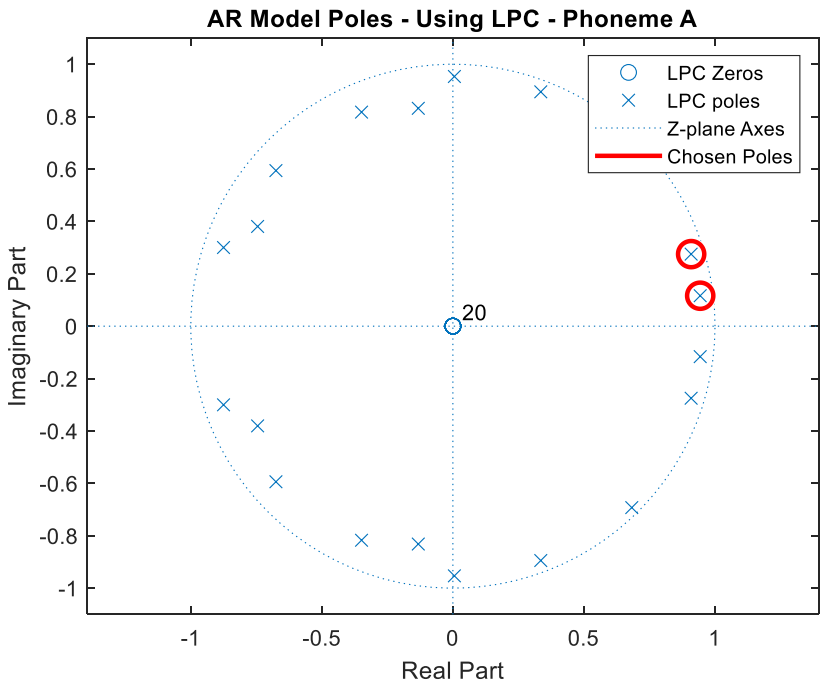
איור 23 : גרף שערוך ספקטרלי (פרמטרי ולא פרמטרי) של 'SH'



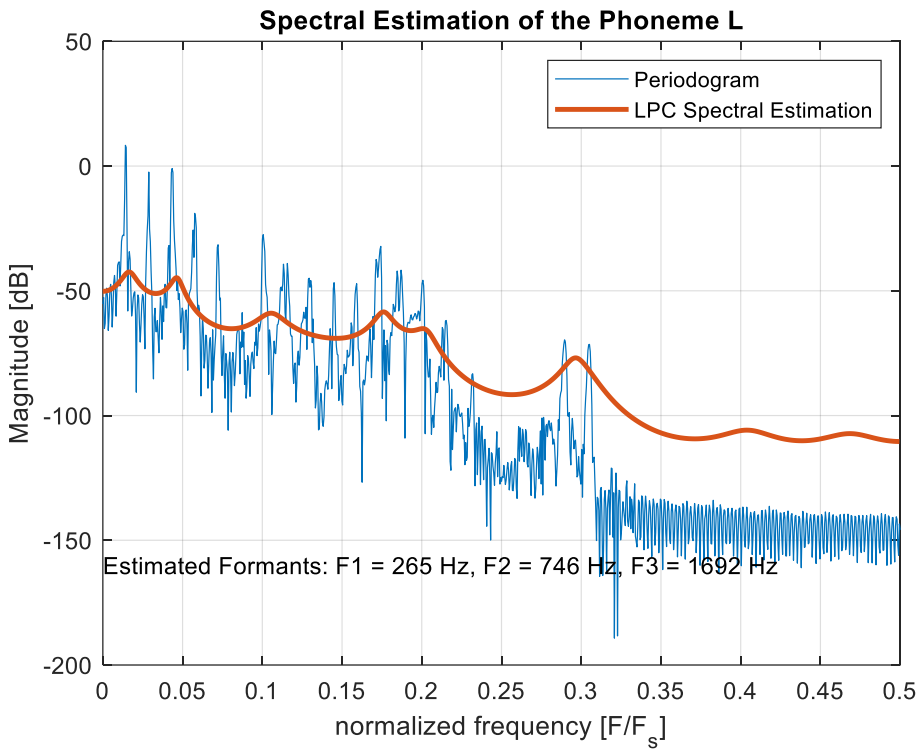
איור 24 : גרף מפת קטבים ואפסים כולל סימון שלושת הקטבים הקשורים לשלוש הפורמנטות הראשונות ל'SH'



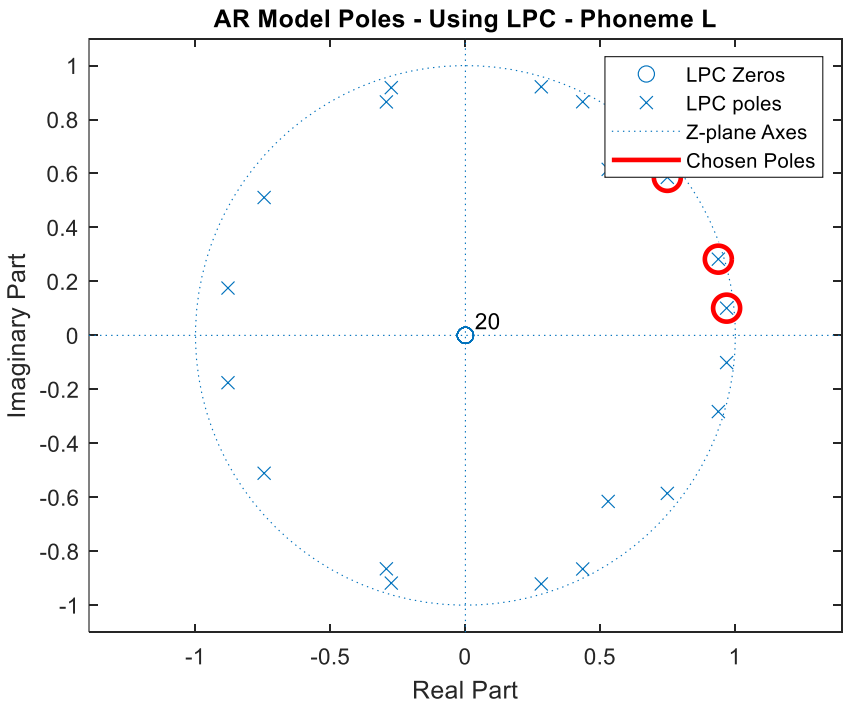
איור 25 : גרף שערוך ספקטרלי (פרמטרי ולא פרמטרי) של 'A'



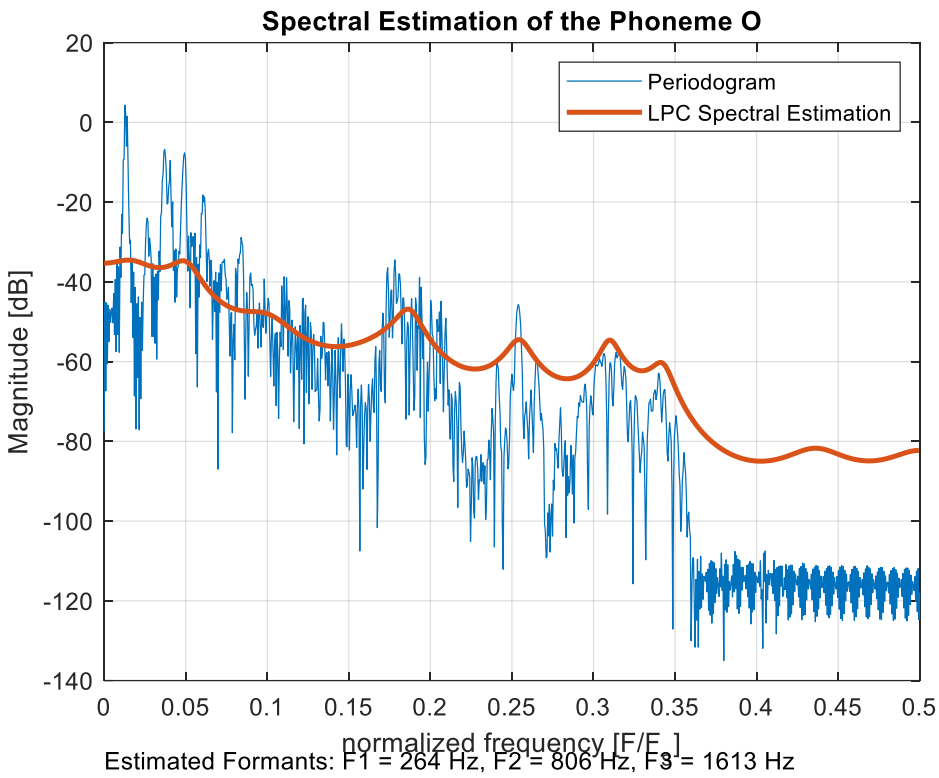
איור 26 : גרף מפת קטבים ואפסים כולל סימון שלושת הקטבים הקשורים לשלוש הפורמנטות הראשונות ל'A'



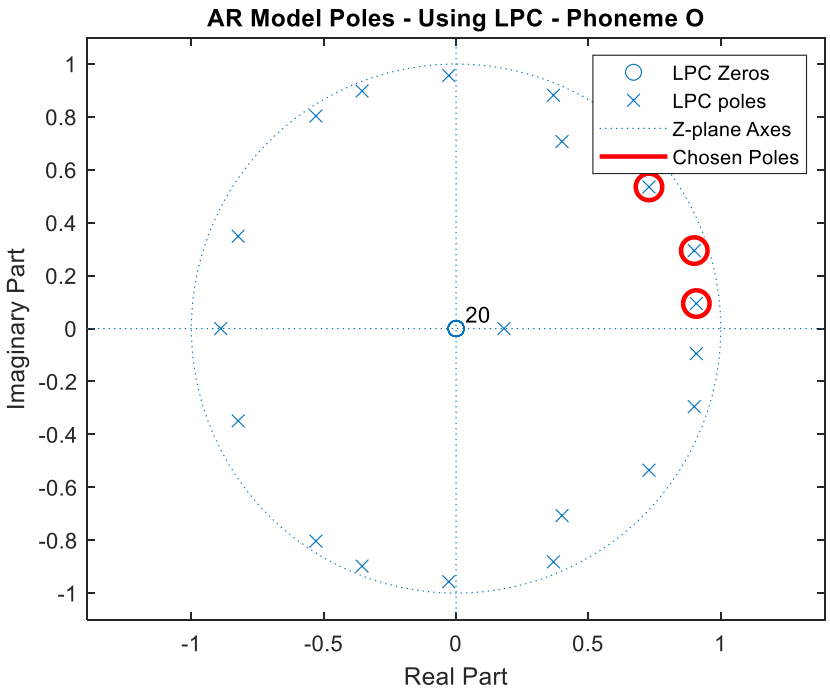
איור 27 : גרף שערוך ספקטרלי (פרמטרי ולא פרמטרי) של 'L'



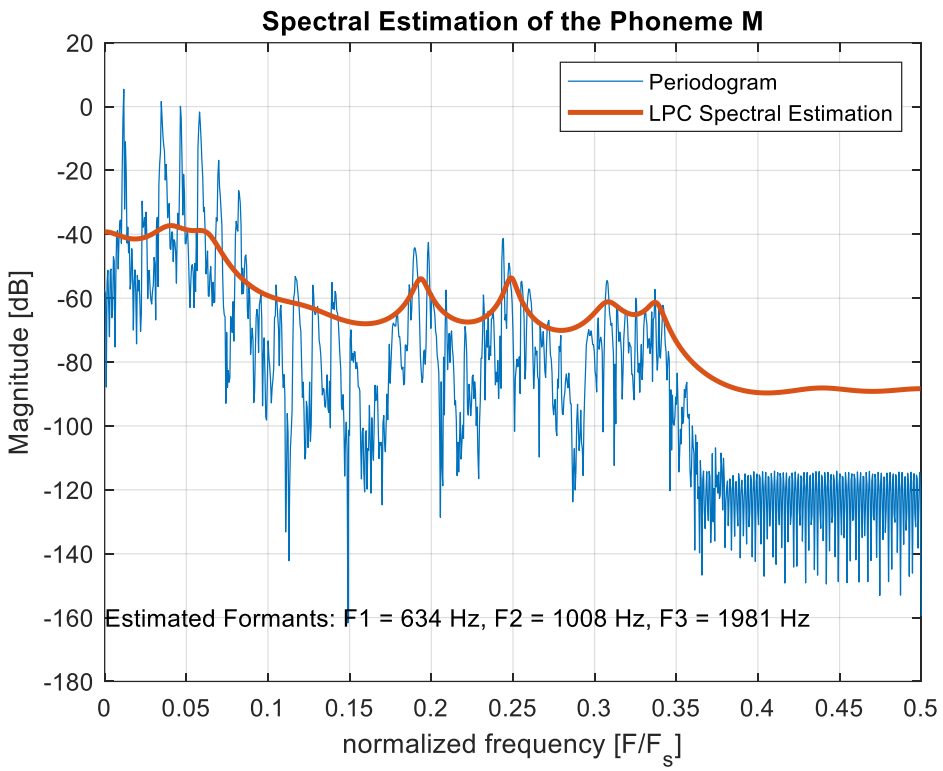
איור 28 : גרף מפת קטבים ואפסים כולל סימון שלושת הקטבים הקשורים לשלוש הפורמנטות הראשונות ל'L'



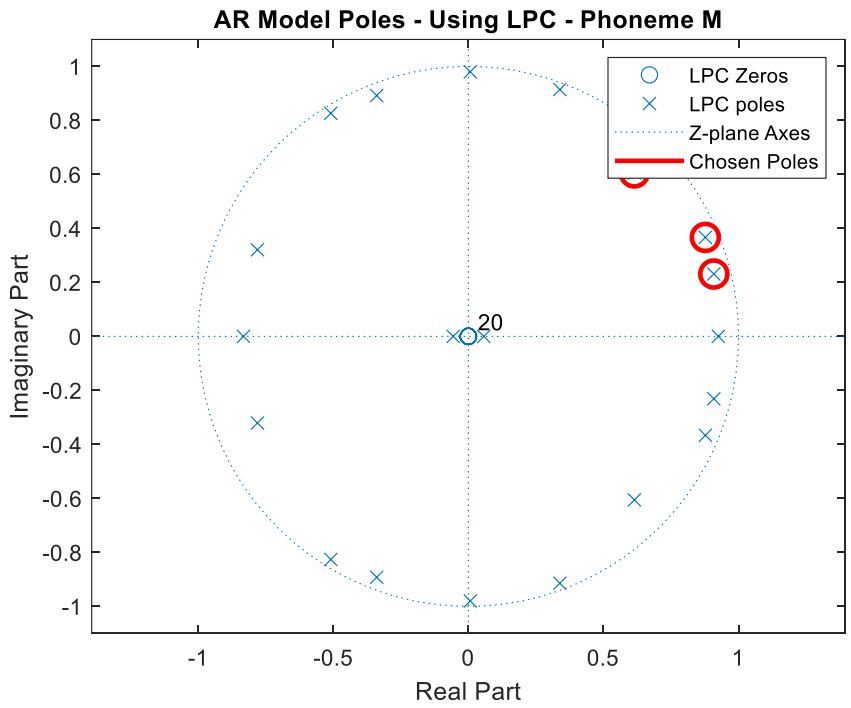
איור 29 : גרף שערוך ספקטרלי (פרמטרי ולא פרמטרי) של'O'



איור 30 : גרף מפת קטבים ואפסים כולל סימון שלושת הקטבים הקשורים לשלוש הפורמנטות הראשונות ל'O'



איור 31 : גרף שערוך ספקטרלי (פרמטרי ולא פרמטרי) של 'M'



איור 32 : גרף מפת קטבים ואפסים כולל סימון שלושת הקטבים הקשורים לשלוש הפורמנטות הראשונות ל'M'

מסקנות כלליות

במעבדה זו הכרנו תכונות בסיסיות של אותות דיבור וביצענו ניתוחים שונים של אותות אלה. המטרה הכללית הייתה לזהות פונמות שונות ומאפייניהן במילה 'שלום'. תחילה למדנו כיצד לבצע סינון מקדים לאות דיבור הכולל הדגשת תדרים גבוהים לשם הדגשת פורמנטות וחלוקה לחלונות זמניים קטנים. לאחר מכן, יצרנו אלגוריתם לביצוע סגמנטציה של האות שמטרתו להסיר חלקים שקטים באות, ולחלק את לפונמות שונות (מקטעים קוליים וא-קוליים). את החלוקה עושים ע"י מדדי שוני ספקטרלי, כאשר ניתן לבחור מדדים שונים, אך מהות המדדים הוא מציאת שוני באופי התדרי של מקטעים שונים באות כדי להבדיל ביניהם. מאפיין חשוב של אות דיבור הינו הפורמנטות ממנו הוא מורכב. לאחר ביצוע הסגמנטציה אפיינו כל מקטע באמצעות שערוך הספקטרום, ומציאת הפורמנטות מתוך שערוך זה. נוכחנו לראות כי באמצעות שערוך פרמטרי – LPC ניתן לשערך את האופי התדרי בצורה די טובה ובאמצעות חישובים מועטים יחסית, וכן לחלץ משערוך זה את ערכי הפורמנטות. לאחר מכן, ביצענו שערוך של האנרגיה ושל קצב חציית האפס (ZCR), כאשר מדדים אלה מאפשרים להבחין בין פונמות קוליות לא-קוליות. פונמות קוליות מתאפיינות באנרגיה גבוהה ו – ZCR נמוך, לעומת פונמות א-קוליות להן אנרגיה נמוכה ו – ZCR גבוה. כמו כן, ביצענו ניתוח STFT של אותות הדיבור. ניתוח זה מאפשר לראות את השינויים התדריים בזמן בו הם מתרחשים בניגוד להתמרת פורייה מלאה על האות, בה ניתן לראות את המרכיבים התדריים של האות לאורך כל ההקלטה, בלי להבחין בשינויים זמניים של האופי התדרי. כלי זה חשוב לניתוח אותות דיבור כיוון שלכל פונמה יש אופי תדרי שונה ובייחוד מדגיש את ההבדלים בין פונמות קוליות לא-קוליות. בפונמות קוליות ניתן לראות רכבת הלמים (pitch), ואילו בפונמות א-קוליות נראה מריחה או ביטוי של טווח תדרים ספציפיים. לאחר מכן, השתמשנו בכל המאפיינים שהסברנו עד כה ליצירת מודל שיאפשר הקלטה של המילה 'שלום', ביצוע סגמנטציה אוטומטית של האות וסיווג הדובר. המודל מאפשר גם שמירה של גרפים עבור המאפיינים המתוארים לעיל, כגון – ZCR, אנרגיית האות, זיהוי פורמנטות, STFT. ראינו כי המערכת התקשתה בסיווג הדוברים. תהליך הסגמנטציה שביצענו נתן תוצאות לא מדויקות מאוד. כיוון שהמסווג לדוברים משתמש בפיצ'רים לכל אחת מהפונמות, ייתכן כי העובדה שהפונמות לא מופרדות בצורה מספיק טובה, הובילה לאיכות נמוכה של פיצ'רים כך שהסיווג לא בוצע בצורה טובה. ייתכן כי הקושי נובע משימוש באלגוריתם naïve-bayes הבונה התפלגות להסתברות שההקלטה שייכת לדובר מסוים על סמך כלל הפיצ'רים שייצרנו במודל. ניתן לשפר את המודל ע"י הגדלת כמות הדגימות, או שיפור הסגמנטציה.

מקורות

[1] A. Behrman and D. Finan, *Speech and Voice Science*. Plural Publishing, Incorporated, 2021.

[2] L. R. Rabiner and R. W. Schafer, *Introduction to Digital Speech Processing*. Now Publishers Inc, 2007.

[3] "A Course in Phonetics - Peter Ladefoged, Keith Johnson - Google ספרים." https://books.google.co.il/books?id=9a0JzgEACAAJ&dq=a+course+in+phonetics&hl=iw&sa=X&redir_esc=y (accessed Jan. 09, 2023).

[4] J. R. D. Jr, J. H. L. Hansen, and J. G. Proakis, *Discrete-Time Processing of Speech Signals*. Wiley, 2000.

[5] L. Sörnmo and P. Laguna, Eds., "Copyright," in *Bioelectrical Signal Processing in Cardiac and Neurological Applications*, Burlington: Academic Press, 2005, p. iv. doi: 10.1016/B978-0-12-437552-9.50013-1.

נספחים

```
%% speech

%% 1.

% 1.1

% Pre-emphasis filter
alpha = 0.95;
figure, zplane([1 -alpha], [1 0]), title 'Pole-Zero plot, alpha = 0.95'

% plotting hamming window
wvtool(hamming(64));

% 1.2 - see PreProcess.m

%% 2.
% 2.2 - see FindWordIdx.m
% 2.3 - see segmentation.m


%2.4
% First lets present the signal before and after preprocessing
[signal,Fs] = audioread("shalom_example.wav"); %reading the audio file
alpha = 0.95; WindowLength = 30*10^-3; Overlap = 50; % standart parameters
for framing
[ProcessedSig,FramedSig] =
PreProcess(signal,Fs,alpha,WindowLength,Overlap); %preprocessing

% plotting the original and preprocessed signal
t = 0:1/Fs:((length(signal)-1)/Fs);
figure, subplot(211), plot(t,signal)
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Original audio signal'
subplot(212), plot(t,ProcessedSig)
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Preprocessed audio
signal'


%finding quite parts and removing them:
Idx = FindWordIdx(FramedSig,Fs,WindowLength,Overlap);

% Deleting the quite parts of the signal:
% removing overlaping indicies from Idx
Idx_new = zeros(size(Idx));
Idx_new(1,:) = Idx(1,:);
j =1; %index of iterations
for i=2:length(Idx)
    % if there is overlap we will combine the two overlapping raws into one
    if Idx(i,1)<=Idx(i-1,2)
        Idx_new(j,2) = Idx(i,2);
    else
        j = j + 1;
        Idx_new(j,:) = Idx(i,:);
    end
end
Idx_new(Idx_new == 0) = []; %deleting raws of zero
```

```
% the number of samples in each speech frame
num_frm_smp = (Idx_new(:,2)-Idx_new(:,1))+1;
% length of all speech frames combined
len_loud_signal = sum(num_frm_smp);
% loud_signal will contain only the speech parts of the original signal
loud_signal = zeros(1,len_loud_signal);
j=1;

for i=1:(size(Idx_new,1))
    loud_signal(j:(j+num_frm_smp(i)-1)) =
ProcessedSig(Idx_new(i,1):Idx_new(i,2));
    j = j + num_frm_smp(i) ;
end

% plotting the detected loud parts vs original signal
figure, subplot(211), plot(t(Idx_new(1):Idx_new(2)),loud_signal)
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Loud part detection of
audio signal'
xlim([t(1) t(end)])
subplot(212), plot(t,ProcessedSig)
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Preprocessed audio
signal'
xlim([t(1) t(end)])


% using the segmentation algorithm on the processed signal:
eta = 33.66; dt = 0.047; %threshold and minimum time of crossing it
WindowLength = 0.055;
[seg_ind,delta] = segmentation(ProcessedSig,WindowLength,eta,dt,Fs,Idx);

% plotting the sementation results
new_t = 0:1/Fs:(length(loud_signal)-1)/Fs; %time vector for loud_signal
phonems = ["SH", "A", "L", "O", "M"]; %phonems of 'SHALOM' vector
figure, subplot(211), plot(new_t,loud_signal); xlim([-0.05 1.1])
hold on;
for i=1:length(seg_ind)
    if i<=length(phonems)
        xline(new_t(seg_ind(i)),'r',phonems(i),'LineWidth',2);
    else
        xline(new_t(seg_ind(i)),'r','LineWidth',2)
    end
end
hold off
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Signal segmentation'
subplot(212), plot(new_t,delta); xlim([-0.05 1.1])
hold on; yline(eta,'--',['\eta = ',
num2str(eta)],'LabelHorizontalAlignment','left','LineWidth',2);
for i=1:length(seg_ind)
    xline(new_t(seg_ind(i)),'r','LineWidth',2)
end
hold off
xlabel 'time [sec]' , ylabel '\Delta_1(n)' , title 'spectral error
measurement'


%% 3.

% 3.1
% plotting the phonem 'a' on the 'shalom' signal
```

```
figure, plot(new_t,loud_signal); hold on
xline(new_t(seg_ind(2)-420),'r','LineWidth',2); % start of phonem 'a'
xline(new_t(seg_ind(3)-380),'r','LineWidth',2); % end of phonem 'a'
xlabel 'time [sec]' , ylabel 'amplitude' , title ' ''a'' phonem'
hold off

% spectral estimation from preiodogram
phonem_a = loud_signal((seg_ind(2)-420):(seg_ind(3)-380));
[Pxx,w] = periodogram(phonem_a);
%plotting the spectral estimation
figure, plot(w/(2*pi),db(Pxx))
xlabel 'normalized frequency [F/F_s]' , ylabel('amplitude [dB]')
title('Spectral estimation of phonem ''a'' using periodogram')

% 3.3
% LPC:
p = Fs/1000 + 4; % LPC model oreder
[a,g] = lpc(phonem_a,p); % a = model coeeficients, g = gain
[H,w_LPC] = freqz(g,a,1024); % spectrum

% comparing the unparametric and paramteric spectrum estimators
figure, plot(w/(2*pi),db(Pxx))
xlabel 'normalized frequency [F/F_s]' , ylabel 'Magnitude [dB]'
title('Spectral estimation of phonem ''a'' using periodogram and LPC')
hold on; plot(w_LPC/(2*pi),db(H),'LineWidth',2)
legend('Periodogram','LPC')
hold off

% 3.5
poles = roots(a);
figure, zplane(g,a);
title 'Pole-Zero plot of phoneme ''a'''

%finding the three poles that are closest to 0 frequency but arent actually
% 0 frequency
% thetas = sorted list of the positive angles of the poles
[thetas,ind] = unique(abs(angle(poles)));
thetas(thetas == 0) = []; %if we have pole on the real axis
if length(thetas)<length(ind)
    ind(1) = [];
end

% marking the formants on the plot
viscircles([real(poles(ind(1:3))), imag((poles(ind(1:3))))]
,0.05,'Color','r');


% the 3 smallest thetas represent the 3 Formants
formants = thetas(1:3)/pi*(Fs/2);


[h1,h2] = estimatePhonemeFormants(phonem_a,Fs,'a');

%% 4:
%4.5:
[signal,Fs] = audioread("shalom_example.wav");
alpha = 0.95; WindowLength = 30*10^-3; Overlap = 50; % standart parameters
for framing
```

```
[ProcessedSig,FramedSig] =
PreProcess(signal,Fs,alpha,WindowLength,Overlap); %preprocessing
%finding quite parts and removing them:
Idx = FindWordIdx(FramedSig,Fs,WindowLength,Overlap);
% Deleting the quite parts of the signal:
% removing overlaping indicies from Idx
Idx_new = zeros(size(Idx));
Idx_new(1,:) = Idx(1,:);
j =1; %index of iterations
for i=2:length(Idx)
    % if there is overlap we will combine the two overlapping raws into one
    if Idx(i,1)<=Idx(i-1,2)
        Idx_new(j,2) = Idx(i,2);
    else
        j = j + 1;
        Idx_new(j,:) = Idx(i,:);
    end
end
Idx_new(Idx_new == 0) = []; %deleting raws of zero

% the number of samples in each speech frame
num_frm_smp = (Idx_new(:,2)-Idx_new(:,1))+1;
% length of all speech frames combined
len_loud_signal = sum(num_frm_smp);
% loud_signal will contain only the speech parts of the original signal
loud_signal = zeros(1,len_loud_signal);
j=1;

for i=1:(size(Idx_new,1))
    loud_signal(j:(j+num_frm_smp(i)-1)) =
ProcessedSig(Idx_new(i,1):Idx_new(i,2));
    j = j + num_frm_smp(i) ;
end


% using the segmentation algorithm on the processed signal:
eta = 33.66; dt = 0.047; %threshold and minimum time of crossing it
WindowLength = 0.055;
[seg_ind,delta] = segmentation(ProcessedSig,WindowLength,eta,dt,Fs,Idx);

%calculate energy and zcr:
Energy=calcNRG(FramedSig);
ZCR=calcZCR(FramedSig);

%Overlap is 50% so every new frame is after 240 sampels.
ind=ceil(seg_ind/240)+ceil(Idx_new(1)/240);
ind(end+1)=ceil(Idx_new(2)/240);
% plotting the sementation results
new_t = 0:1/Fs:(length(loud_signal)-1)/Fs; %time vector for loud_signal
phonems = ["SH", "A", "L", "O", "M"]; %phonems of 'SHALOM' vector
figure, subplot(311), plot(new_t,loud_signal); xlim([-0.05 1.1])
hold on;
for i=1:length(seg_ind)
    if i<=length(phonems)
        xline(new_t(seg_ind(i)),'r',phonems(i),'LineWidth',2);
    else
        xline(new_t(seg_ind(i)),'r','LineWidth',2)
    end
end
```

```
hold off
xlabel 'time [sec]' , ylabel 'amplitude' , title 'Signal segmentation'
subplot(312); scatter(1:length(Energy),Energy,'filled');
hold on;
xline(ind(1),'r','SH','LineWidth',2);xline(ind(2),'r','A','LineWidth',2);
xline(ind(3),'r','L','LineWidth',2);xline(ind(4),'r','O','LineWidth',2);xli
ne(ind(5),'r','M','LineWidth',2);
xline(ind(6),'r','LineWidth',2);
subplot(313);scatter(1:length(ZCR),ZCR,'filled');
hold on;
xline(ind(1),'r','SH','LineWidth',2);xline(ind(2),'r','A','LineWidth',2);
xline(ind(3),'r','L','LineWidth',2);xline(ind(4),'r','O','LineWidth',2);xli
ne(ind(5),'r','M','LineWidth',2);
xline(ind(6),'r','LineWidth',2);

%4.6:
% 'SH'
SH_Phoneme=loud_signal(seg_ind(1):seg_ind(2));
SH_framedPhoneme=FramedSig(ind(1):ind(2),:);
[SH_FeatsVector,Feat_title]=FeatExt(SH_Phoneme,Fs,SH_framedPhoneme);

% 'A'
A_Phoneme=loud_signal(seg_ind(2):seg_ind(3));
A_framedPhoneme=FramedSig(ind(2):ind(3),:);
[A_FeatsVector,Feat_title]=FeatExt(A_Phoneme,Fs,A_framedPhoneme);


% 'L'
L_Phoneme=loud_signal(seg_ind(3):seg_ind(4));
L_framedPhoneme=FramedSig(ind(3):ind(4),:);
[L_FeatsVector,Feat_title]=FeatExt(L_Phoneme,Fs,L_framedPhoneme);


% 'O'
O_Phoneme=loud_signal(seg_ind(4):seg_ind(5));
O_framedPhoneme=FramedSig(ind(4):ind(5),:);
[O_FeatsVector,Feat_title]=FeatExt(O_Phoneme,Fs,O_framedPhoneme);


% 'M'
M_Phoneme=loud_signal(seg_ind(5):end);
M_framedPhoneme=FramedSig(ind(5):ind(6),:);
[M_FeatsVector,Feat_title]=FeatExt(M_Phoneme,Fs,M_framedPhoneme);

AllFeat=table(Feat_title,SH_FeatsVector,A_FeatsVector,L_FeatsVector,O_Feats
Vector,M_FeatsVector);


%% 5:


%% 6:
[shalom,Fs] = audioread('shalom_example.wav');
sound(shalom,Fs)

[S,F,T] = spectrogram(shalom,hamming(500),250,[],Fs);

figure;imagesc(T,F,10*log10(abs(S)));
axis xy; xlabel('Time (s)');ylabel('Frequency (Hz)');
```

```
c = colorbar; c.Label.String = 'Power/frequency (dB/Hz)';

[S,F,T] = spectrogram(shalom,hamming(250),125,[],Fs);

figure;imagesc(T,F,10*log10(abs(S)));
axis xy; xlabel('Time (s)');ylabel('Frequency (Hz)');
c = colorbar; c.Label.String = 'Power/frequency (dB/Hz)';
```