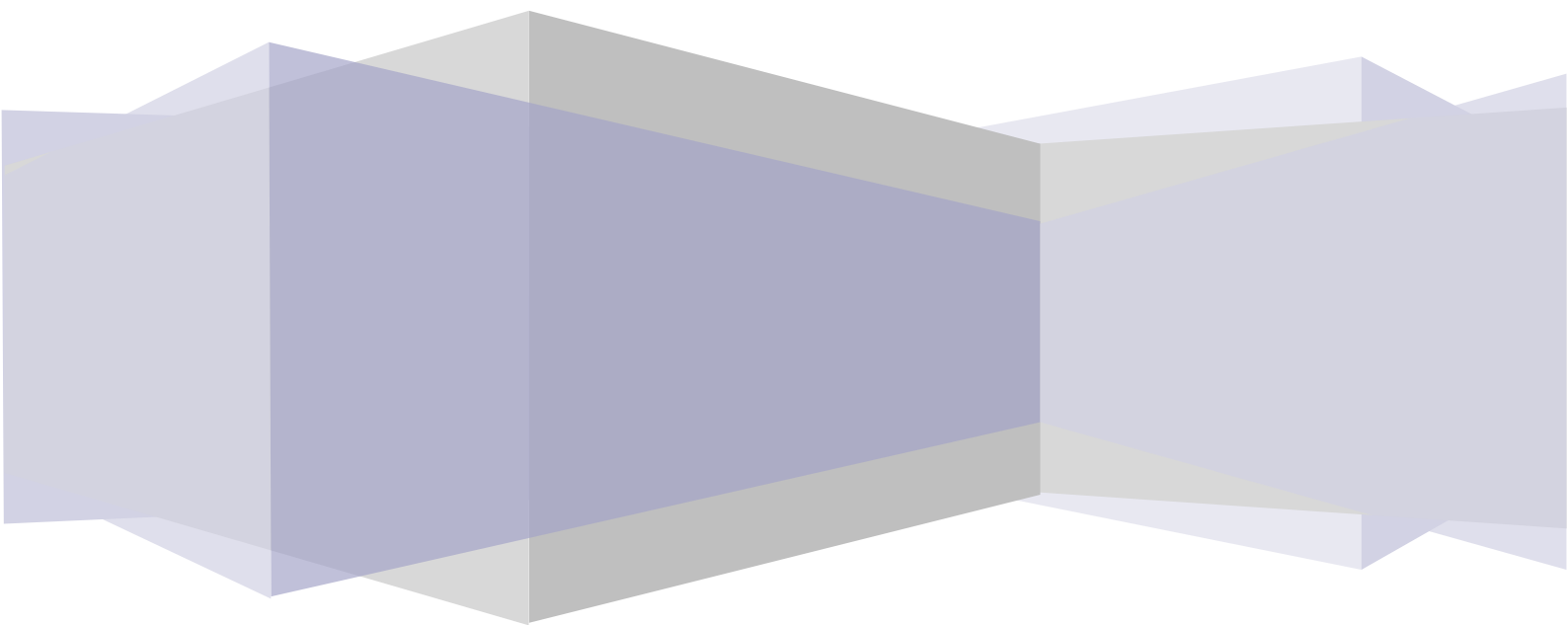


Digital Signal Processing

1η Εργαστηριακή Άσκηση: Εισαγωγή στην Ψηφιακή
Επεξεργασία Σημάτων με Matlab και Εφαρμογές σε
Ακουστικά Σήματα

Διδυμιώτου Καουκάκη Κωνσταντίνα AM 03115726

Κυριάκου Αθηνά AM 03117405



Ανάγκη για ανάλυση βραχέως χρόνου

Η ομιλία είναι ένα ακουστικό σήμα που παράγεται από ένα χρονικά μεταβαλλόμενο σύστημα με διέγερση που μεταβάλλεται στον χρόνο, αυτό των φωνητικών οδών. Καθώς λοιπόν το σύστημα παραγωγής δεν είναι χρονικά αμετάβλητο, δεν είναι δυνατή η μελέτη του σήματος με τις κλασικές μεθόδους ανάλυσης σημάτων που υποθέτουν ότι η συχνότητα και το φασματικό του περιεχομένου των παραγόμενων σημάτων είναι ανεξάρτητα του χρόνου.

Ωστόσο, μπορεί να θεωρηθεί ότι το σήμα της ομιλίας έχει χρονοανεξάρτητο φασματικό περιεχόμενο όταν αναλύεται σε πλαίσια χρονικής διάρκειας 10-30msec, οπότε και μπορεί να μελετηθεί με τις κλασικές μεθόδους. Η διαδικασία αυτή ονομάζεται ανάλυση βραχέως χρόνου STP (Short Term Processing) και πραγματοποιείται τόσο στο πεδίο του χρόνου όσο και της συχνότητας. Στο πεδίο του χρόνου υπολογίζονται χαρακτηριστικές παράμετροι του σήματος όπως η Ενέργεια Βραχέως Χρόνου STE (Short Time Energy) και ο Ρυθμός Μεταβολής Προσήμου ZCR (Zero Crossing Rate), ενώ στο πεδίο της συχνότητας ο Μετασχηματισμός Fourier Βραχέως Χρόνου STFT (Short Time Fourier Transform).

Τα μεγέθη αυτά θα προσδιοριστούν στα δύο πρώτα μέρη της άσκησης με χρήση του Matlab για να εξάγουμε χρήσιμα συμπεράσματα για το σήμα φωνής στο αρχείο `speech_utterance.wav` και το `music.wav`.

Επίσης στο τρίτο μέρος της άσκησης θα μελετηθεί η φασματική ανάλυση ημιτονοειδών non-stationary σημάτων με χρήση χρονό-συχνοτικών μετασχηματισμών του STFT και Wavelets Transform και θα παρατηρηθούν τα σπεκτρογράφηματά τους.

***Σημείωση:** Σας επισυνάπτουμε τρία αρχεία κώδικα έναν για κάθε άσκηση που εμφανίζουν σε διαφορετικά figures όλα τα ζητούμενα διαγράμματα. Επίσης επισυνάπτεται το ανακατασκευασμένο αρχείο μουσικής με (`speech_utterance_rec.wav`) και αρχείο κώδικα της συνάρτησης `mySTFT`. Για να εμφανιστούν τα διαγράμματα για το αρχείο μουσικής αλλάζουμε το πεδίο της `audioread()` με όρισμα το `music.wav`.

Μέρος 1ο - Χαρακτηριστικά Βραχέος Χρόνου Σημάτων Φωνής και Μουσικής (Ενέργεια και Ρυθμός Εναλλαγής Προσήμου).

Σκοπός της άσκησης είναι ο διαχωρισμός φωνής από σιωπής και έμφωνους (όπως /aa/, /ih/) από άφωνους (όπως /f/, /p/) ήχους μετρώντας την ενέργεια βραχέος χρόνου και στη συνέχεια μετρώντας το ρυθμό εναλλαγής πρόσημου.

Για τα ακουστικά σήματα υπολογίστηκαν η Ενέργεια Βραχέος Χρόνου (STE) και ο Ρυθμός Εναλλαγής Προσήμου (ZCR) με συχνότητα δειγματοληψίας $fs=16kHz$ και παράθυρο Hamming βάσει των τύπων:

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

$$Z_n = \sum_{m=-\infty}^{\infty} |sgn[x[m]] - sgn[x[m-1]]|w[n-m]$$

Ερώτημα 1.1:

Αρχικά για το ακουστικό σήμα '**speech_utterance.wav**', που παίρνουμε με την `audioread()`, έχουμε για παράθυρο Hamming μήκους 20msec:

```
%~~~~~1.1 Short Time Energy ~~~~~
sig_framed=buffer(y,w,w-1,'nodelay'); %the framed signal
[n,m]=size(sig_framed); %n=rows, m=columns
for i=1:m
    sig_windowed(:,i)=sig_framed(:,i).*h; %windowing
end;

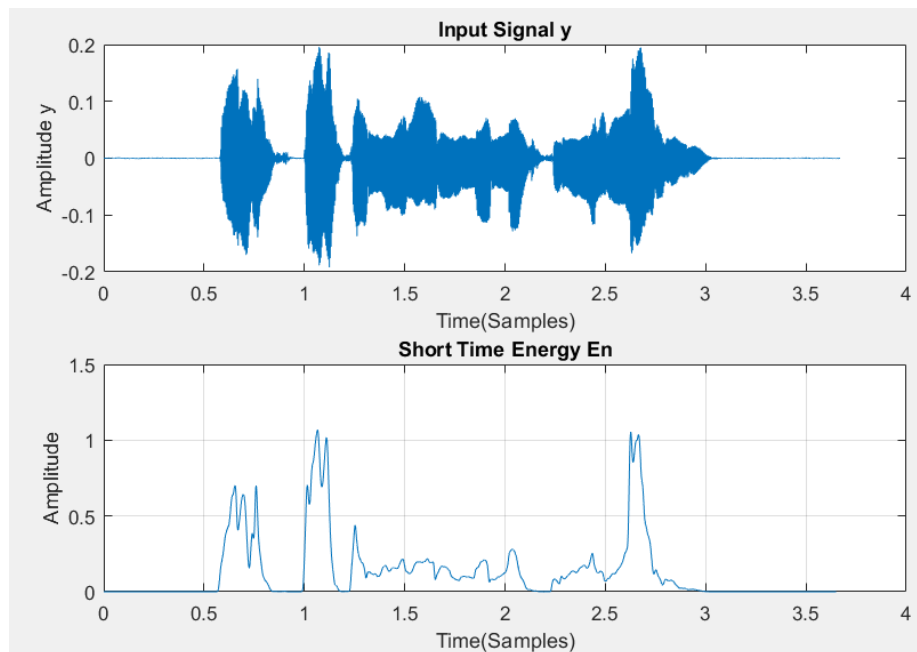
En=sum(sig_windowed.^2,1); %Short Time Energy (STE)

figure;
subplot(2,1,1);
t2=[0:length(y)-1]/fs;
plot(t2,y);
title(['Input Signal y']);
xlabel('Time(Samples)');
ylabel('Amplitude y');

subplot(2,1,2);
t1=[0:length(En)-1]/fs;
plot(t1,En);
title(['Short Time Energy En']);
xlabel('Time(Samples)');
ylabel('Amplitude');
grid on;
```

Σχήμα 1.1a: Κομμάτι κώδικα για την εμφάνιση του σήματος
και του STE για το σήμα ομιλίας

Και τα διαγράμματα που παίρνουμε είναι τα ακόλουθα:



Σχήμα 1.1b: STE για το σήμα ομιλίας για μήκος παραθύρου 20msec

Για το Zero Crossing Rate για το ακουστικό σήμα 'speech_utterance.wav' έχουμε για παράθυρο Hamming μήκους 20msec:

```
%~~~~~1.1 Zero Crossing Rate ~~~~~
sig_signed(1)=abs(sign(y(1)));
for i=2:length(y)
    sig_signed(i)=abs(sign(y(i))-sign(y(i-1)));    %signal=|sgn(y)-sgn(y-1)|
end;

sig_signed=sig_signed';

sig_framed=buffer(sig_signed,w,w-1,'nodelay');    %new framed signal

[n,m]=size(sig_framed);
for i=1:m
    sig_windowed(:,i)=sig_framed(:,i).*h;
end;

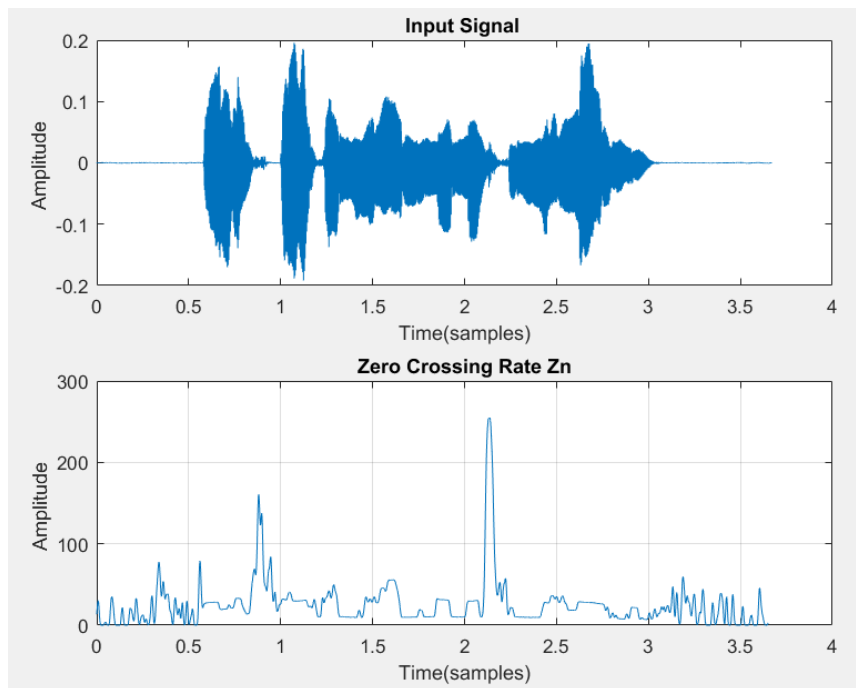
Zn=sum(sig_windowed,1);    %Zero Crossing Rate (ZCR)

figure;
subplot(2,1,1);
t2=[0:length(y)-1]/fs;
plot(t2,y);
title(['Input Signal ']);
xlabel('Time(samples)');
ylabel('Amplitude ');

subplot(2,1,2);
t1=[0:length(Zn)-1]/fs;
plot(t1,Zn);
title(['Zero Crossing Rate Zn']);
xlabel('Time(samples)');
ylabel('Amplitude ');
grid on;
```

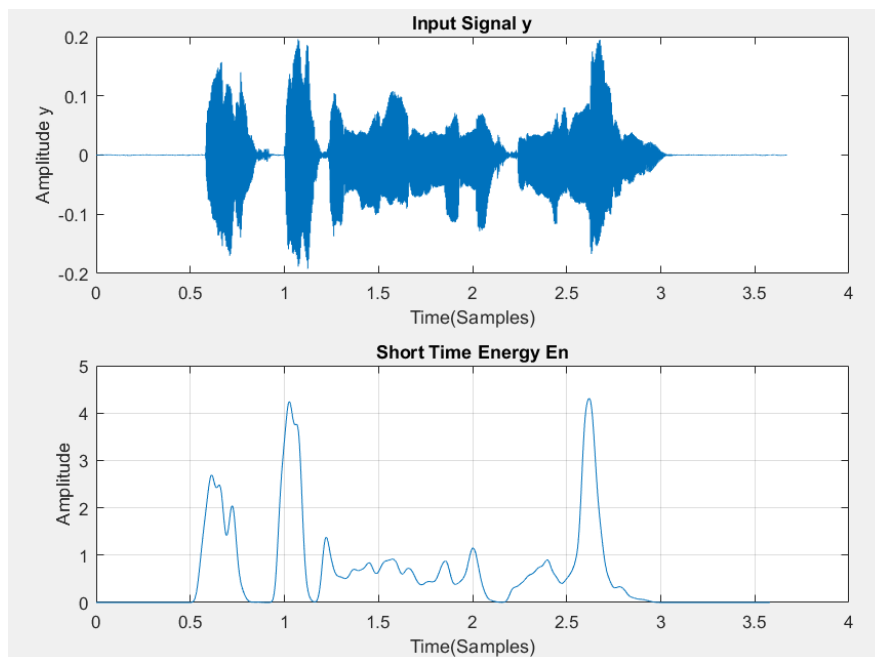
Σχήμα 1.1c: Κομμάτι κώδικα για υλοποίηση ZCR για το σήμα ομιλίας

Και έχουμε τα εξής διαγράμματα:

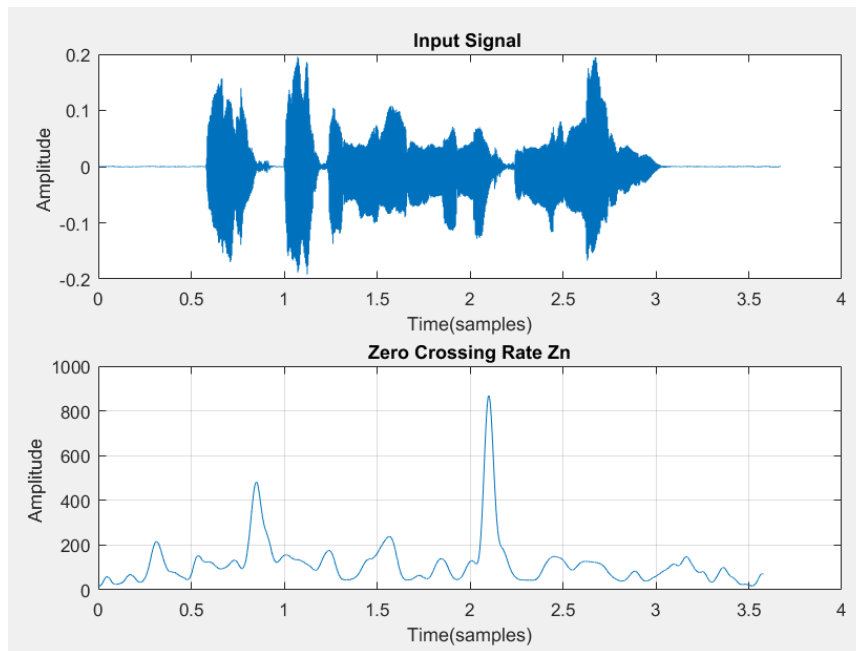


Σχήμα 1.1d: ZCR για το σήμα φωνής με μήκος παραθύρου 20 msec

→ Όμοια μεγαλώνοντας στον κώδικα του σχήματος 1.1a και 1.1c το μήκος του παραθύρου σε 90 msec προκύπτουν τα εξής διαγράμματα για STE και ZCR:



Σχήμα 1.1e: STE για το σήμα φωνής με μήκος παραθύρου 90 msec



Σχήμα 1.1f: ZCR για το σήμα ομιλίας με μήκος παραθύρου 90msec

Παρατηρήσεις:

- Από τη θεωρία γνωρίζουμε ότι ανάλογα με τη διέγερση του συστήματος των φωνητικών οδών, το σήμα της ομιλίας μπορεί να κατηγοριοποιηθεί σε τρεις περιοχές. Αν η διέγερση είναι σχεδόν περιοδική τότε πρόκειται για εμφωνία, αν είναι τυχαία για αφωνία και αν δεν υπάρχει διέγερση πρόκειται για σιωπή. Στην πράξη όμως μπορεί να υπάρχουν και περιοχές τόσο εμφωνίας όσο και αφωνίας.

Οπότε, σε ένα διάγραμμα του πλάτους του ακουστικού σήματος ως προς τον χρόνο, οι περιοχές που εμφανίζουν υψηλή περιοδικότητα αναπαριστούν έμφωνους ήχους, αυτές με μη μηδενικό πλάτος και χωρίς περιοδικότητα τους άφωνους και αυτές με σχεδόν αμελητέο πλάτος αναπαριστούν τη σιωπή. Η σιωπή συνήθως βρίσκεται ανάμεσα σε έμφωνες και άφωνες περιοχές. Με βάση αυτά, λοιπόν με βάση το STE μπορούμε να ξεχωρίσουμε έμφωνους και άφωνους ήχους αφού οι έμφωνοι αντιστοιχούν σε μεγάλη ενέργεια, οι άφωνοι σε χαμηλότερες τιμές ενέργειας και η σιωπή σε σχεδόν μηδενική (Σχήμα 1.1b).

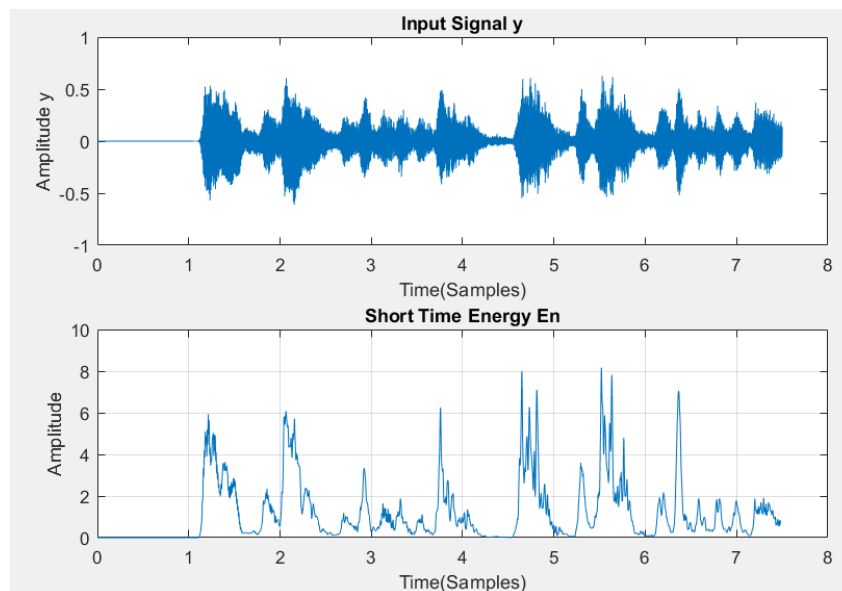
- Όσον αφορά τον ρυθμό εναλλαγής προσήμου (ZCR) θα ισχύει το αντίθετο από την STE, όπου η μία λαμβάνει σχετικά μεγάλη τιμή η άλλη λαμβάνει αντίστοιχα μικρή τιμή. Αν η τιμή του ZCR είναι υψηλή σημαίνει ότι το σήμα μεταβάλλεται γρήγορα και για αυτό εμφανίζει υψηλό συχνοτικό περιεχόμενο, αντίθετα αν είναι χαμηλή έχουμε περισσότερο σταθερότητα. Άφωνοι έχουν υψηλό ZCR, έμφωνοι χαμηλότερο ενώ τα σημεία σιωπής όπου παρουσιάζεται μηδενική ενέργεια, ενώ

χαρκτηρίζονται από μεγάλο ρυθμό ZCR. Αυτό οφείλεται στον τυχόν θόρυβο που μπορεί να υπάρχει κατά την ομιλία και κατά την σιωπή.

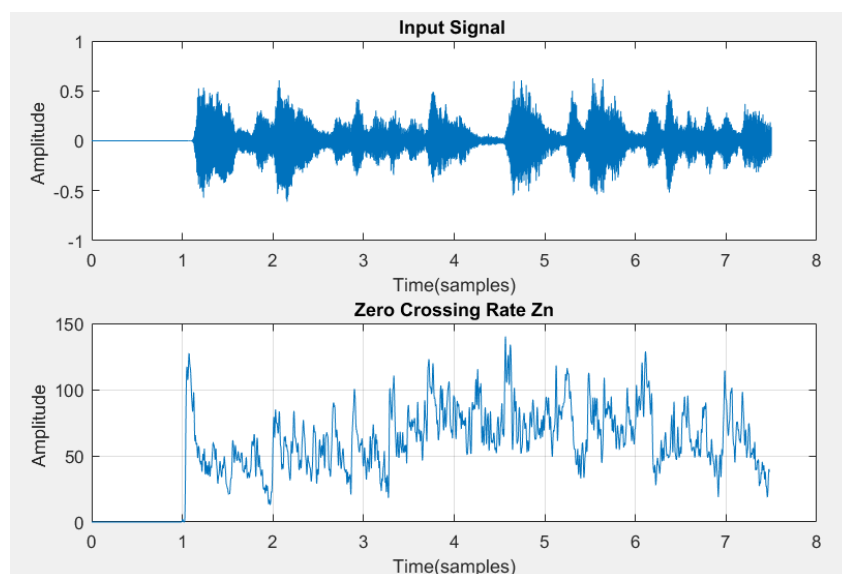
- Όταν αυξήσαμε το μήκος παραθύρου σε 90 msec (σχήματα 1.1e και 1.1f) παρατηρούμε ότι οι μεταβάσεις και μεταβολές είναι πιο ομαλές και δεν έχουμε μεγάλη λεπτομέρεια στις μετρήσεις μας και άρα ακρίβεια για να βγάλουμε συμπεράσματα. Αυτό συμβαίνει γιατί σε διάστημα 90 msec έχουμε ανάλυση σε περισσότερα δείγματα και είναι λογικό να μειώνεται η ακρίβεια.

Ερώτημα 1.2:

Ομοίως με το ερώτημα 1.1, αλλάζοντας τον κώδικα παίρνουμε το σήμα 'music.wav' με την συνάρτηση `audioread()` και χρησιμοποιώντας παράθυρο Hamming αρχικά 20 msec και με 90 msec στην συνέχεια παίρνουμε τα εξής διαγράμματα:

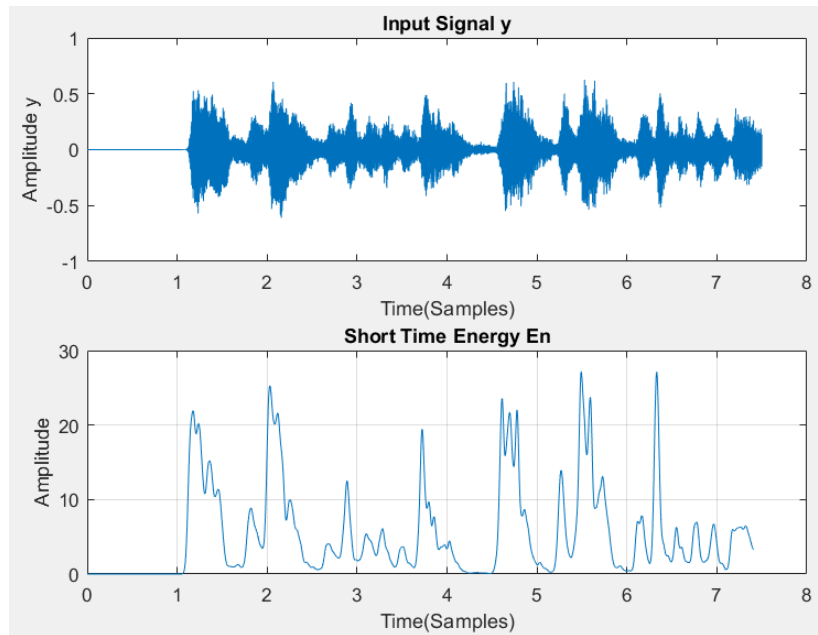


Σχήμα 1.2a: STE για το σήμα μουσικής για μήκος παραθύρου 20 msec

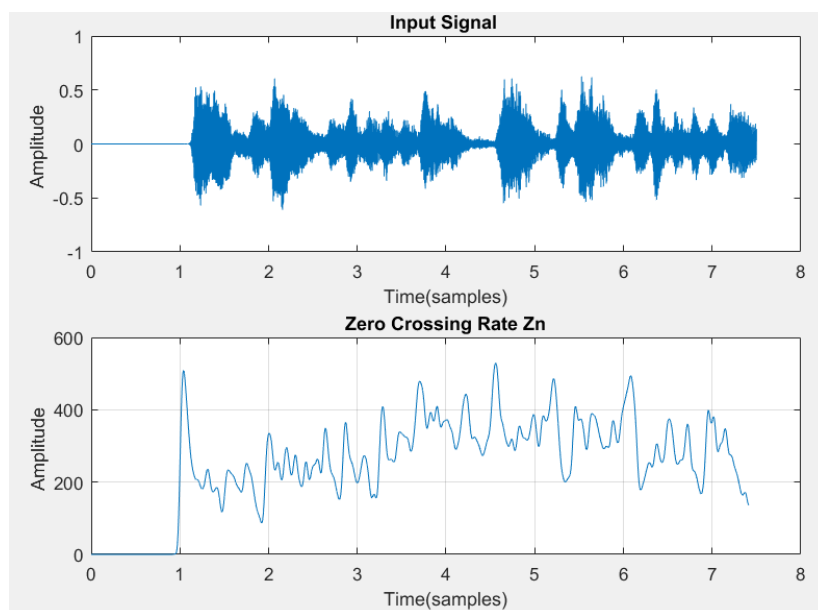


Σχήμα 1.2b: ZCR για το σήμα μουσικής για μήκος παραθύρου 20 msec

→ Όμοια μεγαλώνοντας στον κώδικα το μήκος του παραθύρου σε 90 msec προκύπτουν τα εξής διαγράμματα για STE και ZCR για το σήμα μουσικής:



Σχήμα 1.2c: STE για το σήμα μουσικής για μήκος παραθύρου 90 msec



Σχήμα 1.2d: ZCR για το σήμα μουσικής για μήκος παραθύρου 90 msec

Μέρος 2ο - Ανάλυση και Σύνθεση Σήματος με τον Μετασχηματισμό Fourier Βραχέος Χρόνου (STFT)

Στο δεύτερο μέρος της εργαστηριακής άσκησης, υπολογίστηκε με χρήση του Matlab ο STFT για δύο ηχητικά σήματα, ένα ομιλίας (speech_utterance.wav) και ένα μουσικής (music.wav) καθώς και ανακατασκευάστηκε το σήμα ομιλίας. Ο STFT χρησιμοποιείται για την ανάλυση του φασματικού περιεχομένου σημάτων όταν αυτό μεταβάλλεται ως προς τον χρόνο και τα αποτελέσματα του αναπαρίστανται σε ένα τριαξονικό διάγραμμα (χρόνος, συχνότητα, μέγεθος), το σπεκτρογράφημα.

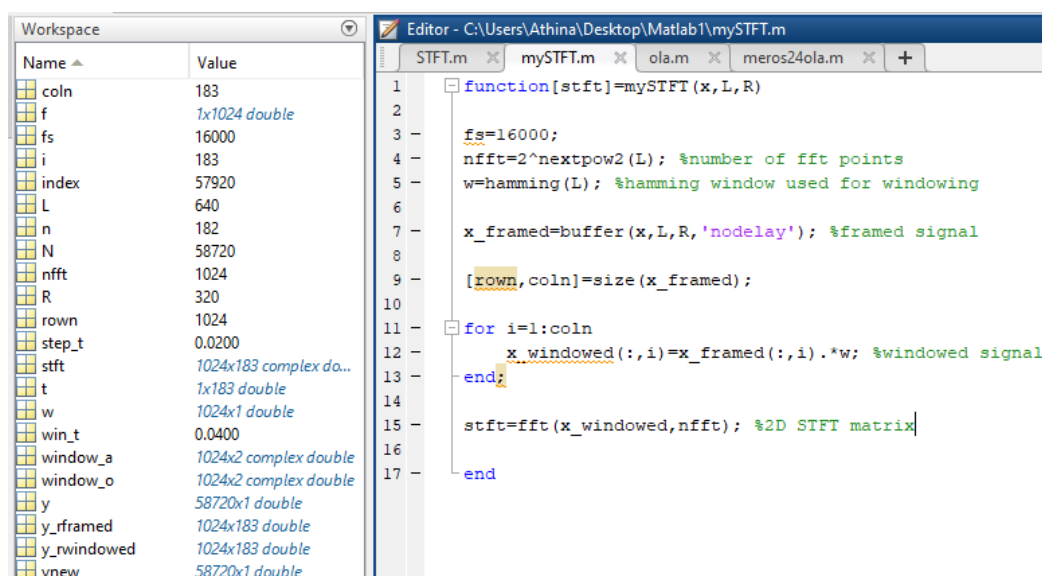
Ο τύπος που χρησιμοποιήθηκε είναι :

$$\text{STFT}(\tau, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n - \tau]e^{-j\omega n}, \text{ όπου } \tau \in Z.$$

Ερώτημα 2.1:

Ο κώδικας για τον υπολογισμό του STFT βρίσκεται στο επισυναπτόμενο αρχείο STFT.m ο οποίος καλεί τη συνάρτηση mySTFT.

Η συνάρτηση παίρνει ως ορίσματα το διάνυσμα x, που αναπαριστά το δειγματοληπτημένο σήμα από το αρχείο .wav με τη συνάρτηση audioread() του Matlab με συχνότητα δειγματοληψίας fs=16kHz, το μήκος του παραθύρου L και το βήμα ολίσθησης R και επιστρέφει τον STFT του σήματος σε μορφή διδιάστατου πίνακα 1024x183.



The screenshot shows the MATLAB Editor and Workspace. The Editor window displays the script `STFT.m` with the following code:

```
1 function [stft]=mySTFT(x,L,R)
2
3     fs=16000;
4     nfft=2^nextpow2(L); %number of fft points
5     w=hamming(L); %hamming window used for windowing
6
7     x_framed=buffer(x,L,R,'nodelay'); %framed signal
8
9     [rown,coln]=size(x_framed);
10
11     for i=1:coln
12         x_windowed(:,i)=x_framed(:,i).*w; %windowed signal
13     end;
14
15     stft=fft(x_windowed,nfft); %2D STFT matrix
16
17 end
```

The Workspace window on the left shows the following variables and their values:

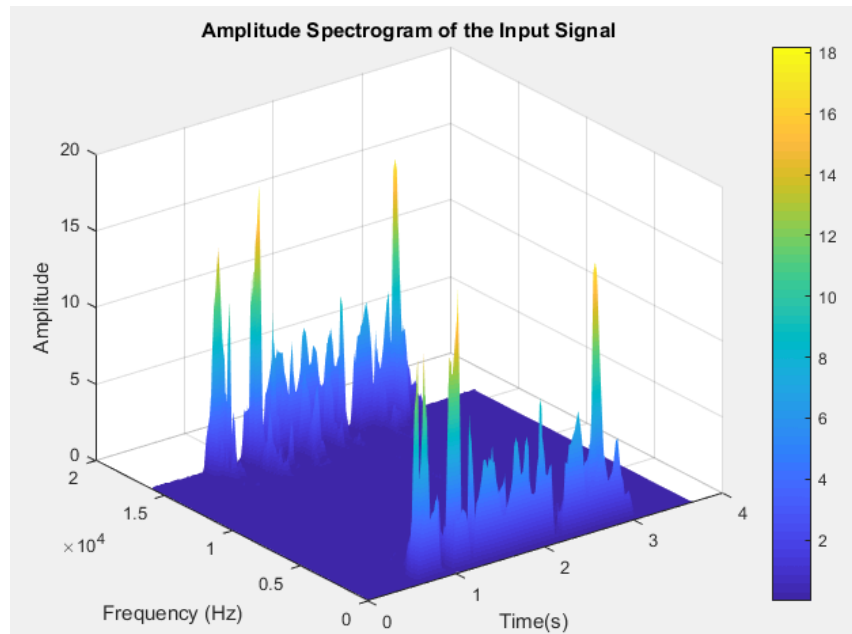
Name	Value
coln	183
f	1x1024 double
fs	16000
i	183
index	57920
L	640
n	182
N	58720
nfft	1024
R	320
rown	1024
step_t	0.0200
stft	1024x183 complex da...
t	1x183 double
w	1024x1 double
win_t	0.0400
window_a	1024x2 complex double
window_o	1024x2 complex double
y	58720x1 double
y_framed	1024x183 double
y_rwindowed	1024x183 double
ynew	58720x1 double

Σχήμα 2.2: Υλοποίηση της συνάρτησης mySTFT

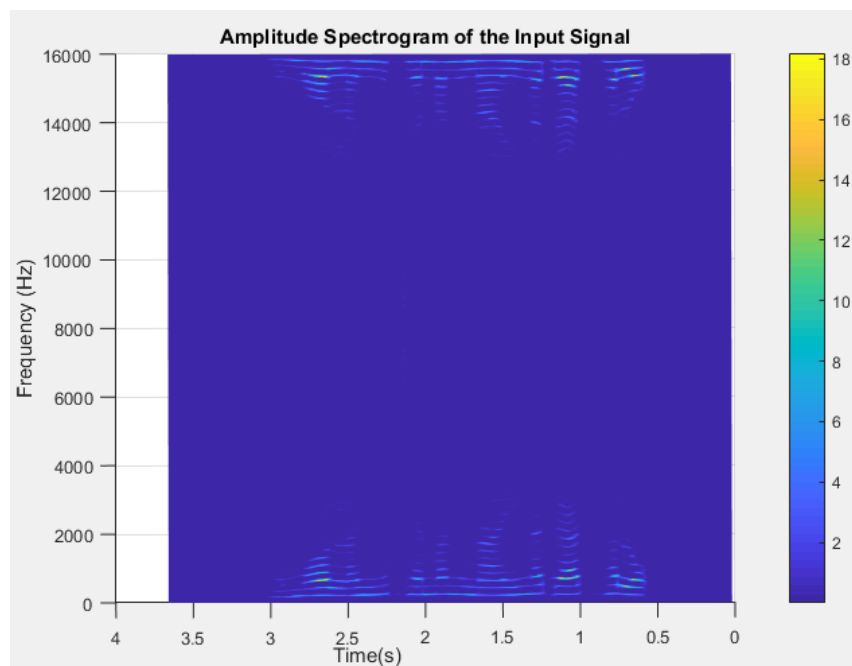
Η συνάρτηση κλήθηκε με όρισμα το διάνυσμα y που προκύπτει από την συνάρτηση `[y,fs] = audioread('speech_utterance.wav');` .

Ερώτημα 2.2:

Για την αναπαράσταση του πλάτους του STFT χρησιμοποιήθηκε η συνάρτηση `surf()`. Το διάγραμμα που προκύπτει φαίνεται στο Σχήμα 2.2α,b.



Σχήμα 2.2α: Σπεκτρογράφημα πλάτους STFT





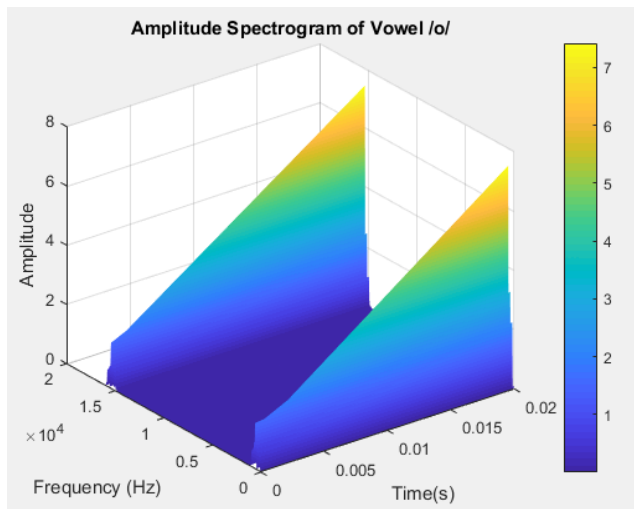
Σχήμα 2.2b: Εξάρτηση συχνότητας από τον χρόνο

Στο αρχείο `speech_utterance.wav`, παρατηρήθηκε ότι το φωνήεν της δοσμένης πρότασης /o/ εμφανίζεται για χρόνους [0.570688, 0.662527] και το /a/ για χρόνους [0.748243, 0.797224].

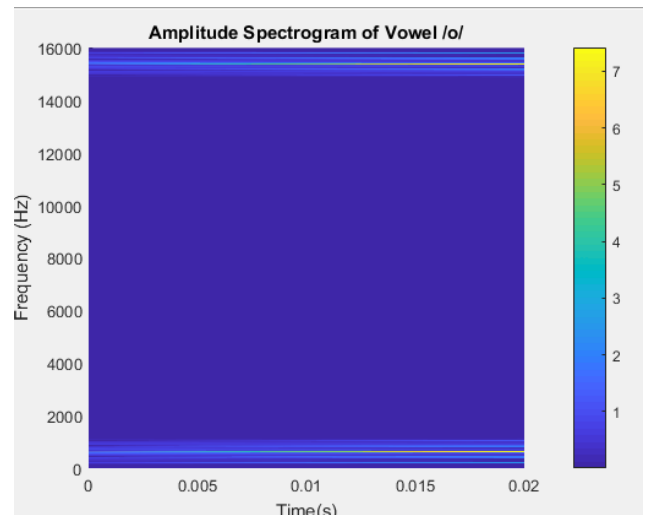
Απομονώθηκαν δύο παράθυρα για κάθε φωνήεν και κλήθηκε η συνάρτηση `surf()` για να προκύψει το σπεκτρογράφημα του πλάτους του STFT αυτών των παραθύρων (Σχήματα 2.2c,d και 2.2e,f).

```
% _____ 2.2 _____
%vowel /a/ and /o/
window_o=stft(:,round(0.580/step_t):round(0.600/step_t));
window_a=stft(:,round(0.770/step_t):round(0.790/step_t));
```

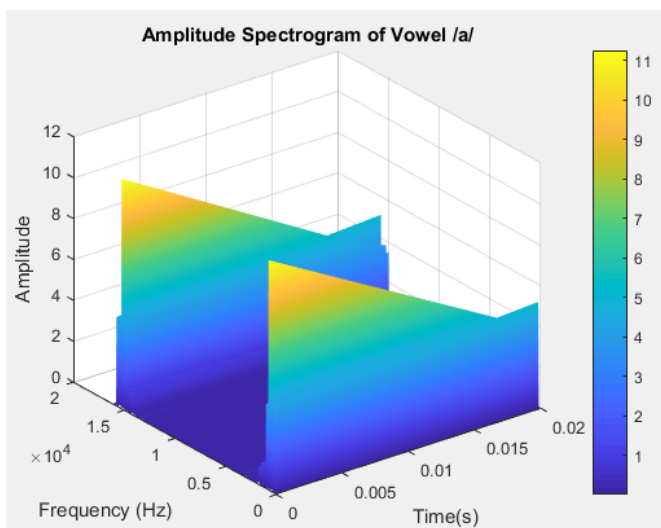
 `window_a` *1024x2 complex double*
 `window_o` *1024x2 complex double*



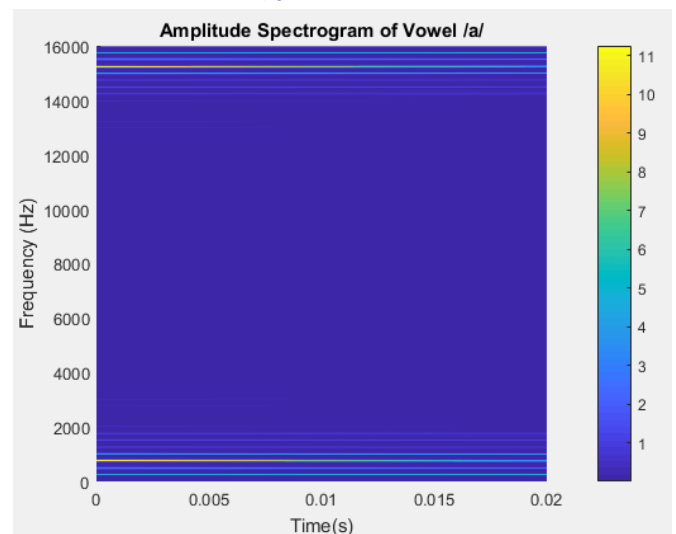
Σχήμα 2.2c: Σπεκτρογράφημα πλάτους /o/



Σχήμα 2.2d: Ανεξαρτησία συχνότητας από τον χρόνο /o/



Σχήμα 2.2e: Σπεκτρογράφημα πλάτους /a/



Σχήμα 2.2f: Ανεξαρτησία συχνότητας από τον χρόνο /a/

Από τα διαγράμματα παρατηρούμε ότι επειδή μελετάμε το ηχητικό σήμα κάθε φωνήεντος σε χρονικό διάστημα 0.02sec, η συχνότητα είναι ανεξάρτητη των χρονικών μεταβολών και άρα η γραφική παράσταση είναι ευθεία γραμμή στα διαγράμματα χρόνου-συχνότητας (Σχήματα 2.2d και 2.2f).

Επίσης, από τα παραπάνω σχήματα προκύπτει ότι τα φωνήεντα μπορούν να διακριθούν βάσει του πλάτους τους και της συχνότητάς τους. Συγκεκριμένα, παρατηρούμε ότι τόσο το peak του πλάτους του /a/ όσο και η συχνότητά του (pitch) είναι μεγαλύτερα από του /o/ .

Ερώτημα 2.3:





Για την ανακατασκευή του αρχικού σήματος από τον STFT με ακρίβεια μίας πολλαπλασιαστικής σταθεράς υλοποιήθηκε η αντίστροφη διαδικασία με χρήση της τεχνικής OverLap-Add και της συνάρτησης `ifft()` του Matlab. Το ανακατασκευασμένο σήμα αποθηκεύτηκε στο αρχείο ήχου `speech_utterance_rec.wav` με χρήση της `audiowrite()`.

```
% 2.3
%signal reconstruction
y_rwindowed=ifft(stft,nfft); %reconstructed windows

for i=1:coln
    y_rframed(:,i)=real(y_rwindowed(:,i)).*w; %reconstructed frames
end;

ynew=zeros(N,1); %initialization reconstructed signal

%time_shift
for n=1:coln
    index=(n-1)*R;
    if index+nfft>N, break; end
    ynew(index+1:index+nfft)=ynew(index+1:index+nfft)+y_rframed(:,n);
end
```

	y	58720x1 double
	y_rframed	1024x183 double
	y_rwindowed	1024x183 double
	ynew	58720x1 double

Ερώτημα 2.4:

Σύμφωνα με τη θεωρία για την Overlap-Add Ανάλυση, για να μπορέσουμε να ανακατασκευάσουμε το σήμα από τα επιμέρους επικαλυπτόμενα παράθυρα θα πρέπει να ισχύει, όπου $x_m[n]$ το παραθυρωποιημένο πλαίσιο δεδομένων και $w[n]$ το μηδενικής φάσης παράθυρο που ολισθαίνουμε:

$$x[n] = \sum_{m=-\infty}^{\infty} x_m[n] = x[n] \sum_{m=-\infty}^{\infty} w[n - mR]$$

Οπότε, $x[n] = \sum_m x_m[n]$ αν και μόνο αν

$$\sum_{m \in \mathbf{Z}} w[n - mR] = 1, \forall n \in \mathbf{Z}.$$

Άρα, η ανακατασκευή του σήματος είναι επιτυχής αν το άθροισμα των μετατοπισμένων παραθύρων είναι 1.

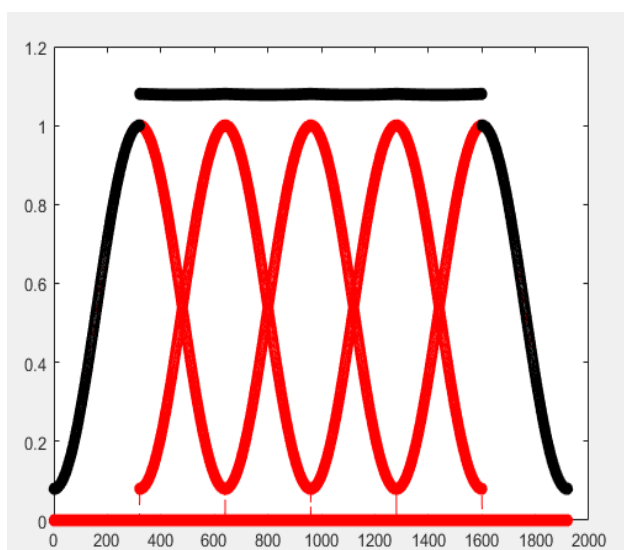
Καλώντας τη συνάρτηση ola.m (ola_call.m) του συμπληρωματικού υλικού για $M=640$, $R=320$ και $N=1920$, προέκυψε το Σχήμα 2.4a.

```
1 %ola_call
2
3 - fs=16000;
4 - win_t=0.04;
5 - step_t=0.02;
6
7 - M=win_t*fs; %window length
8 - R=step_t*fs; %hop size
9 - N=3*M; % winoverlap-add span
10
11 - ola(M,R,N);
12 |
```

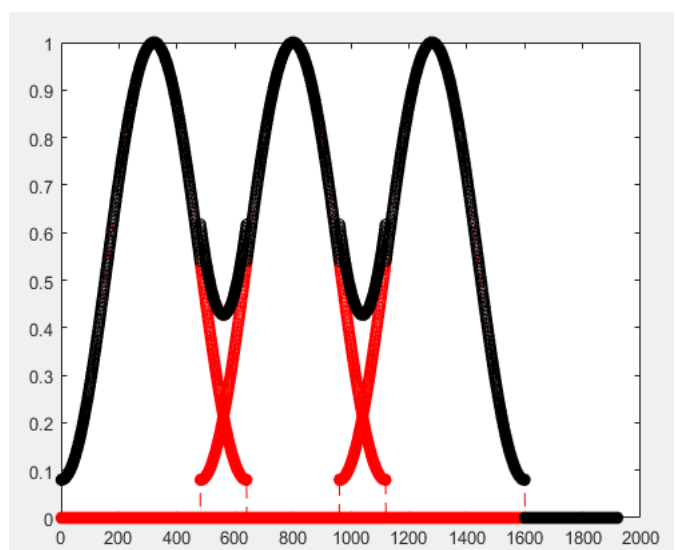
Καλώντας τη συνάρτηση ola.m (ola_call.m) του συμπληρωματικού υλικού για $M=640$, $R=480$ και $N=1920$, προέκυψε το Σχήμα 2.4b.

Ο οριζόντιος άξονας αναπαριστά τα δείγματα και ο κατακόρυφος το πλάτος του παραθύρου Hamming. Τα μετατοπισμένα παράθυρα Hamming απεικονίζονται με κόκκινο χρώμα ενώ με μαύρο το άθροισμά τους για το συγκεκριμένο πλήθος ολισθήσεων.

Στην πρώτη περίπτωση, παρατηρούμε ότι το άθροισμα των παραθύρων είναι 1 (με εξαίρεση τα άκρα) για αυτό είναι επιτυχής η ανακατασκευή του σήματος, ενώ στη 2^η δεν είναι και για αυτό το ανακατασκευασμένο σήμα ακούγεται παραμορφωμένο.



Σχήμα 2.4a: Εκτέλεση της ola_call για $M=640$, $R=320$ και $N=1920$



Σχήμα 2.4b: Εκτέλεση της ola_call για $M=640$, $R=480$ και $N=1920$

Μέρος 3ο - Φασματική Ανάλυση Ημιτονοειδών και Ανίχνευση Απότομων Μεταβάσεων με τον Μετ/σμό Fourier Βραχέος Χρόνου (STFT) και τον Μετ/σμό Wavelets (διακριτοποιημένο DT-CWT).

Σκοπός της άσκησης είναι η Φασματική Ανάλυση Ημιτονοειδών και μελέτη συχνοτικού περιεχομένου με χρήση του χρονο-συχνοτικού μετασχηματισμού Short Time Fourier Transform (STFT) αλλά και με τον επίσης χρονο-συχνοτικό μετασχηματισμό Wavelets σε διακριτά σήματα Discrete-time Continuous Wavelet Transform (DT-CWT).

$$\text{STFT}(\tau, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n - \tau]e^{-j\omega n}, \text{ όπου } \tau \in \mathbb{Z}.$$

$$\text{CWT}(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t)\psi^*\left(\frac{t - \tau}{s}\right) dt,$$

Σχήμα 3.0: Μετασχηματισμοί Fourier SFTP & CWT

Ερώτημα 3.1

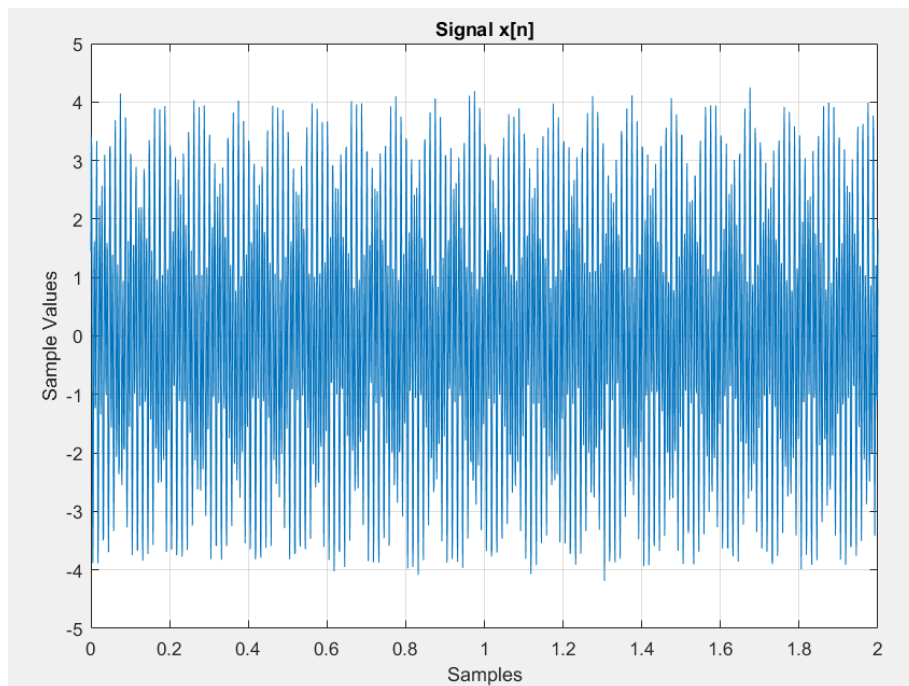
Έχουμε το σήμα $x(t) = 1.5 \cos(2\pi 80t) + 2.5 \sin(2\pi 150t) + 0.15u(t)$, όπου $u(t)$ λευκός Gaussian θόρυβος μηδενικής μέσης τιμής, και το δειγματοληπτούμε με συχνότητα δειγματοληψίας $f_s=1000\text{Hz}$ στο διάστημα $t \in [0,2]$ sec και προκύπτει το $x[n]$.

```
fs=1000;
window_sec=0.04;
L=window_sec*fs;    %the window length

%~~~~~3.1a Signal x[n]~~~~~
t=[0:1/fs:2];      %sampling
u=randn(1,length(t)); %gaussian white noise
xn=1.5*cos(2*pi*80*t)+2.5*sin(2*pi*150*t)+0.15*u;
figure;
plot(t,xn);
title(['Signal x[n]']);
xlabel('Samples');
ylabel('Sample Values');
grid on;
```

Σχήμα 3.1: Κομμάτι κώδικα για την υλοποίηση του σήματος $x[n]$

(α) Το σήμα $x[n]$ που προκύπτει μετά την δειγματοληψία είναι:



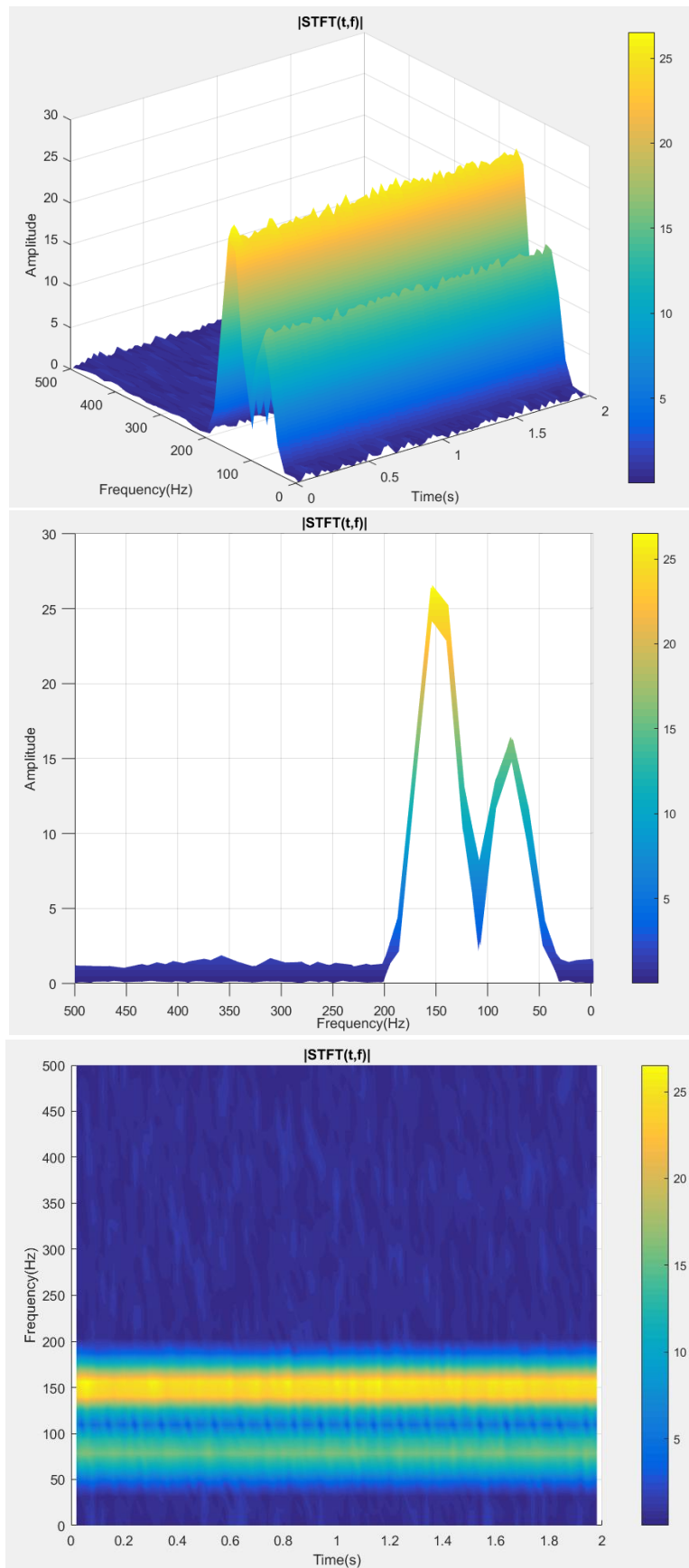
Σχήμα 3.1a: Λαμβανόμενο σήμα $x[n]$

(β) Θα υπολογίσουμε τον SFTP με χρήση της ρουτίνας `spectrogram` και θα εμφανίσουμε το πλάτος $|STFT(\tau, f)|$ σε τριαξονικό διάγραμμα (σπεκτρογράφημα) με χρήση της συνάρτησης `surf()` του Matlab που σαν ορίσματα θα παίρνει το T, F, S της συνάρτησης `spectrogram`. Χρησιμοποιήθηκε μήκος παραθύρου 0.04 sec και επικάλυψη 0.02 sec.

```
%~~~~~3.1b Short Time Fourier Transform ~~~~~~
[S,F,T]=spectrogram(xn,L,L/2,2^(nextpow2(L)),fs,'yaxis');
figure;
surf(T,F,abs(S),'edgecolor','none');
shading interp;
title(['|STFT(t,f)|']);
xlabel('Time(s)');
ylabel('Frequency(Hz)');
zlabel('Amplitude');
grid on;
colorbar;
```

Σχήμα 3.1b: Κομμάτι κώδικα για την υλοποίηση του STFT και την απεικόνιση του σπεκτρογραφήματος

Και θα έχουμε τις παρακάτω απεικονίσεις για το πλάτος του μετασχηματισμού STFT (πήραμε διάφορα στιγμιότυπα με την χρήση του `Rotate 3D`):



Σχήμα 3.1c: Σπεκτρογραφήματα για το πλάτος του Μετ/σμού STFT

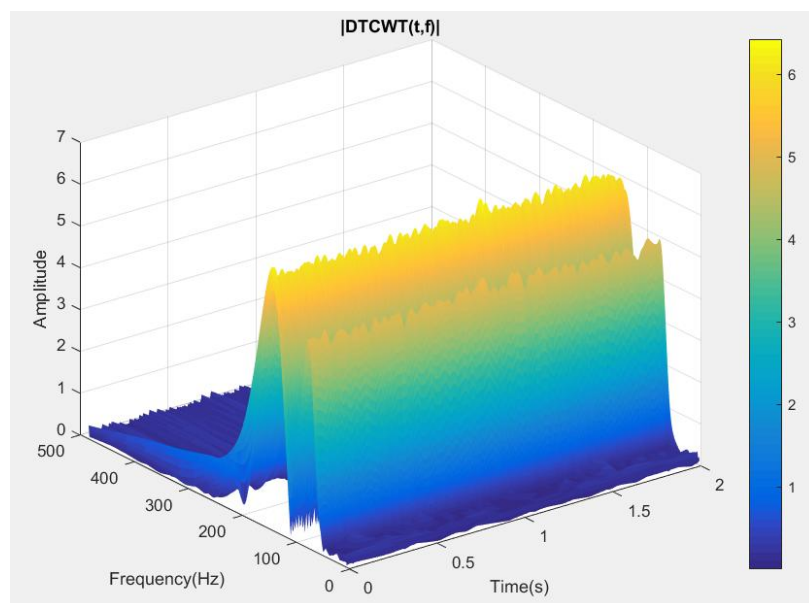
(γ) Τώρα θα υπολογίσουμε τον DT-CWT με χρήση των ρουτινών `cwtft` και `wavescales` του συμπληρωματικού υλικού και θα εμφανίσουμε το πλάτος $|DT_CWT(\tau, f)|$ σε τριαξονικό διάγραμμα (σπεκτρογράφημα) με χρήση της συνάρτησης `surf()` του Matlab που σαν ορίσματα θα παίρνει το t, f και `cfs` της συνάρτησης `cwtft()`. Χρησιμοποιήθηκε μήκος παραθύρου 0.04 sec και επικάλυψη 0.02 sec και σαν wavelet χρησιμοποιήθηκε το Morlet Wavelet (ή Gabor Wavelet).

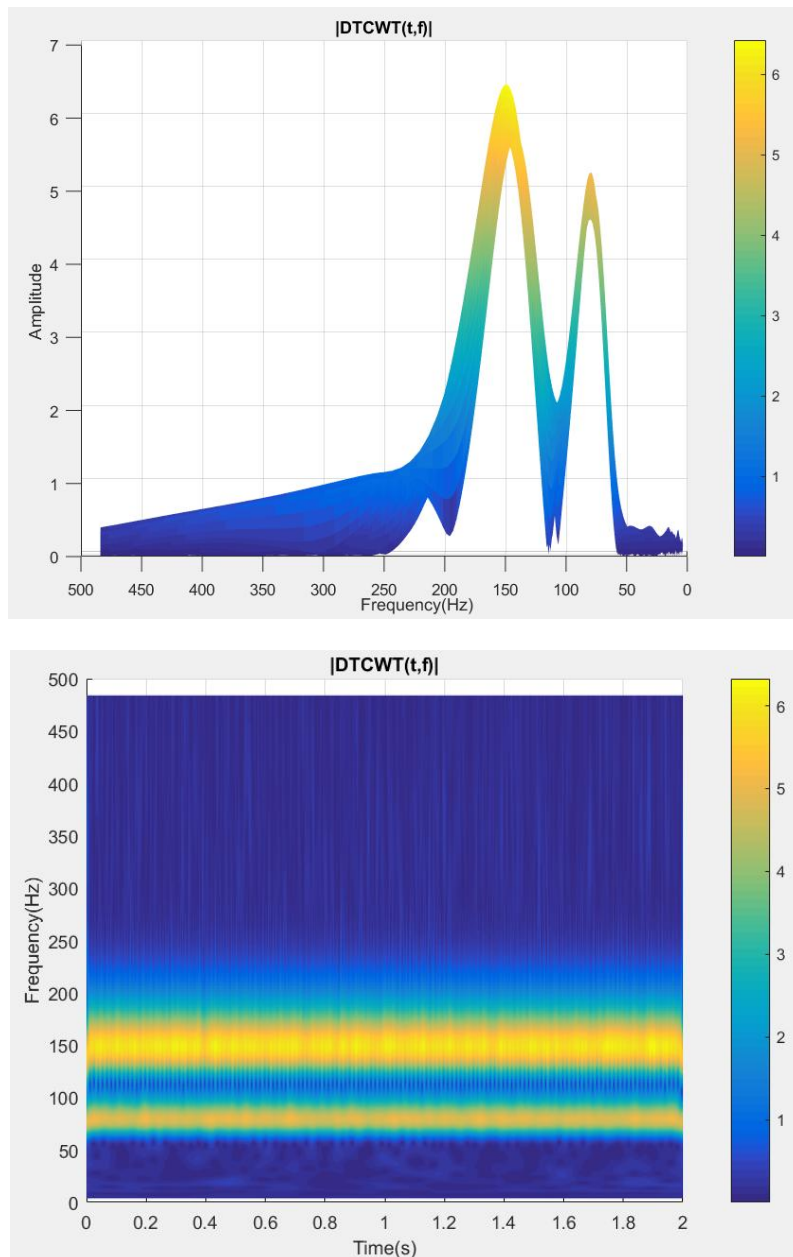
```
%~~~~~3.1c Discrete Time Continuous Wavelet Transform~~~~~
[s,f]=wavescales('morl',fs); %use function [s,f]= wavescales(wavelet,Fs) with wavelet=Morlet
% CWTSTRUCT = CWTFT(SIG,'scales',SCA,'wavelet',WAV)
%from the file
%==== To obtain the CWT coefficients =====
% cwtstruct = cwtft({x,1/Fs},'Scales',scales,'Wavelet','morl');
% cfs = cwtstruct.cfs;

cwtstruct=cwtft({xn,1/fs},'scales',s,'wavelet','morl'); |
figure;
cfs=cwtstruct.cfs; %cfs:coefficients of wavelet transform.
surf(t, f, abs(cfs) , 'edgecolor' , 'none');
title(['|DTCWT(t,f)|']);
xlabel('Time (s)');
ylabel('Frequency (Hz)');
zlabel('Amplitude');
grid on;
colorbar;
```

Σχήμα 3.1d: Κομμάτι κώδικα για την υλοποίηση του Μετ/σμού DT-CWT και την απεικόνιση του σπεκτρογραφήματος

Και θα έχουμε τις παρακάτω απεικονίσεις για το πλάτος του μετασχηματισμού DT_CWT (πήραμε διάφορα στιγμιότυπα με την χρήση του Rotate 3D):





Σχήμα 3.1ε: Σπεκτρογραφήματα για το πλάτος του Μετ/σμού DT_CWT

(δ) Παρατηρήσεις :

Από θεωρία γνωρίζουμε ότι το μήκος του παραθύρου που θα χρησιμοποιηθεί καθορίζει την σχέση μεταξύ διακριτικής ικανότητας και ανάλυσης στο χρόνο. Ένα στενό παράθυρο είναι κατάλληλο για χαμηλές συχνότητες και πετυχαίνει καλή ανάλυση στο χρόνο με όμως χειρότερη διακριτική ικανότητα στην συχνότητα. Αντίθετα ένα πιο φαρδύ παράθυρο στο χρόνο είναι κατάλληλο για υψηλές συχνότητες και πετυχαίνει καλή διακριτική ικανότητα στο πεδίο των συχνοτήτων χάνοντας σε ανάλυση στον χρόνο. Επίσης γνωρίζουμε ότι ο Μετασχηματισμός στη βάση των Κυματιδίων έχει το πλεονέκτημα του παραθύρου μεταβλητού μεγέθους εν αντιθέσει με του σταθερού παραθύρου του STFT. Αυτή η δυνατότητα οφείλεται

στο ότι μπορούμε να μεταβάλουμε την κλίμακα s , μεγέθυνση ή σμίκρυνση, (δηλαδή με την wavelet ανάλυση πετυχαίνουμε time-scale απεικόνιση του σήματος εισόδου). Με βάση αυτά ο μετασχηματισμός DT-CWT έχει καλύτερη ανάλυση στο χρόνο ή στη συχνότητα και με την επιλογή κατάλληλου παραθύρου μπορούμε να έχουμε καλύτερη ανάλυση για τις συχνοτικές περιοχές που μας ενδιαφέρουν. Παρατηρώντας τα σχήματα 3.1c και 3.1e και συγκρίνοντας, συμπεραίνουμε ότι στον μετασχηματισμό με wavelets (σχήμα 3.1e) δεν έχουμε απότομες μεταβολές στην συχνότητα, πράγμα που δεν ισχύει στον μετασχηματισμό STFT (σχήμα 3.1c). Ακόμη ο wavelet μετασχηματισμός απαλείφει το θόρυβο του σήματος εισόδου (denoising) σε αντίθεση με τον STFT που δεν τον εξομαλύνει. Αυτό φαίνεται πάλι από τα σχήματα 3.1c και 3.1e με το δεύτερο να είναι πιο καθαρό και πιο ομαλό χωρίς παραμορφώσεις και απότομες μεταβολές όπως του σπεκτρογραφήματος του σήματος 3.1c.

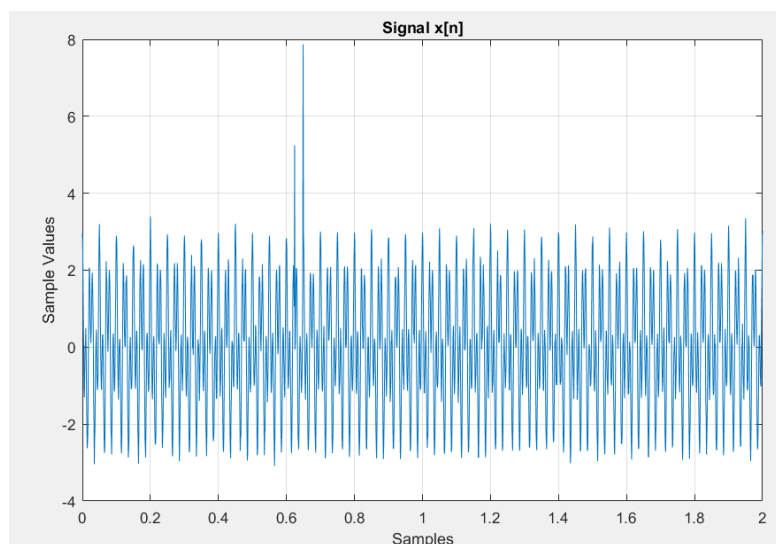
Ερώτημα 3.2

Έχουμε το σήμα $x(t) = 1.5 \cos(2\pi 40t) + 1.5 \sin(2\pi 100t) + 0.15u(t) + 5(\delta(t - 0.625) + \delta(t - 0.650))$, όπου $u(t)$ λευκός Gaussian θόρυβος μηδενικής μέσης τιμής και δ οι dirac, και το δειγματοληπτούμε με συχνότητα δειγματοληψίας $f_s=1000\text{Hz}$ στο διάστημα $t \in [0,2]$ sec και προκύπτει το $x[n]$.

```
%~~~~~3.2a Signal x[n]~~~~~
xn=1.5*cos(2*pi*40*t) +1.5*cos(2*pi*100*t)+0.15*u;
xn(625) = xn(625) + 5;      %5δ(t-0.625) dirac
xn(650) = xn(650) + 5;      %5δ(t-0.650) dirac
figure;
plot(t,xn);
title(['Signal x[n]']);
xlabel('Samples');
ylabel('Sample Values');
grid on;
```

Σχήμα 3.2: Κομμάτι κώδικα για την υλοποίηση του σήματος $x[n]$

(α) Το σήμα $x[n]$ που προκύπτει μετά την δειγματοληψία είναι:



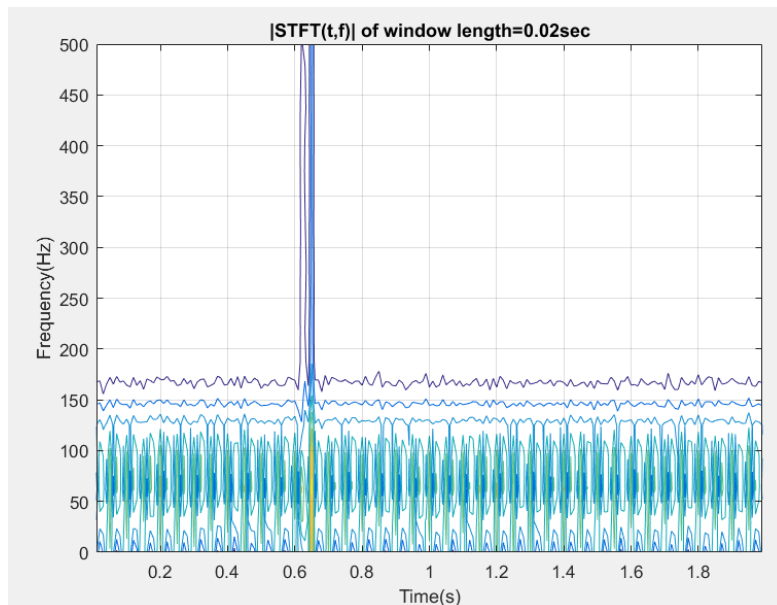
Σχήμα 3.2α: Λαμβανόμενο Σήμα $x[n]$

(β) Θα υπολογίσουμε τον SFTP με χρήση της ρουτίνας spectrogram και θα εμφανίσουμε το πλάτος $|STFT(t, f)|$ σε δισδιάστατο διάγραμμα, αυτή την φορά με χρήση της συνάρτησης contour του Matlab που σαν ορίσματα θα παίρνει το T,F,S της συνάρτησης spectrogram. Χρησιμοποιήθηκε μήκος παραθύρου 0.02sec, 0.04sec και 0.06 sec και επικάλυψη 50% εκάστοτε.

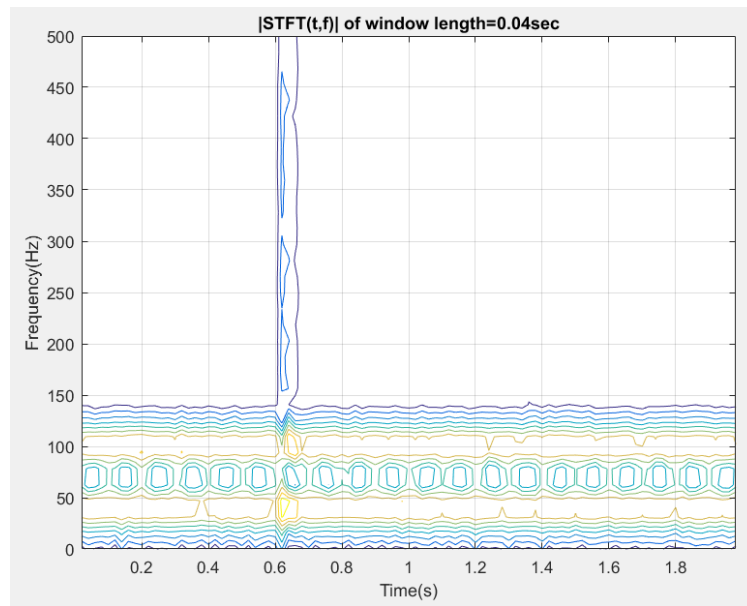
```
%~~~~~3.2b Short Time Fourier Transform ~~~~~
for i=0.02:0.02:0.06
    [S,F,T] = spectrogram(xn,i*fs,i*fs/2,2^(nextpow2(i*fs)),fs,'yaxis');
    figure;
    contour(T,F,abs(S));
    title(['|STFT(t,f)| of window length=' num2str(i) 'sec']);
    xlabel('Time(s)');
    ylabel('Frequency(Hz)');
    grid on;
end
```

Σχήμα 3.1b: Κομμάτι κώδικα για την υλοποίηση του STFT και την απεικόνιση του σπεκτρογραφήματος

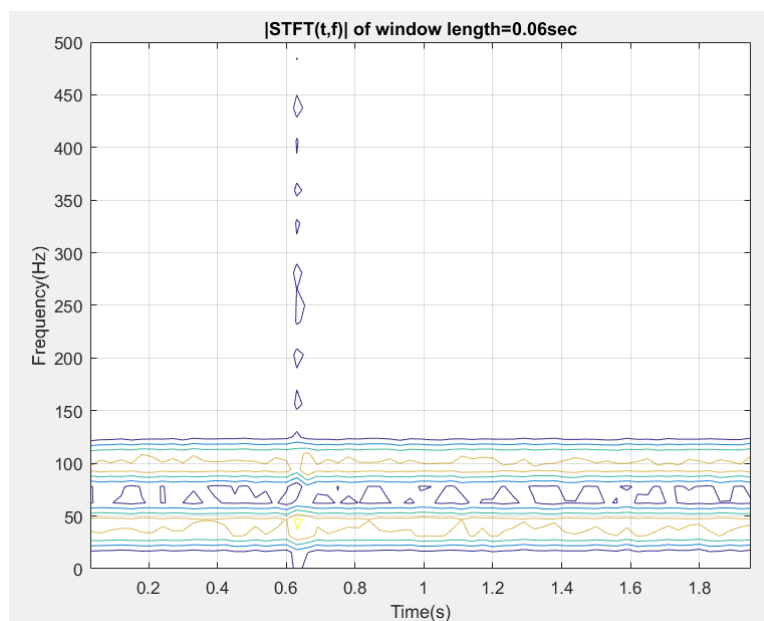
Και θα έχουμε τις παρακάτω απεικονίσεις για το πλάτος του μετασχηματισμού STFT για τις διάφορες τιμές του μήκους παραθύρου:



Σχήμα 3.1ci: Απεικόνιση του πλάτους του STFT με συνάρτηση contour και μήκος παραθύρου 0.02sec



Σχήμα 3.1γ=cii: Απεικόνιση του πλάτους του STFT με συνάρτηση contour και μήκος παραθύρου 0.04sec



Σχήμα 3.1cii: Απεικόνιση του πλάτους του STFT με συνάρτηση contour και μήκος παραθύρου 0.06sec

(γ) Τώρα θα υπολογίσουμε τον DT-CWT με χρήση των ρουτινών `cwtft` και `wavescales` του συμπληρωματικού υλικού και θα εμφανίσουμε το πλάτος $|DT_CWT(\tau, f)|$ σε διδιάστατο διάγραμμα με χρήση της συνάρτησης `contour` του Matlab που σαν ορίσματα θα παίρνει το τ, f και `cfs` της συνάρτησης `cwtft`. Χρησιμοποιήθηκε και πάλι μήκος παραθύρου 0.04 sec και επικάλυψη 0.02 sec ενώ σαν wavelet χρησιμοποιήθηκε το Morlet Wavelet (ή Gabor Wavelet).

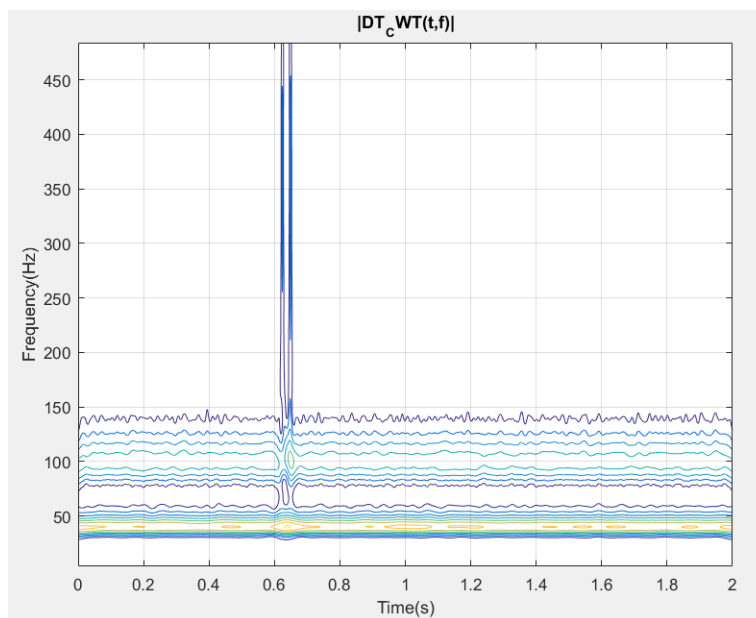
```

%%~~~~~3.2c Discrete Time Continuous Wavelet Transform~~~~~
cwtstruct = cwtft({xn,1/fs},'scales',s,'wavelet','morl');
cfs=cwtstruct.cfs;
figure;
contour(t, f, abs(cfs));
title(['|DT_CWT(t,f) |']);
xlabel('Time(s) ');
ylabel('Frequency(Hz) ');
grid on;

```

Σχήμα 3.1d: Κομμάτι κώδικα για την υλοποίηση του Μετ/σμού DT-CWT και την απεικόνιση του σπекτρογραφήματος

Θα έχουμε την παρακάτω απεικόνιση για το πλάτος του μετασχηματισμού DT_CWT με χρήση της συνάρτησης contour:



Σχήμα 3.1e: Απεικόνιση του πλάτους του STFT με συνάρτηση contour και μήκος παραθύρου 0.04sec

(δ) Παρατηρήσεις :

Από τα Σχήματα 3.1c και 3.1e παρατηρούμε ότι και οι δύο μετασχηματισμοί εμφανίζουν και τους dirac όρους. Όσον αφορά τον μετασχηματισμό STFT μεταβάλλοντας το μήκος παραθύρου βλέπουμε ότι οι μεταβάσεις και μεταβολές είναι πιο αραιές και οι dirac σχεδόν εξαφανίστηκαν. Επομένως, δεν έχουμε μεγάλη λεπτομέρεια στις μετρήσεις μας. Συνεπώς και οι δύο μετασχηματισμοί μπορούν να πετύχουν και τα δύο ζητούμενα. Παρ'όλα αυτά παρατηρούμε ότι με τον Wavelet Transform (Σχήμα 3.1e) μπορούμε να πετύχουμε καλύτερη χρονική ανάλυση σε υψηλότερες συχνότητες σε σχέση με τον Short Time Fourier Transform.

Βιβλιογραφία

- Short Term Time Domain Processing of Speech (Theory): Speech Signal Processing Laboratory: Electronics & Communications: IIT GUWAHATI Virtual Lab.
- Non-Stationary Nature of Speech Signal (Theory): Speech Signal Processing Laboratory: Electronics & Communications: IIT GUWAHATI Virtual Lab.
- Identification of Voice/Unvoiced/ Silence regions of Speech(Theory): Speech Signal Processing Laboratory: Electronics & Communications: IIT GUWAHATI Virtual Lab .
- Speech Processing using MATLAB, Part 1, KatsamaniS Nassos, Pitsikalis Vasilis,Oct 29, 2004
- Comparison of Short-time Fourier Transform and Wavelet Transform of Transient and Tone Burst Wave Propagation Signals For Structural Health Monitoring.
- Comparison of STFT and Wavelet Transform methods in determining epileptic seizure activity in EEG (electroencephalography) signals for real-time application.
- FFT Resynthesis