Μέρος 1: Φασματική ανάλυση με Γραμμική πρόβλεψη και Cepstrum

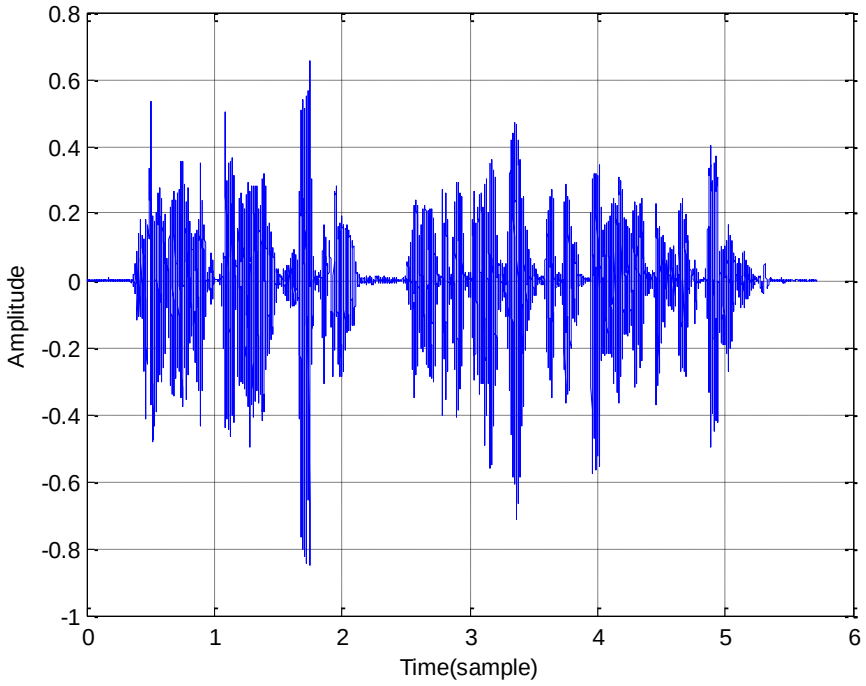
Για να διαβάσουμε το σήμα φωνής χρησιμοποιούμε την συνάρτηση του **MATLAB** την **wavread.** Επομένως στο **Command Window** γράφουμε :

v = wavread('s022ad165');

Το σήμα αυτό έχει περίοδο δειγματοληψίας $\boldsymbol{T_s = \frac{1}{F_s} = 6.25 * 10^{-5}}$ και γράφοντας στο **Command Window :**

vs = linspace(0,5.72,91510);
plot(vs,v),xlabel('Time(sample)'),ylabel('Amplitude'),grid;

παίρνουμε το γράφημα του δειγματοληπτιμένου σήματος φωνής, το οποίο είναι :

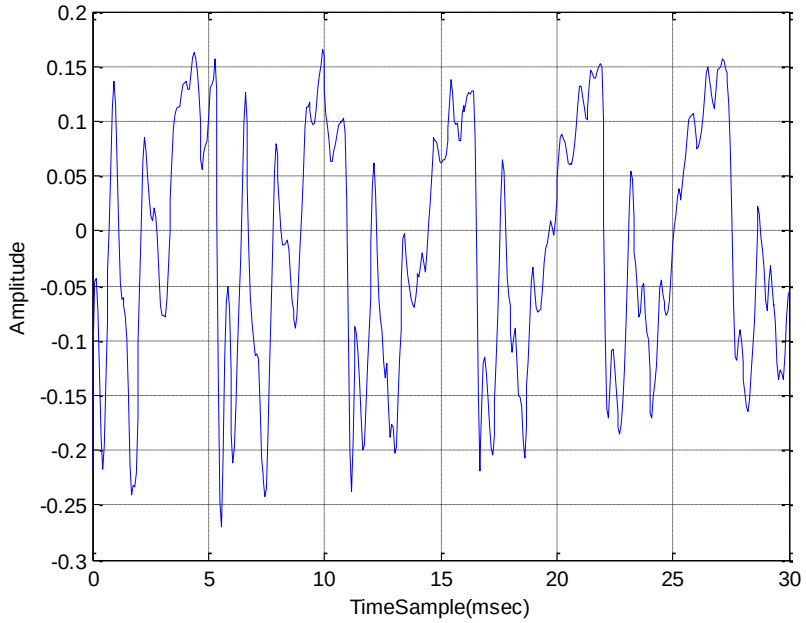


Το τμήμα που θα απομονώσουμε βρίσκεται (περίπου) στο διάστημα [2 , 2.1] και επειδή θέλουμε να έχει διάρκεια 30 msec έχουμε περίοδο $\boldsymbol{N_p = \frac{0.03}{T_s}}$ **= 480** δείγματα.
Για να πάρουμε αυτό το κομμάτι γράφουμε στο **Command Window :**
svoice = v(32001:32480);

Για να πάρουμε το γράφημα του απομονωμένου κομματιού γράφουμε στο **Command Window :**

ivs = linspace(0,30,480);
plot(ivs,svoice),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;

Επομένως η γραφική παράσταση του απομονωμένου κομματιού θα είναι :



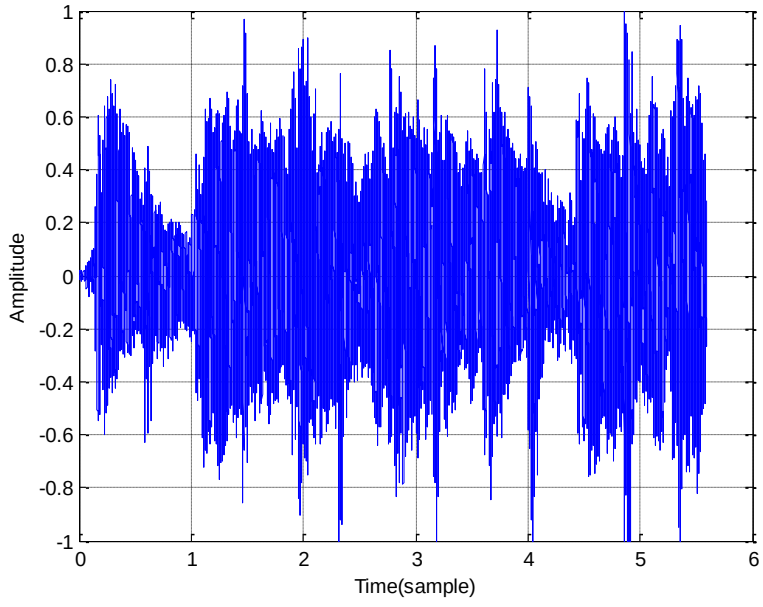
Για να διαβάσουμε το σήμα μουσικής χρησιμοποιούμε την συνάρτηση του **MATLAB** την **wavread.** Επομένως στο **Command Window** γράφουμε :

m = wavread('music');

Το σήμα αυτό έχει περίοδο δειγματοληψίας $\boldsymbol{T_s = \frac{1}{F_s} \cong 2.2676 * 10^{-5}}$ και γράφοντας στο **Command Window :**

ms = linspace(0,5.59,246535);
plot(ms,m),xlabel('Time(sample)'),ylabel('Amplitude'),grid;

παίρνουμε το γράφημα του δειγματοληπτιμένου σήματος μουσικής, το οποίο είναι :



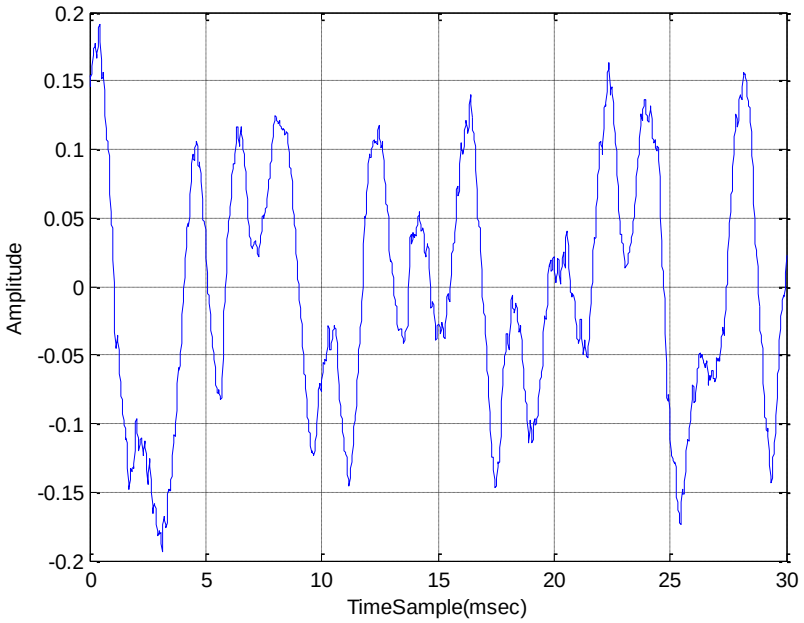
Το τμήμα που θα απομονώσουμε βρίσκεται (περίπου) στο διάστημα [0.9 , 1] και επειδή θέλουμε να έχει διάρκεια 30 msec έχουμε περίοδο $\boldsymbol{N_p = \frac{0.03}{T_s} \cong 1322}$ δείγματα.
Για να πάρουμε αυτό το κομμάτι γράφουμε στο **Command Window :**

smusic = m(39690:41011);

Για να πάρουμε το γράφημα του απομονωμένου κομματιού γράφουμε στο **Command Window :**

ims = linspace(0,30,1322);
plot(ims,smusic),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;

Επομένως η γραφική παράσταση του απομονωμένου κομματιού θα είναι :

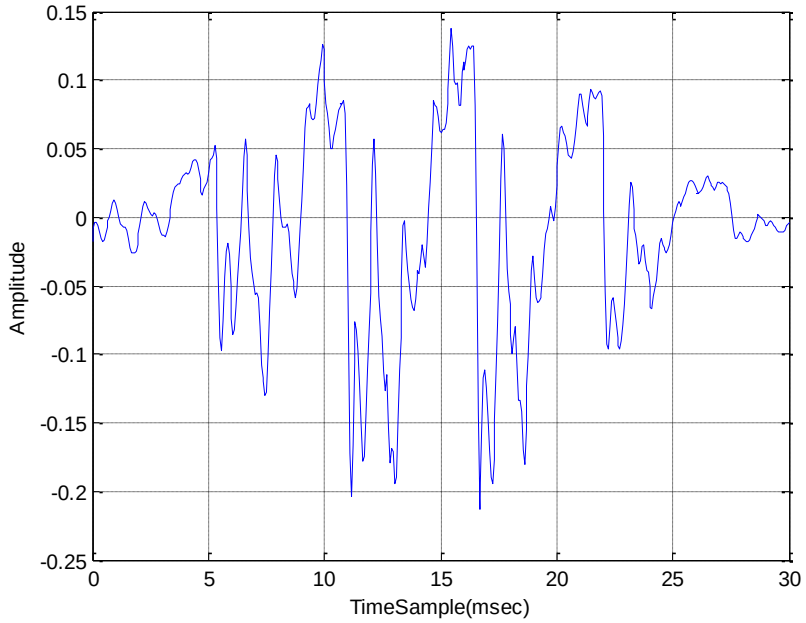


1.1 Cepstrum

(1) Για να σχεδιάσουμε το γράφημα του παραθυρωμένου σήματος φωνής θα γράψουμε στο **Command Window :**

x = svoice.*hamming(480);
plot(ivs,x),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;

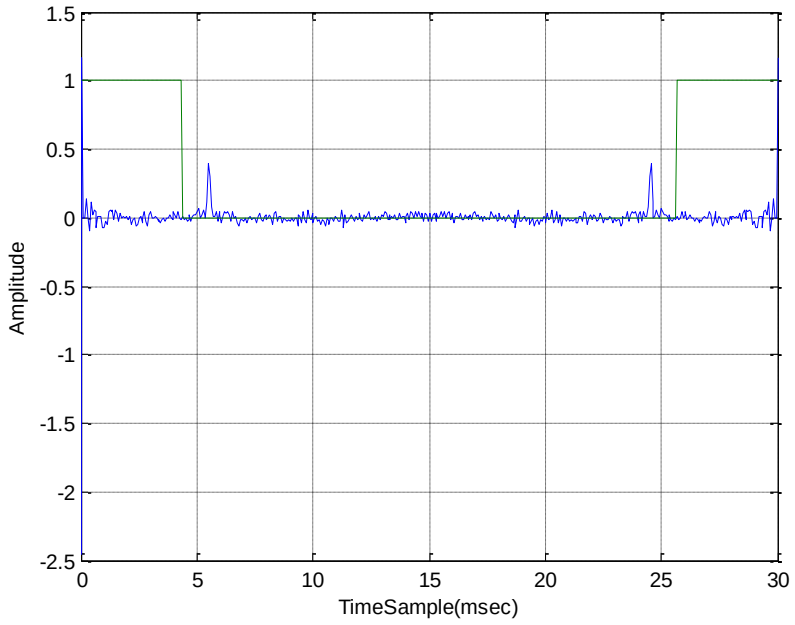
Επομένως η γραφική παράσταση του παραθυρωμένου σήματος φωνής θα είναι :



(2) Για τον σχεδιασμό του **Cepstrum** καθώς και του ορθογώνιου παραθύρου που αποκόπτει το χαμηλό-χρονο τμήμα του γράφουμε στο **Command Window :**

```
cepstrum = rceps(x);
wdw = [ones(1,70) zeros(1,340) ones(1,70)];
wdw = wdw';
plot(ivs,cepstrum,ivs,wdw),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;
```

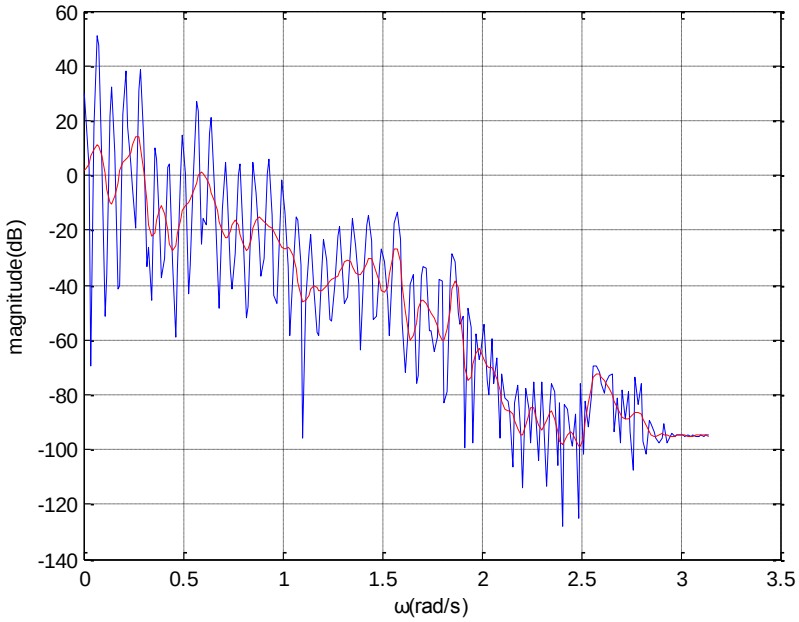
Επομένως η γραφική παράσταση του **Cepstrum** καθώς και του ορθογώνιου παραθύρου θα είναι :



(3)Για να σχεδιάσουμε το ομαλό φάσμα για το χαμηλό-χρονο τμήμα του **Cepstrum** σε κοινό διάγραμμα με το λογαριθμικό γράφουμε στο **Command Window :**

```
c = 20*log(abs(fft(x)));
c = c(1:240);
cepswdw = cepstrum.*wdw;
e = 20*log(abs(exp(fft(cepswdw))));
e = e(1:240);
pce = linspace(0,pi,240);
plot(pce,c,pce,e ,'r'),xlabel('ω(rad/s)'),ylabel('magnitude(dB)'),grid;
```

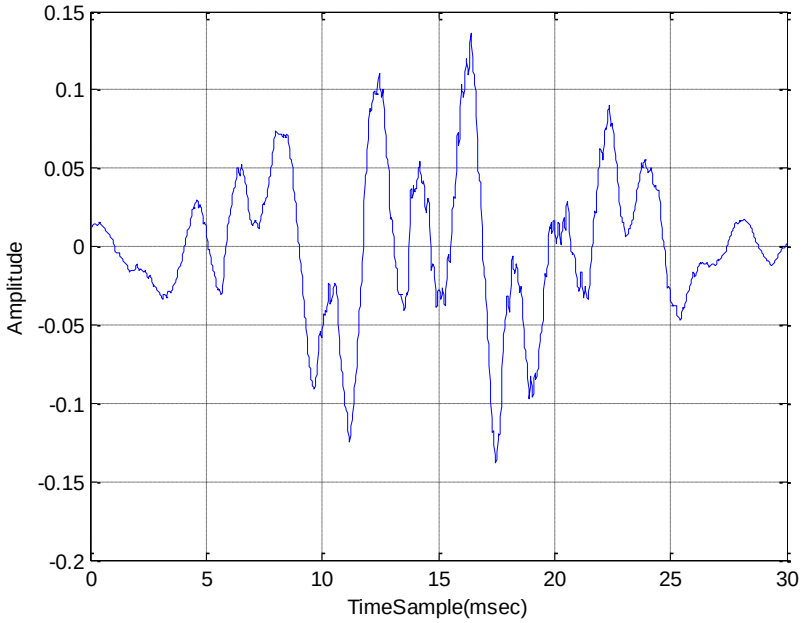
Επομένως το γράφημα των φασμάτων θα είναι :



(1) Για να σχεδιάσουμε το γράφημα του παραθυρωμένου σήματος μουσικής θα γράψουμε στο **Command Window :**

```
x = smusic.*hamming(1322);
plot(ims,x),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;
```

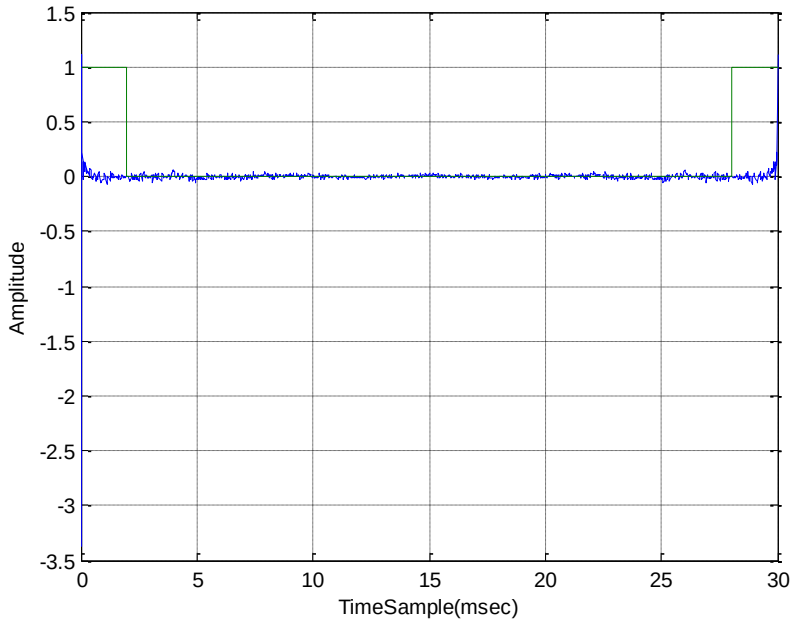
Επομένως η γραφική παράσταση του παραθυρωμένου σήματος μουσικής θα είναι :



(2) Για τον σχεδιασμό του **Cepstrum** καθώς και του ορθογώνιου παραθύρου που αποκόπτει το χαμηλό-χρονο τμήμα του γράφουμε στο **Command Window :**

```
cepstrum = rceps(x);
wdw = [ones(1,88) zeros(1,1146) ones(1,88)];
wdw = wdw';
plot(ims,cepstrum,ims,wdw),xlabel('TimeSample(msec)'),ylabel('Amplitude'),grid;
```

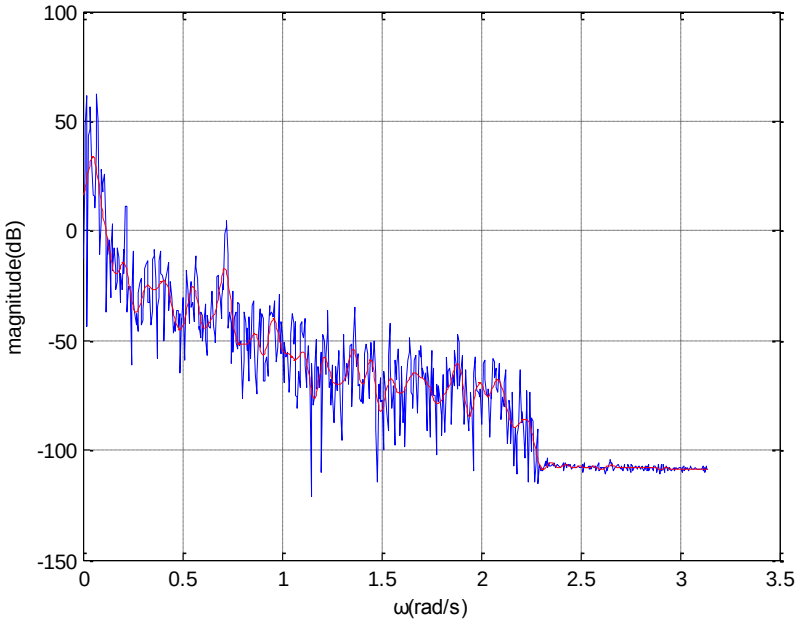
Επομένως η γραφική παράσταση του **Cepstrum** καθώς και του ορθογώνιου παραθύρου θα είναι :



(3)Για να σχεδιάσουμε το ομαλό φάσμα για το χαμηλό-χρονο τμήμα του **Cepstrum** σε κοινό διάγραμμα με το λογαριθμικό γράφουμε στο **Command Window :**

```
c = 20*log(abs(fft(x)));
c = c(1:661);
cepswdw = cepstrum.*wdw;
e = 20*log(abs(exp(fft(cepswdw))));
e = e(1:661);
pce = linspace(0,pi,661);
plot(pce,c,pce,e),xlabel('ω(rad/s)'),ylabel('magnitude(dB)'),grid;
```

Επομένως το γράφημα των φασμάτων θα είναι :



Το παράθυρο του **cepstrum** πριν από το πρώτο **peak** του μας δίνει στην έξοδο ένα σήμα «ομαλοποιημένο». Ο λόγος για τον οποίο παίρνουμε αυτό το συγκεκριμένο παράθυρο είναι ότι μας παρέχει μια πολύ ικανοποιητική προσέγγιση του σήματος, «γλιτώνοντας» πολλά δείγματα, ενώ δεν αφήνει πολλές αρμονικές να εισχωρήσουν στην έξοδο.

Όσον αφορά το μήκος του παραθύρου, τόσο για το σήμα φωνής όσο και για το σήμα μουσικής, όσο μικραίνουμε το παράθυρο(πάντα κάτω από την $1^{η}$ κορυφή του **cepstrum**) εξομαλύνεται η καμπύλη του χαμηλό-χρονου κομματιού και αυτό διότι μειώνονται τα δείγματα του σήματος.

1.2 Γραμμική Πρόβλεψη

Για να πάρουμε το παραθυρωμένο σήμα φωνής γράφουμε στο **Command Window :**

```
v = wavread('s022ad165');
svoice = v(32001:32480);
x = svoice.*hamming(480);
```

Για να βρούμε τους συντελεστές πρόβλεψης γράφουμε στο **Command Window :**

```
a1 = lpc(x,20);
a2 = lpc(x,40);
```

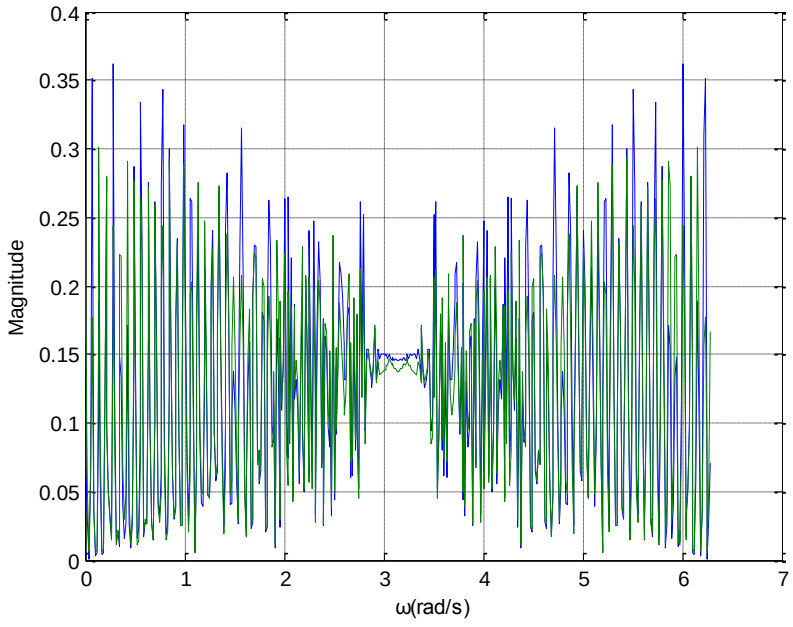
Για την εύρεση του σφάλματος πρόβλεψης για δύο τιμές της τάξης μοντέλου p γράφουμε στο **Command Window :**

```
est_x = filter([0 -a1(2:end)],1,x);
error1 = x - est_x;
est_x = filter([0 -a2(2:end)],1,x);
error2 = x - est_x;
```

Για τον σχεδιασμό του γραφήματος του φάσματος των λαθών πρόβλεψης γράφουμε στο **Command Window :**

```
ers = linspace(0,2*pi,480);
plot(ers,abs(fft(error1)),ers,abs(fft(error2))),xlabel('ω(rad/s)'),ylabel('Magnitude'),grid;
```

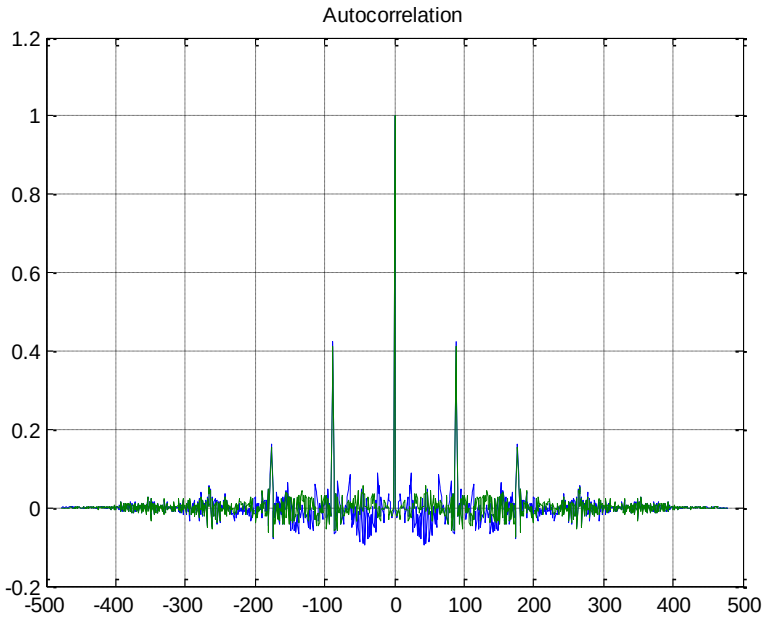
Η γραφική παράσταση είναι η παρακάτω:



Για τον σχεδιασμό της αυτοσυσχέτισης του σφάλματος και για τις δύο περιπτώσεις τάξης προβλέπτη γράφουμε στο **Command Window :**

```
[acs1,lags1] = xcorr(error1,'coeff');
[acs2,lags2] = xcorr(error2,'coeff');
plot(lags1,acs1,lags2,acs2);grid;title('Autocorrelation');
```

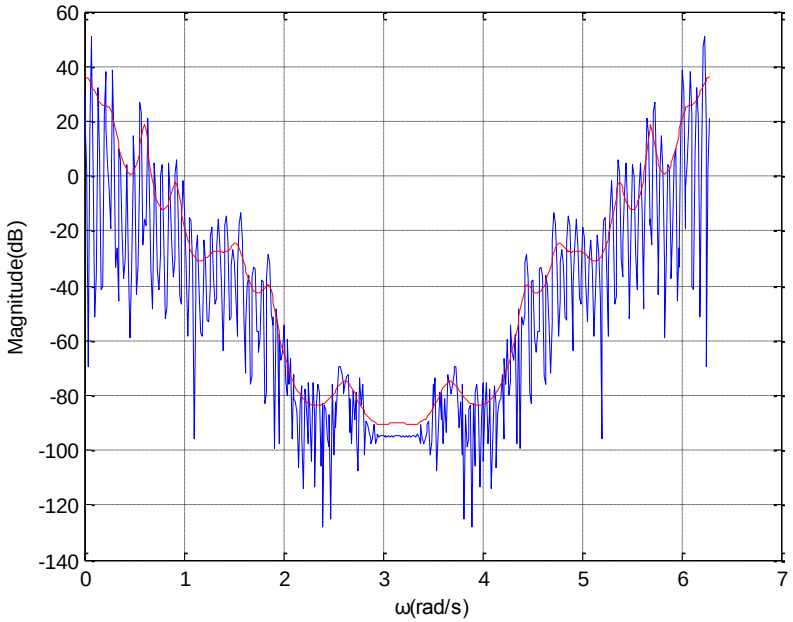
Η γραφική παράσταση θα είναι η εξής:



Για τον σχεδιασμό του γραφήματος του φάσματος του μοντέλου lpc και του φάσματος του παραθυρωμένου σήματος για τάξη πρόβλεψης **p=20** γράφουμε στο **Command Window :**

```
gain = GainG(error1);
[H]=freqz(gain,a1,480,'whole');
plot(ers,20*log(abs(fft(x))),ers,20*log(abs(H)),'r'),xlabel('ω(rad/s)'),ylabel('Magnitude(dB)'),grid;
```

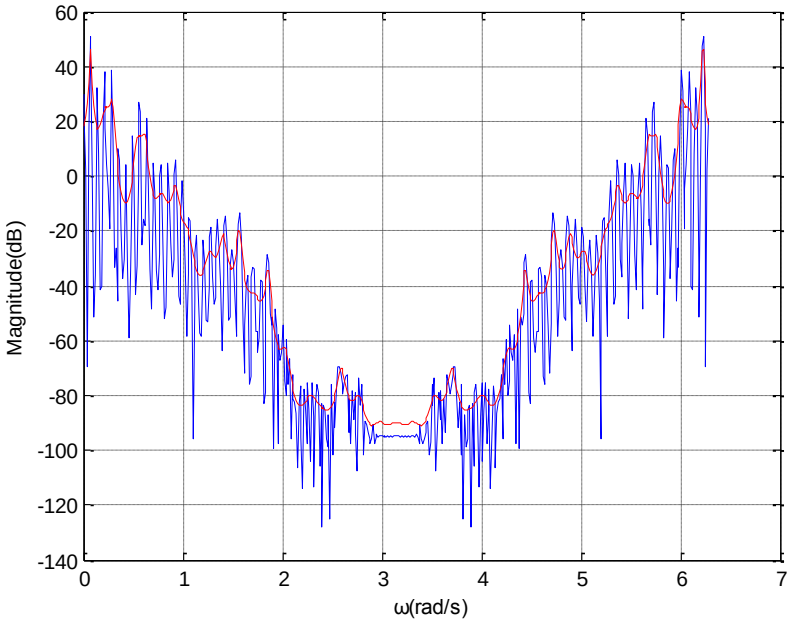
Η γραφική παράσταση είναι η παρακάτω:



Για τον σχεδιασμό του γραφήματος του φάσματος του μοντέλου lpc και του φάσματος του παραθυρωμένου σήματος για τάξη πρόβλεψης **p=40** γράφουμε στο **Command Window :**

```
gain = GainG(error2);
[H]=freqz(gain,a2,480,'whole');
plot(ers,20*log(abs(fft(x))),ers,20*log(abs(H)),'r'),xlabel('ω(rad/s)'),ylabel('Magnitude(dB)'),grid;
```

Η γραφική παράσταση είναι η παρακάτω:



Για να πάρουμε το παραθυρωμένο σήμα μουσικής γράφουμε στο **Command Window :**

```
m = wavread('music');
smusic = m(39690:41011);
x = smusic.*hamming(1322);
```

Για να βρούμε τους συντελεστές πρόβλεψης γράφουμε στο **Command Window :**

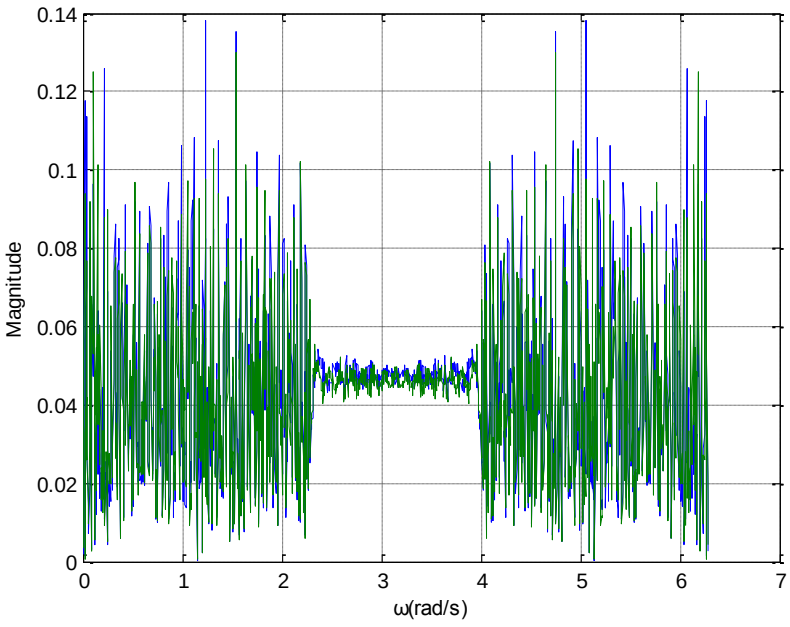
```
a1 = lpc(x,40);
a2 = lpc(x,60);
```

Για την εύρεση του σφάλματος πρόβλεψης για δύο τιμές της τάξης μοντέλου p γράφουμε στο **Command Window :**

```
est_x = filter([0 -a1(2:end)],1,x);
error1 = x - est_x;
est_x = filter([0 -a2(2:end)],1,x);
error2 = x - est_x;
```

Για τον σχεδιασμό του γραφήματος του φάσματος των λαθών πρόβλεψης γράφουμε στο **Command Window :**

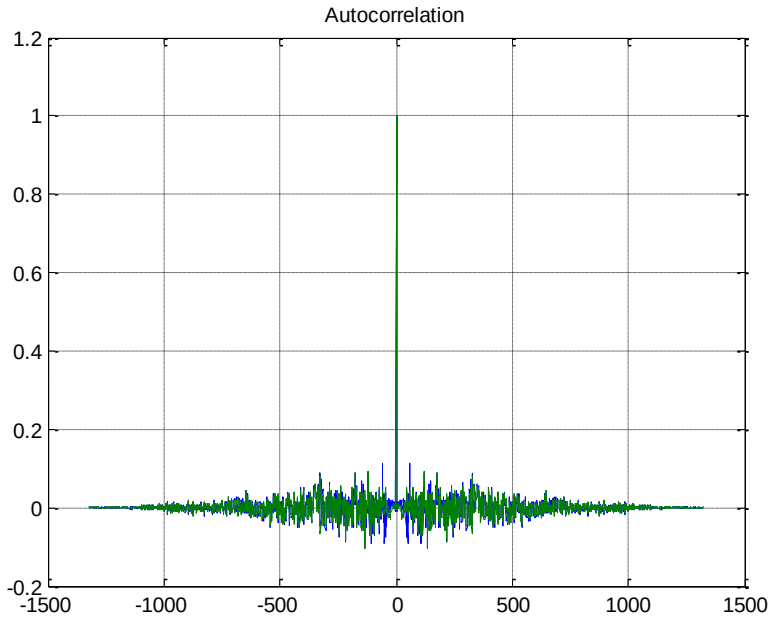
```
ers = linspace(0,2*pi,1322);
plot(ers,abs(fft(error1)),ers,abs(fft(error2))),xlabel('ω(rad/s)'),ylabel('Magnitude'),grid;
```



Για τον σχεδιασμό της αυτοσυσχέτισης του σφάλματος και για τις δύο περιπτώσεις τάξης προβλέπτη γράφουμε στο **Command Window :**

```
[acs1,lags1] = xcorr(error1,'coeff');
[acs2,lags2] = xcorr(error2,'coeff');
plot(lags1,acs1,lags2,acs2);grid;title('Autocorrelation');
```

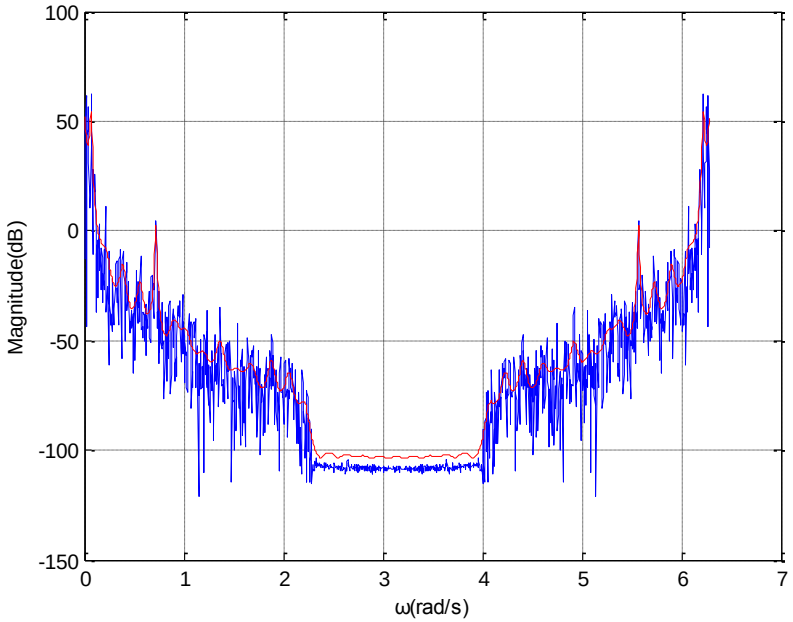
Η γραφική παράσταση θα είναι η εξής:



Για τον σχεδιασμό του γραφήματος του φάσματος του μοντέλου lpc και του φάσματος του παραθυρωμένου σήματος για τάξη πρόβλεψης **p=40** γράφουμε στο **Command Window :**

```
gain = GainG(error1);
[H]=freqz(gain,a1,1322,'whole');
plot(ers,20*log(abs(fft(x))),ers,20*log(abs(H)),'r'),xlabel('ω(rad/s)'),ylabel('Magnitude(dB)'),grid;
```

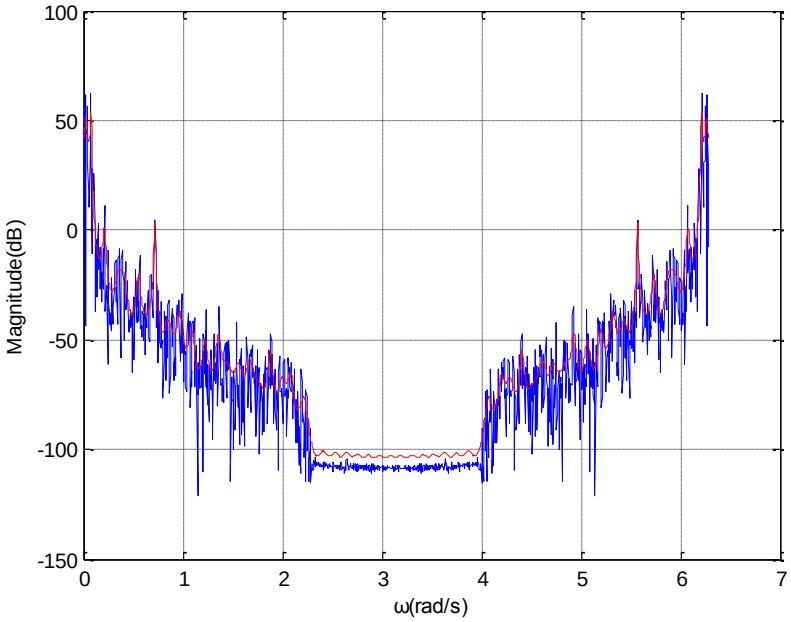
Η γραφική παράσταση είναι η παρακάτω:



Για τον σχεδιασμό του γραφήματος του φάσματος του μοντέλου lpc και του φάσματος του παραθυρωμένου σήματος για τάξη πρόβλεψης **p=60** γράφουμε στο **Command Window :**

```
gain = GainG(error2);
[H]=freqz(gain,a2,1322,'whole');
plot(ers,20*log(abs(fft(x))),ers,20*log(abs(H)),'r'),xlabel('ω(rad/s)'),ylabel('Magnitude(dB)'),grid;
```

Η γραφική παράσταση είναι η παρακάτω:



Όπως παρατηρούμε από τα γραφήματα των αυτοσυσχετίσεων των σφαλμάτων το σφάλμα πρόβλεψης είναι πράγματι λευκός θόρυβος, επειδή, όπως βλέπουμε, το επίπεδο του φάσματος προσεγγίζει το μηδέν.

Γενικά, παρατηρούμε ότι η αύξηση της τάξης του προβλέπτη αυξάνει την ακρίβεια της πρόβλεψης και μειώνει το ύψος του φάσματος του σφάλματος πρόβλεψης. Στην περίπτωση του σήματος μουσικής επιλέγουμε τάξη προβλέπτη p=60 που φαίνεται ότι μας δίνει μικρό σφάλμα πρόβλεψης και ικανοποιητική ακρίβεια.

Μέρος 2. Ψηφιακή Κωδικοποίηση/ Σύνθεση Φωνής με Γραμμική Πρόβλεψη LPC Vocoder

2.1 Σύνθεση Φωνής με Γραμμική Πρόβλεψη

Για να διαβάσουμε το σήμα φωνής γράφουμε στο **Command Window :**

[v,Fs,nbits] = wavread('s022ad165');

Για να παραθυροποιήσουμε το σήμα μας σε παράθυρα μήκους **480** και επικάλυψης **320** γράφουμε στο **Command Window :**

Y = buffer(v,480,320,'nodelay');

Για να διαβάσουμε το αρχείο pitch και για να αποθηκεύσουμε τα pitches όλων των παραθύρων γράφουμε στο **Command Window :**

Pitch = load('pitch');
PitchM = Pitch.pitch;

Παρατηρώντας τον πίνακα που περιέχει το παραθυροποιημένο σήμα και τον πίνακα που περιέχει τα pitch βλέπουμε πως για το τελευταίο παράθυρο δεν υπάρχει pitch(ο πίνακας pitch έχει 569 στήλες ενώ ο πίνακας με τα παράθυρα 570).Επομένως επειδή υποτίθεται ότι το σήμα μας τελειώνει στο τελευταίο παράθυρο θα βάλουμε pitch=0 που υποδηλώνει άφωνο ήχο. Άρα γράφουμε στο **Command Window :**

PitchM = [PitchM 0];

Για να κατασκευάσουμε ένα πίνακα ο οποίος θα περιέχει όλες τις διεγέρσεις που προκαλούν τα διάφορα παράθυρα(δηλαδή αν πρόκειται για περιοδική κρουστική παλμοσειρά ή για τυχαία ακολουθία δειγμάτων ανάλογα με το αν έχουμε έμφωνο ή άφωνο ήχο αντίστοιχα) γράφουμε στο **Command Window :**

u = GeneratorM(PitchM,Fs);

Για να δημιουργήσουμε έναν πίνακα κάθε στήλη του οποίου θα περιλαμβάνει τα παράθυρα που προκύπτουν ύστερα από lpc ανάλυση στα αρχικά γράφουμε στο **Command Window :**

pSignal = CrePreSig(u,Y);

Για να δημιουργήσουμε το συνθετικό σήμα φωνής θα πρέπει τα διάφορα παράθυρα να προστεθούν σύμφωνα μα την διαδικασία **OverlapAdd** κατά την οποία τα δείγματα ενός παραθύρου πρέπει να προστεθούν με τα πρώτα επικαλυπτόμενα δείγματα του επόμενου παραθύρου και τα υπόλοιπα να μείνουν ως έχουν. Στην περίπτωσή μας έχουμε επικάλυψη **320** δείγματα από τα συνολικά **480**, αυτό σημαίνει πως το πρώτο παράθυρο έχει τα πρώτα **160** δείγματα κανονικά και τα υπόλοιπα **320**, επικαλυπτόμενα με τα **320** πρώτα δείγματα τού δεύτερου παραθύρου και ούτο καθ'εξής. Επομένως παρατηρώντας τα γραφήματα στο **σχήμα.4** της εκφώνησης βλέπουμε πως για να πάρουμε το αρχικό σήμα από τα διάφορα παράθυρα θα πρέπει τα επικαλυπτόμενα δείγματα δύο διαδοχικών παραθύρων να προστίθενται και τα υπόλοιπα να παραμένουν αμετάβλητα στο τελικό σήμα. Οπότε για να κατασκευάσουμε το τελικό σήμα γράφουμε στο **Command Window :**

FinalSig = CreFinSig(pSignal);

Για να ακούσουμε το κατασκευασμένο σήμα στο **matlab** γράφουμε στο **Command Window:**

sound(FinalSig,Fs);

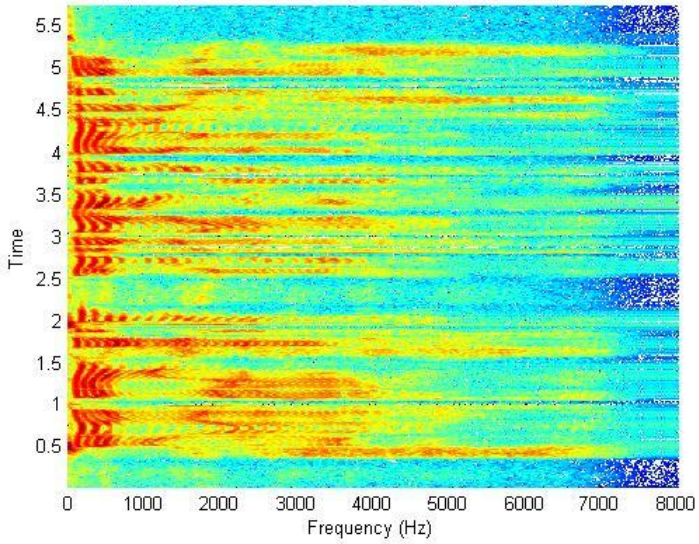
Για δημιουργία αρχείου .wav γράφουμε στο **Command Window:**

wavwrite(FinalSig,Fs,nbits,'Final_Signal.wav');

Για να σχεδιάσουμε το φασματογράφημα του αρχικού σήματος φωνής γράφουμε στο **Command Window:**

spectrogram(v,480, 320, 1024, Fs);

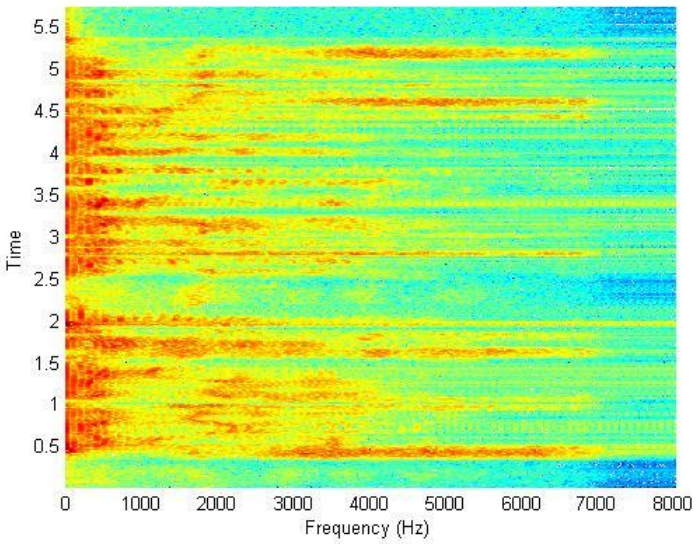
Το φασματογράφημα είναι το εξής:



Για να σχεδιάσουμε το φασματογράφημα του τελικού σήματος φωνής γράφουμε στο **Command Window:**

spectrogram(FinalSig,480, 320, 1024, Fs);

Το φασματογράφημα είναι το εξής:



m-files

```
function [G]= GainG(error)
%Compute the gain of the lpc model.
%input:
%error = PredictionError
%Output:
%G = gain
%Usage:
%gain = GainG(PrEr);
G = 0;
error = error.^2;
for m = 1:length(error)
    G = G+error(m);
end
G = sqrt(G);

function [U] = GeneratorM(pm,fs)
%It creates a matrix from which we can
%determine if the windowed signal
%is voiced or unvoiced.
%Input:
%pm = the PitchMatrix
%fs = SamplingFrequensy
%Output:
%U = StimulatedMatrix
%Usage:
%u = GeneratorM(PitchM,Fs);
U = zeros(480,570);
nodeM = (1:480)';
for i = 1:570
    if pm(i) == 0
        U(:,i) = rand(480,1);
        U(:,i) = U(:,i)-0.5;
    else
        U(:,i) = mod(nodeM,round(fs/pm(i))) == 0;
    end
end

function [Psignal] = CrePreSig(U,y)
%It creates a matrix(using lpc) who's
%columns contain the predicted signal
%for every window.
%Input:
%U = StimuladedMatrix
%y = WindowedSignalMatrix
%Output:
%Psignal = PredictedSignalMatrix
%Usage:
%pSignal = CrePreSig(u,Y);
Psignal = zeros(480,570);
for i = 1:570
    y1 = y(:,i).*hamming(480);
    a = lpc(y1,20);
    est_y = filter([0 -a(2:end)],1,y1);
    error = y1-est_y;
    G = sqrt(sum(error.^2));
    Psignal(:,i) = filter(G,[1 a(2:end)],U(:,i));
end
```

```
function [finalsig] = CreFinSig(Psignal)
%It creates the final speech signal
%using the predicted version(lpc) of
%the initial windowed signals and the
%OverlapAdd method.
%Input:
%Psignal = PredictedSignalMatrix
%Output:
%finalsig = FinalSpeechSignal
%Usage:
%FinalSig = CreFinSig(pSignal);
finalsig = zeros(1,91520);
for i = 1:570
    OvAdd = 160*(i-1);
    for j = 1:480
        finalsig(OvAdd+j) = finalsig(OvAdd+j)+Psignal(j,i);
    end
end
```