Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Μάθημα: Ψηφιακή Επεξεργασία Σημάτων

2η Εργαστηριακή Άσκηση

Θέμα: Γραμμική Πρόβλεψη (LPC) και Ομομορφική Cepstrum Επεξεργασία Σημάτων με MATLAB και Εφαρμογές στη Σύνθεση και Συμπίεση Φωνής

Ονοματεπώνυμο: **Σταυρακάκης Δημήτριος** ΑΜ: **03112017**

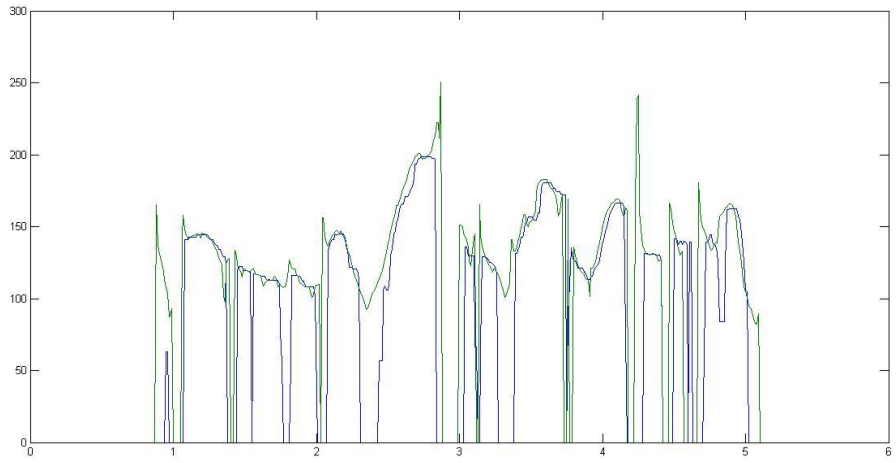
Ημερομηνία Παράδοσης: 21/5/2015

***Μέρος 1*: Εξαγωγή pitch φωνής με χρήση Cepstrum**

Στόχος της άσκησης : Εξαγωγή της θεμελιώδους συχνότητας της φωνής που υπάρχει στο αρχείο speech.wav με χρήση ομομορφικής επεξεργασίας σημάτων.

Διαδικασία παραγωγής των ζητούμενων γραφημάτων:

- Με τη συνάρτηση wavread() του Matlab εισάγουμε το σήμα φωνής που μας ενδιαφέρει να επεξεργαστούμε. Η διάκεια του είναι 5.83 sec και συχνότητα δειγματοληψίας 48kHz.
- Με τη συνάρτηση buffer() δημιουργώ τα επικαλυπτόμενα πλαίσια του σήματος. Με χρήση της συνάρτησης hamming() δημιουργώ ένα hamming παράθυρο για την παραθύρωση των πλαισίων.
- Για 580 επαναλήψεις παραθυρώνω κάθε πλαίσιο και δημιουργώ το cepstrum του. Επειδή το αρχείο αυτό περιέχει αντρική φωνή (50 ως 250 Hz) θα πρέπει να κόψω μερικά δείγματα για να έχω σωστό αποτέλεσμα. Έτσι κρατάω το cepstrum ανάμεσα σε δυο όρια(τα οποία έχω υπολογίσει) και κρατάω για κάθε πλαίσιο την μέγιστη κορυφή και τον δείκτη στον οποίο εμφανίζεται αυτή σε ένα πίνακα.
- Για να δημιουργήσω τον πίνακα συχνοτήτων τρέχω ένα επαναληπτικό βρόχο 580 επαναλήψεων και ανάλογα με την τιμή του μεγίστου που υπάρχει στον πίνακα αποφασίζω αν ο αντίστοιχος ήχος είναι έμφωνος ή άφωνος. Αν είναι άφωνος βάζω στον πίνακα συχνοτήτων 0 ενώ αν είναι έμφωνος διαιρώ την συχνότητα δειγματοληψίας με τον αντίστοιχο δείκτη προσαυξημένο κατά 191 όπου 191 είναι τα δείγματα που έκοψα, για να έχω σωστό αποτέλεσμα στο τέλος.
- Αφού δημιούργησα τον πίνακα των συχνοτήτων είμαι σε θέση να φορτώνω το αρχείο με όνομα pitch.mat με χρήση της συνάρτησης load() το οποίο περιέχει τον πίνακα συχνοτήτων όπως αυτός βρέθηκε από την μέθοδο autocorrelation και αναπαριστώ σε κοινό διάγραμμα τις δυο χρονοσειρές συχνοτήτων. Το γράφημα αυτό παρατίθεται παρακάτω:



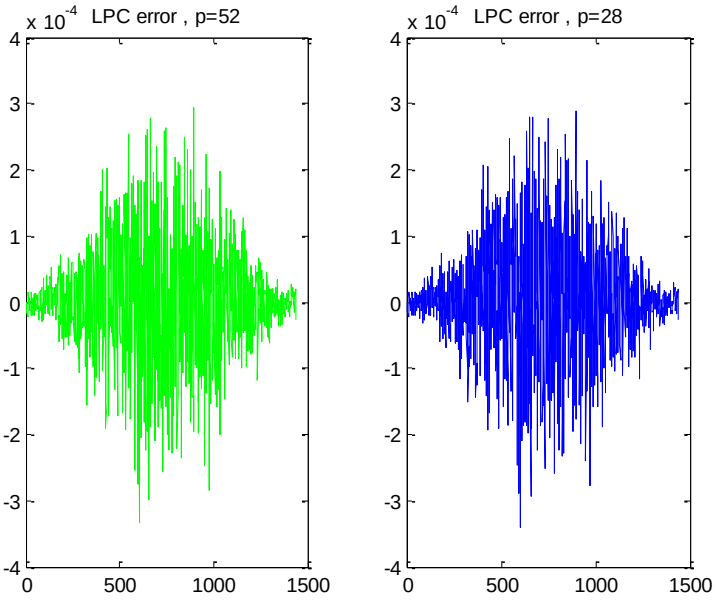
Σχολιασμός : Παρατηρούμε ότι τα αποτελέσματα είναι πάρα πολύ κοντά, άρα η μέθοδος που χρησιμοποιήσαμε είναι αρκετά ικανοποιητική από άποψη ποιότητας.

Μέρος 2: Ψηφιακή Σύνθεση Φωνής με Γραμμική Πρόβλεψη (LPC Vocoder)

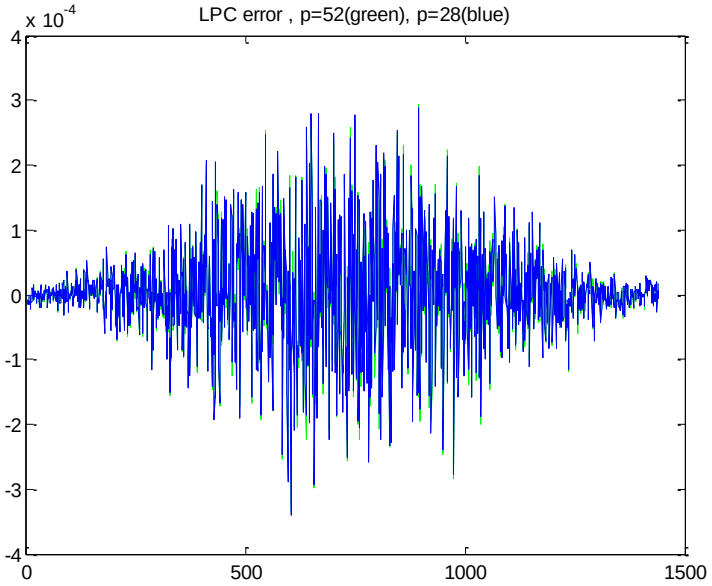
2.1: Ανάλυση Φωνής με Γραμμική Πρόβλεψη

Υλοποιούμε ένα σύστημα LPC μοντελοποίησης με χρήση του autocorrelation. Για παράθυρο χρησιμοποιούμε hamming window και η τάξη που προβλέπτη που χρησιμοποιούμε είναι p = Fs(kHz) +4 η οποία κρίνεται επαρκής για ανάλυση φωνής και μοντελοποίηση της φασματικής πολυπλοκότητας. Η πορεία που ακολουθήσαμε είναι η εξής:

- Με τη συνάρτηση wavread() του Matlab εισάγουμε το σήμα φωνής που μας ενδιαφέρει να επεξεργαστούμε. Η συχνότητα δειγματοληψίας είναι 48kHz.
- Το hamming window που χρειάζομαι για να παραθυροποιήσω το σήμα μου υπάρχει μέσα στη συνάρτηση lpc_analysis() που μας δίνεται έτοιμη και για τον λόγο αυτό δεν κάνω καμία παραθυροποίηση του σήματός μου.
- Επιλέγω τυχαία ένα από τα πλαίσια ανάλυσης του αρχικού σήματος μου για να κάνω τις ζητούμενες γραφικές παραστάσεις, αφού δεν έχει ιδιαίτερη ουσία να γίνει αυτό για καθένα από τα πλαίσια λόγω του μεγάλου τους αριθμού. Στέλνω το τυχαίο πλαίσιο ανάλυσης στη συνάρτηση lpc_analysis(), παίρνω τους συντελεστές γραμμικής πρόβλεψης α_k, το κέρδος G και το σήμα λάθους, τα οποία θέλω, όπως επίσης και το reconstructed σήμα. Η παραπάνω διαδικασία γίνεται για 2 τιμές της τάξης του προβλέπτη p=52 και p=28.
  Τα γραφήματα του λάθους γραμμικής πρόβλεψης φαίνονται παρακάτω :



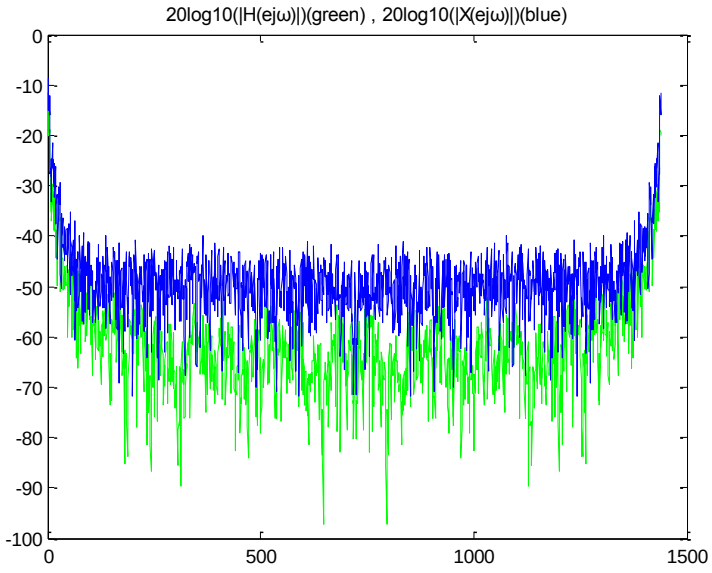
Σε κοινό διάγραμμα :



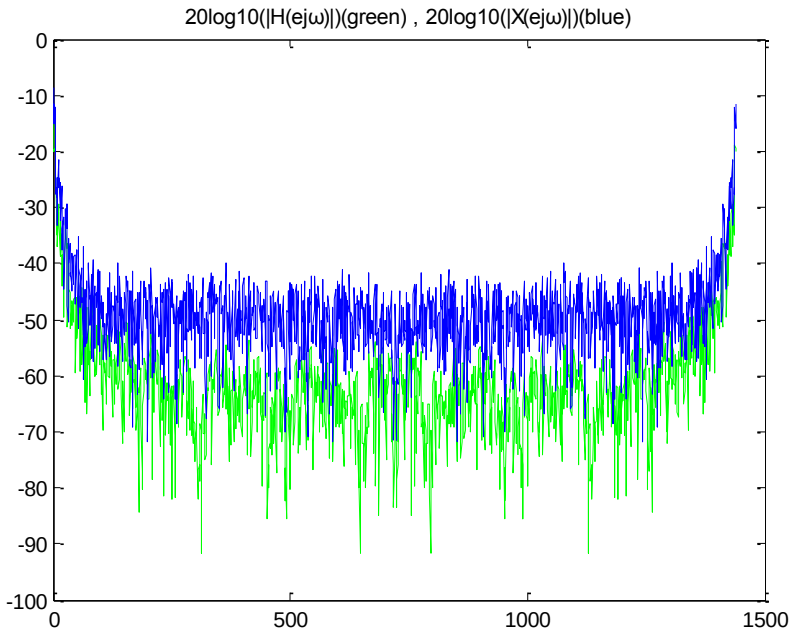
Παρατήρηση : Στο κοινό διάγραμμα παρατηρώ ότι η διαφορές είναι ελάχιστες. Για να παρατηρήσουμε τις διαφορές πρέπει να γίνει κατάλληλη μεγέθυνση . Για τον λόγο αυτό παρέδωσα και τα δυο γραφήματα για να γίνει καλύτερα αντιληπτή η μορφή τους.

- Τέλος, θέλω να αναπαραστήσω γραφικά σε κοινό διάγραμμα για τις παραπάνω τιμές του προβλέπτη p το φάσμα του LPC μοντέλου (20log10(|H(ejω)|)) και το φάσμα του παραθυρομένου σήματος x[n] (20log10(|X(ejω)|)). Τα ζητούμενα γραφήματα παρατίθενται παρακάτω :

<u>Για p=52</u> **:**



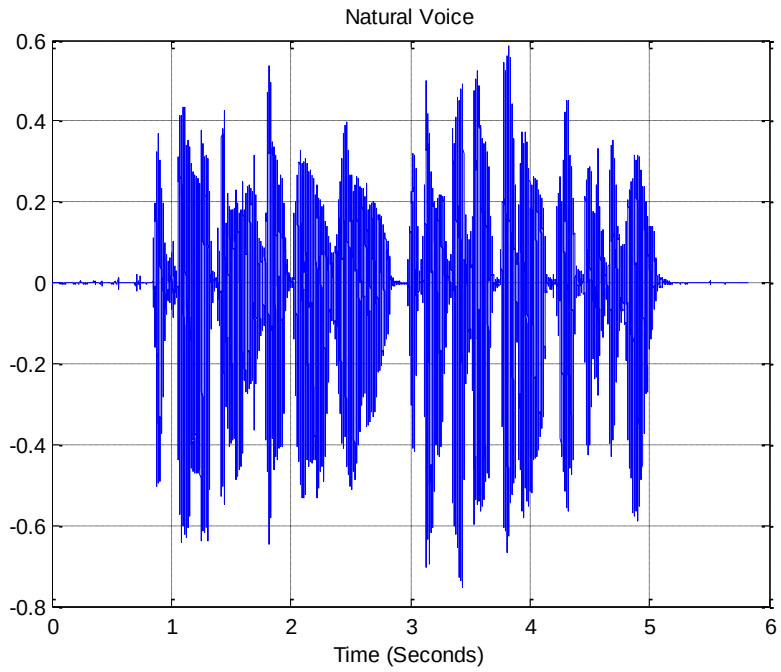
Για p=26 **:**

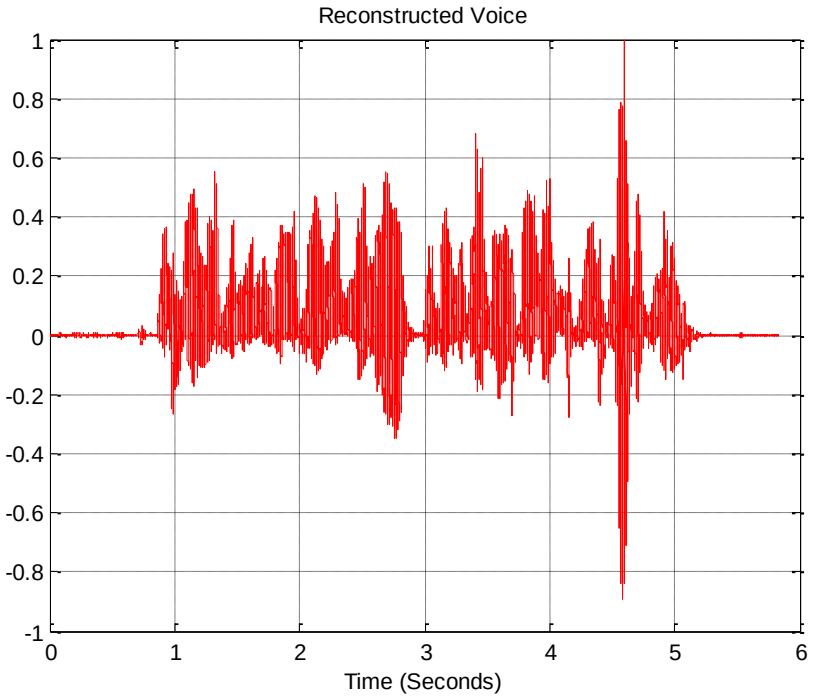


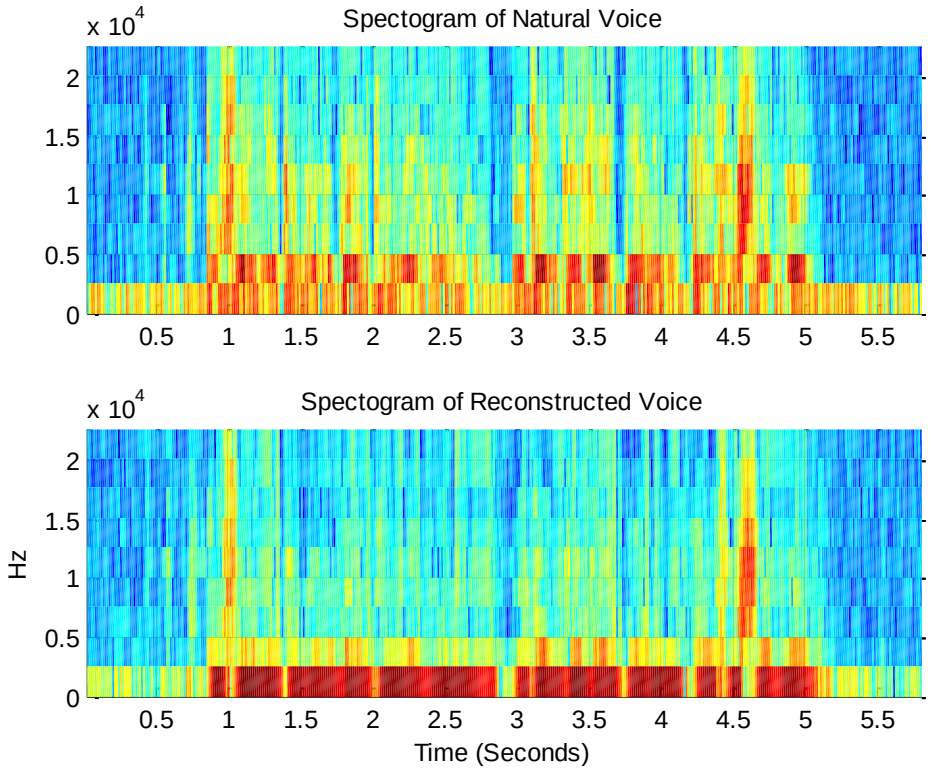
2.2 : Σύνθεση Φωνής με Γραμμική Πρόβλεψη

Στο μέρος αυτό κάνουμε ανάλυση φωνής με γραμμική πρόβλεψη, η οποία γίνεται σε μικρά επικαλυπτόμενα πλαίσια και πρέπει να υπάρχει επικάλυψη ώστε να διασφαλίσουμε τη συνέχεια και την ομαλότητα της ανάλυσης. Συγκεκριμένα θεωρούμε ότι κάθε πλαίσιο ανάλυσης έχει μήκος 30 ms και η επικάλυψη που έχει με το προηγούμενο πλαίσιο είναι ίση με 2L/3 όπως αυτό ζητείται. Η πορεία που ακολουθείται στο μέρος αυτό είναι η εξής :

- Με τη συνάρτηση wavread() του Matlab εισάγουμε το σήμα φωνής που μας ενδιαφέρει να επεξεργαστούμε. Η συχνότητα δειγματοληψίας είναι 48kHz.
- Φορτώνουμε στο Matlab το αρχείο με όνομα pitch.mat με χρήση της συνάρτησης load().
- Περνάμε τις παραμέτρους που μας υποδεικνύει η εκφώνηση στη συνάρτηση με όνομα ***lpc_vocoder()*** και παίρνουμε τα παρακάτω γραφήματα:


Natural Voice
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
Time (Seconds)


Reconstructed Voice
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6
Time (Seconds)



Το ζητούμενο αρχείο το οποίο περιέχει την συνθετική φωνή όπως αυτή προέκυψε από την συνάρτηση lpc_vocoder() υπάρχει στο παραδοτέο φάκελο με όνομα synthesis.wav

Μέρος 3: *Βελτιωμένη Σύνθεση Φωνής με Χρήση Μακροπρόθεσμης Πρόβλεψης και Κατάλληλα Σχεδιασμένης Βάσης Διεγέρσεων (CELP)*

3.1: Εισαγωγή Προβλέπτη μακράς καθυστέρησης

Σκοπός της άσκησης αυτής είναι να δημιουργήσουμε έναν προβλέπτη ο οποίος θα προβλέπει την περιοδικότητα της φωνής, όπου αυτή υπάρχει, και θα ακολουθεί τον πρώτο προβλέπτη κατά την ανάλυση. Όπως αναφέρεται και στην εκφώνηση, οι διαδοχικές περίοδοι όταν έχουμε σήματα έμφωνων ήχων έχουν αρκετή ομοιότητα. Παρόμοια περιοδικότητα εμφανίζεται επίσης και στο λάθος του προβλέπτη (lpc). Προκειμένου να αφαιρεθεί αυτή η περιοδικότητα στην άσκησή μας, δημιουργούμε έναν νέο προβλέπτη $1^{ης}$ τάξης στην ουσία με τύπο:

$$P_e = \beta \cdot z^{-M}$$

Στην ουσία περνάμε δηλαδή το σφάλμα της ανάλυσης του γραμμικού προβλέπτη (lpc_analysis_error) από ένα φίλτρο της μορφής:

$$P_e = \frac{1}{1 - \beta \cdot z^{-M}}$$

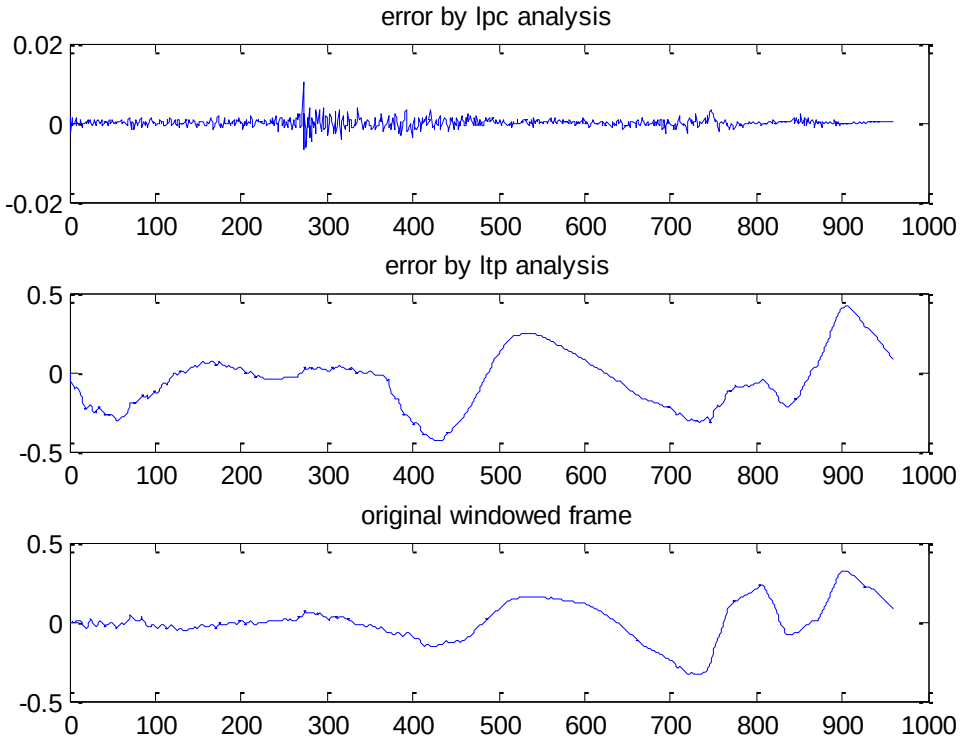
Όπου

- Μ είναι η μακρά καθυστέρηση η οποία αντιστοιχεί σε ακέραια πολλαπλάσια της περιόδου του e[n]
- β ο συντελεστής γραμμικής πρόβλεψης που μπορεί να βρεθεί με το να ελαχιστοποιήσουμε το τετράγωνο του λάθους της νέας πρόβλεψης.

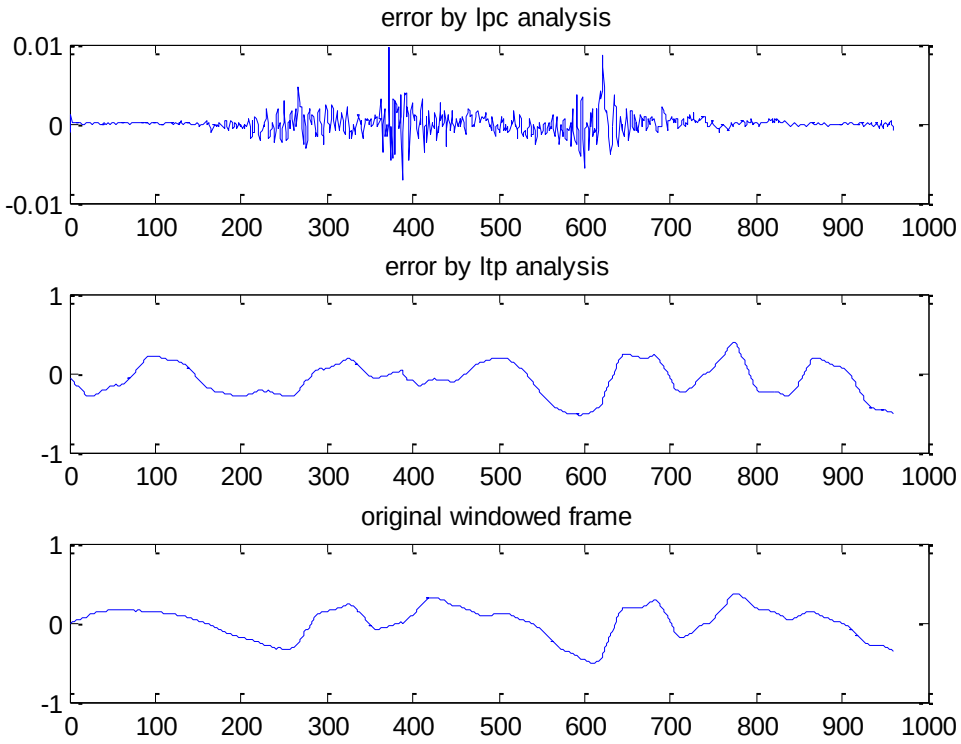
Ο τρόπος με τον οποίο θα υπολογιστούν τα παραπάνω μέσω του matlab θα επεξηγηθεί εκτενώς στη συνέχεια.

Στο πρώτο ερώτημα καλούμαστε να υπολογίσουμε και να υλοποιήσουμε το παραπάνω φίλτρο που αναφέρθηκε καθώς και να παρατεθούν σε διαγράμματα των λαθών: $e[n], \widetilde{e[n]}$ σε αντιπαραβολή με το αρχικό σήμα για έναν έμφωνο ήχο. Προκειμένου να το επιτύχουμε αυτό, σε έναν επαναληπτικό βρόχο , ο οποίος απαιτείται για την υλοποίηση της άσκησης, επιλέγουμε 4 διαδοχικά παράθυρα (i από 88 έως 91) όπου παρατηρήθηκαν και έμφωνοι και άφωνοι ήχοι. Για αυτά τα παράθυρα αφού κάνουμε πρώτα lpc ανάλυση, μέσω της συνάρτησης που μας δόθηκε έτοιμη (lpc analysis), λάβαμε το λάθος (e[n]). Στη συνέχεια αυτό το λάθος το περάσαμε από το φίλτρο $P_e = \frac{1}{1-\beta \cdot z^{-M}}$, όπως αναφέρθηκε και προηγουμένως. Την έξοδο αυτού του φίλτρου, την αφαιρούμε από το αρχικό παράθυρο , που περιέχει τις πραγματικές τιμές του σήματος φωνής που επεξεργαζόμαστε, και έτσι λαμβάνουμε το $\widetilde{e[n]}$. Παρακάτω παρατίθενται τα διαγράμματα των σφαλμάτων αυτών, καθώς και οι τιμές του παραθυροποιημένου σήματος.,

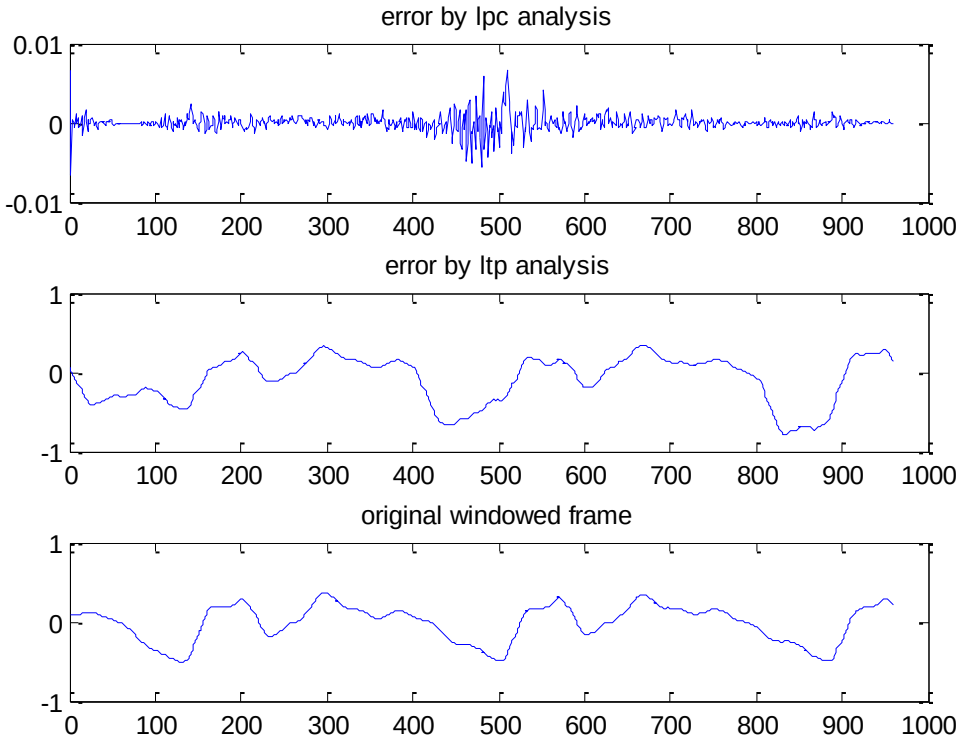
Οι τιμές των $e[n], \widetilde{e[n]}$ και σήματος για το 88^{o} παράθυρο:



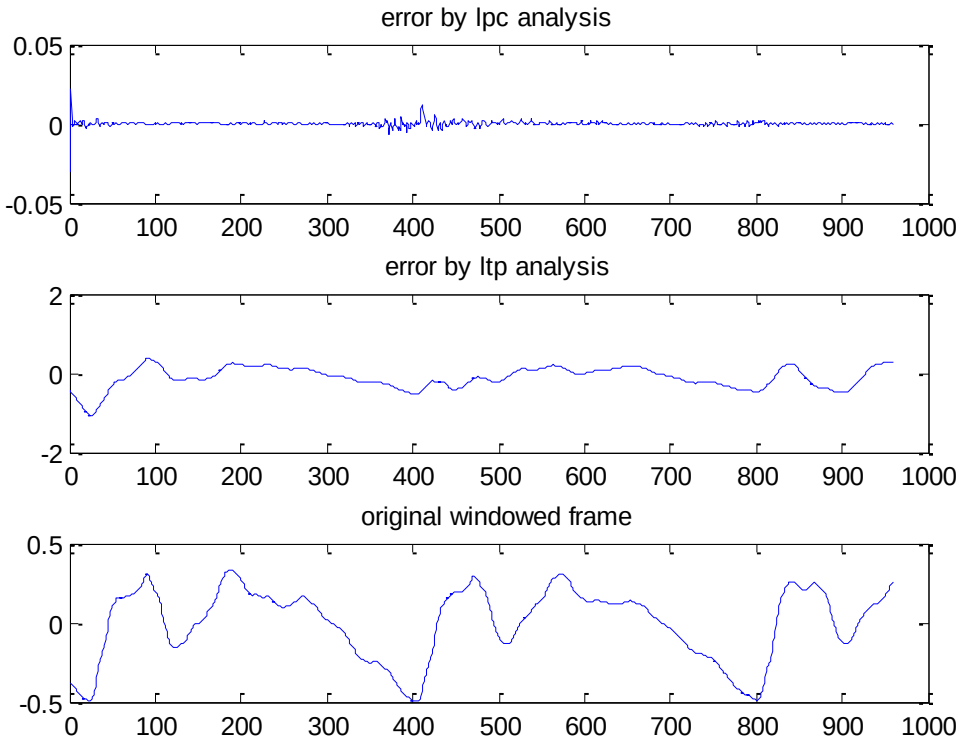
Οι τιμές των $e[n], \widetilde{e[n]}$ και σήματος για το 89^{o} παράθυρο:



Οι τιμές των $e[n], \widetilde{e[n]}$ και σήματος για το 90^{o} παράθυρο:



Οι τιμές των $e[n], \widetilde{e[n]}$ και σήματος για το 91^{o} παράθυρο:



Στη συνέχεια της άσκησης, εισάγουμε κατάλληλα το παραπάνω φίλτρο μακράς πρόβλεψης

$$P_e = \frac{1}{1 - \beta \cdot z^{-M}}$$

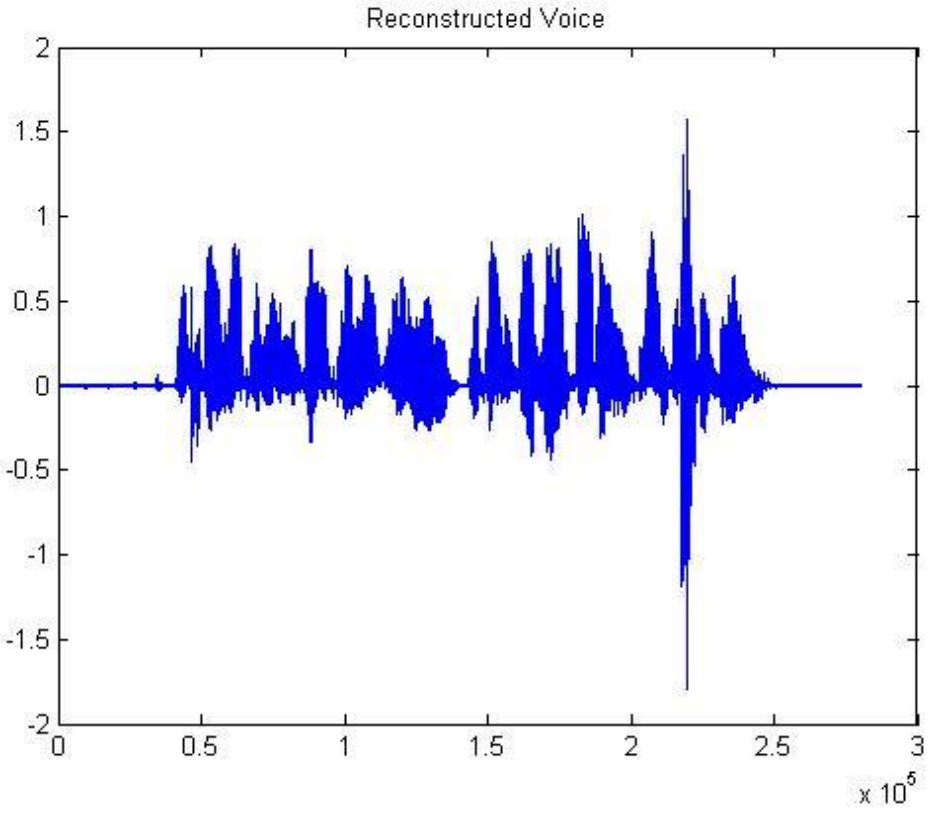
Στον κώδικα της άσκησης 2.2 . Δηλαδή αφού στην ουσία κάνουμε την lpc ανάλυση στο σήμα μας, λαμβάνουμε το error που μας δίνει αυτή. Για να βρούμε τα β και τα M ακολουθούμε την εξής διαδικασία:

Παίρνουμε την αυτοσυσχέτιση του lpc_error με τη συνάρτηση xcorr. Επειδή όμως η αυτοσυσχέτιση είναι συμμετρική επιλέγουμε να πάρουμε τις τιμές της από το peak της και μετά , καθώς αυτές αντιστοιχούν σε θετικές συχνότητες. Σε αυτό το πεδίο λοιπόν της υπολογισμένης αυτοσυσχέτισης, αφού κόψουμε έναν αριθμό δειγμάτων από την αρχή (στην περίπτωση μας 100 αφού η αυτοσυσχέτιση έχει τελικά μήκος 960 δειγμάτων) αναζητούμε το μέγιστο αυτής καθώς και το δείκτη που μας δίνει τη θέση αυτού του μεγίστου. Με δεδομένες τις τιμες αυτές τα β, M υπολογίζονται με τον παρακάτω τρόπο:

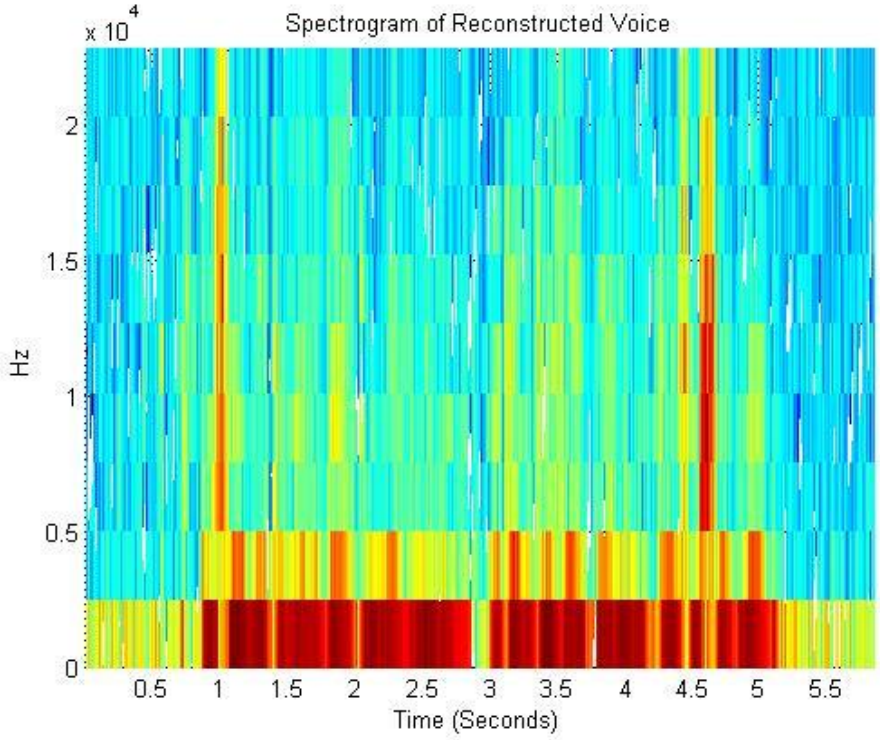
- M= ο δείκτης του μεγίστου αυτού που βρήκαμε
- Και $\beta = \frac{Τιμή\ της\ αυτοσυσχέτισης\ στη\ θέση\ M}{Μέγιστη\ τιμή\ της\ αυτοσυσχέτισης\ του\ σφάλματος\ (peak)}$

Τα παραπάνω φαίνονται και στους κώδικες matlab που παραδίδονται μαζί με την παρούσα αναφορά.

Μετά την έξοδο του φίλτρου, μπορούμε να παρατηρήσουμε ότι ο ήχος που ανασυνθέσαμε έχει κάπως καλύτερη ποιότητα (βέβαια υπάρχει μια μικρή προσθήκη θορύβου) έχοντας επίσης μια αμελητέα διαφορά του κόστους σε bits. Η τελική εικόνα του ήχου μας είναι:

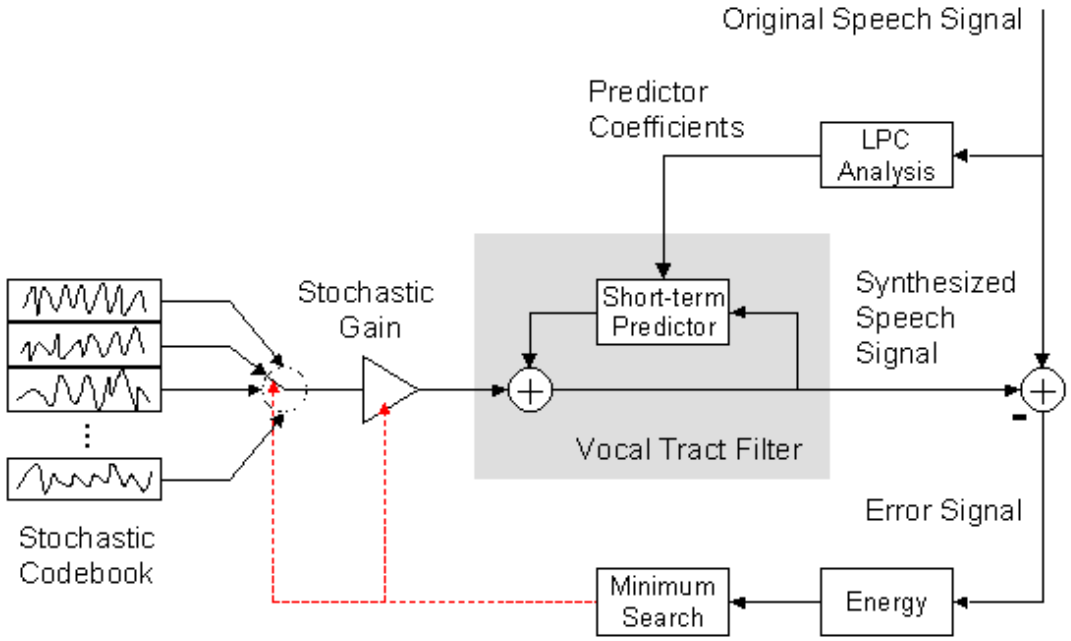


Το φασματογράφημα του ήχου που ανασυνθέσαμε είναι το ακόλουθο: (παρατίθεται και στο zip αρχείο)



3.2 Δημιουργία Βάσης Τυχαίων Διεγέρσεων και Επιλογή της βέλτιστης για σύνθεση

Στο κομμάτι αυτό της άσκησης, θα εισάγουμε ένα codebook (βάση τυχαίων διεγέρσεων) , από τις οποίες θα επιλέγουμε την καταλληλότερη για την ανασύνθεση της φωνής. Αυτό θα βελτιώσει αισθητά την ποιότητα του ήχου μας. Όσον αφορά τη διαδικασία επιλογής της καταλληλότερη διέγερσης, αυτή υλοποιείται με το ακόλουθο σύστημα που θα επεξηγηθεί στη συνέχεια:



Ακολουθούμε σε αυτό το ερώτημα την ίδια διαδικασία που ακολουθήθηκε και στο προηγούμενο με μια σημαντική διαφορά . Αντί να χρησιμοποιήσουμε παλμούς της συνάρτησης ugenerator για την ανασύνθεση της φωνής μέσω του lpc_vocoder δημιουργούμε 1024 διαφορετικές διεγέρσεις της μορφής:

$$v_n = \sum_{k=0}^{N-1} c_k \cos(\pi k n/N + \phi_k), n = 0, 1, \ldots, 2N-1$$

Όπου Ck και Φk είναι τυχαίες μεταβλητές. Η Φk είναι ομοιόμορφα κατανεμημένη μεταξύ 0 και 2π ενώ η Ck είναι Rayleigh – κατανεμημένη με σ.π.π.:

$$p(c_k) = c_k \exp(-c_k^2/2), c_k > 0$$

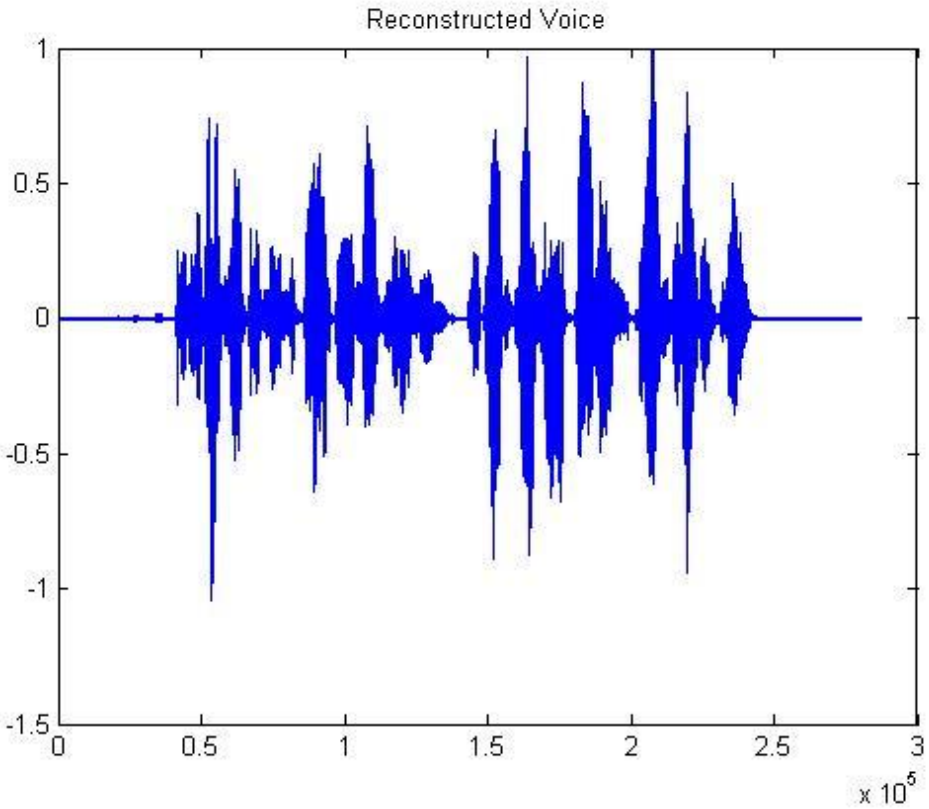
Πιο αναλυτικά, η διαδικασία που ακολουθήσαμε για τη βελτιωμένη ανασύνθεση φωνής σε αυτό το ερώτημα είναι η ακόλουθη:

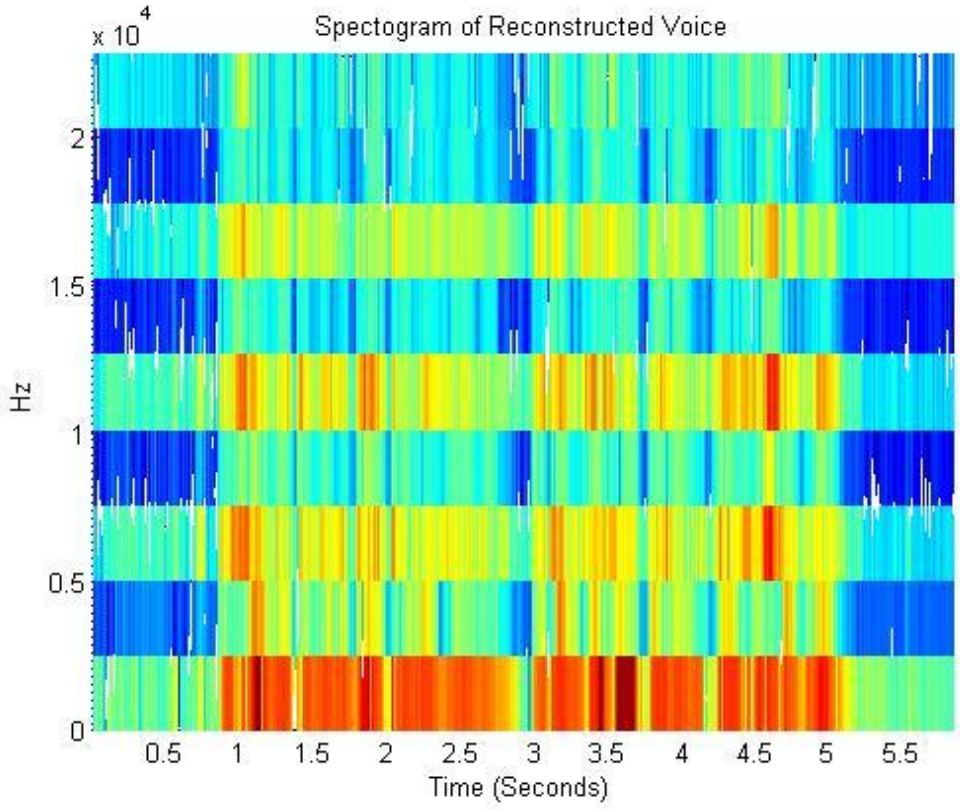
Αρχικά, διαβάζουμε μέσω του wavread το σήμα μας. Παίρνουμε, όπως και στην άσκηση 3.1 παράθυρα 20 ms με overlap 10 ms. Παράγουμε το codebook που θα χρησιμοποιήσουμε όπως αυτό ορίζεται από την εκφώνηση, με βάση δηλαδή τις παραπάνω συναρτήσεις. Στη συνέχεια για κάθε παράθυρο του σήματός μας, κάνουμε lpc ανάλυση λαμβάνοντας τους συντελεστές a,G καθώς και το lpc_error. Με τη διαδικασία που περιγράφηκε και στο προηγούμενο ερώτημα (xcorr(lpc_error) κτλ.) υπολογίζουμε τις τιμές των β, M του κάθε παραθύρου. Στη συνέχεια, για κάθε παράθυρο περνάμε όλες τις διεγέρσεις Vn από τα εξής φίλτρα:

LTP Φίλτρο: $P_e = \frac{1}{1-\beta \cdot z^{-M}}$, με τα β, M κάθε παραθύρου

LPC Φίλτρο: $H(z) = \frac{G}{A(z)}, \text{όπου } A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}$

Στη συνέχεια, αφού πάρουμε τις εξόδους των παραπάνω φίλτρων, υπολογίζουμε τη διαφορά τους από το παραθυροποιημένο σήμα. Για αυτό το σήμα που θα μας προκύψει, υπολογίζουμε την ενέργεια του. Επιλέγουμε για κάθε παράθυρο τη διέγερση που δίνει τη μικρότερη ενέργεια. Έτσι βρίσκεται η καταλληλότερη διέγερση για κάθε παράθυρο. Αφού λοιπόν, εντοπίσουμε αυτά τα σήματα, κάνουμε overlap and add και συνθέτουμε αυτές τις καταλληλότερες διεγέρσεις ώστε να πάρουμε το τελικό μας σήμα που επιθυμούμε. Το ηχογραφημένο σήμα βρίσκεται στο zip αρχείο. Η τελική εικόνα του σήματος που συνθέσαμε όπως και το φασματογράφημά του βρίσκονται και αυτά στο zip αρχείο όμως παραρατίθενται και εδώ άμεσα για πληρότητα:





Η ποιότητα του ήχου που ανασυνθέσαμε είναι καλύτερη, ωστόσο παρατηρούνται κάποια σημάδια θορύβου που μας την επηρεάζουν κάπως.

Μέρος 4: Κωδικοποίηση και Συμπίεση

Στην άσκηση αυτή θα εφαρμόσουμε κωδικοποίηση και συμπίεση του αρχείου ήχου μας. Προκειμένου να το κάνουμε αυτό θα κβαντίσουμε τις παραμέτρους του LPC συνθέτη. Ωστόσο αντί για τους συντελεστές της γραμμικής πρόβλεψης $\{\alpha_i\}$, θα χρησιμοποιήσουμε τις παρακάτω παραμέτρους: $g_i = log\frac{1-k_i}{1+k_i}$ (log area ratios). Αυτό το πράττουμε γιατί ενίοτε οι συντελεστές γραμμικής πρόβλεψης μπορεί να παρουσιάσουν προβλήματα αστάθειας.Οι συντελεστές k_i που χρησιμοποιούμε παραπάνω είναι οι συντελεστές ανάκλασης (PARCOR) οι οποίοι προκύπτουν από τους συντελεστές $\{\alpha_i\}$ με τον αλγόριθμο Levinson-Durbin.

Από την εκφώνηση δίνεται ότι η συχνότητα με την οποία γίνεται η lpc ανάλυση είναι 100 frames per second.Για την κβάντιση των παραπάνω συντελεστών, όπως υποδεικνύεται θα χρησιμοποιήσουμε:

- *5 bits για τον κβαντισμό (σε γραμμική κλίμακα) κάθε παραμέτρου g_i*
- *6 bits για τον κβαντισμό (σε γραμμική κλίμακα) της θεμελιώδους περιόδου του pitch (στον κώδικα του matlab κάνουμε load το pitch που μας δίνεται)*
- *1 bit για το αν ο ήχος είναι έμφωνος ή άφωνος*
- *5 bits για τον κβαντισμό (σε λογαριθμική κλίμακα) του κέρδους G του μοντέλου γραμμικής πρόβλεψης.*

Ζητείται ο υπολογισμός του αρχικού ρυθμού πληροφορίας (kbits/sec) του σήματος φωνής, εάν υποθέσουμε 16 bits ανά δείγμα καθώς και τον τελικό ρυθμό πληροφορίας μετά την ανωτέρω κωδικοποίηση με κβαντισμένες παραμέτρους.

Ο αρχικός ρυθμός πληροφορίας είναι ο ακόλουθος:

Έχουμε 16 Bits/sample και 279563 samples στο σήμα φωνής μας.

Η διάρκεια του σήματος φωνής είναι 5.87 sec.

Άρα καταλήγουμε ότι:

Αρχικός ρυθμός πληροφορίας$= \frac{16*279563}{5{,}87} = 762011{,}5843\frac{bits}{sec} = 762\ kbits/sec$

Ο τελικός ρυθμός πληροφορίας θα είναι ο ακόλουθος:

Έχουμε:

- 5bits για κάθε ένα από τους 52 όρους g_i του κάθε frame
- 1bit για κάθε frame για το αν ο ήχος είναι έμφωνος ή άφωνος
- 5bits για τον συντελεστή G(Gain) κάθε frame
- 6bits για τη θεμελιώδη περίοδο του Pitch σε κάθε frame

Άρα καταλήγουμε ότι : (το σύνολο των frames είναι 584)

Τελικός ρυθμός πληροφορίας$= \frac{(5*52+1+5+6)}{5.87} * 584 \cong 27\frac{kbits}{sec}$

Ο λόγος συμπίεσης είναι λοιπόν:

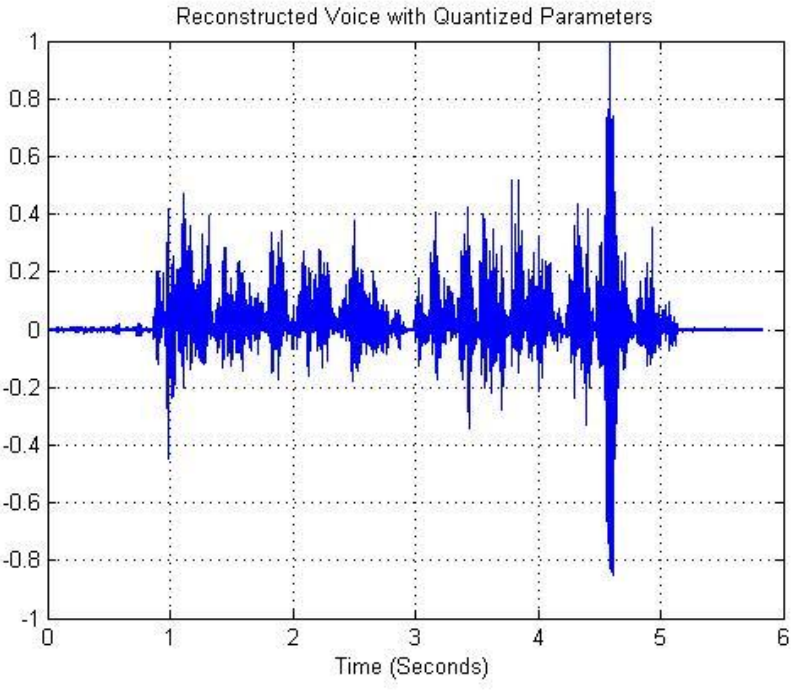
$$compress_{ratio} = \frac{762}{27} = 28.22$$

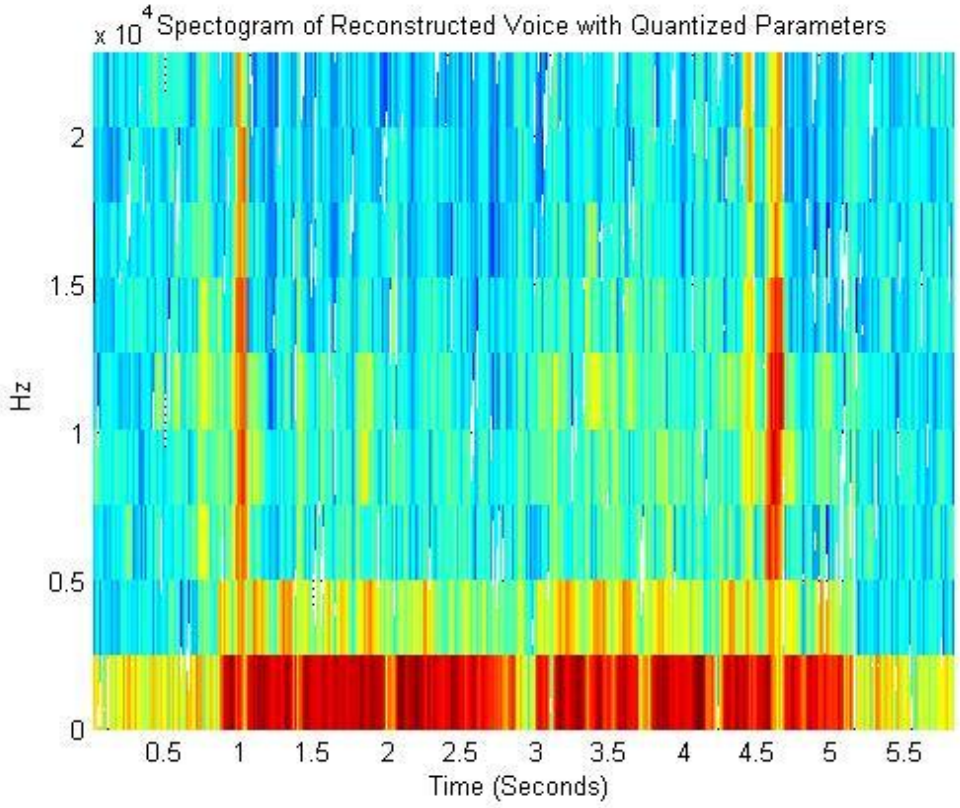
Πρακτικά, όσα αναφέρθηκαν παραπάνω υλοποιούνται με τις συναρτήσεις και τα προγράμματα του matlab που βρίσκονται στο zip αρχείο.

Η υλοποίηση όλης της διαδικασίας γίνεται ως εξής:

Παίρνουμε και πάλι το σήμα και το «σπάμε» σε παράθυρα. Αυτή τη φορά χρησιμοποιούμε τη συνάρτηση Buffer για να πάρουμε παράθυρα μήκους 30 ms με overlap 20 ms. Στη συνέχεια περνάμε κάθε παράθυρο από τη συνάρτηση lpc analysis και κρατάμε τους συντελεστές γραμμικής πρόβλεψης καθώς και το G κάθε παραθύρου σε έναν πίνακα. Με χρήση της συνάρτησης **poly2rc** του matlab υπολογίζουμε τους συντελεστές Ki από τους οποίους τελικά λαμβάνουμε τους συντελεστές που ζητάμε (gi) μέσω του τύπου που δίνεται στην εκφώνηση. Τους αποθηκεύουμε και αυτούς σε πίνακες ώστε να τους έχουμε στη διάθεση μας συνολικά για όλα τα παράθυρα για να είμαστε σε θέση να τους κβαντίσουμε. Ακόμη για κάθε παράθυρο τσεκάρουμε την τιμή του pitch για να δούμε αν ο ήχος είναι έμφωνος ή άφωνος. Αφού κάνουμε όλα τα παραπάνω, υλοποιούμε μια συνάρτηση linearQ που στην ουσία είναι αυτή που κβαντίζει τις τιμές που θέλουμε. Υπολογίζει εσωτερικά η συνάρτηση, τον αριθμό των διαστημάτων ανάλογα με τον αριθμό των Bits που έχουμε στη διάθεσή μας, τις τιμές των επιπέδων καθώς και τις σχετικές αποστάσεις. Με αυτή τη συνάρτηση και τα κατάλληλα ορίσματα υλοποιούμε τόσο τις κβαντίσεις σε γραμμική όσο και σε λογαριθμική κλίμακα (απλά εδώ δίνουμε ως όρισμα το log του μεγέθους που θέλουμε να κβαντίσουμε και παίρνουμε πίσω το 10^(τιμές της linearQ)).

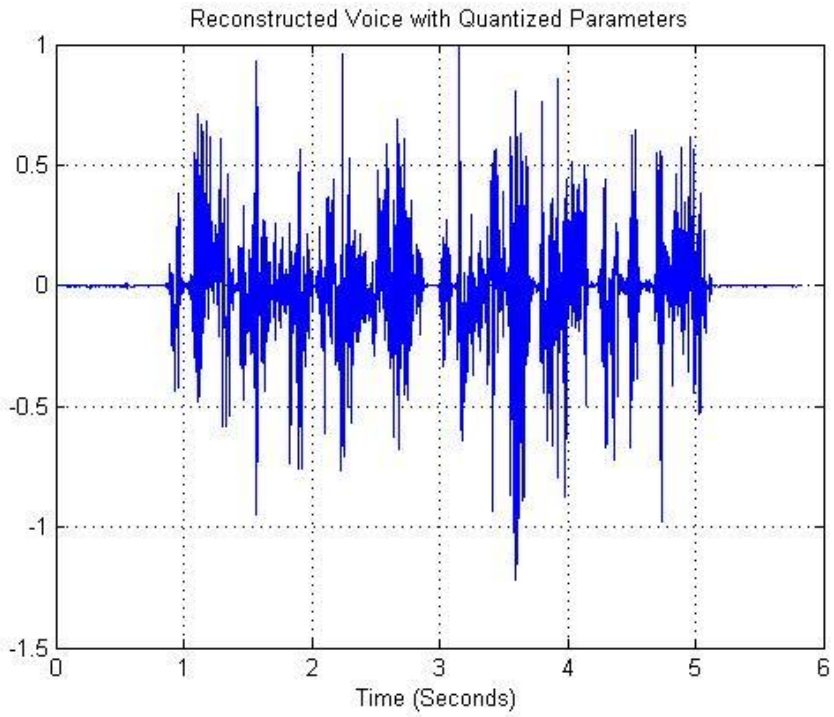
Έτσι έχουμε κβαντίσει όλα τα μεγέθη που επιθυμούμε. Στη συνέχεια, περνάμε τις κβαντισμένες παραμέτρους σε μια τροποποιημένη συνάρτηση του vocoder, την `Quantized_Values_Voc` η οποία αναλαμβάνει να κάνει την ανασύνθεση του σήματος φωνής έχοντας ως βάση για τα φίλτρα της, όχι τις πραγματικές παραμέτρους που είχε στις προηγούμενες ασκήσεις της παρούσας σειράς (2,3) , αλλά τις νέες κβαντισμένες τιμές που υπολογίσαμε. Για την απλή ανασύνθεση φωνής με χρήση του φίλτρου της απλής LPC Analysis αλλά με κβαντισμένες παραμέτρους, παίρνουμε το ακόλουθο σήμα φωνής καθώς και το φασματογράφημά του:

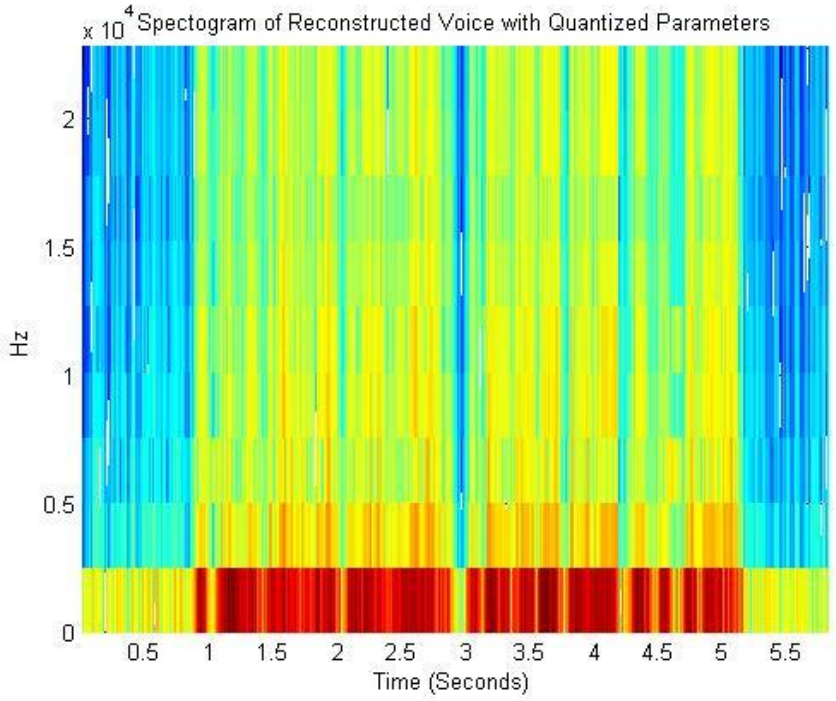




Οι παραπάνω εικόνες βρίσκονται και στο zip αρχείο. Το σήμα που εξέρχεται από αυτή τη σύνθεση είναι το `'synthesis_encoded_lpc.wav'`

Στη συνέχεια, για κάθε παράθυρο, όπως κάναμε και στην 3.2 , υπολογίζουμε τους συντελεστές β και M. Τους κρατάμε και αυτούς σε πίνακες για να τους έχουμε συνολικά για όλο το σήμα. Με ανάλογο τρόπο, όπως τους συντελεστές του προηγούμενου ερωτήματος, κβαντίζουμε και αυτούς. (5 bits για το β σε γραμμική κλίμακα και 5 bits για το M σε λογαριθμική κλίμακα)
Αφού το κάνουμε αυτό καλούμε με τις κατάλληλες παραμέτρους τη συνάρτηση `Quantized_Values_Celp_Voc` η οποία περνάει το σήμα μας στην ουσία από δύο φίλτρα προτού το ανασυνθέσει. Τα 2 αυτά φίλτρα είναι αυτά που αναφέραμε και στο ερώτημα 3.2 απλά με τις νέες κβαντισμένες παραμέτρους. Με τις εξόδους αυτών ανασυνθέτουμε με overlap and add το τελικό σήμα φωνής μας. Η εικόνα του σήματος εξόδου και το φασματογράφημά του είναι οι παρακάτω:





Οι παραπάνω εικόνες βρίσκονται και στο zip αρχείο. Το σήμα που εξέρχεται από αυτή τη σύνθεση είναι το `'synthesis_encoded_celp.wav'`