Νίκος Παγώνας, ΑΜ el18175
Εργασία MATLAB στα πλαίσια του μαθήματος Σήματα και Συστήματα

Σημείωση: Όλα τα σχήματα, διαγράμματα και φάσματα για τα οποία γίνεται συζήτηση στην παρούσα εργασία μπορούν να παρατηρηθούν τρέχοντας το program.m (και έχοντας τα αρχεία υπό συζήτηση στο ίδιο path με το program.m για να μην δημιουργηθούν conflicts). Κρίθηκε επομένως σκόπιμο για οικονομία χώρου να μην προστεθούν σχήματα στην αναφορά, καθώς αυτά μπορούν να ελεγχθούν κατά βούληση από τον κώδικα matlab. Όλα τα figures έχουν κατάλληλους τίτλους ώστε να καθίσταται σαφές το μέγεθος στο οποίο αναφέρονται.

1 Σχεδίαση Φίλτρων

1.1 Σχεδίαση Βαθυπερατών Φίλτρων (Κινούμενου Μέσου)

α. Σύμφωνα με την εξίσωση που περιγράφει τα φίλτρα κινούμενου μέσου, ορίζουμε τα διανύσματα b = [1/(N+1) 1/(N+1) … 1/(N+1)] και a = [1 0 0 0 … 0] για N = 2,4,10.

β.Σχεδιάζουμε την απόκριση πλάτους και φάσης των φίλτρων αυτών χρησιμοποιώντας την εντολή freqz(). Παρατηρούμε κάποιου είδους “περιοδικότητα” και στις δύο αποκρίσεις η συχνότητα της οποίας αυξάνεται όσο αυξάνεται και το N.

γ.Σχεδιάζουμε τα διαγράμματα πόλων και μηδενικών μέσω της συνάρτησης zpd(), αφού πρώτα μετατρέψουμε τις συναρτήσεις μεταφοράς των φίλτρων σε μορφή πόλων-μηδενικών, μέσω της συνάρτησης tf2zp(). Έχουμε κάθε φορά τόσους πόλους όσους και μηδενικά (αν θεωρήσουμε και την πολλαπλότητα του κάθε πόλου). Οι πόλοι είναι πάντα μηδενικοί, ενώ τα μηδενικά βρίσκονται πάνω στον μοναδιαίο κύκλο.

1.2 Σχεδίαση Ζωνοπερατών Φίλτρων

α. Σχεδιάζουμε με τη βοήθεια του MATLAB ένα φίλτρο με πόλους στις θέσεις {0.68±0.51i} και μηδενικά στις θέσεις {1.2,-0.6}. Σχεδιάζουμε το διάγραμμα πόλων και μηδενικών με την συνάρτηση zpd.m. Ακολουθούμε την αντίστροφη διαδικασία με πριν, και πλέον βρίσκουμε τους συντελεστές του φίλτρου μέσω της συνάρτησης zp2tf.

β. Σχεδιάζουμε την απόκριση πλάτους και φάσης του φίλτρου μέσω της συνάρτησης freqz(). Βλέπουμε ότι το φίλτρο είναι (κατά προσέγγιση) υψιπερατό, καθώς οι συχνότητες κοντά στο π ενισχύονται ενώ οι συχνότητες κοντά στο 0 σχεδόν αποκόπτονται.

γ. Σχεδιάζουμε την κρουστική και την βηματική απόκριση του συστήματος με την χρήση των συναρτήσεων impz() και stepz() αντίστοιχα.

δ. Αν μετακινήσουμε τους πόλους στις θέσεις {0.76±0.57i}, {0.8±0.6i}, {0.84±0.63i}, κρατώντας τα μηδενικά στις ίδιες θέσεις με πριν, και σχεδιάζοντας την βηματική απόκριση για κάθε περίπτωση, αλλά και την απόκριση πλάτους για την πρώτη περίπτωση, παρατηρούμε μεγάλη ενίσχυση κοντά στο ω = 0.65. Όσον αφορά τις βηματικές αποκρίσεις, αντίστοιχα, παρατηρούμε, απόσβεση στην πρώτη περίπτωση, σταθερή ημιτονοειδή μορφή στην δεύτερη, και σταδιακά ενισχυόμενη ημιτονοειδή στην τρίτη.

ε. Διεγείρουμε το σύστημα χρησιμοποιώντας μια κρουστική παλμοσειρά (με την βοήθεια της συνάρτησης gensig(), την πρώτη φορά με περίοδο 50s ενώ την δεύτερη με 5s. Η παλμοσειρά με περίοδο 50s έχει φαίνομενα απόσβεσης προσωρινά ενώ η 5s διατηρεί σταθερό το πλάτος της εξόδου (σε ημιτονοειδή μορφή πάντα)

στ. επαναλαμβάνουμε τα α και β για πόλους στις θέσεις ±0.8i αυτή τη φορά. Το φίλτρο είναι ζωνοπερατό με κεντρική συχνότητα περίπου ίση με 1.5.

2. Ανάλυση και Σύνθεση Μουσικών Σημάτων

2.1 Ανάλυση Μουσικών Σημάτων

α. Φορτώνουμε στο MATLAB, με την χρήση της audioread() τα αρχεία που μας έχουν δοθεί, και αντιστοιχούν σε ήχους από φλάουτο, κλαρινέτο και βιολοντσέλου. Και τα τρία δείγματα έχουν συχνότητα δειγματοληψίας 44.1 kHz. Ακούμε τα σήματα με την χρήση της εντολής sound().

β. Μέσω της plot, απεικονίζουμε τα τρία σήματα στο πεδίο του χρόνου. Αυτά είναι (κατά προσέγγιση) περιοδικά με περιόδους:
φλάουτο: 1.9 ms άρα 44.1*10^3 δείγμ./s * 1.9*10^-3 s = 83 δείγματα
κλαρινέτο: 3.8 ms άρα 44.1*10^3 δείγμ/s * 3.8*10^-3 s = 167 δείγματα
τσέλο: 4.18 ms άρα 44.1*10^3 δείγμ/s *4.18*10^-3 s = 184 δείγματα

γ. Μέσω της εντολής fft() σχεδιάζουμε το φάσμα του κάθε σήματος, και βρίσκουμε την θεμελιώδη του συχνότητα.

φλάουτο: 528Hz, κλαρινέτο: 261Hz, τσέλο: 240Hz
άρα 1/528 = 1.89ms, 1/261=3.83ms, 1/240=4.16ms άρα επαληθεύονται οι παραπάνω (εποπτικές) περίοδοι.

Το τσέλο έχει περισσότερες αρμονικές που “παίζουν ρόλο” (υπολογίσιμο πλάτος που αντιστοιχεί στην συχνότητα της κάθε αρμονικής) από το κλαρινέτο, και το κλαρινέτο περισσότερες από το φλάουτο. Αυτό προφανώς αλλάζει και το ηχόχρωμα του κάθε οργάνου, με αποτέλεσμα ακόμα και την ίδια νότα να έπαιζαν τα όργανα, το ηχόχρωμα κάθε οργάνου θα ήταν αρκετό για να τα ξεχωρίσουμε.

ε. Φορτώνουμε στο matlab το αρχείο cello_noisy.wav, που είναι το προηγούμενο σήμα νότας τσέλου με προσθήκη Γκαουσιανού θορύβου. Υπολογίζοντας το φάσμα του με την εντολή fft(), παρατηρούμε ότι έχουμε τις ίδιες συχνότητες να ενισχύονται με πριν, αλλά βλέπουμε ότι όλες σχεδόν οι συχνότητες επιπλέον από αυτές που είχαμε πριν, έχουν ενισχυθεί κατά ένα μικρό ποσό. Αυτό είναι το αποτέλεσμα του θορύβου και η μικρή ενίσχυση είναι ο λόγος που μπορούμε ακόμα να ακούμε καθαρά το τσέλο παρά τον θόρυβο.

2.2 Σύνθεση Μουσικών Σημάτων ως Άθροισμα Ημιτονοειδών

α. Επιλέγουμε το cello_note.wav για την ανάλυσή μας. Απομονώνουμε ένα κομμάτι του ίσα με 10 φορές την θεμελιώδη του περίοδο.

β. Υπολογίζουμε για αυτό το απόσπασμα, το διακριτό μετασχηματισμό Fourier με την χρήση της συνάρτησης fft και πλοτάρουμε το πλάτος του. Υπολογίζουμε προσεγγιστικά του συντελεστές cn=An/A1 και **γ.** φn (τα φn υπολογίζονται από το φάσμα φάσης)

δ. προσθέτουμε τα επιμέρους ημιτονοειδή που προέκυψαν από τις παραμέτρους cn και φn

ε. Το ανακατασκευασθέν σήμα παρατηρείται ότι δεν είναι ίδιο με το αρχικό. Αυτό μπορεί να οφείλεται σε αριθμητικό σφάλμα, ή σε λανθασμένους υπολογισμούς. Το μόνο σίγουρο είναι ότι μετά από αρκετές αρμονικές, το σήμα πλησιάζει (θεωρητικά) όλο και περισσότερο στο αρχικό. Το αποτέλεσμα αρχικά διορθώνεται με γρήγορο ρυθμό, αλλά ο ρυθμός διόρθωσης όλο και μειώνεται (λόγω της όλο και μειούμενης συμβολής των μεγαλύτερων αρμονικών στην διαμόρφωση του σήματος)

στ. αποθηκεύουμε το ανακατασκευασμένο σήμα σε ένα αρχείο τύπου .wav μέσω της ρουτίνας audiowrite().