ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
Τμήμα Επιστήμης Υπολογιστών

ΗΥ-215: Εφαρμοσμένα Μαθηματικά για Μηχανικούς
Εαρινό Εξάμηνο 2020-21
Διδάσκοντες: Γ. Στυλιανού, Γ. Καφεντζής

Τρίτη Σειρά Ασκήσεων

Ημερομηνία Ανάθεσης: 19/3/2021 Ημερομηνία Παράδοσης: 3/4/2021, 23:59

Οι ασκήσεις με [$\star$] είναι **bonus**, $+10$ μονάδες η καθεμία στο βαθμό αυτής της σειράς ασκήσεων (δηλ. μπορείτε να πάρετε μέχρι $90/70$ σε αυτή τη σειρά.)

Άσκηση 1 - Διαφορικές Εξισώσεις
Υπολογίστε τη συνολική έξοδο του συστήματος που περιγράφεται από τη διαφορική εξίσωση

$$\frac{d^2}{dt^2}y(t) + 7\frac{d}{dt}y(t) + 10y(t) = x(t) - \frac{d}{dt}x(t) \tag{1}$$

με αρχικές συνθήκες $y(0^-) = 0$, $y'(0^-) = -1$ και είσοδο $x(t) = e^{-t}u(t)$. Δηλαδή

(α') βρείτε την απόκριση μηδενικής εισόδου, $y_{zi}(t)$

(β') βρείτε την κρουστική απόκριση, $h(t)$

(γ') βρείτε την απόκριση μηδενικής κατάστασης, $y_{zs}(t)$

(δ') τέλος, βρείτε τη συνολική έξοδο $y_{tot}(t) = y_{zi}(t) + y_{zs}(t)$

Απ.: (α) $y_{zi}(t) = \Big(-\frac{1}{3}e^{-2t} + \frac{1}{3}e^{-5t}\Big)u(t)$, (β) $h(t) = (e^{-2t} - 2e^{-5t})u(t)$

Άσκηση 2 - Συνέλιξη
Υπολογίστε αναλυτικά τη συνέλιξη μεταξύ των σημάτων

(α') $x(t) = u(1-t)$, $y(t) = e^{-t}u(t-1)$

(β') $x(t) = e^{-t}u(t-1)$, $y(t) = 2u(t-1)$

Απ.: (α) $c(t) = \frac{1}{e}u(2-t) + e^{-(t-1)}u(t-2)$, (β) $c(t) = 2\Big(\frac{1}{e} - e^{-(t-1)}\Big)u(t-2)$

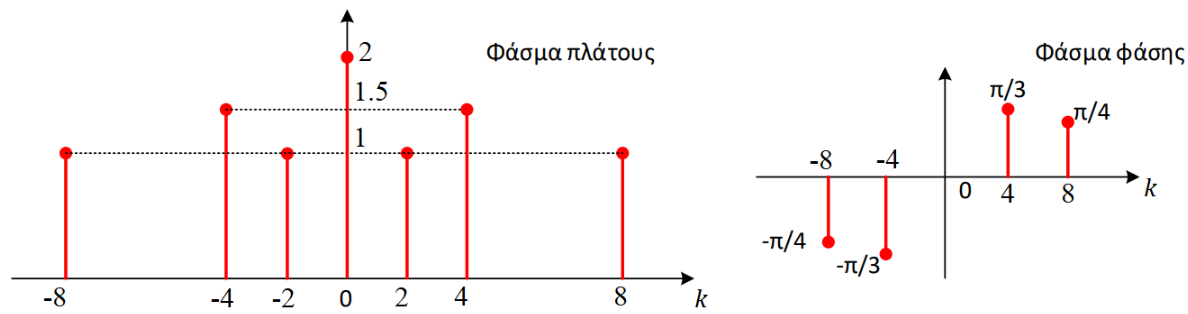
Άσκηση 3 - Σειρές Fourier - I
Αναπτύξτε το παρακάτω σήμα σε εκθετική Σειρά Fourier και σχεδιάστε τα φάσματα πλάτους και φάσης της.

$$x(t) = 2\sin^2(2\pi t) - \cos(4\pi t) \tag{2}$$

Ποιά είναι η περίοδος T_0 και η θεμελιώδης συχνότητα f_0 του περιοδικού σήματος;

Απ.: $T_0 = 0.5$ s

Άσκηση 4 - Σειρές Fourier - II
Σας δίνεται το φάσμα πλάτους και φάσης ενός περιοδικού σήματος με περίοδο $T_0 = 0.005$ s στο Σχήμα 1.



Σχήμα 1: *Φάσματα Άσκησης 4.*

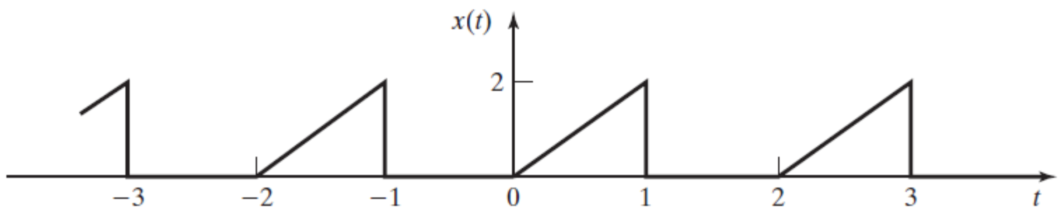
(α΄) Τα παραπάνω φάσματα αντιστοιχούν σε ένα σήμα στο πεδίο του χρόνου. Είναι αυτό πραγματικό ή όχι; Εξηγήστε.

(β΄) Εκφράστε το περιοδικό σήμα σε εκθετική και τριγωνομετρική Σειρά Fourier.

Απ.: (β) $x_{\text{trig}}(t) = 2 + 2\cos(2\pi 400t) + 3\cos(2\pi 800t + \pi/3) + 2\cos(2\pi 1600t + \pi/4)$

Άσκηση 5 - Σειρές Fourier - III

Έστω το περιοδικό σήμα που δίνεται στο Σχήμα 2.



Σχήμα 2: *Περιοδικό σήμα Άσκησης 5.*

Αναπτύξτε το σήμα σε εκθετική Σειρά Fourier με δυο τρόπους:

(α΄) με τον ορισμό, όπου για να είναι λιγότερο επώδυνη η προσπάθειά σας, σας δίνεται ότι

$$\int te^{at}dt = \frac{e^{at}}{a^2}(at-1) \tag{3}$$

(β΄) με χρήση ιδιοτήτων όπως η παραγώγιση/ολοκλήρωση στο χρόνο.

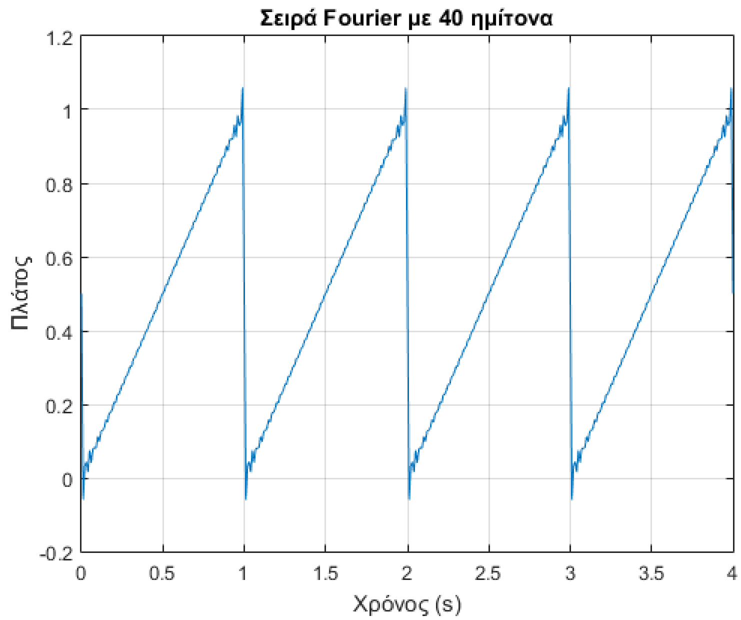
Οι συντελεστές Fourier X_k μπορεί να μην έχουν την ίδια αλγεβρική έκφραση στα δυο υποερωτήματα, ανάλογα με τις πράξεις που θα κάνετε, αλλά ουσιαστικά θα είναι ίδιοι συντελεστές X_k και μπορείτε να το επιβεβαιώσετε στην Άσκηση 6.

Απ.: $X_0 = \frac{1}{2}$, $X_k = \left(\frac{1}{j\pi k}\right)\left(\frac{1}{j\pi k}(1-(-1)^k) - (-1)^k\right)$ (πιθανή μορφή απάντησης)

Άσκηση 6 - Σειρές Fourier στο MATLAB/Octave

Στην Άσκηση 5 υπολογίσαμε στο χαρτί τους συντελεστές μιας Σειράς Fourier. Ας δούμε πώς θα μπορούσαμε να επιβεβαιώσουμε την απάντησή μας στο MATLAB/Octave, και με χρήση αυτού του κώδικα, να μπορούμε να σχηματίζουμε τη σειρά Fourier οποιουδήποτε περιοδικού σήματος. Έστω λοιπόν η τριγωνομετρική σειρά Fourier

$$x(t) = \frac{A}{2} + \sum_{k=1}^{+\infty} \frac{A}{k\pi}\cos(2\pi k f_0 t + \pi/2) \tag{4}$$



Σχήμα 3: *Σειρά Fourier.*

Στο μάθημα έχετε δει κώδικα που υλοποιεί μια Σειρά Fourier με χρήση for loop. Επειδή όμως το MATLAB/Octave είναι κατασκευασμένο ώστε να είναι πολύ πιο γρήγορο όταν γράφουμε κώδικα με πράξεις πινάκων - αντί για for loops - θα δούμε τώρα μια τέτοια υλοποίηση. Ο κώδικας MATLAB/Octave που υλοποιεί την τριγωνομετρική Σειρά Fourier για $A = 1$ και $T_0 = 1$ χωρίς for loop δίνεται στο αρχείο `trigFS.m` και το αποτέλεσμα φαίνεται στο Σχήμα 3. Κατανοήστε γιατί δουλεύει έτσι, σκεπτόμενοι/ες ότι όλες οι μεταβλητές του MATLAB/Octave είναι πίνακες. **Δε χρειάζεται να γράψετε κάτι στην παράδοση σας για αυτό**.

Εναλλακτικά, κι επειδή πολλές φορές η τριγωνομετρική Σειρά δεν υπολογίζεται εύκολα στο χαρτί ώστε να τη μεταφέρουμε στο MATLAB/Octave άμεσα, μπορούμε να υλοποιήσουμε την Εκθετική Σειρά Fourier όπως στο αρχείο `expFS.m` που σας δίνεται. Συμπληρώστε τον κώδικα που σας δίνεται για την εκθετική Σειρά Fourier - εκεί που γράφει `INSERT CODE HERE` **- και προγραμματίστε στο MATLAB/Octave 3 περιόδους από την εκθετική Σειρά Fourier της Άσκησης 5 και παραδώστε τον κώδικα και το σχήμα που προκύπτει**.

Επιπλέον, προγραμματίστε 3 περιόδους από τα περιοδικά σήματα που αντιστοιχούν στους παρακάτω συντελεστές εκθετικής Σειράς Fourier.

I. $T_0 = 1,\ X_0 = 1,\ X_k = \frac{j(-1)^k}{k\pi},\ k \neq 0$

II. $T_0 = 6,\ X_0 = 1/2,\ X_k = \begin{cases} 0, & k \text{ άρτιο} \\ \frac{6}{\pi^2 k^2}\sin(\pi k/2)\sin(\pi k/6), & k \text{ περιττό} \end{cases}$

III. $T_0 = 3,\ X_0 = 1,\ X_k = \dfrac{3j}{2\pi^2 k^2}\left(e^{j2\pi k/3}\sin(2\pi k/3) + 2e^{jk\pi/3}\sin(\pi k/3)\right),\ \ k \neq 0$

Παραδώστε τον κώδικα για τις παραπάνω - 4 συνολικά - Σειρές Fourier.

[$\star$] Άσκηση 7 - Σύνθεση Μουσικής στο MATLAB/Octave

Αν θελήσουμε να δημιουργήσουμε στο MATLAB/Octave ένα ημιτονοειδές σήμα συχνότητας $f = 440$ Hz (νότα ΛΑ), διάρκειας 0.5 sec (δηλ. 500 ms), με μηδενική φάση μετατόπισης ϕ και μοναδιαίου πλάτους A, με συχνότητα δειγματοληψίας $f_s = 16000$ Hz (που αντιστοιχεί σε καλής ποιότητας ηχογράφηση) θα γράψουμε τις ακόλουθες εντολές:

```
fs = 16000;
f = 440;
dur = 500;
t = 0:1/fs:dur/1000;
signal = sin(2*pi*f*t);
```

Για να ακούσουμε αυτό που φτιάξαμε, πληκτρολογούμε

```
soundsc(signal, fs);
```

Στην άσκηση αυτή θα συνθέσουμε τμήμα από το πολύ γνωστό κομμάτι του Beethoven, Fur Elise[1], από νότες πλήκτρων πιάνου.

(α΄) Με χρήση της εντολής

```
load('FurEliseData');
```

φορτώνετε το αρχείο `FurEliseData.mat` που σας δίνεται στο MATLAB/Octave.

Παρατηρήστε ότι δημιουργούνται 4 πίνακες - διανύσματα, `lefthand`, `lhdur`, `righthand`, `rhdur` οι οποίοι ανταποκρίνονται σε αριθμούς πλήκτρων πιάνου που παίζονται από το αριστερό χέρι (`lefthand`), με τις αντίστοιχες διάρκειές τους σε ms (`lhdur`), και σε αριθμούς πλήκτρων πιάνου που παίζονται με το δεξί χέρι (`righthand`), με τις αντίστοιχες επίσης διάρκειές τους σε ms (`rhdur`).

(β΄) Μετατρέψτε σε κάθε χέρι, κάθε αριθμό πλήκτρου πιάνου n στην αντίστοιχη συχνότητα με τη σχέση

$$f(n) = f_{LA} \times 2^{\frac{n-49}{12}} = 440 \times 2^{\frac{n-49}{12}} \tag{5}$$

Στο MATLAB/Octave αυτό γίνεται εύκολα ως

```
freq = 440*2.^((lefthand - 49)/12)
```

με αποτέλεσμα έναν πίνακα - διάνυσμα `freq` με τις αντίστοιχες συχνότητες. Οι παύσεις αναπαρίστανται στους πίνακες με τα πλήκτρα ως (0). Θα μετατραπούν κι αυτές - ενώ δεν πρέπει - όμως αυτό δε θα μας επηρεάσει.

(γ΄) Οι πίνακες `lhdur`, `rhdur` περιέχουν τη διάρκεια κάθε νότας σε ms για το αριστερό και το δεξί χέρι, αντίστοιχα. Ακολουθώντας τον παρακάτω κώδικα, δημιουργήστε τα αντίστοιχα ημίτονα με τη σωστή διάρκεια. Για τις παύσεις, θα δημιουργήσουμε ένα μηδενικό διάνυσμα με την ανάλογη διάρκεια.

```
% Poses notes exoume sto aristero xeri?
N = length(lefthand);

% Syxnothta deigmatolhpsias
fs = 16000;

% As ftia3oume ta hmitona
for i = 1:N
    % o a3onas tou xronou
    t = 0:1/fs:lhdur(i)/1000;
    if lefthand(i) == 0
    % exoume paush, dhmiourgoume ena dianysma me
    % mhdenika stoixeia, diarkeias lhdur(i) ms
         s{i} = zeros(1, length(t));
```

[1] https://www.youtube.com/watch?v=_mVW8tgGY_w

```
    else
    % ftia3te to antistoixo hmitono me xrhsh tou pinaka
    % freq pou dhmiourghsate nwritera
        freq = % INSERT CODE HERE;
        s{i} = % INSERT CODE HERE;
    end
end
```

(δ') Η δομή `s{}` ονομάζεται cell structure, και σε κάθε θέση της `i` μπορεί να αποθηκευτεί οποιαδήποτε μεταβλητή, οποιουδήποτε μεγέθους. Αυτό είναι βολικό γιατί οι νότες δεν έχουν όλες την ίδια διάρκεια, και ένας πίνακας $N \times L$ θα ήταν δύσχρηστος.

(ε') Όμως εδώ κάθε ημίτονο που φτιάξαμε βρίσκεται σε μια διαφορετική θέση στα cells. Πρέπει να τα βάλουμε διαδοχικά για να μπορέσουμε να ακούσουμε τη μελωδία. Άρα θα χρειαστούμε ένα αρκετά μεγάλο πίνακα - διάνυσμα που θα περιέχει το ένα μετά το άλλο τα σήματα που φτιάξαμε (ημίτονα και παύσεις). Ευτυχώς στο MATLAB/Octave αυτό γίνεται εύκολα, χωρίς να χρειάζεται να διαχειριστούμε ρητά τη μνήμη (εν αντιθέσει με τη C και τις αντίστοιχες `malloc`).

```
% Keno dianysma gia to aristero xeri - arxikopoihsh
signal_left = [ ];

% For loop gia na gemisoume to dianysma me ta diadoxika hmitona
for i = 1:N
    % Se ka8e loop, bazoume ena s{i} sto telos tou
    % prohgoumenou signal_left pou exoume ftia3ei
    signal_left = [ signal_left  s{i} ];
end
```

(ϛ') Επαναλάβατε τη διαδικασία για το δεξί χέρι, δηλ. τις μεταβλητές `righthand`, `rhdur` και αποθηκεύστε τα αποτελέσματά σας σε ξεχωριστές μεταβλητές. Παρατηρήστε ότι το δεξί χέρι παίζει λιγότερες νότες από το αριστερό.

(ζ') Δημιουργήστε το συνολικό σήμα, αθροίζοντας τα δυο σήματα ως

```
sig = signal_left + signal_right;
```

Αν τα έχετε κάνει όλα σωστά, το MATLAB/Octave θα σας επιστρέψει σφάλμα, γιατί λόγω αριθμητικών στρογγυλοποιήσεων και του γεγονότος ότι το ένα χέρι παίζει λιγότερες νότες από το άλλο, το ένα σήμα βγήκε λίγο μικρότερο από το άλλο, οπότε δε γίνεται να προστεθούν[2]. Χρησιμοποιήστε την εντολή `length` και μετρήστε το μήκος του κάθε σήματος. Πόση είναι η διαφορά τους σε πλήθος τιμών; **Απαντήστε σε σχόλιο μέσα στον κώδικα.**

(η') Για μια γρήγορη διόρθωση στο παραπάνω πρόβλημα, χρησιμοποιήστε τις εντολές

```
D = length(signal_right)-length(signal_left);
signal_left = [signal_left zeros(1, D)];
sig = signal_left + signal_right;
```

(θ') Μπορείτε τώρα να ακούσετε το σήμα σας!

```
soundsc(sig, fs);
```

[2]Να θυμάστε ότι τα πάντα στο MATLAB/Octave είναι πίνακες, και όλες οι πράξεις πρέπει να τηρούν τους κανόνες διάστασης των πινάκων.

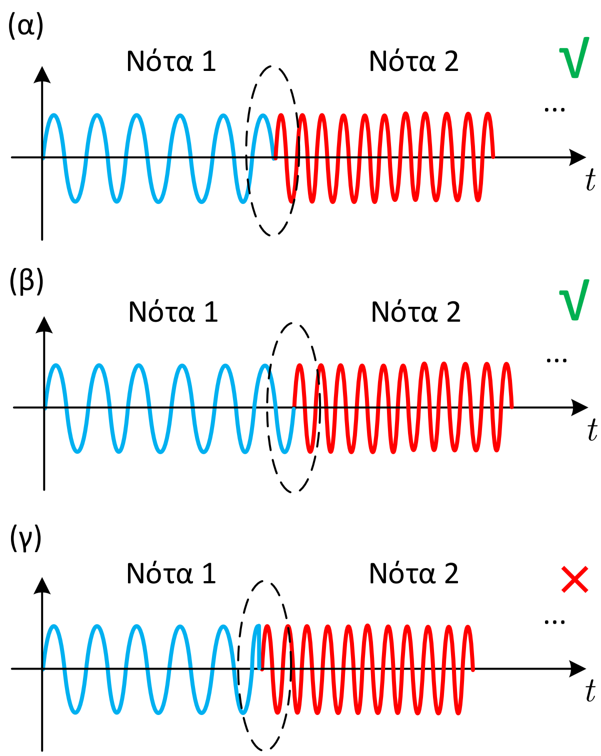
Σχολιάστε το αποτέλεσμα. Να σημειωθεί ότι ένα πραγματικό πλήκτρο πιάνου δεν παράγει ένα ημίτονο, όπως κάναμε εμείς, αλλά ένα άθροισμα από ημίτονα, με συχνότητες σχεδόν ακέραιες πολλαπλάσιες της θεμελιώδους συχνότητας και διαφορετικά μεταξύ τους πλάτη. Με άλλα λόγια, παράγει - προσεγγιστικά - μια *σειρά Fourier*, με πεπερασμένο πλήθος όρων! Επίσης, σε ένα πραγματικό πιάνο, υπάρχουν πάρα πολλές λεπτομέρειες που μας χαρίζουν τον πλούσιο ήχο του, οι οποίες δεν είναι απλό να μοντελοποιηθούν[3].

Παραδώστε πλήρη κώδικα MATLAB/Octave σε script, σκελετό του οποίου θα βρείτε στο αρχείο `fe.m`**, που παράγει τα σήματα του δεξιού και του αριστερού χεριού, δημιουργεί το συνολικό σήμα, και το παίζει ως ήχο.**

[⋆] **Άσκηση 8 - Βελτίωση της προηγούμενης άσκησης**

Όσοι/ες εκτελέσατε με επιτυχία την προηγούμενη άσκηση, θα παρατηρήσατε ότι ο ήχος που δημιουργήσατε είχε κάποια clicks κάθε φορά που άλλαζε μια νότα. Σίγουρα συμφωνείτε ότι είναι πολύ ενοχλητικό στο αυτί μας.

Ο λόγος εμφάνισης αυτών των clicks είναι ο εξής: όταν δημιουργείτε ένα ημίτονο μιας συγκεκριμένης συχνότητας και συγκεκριμένης διάρκειας, δε σημαίνει απαραίτητα ότι το ημίτονο ξεκινά και τερματίζει σε μηδενικό πλάτος. Ο τερματισμός ενός ημιτόνου σε τυχαίο πλάτος και η αρχή του επομένου από μηδενικό πλάτος δημιουργεί μια απότομη αλλαγή ασυνέχεια/αλλαγή φάσης, η οποία ακούγεται σαν click. Δείτε το Σχήμα 4. Σε κάθε παράδειγμα του σχήματος



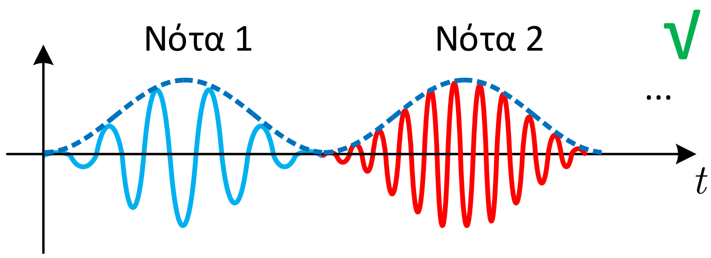
Σχήμα 4: *Σχήμα Άσκησης 8.*

βλέπετε δυο ημίτονα διαφορετικής συχνότητας σαν αυτά που δημιουργήσατε στην προηγούμενη άσκηση, με το ένα να ακολουθεί το άλλο. Αν οι διάρκεια του πρώτου ημιτόνου είναι κατάλληλα επιλεγμένη, τότε η συνένωση της δεύτερης νότας γίνεται ομαλά, όπως στο Σχήμα 4(α). Όμοια και στο Σχήμα 4(β), όπου η συνένωση είναι ακόμα πιο ομαλή και μια τέτοια περίπτωση είναι ιδανική. Και στις δυο περιπτώσεις δεν υπάρχουν clicks. Όμως αυτό που στην πραγματικότητα

[3]Δείτε το - διαθέσιμο στο διαδίκτυο - άρθρο με τίτλο “Perceptual significance of inharmonicity and spectral envelope in the piano bass range”, των Galembo et al., σελ. 6., εξίσωση (7), για να καταλάβετε πόσο πολύπολοκο είναι να συνθέσει κανείς έναν ήχο πλήκτρου πιάνου που να μοιάζει με τον πραγματικό. Η απλή όψη της εξίσωσης είναι αρκετή για να σας πείσει! ;-)

κάνατε στην προηγούμενη άσκηση είναι το Σχήμα 4(γ), καθώς η διάρκεια της κάθε νότας δεν τερμάτιζε απαραίτητα σε σημείο τέτοιο ώστε η ένωση της επόμενης νότας να σχηματίζει συνέχεια στο σήμα. Τις περισσότερες - αν όχι όλες - φορές, ένα ημίτονο που δημιουργούσατε τερμάτιζε σε στιγμή που το πλάτος του δεν ήταν μηδενικό, και αναγκαστικά δημιουργούνταν μια ασυνέχεια στο ηχητικό σήμα λόγω της προσθήκης της επόμενης νότας, που ξεκινούσε από το μηδέν (αφού ήταν συναρτήσεις $\sin(\cdot)$, αλλά το ίδιο θα συνέβαινε αν χρησιμοποιούσαμε $\cos(\cdot)$).

Μια εύκολη λύση αυτού του προβλήματος είναι ο πολλαπλασιασμός του ημιτόνου που μοντελοποιεί κάθε νότα με ένα σήμα *παραθύρου*, ίδιας διάρκειας με το ημίτονο που παριστάνει κάθε νότα. Γνωρίζουμε ήδη δυο τέτοια παράθυρα: τον τετραγωνικό παλμό και τον τριγωνικό παλμό. Ουσιαστικά, ο τετραγωνικός παλμός χρησιμοποιείται έμμεσα κάθε φορά που δημιουργείτε ένα ημίτονο συγκεκριμένης διάρκειας. Όμως μπορείτε να χρησιμοποιήσετε άλλα παράθυρα, τέτοια ώστε να αναγκάζουν το ημιτονοειδές σήμα να κατεβαίνει ομαλά στο μηδέν στα άκρα του, όπως στο Σχήμα 5, όπου χρησιμοποιήσαμε το γνωστό παράθυρο Hanning, που έχει κωδωνοειδή μορφή. Με αυτό τον τρόπο τα clicks μειώνονται



Σχήμα 5: *Εφαρμογή παραθύρου Hanning.*

σημαντικά, ενώ δημιουργούνται και κάποια “εξωτικά” εφέ στον ήχο που παράγεται. ☺

Το MATLAB/Octave έχει έτοιμες συναρτήσεις παραθύρων για να χρησιμοποιήσετε. Όλες συντάσσονται με τον ίδιο τρόπο:

```
window = window_type(duration);
```

με το `window_type` να είναι ένα από τα εξής:

```
hamming
hanning
bartlett
blackman
```

και τη μεταβλητή `duration` να είναι η διάρκεια του παραθύρου σε δείγματα (όχι δευτερόλεπτα!). Χρησιμοποιήστε τη συνάρτηση `length` για να υπολογίσετε τη διάρκεια σε δείγματα κάθε ημιτονοειδούς, και δώστε αυτήν την τιμή ως όρισμα σε κάθε παράθυρο. Πολλαπλασιάστε το παράθυρο που δημιουργήσατε με τα ημίτονα που κατασκευάζετε στην προηγούμενη άσκηση. Θα χρειαστείτε τον τελεστή `.*` για τον πολλαπλασιασμό και ίσως τον τελεστή `'` για να γίνει σωστά η πράξη του γινομένου. Ακούστε και σχολιάστε το αποτέλεσμα.

Παραδώστε τον ίδιο κώδικα MATLAB/Octave σε script που φτιάξατε πριν, κατάλληλα τροποποιημένο ώστε οι νότες που συνθέτετε να είναι πολλαπλασιασμένες με ένα παράθυρο της επιλογής σας από τα παραπάνω. Καταγράψτε τις παρατηρήσεις σας σε σχόλιο στον κώδικα.

Άσκηση 9 - ΓΧΑ Συστήματα και Παραγωγή Ηχούς

Κατά την παραγωγή και καταγραφή ήχου σε ένα χώρο όπου υπάρχουν πολλές ανακλάσεις, εμπόδια, κλπ., το σήμα του ήχου καταγράφεται ως άθροισμα πολλών διαφορετικών “εκδόσεων” (καθυστερήσεων) του σήματος που επιστρέφουν μαζί στο μικρόφωνο, ως ηχώ, κατά την καταγραφή.

Μπορούμε να μοντελοποιήσουμε την ηχώ ως ένα ΓΧΑ σύστημα, το οποίο περιγράφεται από τη σχέση

$$y(t) = x(t) + ax(t - t_d) \tag{6}$$

με a το πλάτος της ηχούς και t_d τη θέση της στο χρόνο, δηλ. τη χρονική στιγμή που εμφανίζεται στο ηχογραφημένο σήμα.

(α΄) Βρείτε στο χαρτί σας την κρουστική απόκριση, $h(t)$, του ΓΧΑ συστήματος, δεδομένου ότι γνωρίζετε ότι η κρουστική απόκριση δίνεται ως η έξοδος ενός συστήματος για είσοδο $x(t) = \delta(t)$.

(β΄) Θα μπορούσαμε να προσθέσουμε κι άλλα αντίγραφα της ηχούς, σε διαφορετικές χρονικές στιγμές και με διαφορετικούς συντελεστές. Όπως μπορείτε εύκολα να καταλάβετε, ένα τέτοιο σύστημα θα είναι της μορφής

$$y(t) = x(t) + \sum_{i=1}^{N} a_i x(t - t_i) \tag{7}$$

Βρείτε στο χαρτί σας την κρουστική απόκριση, $h(t)$, του παραπάνω ΓΧΑ συστήματος.

(γ΄) Θα υλοποιήσουμε το παραπάνω σύστημα παραγωγής ηχούς επάνω σε ένα οποιοδήποτε ηχητικό σήμα εισόδου. Θα συμπληρώσετε μια συνάρτηση στο MATLAB/Octave η οποία θα έχει την παρακάτω σύνταξη:

```
[y_echo, h] = echo_filter_tostudents(signal, times, attenuations, fs)
```

Τα ορίσματα εξηγούνται στα σχόλια στο `echo_filter_tostudents.m` αρχείο που θα βρείτε στο site μαζί με αυτήν την εκφώνηση.

(δ΄) Μια μικρή επεξήγηση για τη γραμμή 25. Επειδή όλα τα σήματα που επεξεργαζόμαστε στον υπολογιστή είναι διακριτού χρόνου, δηλ. ορισμένα για συγκεκριμένες χρονικές τιμές (και όχι για κάθε t), εσείς πρέπει αρχικά να ορίσετε τις τιμές του διανύσματος `times` που θέλετε να ακούγεται η ηχώ (σε δευτερόλεπτα), και να μετατρέψετε στη γραμμή 25 κάθε τιμή του διανύσματος αυτού σε ακέραιες τιμές, δηλ. σε δείγματα. Αυτό γίνεται αν λάβετε υπόψη σας ότι η συχνότητα δειγματοληψίας `fs` ενός σήματος σας λέει ότι σε ένα δευτερόλεπτο ηχογράφησης έχουν παρθεί και αποθηκευτεί `fs` δείγματα (τιμές) του σήματος στον υπολογιστή. Άρα, για παράδειγμα, η χρονική στιγμή $t_0 = 0.5$ s αντιστοιχεί στο δείγμα διακριτού χρόνου `fs/2`. Σε ποιά δείγματα αντιστοιχούν οι δικές σας χρονικές στιγμές της ηχούς που ορίσατε στο διάνυσμα `times`; Αυτή τη μετατροπή πρέπει να γράψετε στη γραμμή 25.

(ε΄) Μπορείτε να χρησιμοποιήσετε ένα οποιοδήποτε σήμα φωνής/μουσικής σε μορφή `.WAV` για να ελέγξετε τη λειτουργία του συστήματός σας. Απλά φροντίστε να μην είναι πολύ μεγάλης διάρκειας για να μην κρασάρετε το MATLAB/Octave. Για δική σας ευκολία, σας δίνονται δυο αρχεία μαζί με τον κώδικα. Οι εντολές για να φορτώσετε ένα `.WAV` σήμα στο MATLAB/Octave είναι:

```
[signal, fs] = audioread('onoma-arxeiou.wav');
```

Αν έχετε παλιότερη έκδοση του MATLAB/Octave και η συνάρτηση αυτή δεν υπάρχει, χρησιμοποιήστε την `wavread`.

Η μεταβλητή `fs` πρέπει να δωθεί ως όρισμα στη συνάρτηση που παράγει την ηχώ.

Παραδώστε συμπληρωμένο τον κώδικα MATLAB/Octave που σας δίνεται.