Συστήματα Πολυμέσων και Εικονική Πραγματικότητα
Εργασία 2025-2026

Ομάδα Κατανόησης Πολυμέσων
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

1 Εισαγωγή

Η παρακάτω εργασία αποτελεί *προαιρετικό* μέρος του μαθήματος "Συστήματα Πολυμέσων και Εικονική Πραγματικότητα" και η εκτέλεσή της συνεισφέρει 1, 3 ή και 4 επιπλέον μονάδες στην τελική βαθμολογία.

Η εργασία θα πρέπει να εκτελεστεί σε ομάδες μέχρι και δύο ατόμων.

Η εργασία αποτελείται από 3 ενότητες. Η υλοποίηση της εργασίας μπορεί κατ' επιλογή να περιλάβει την πρώτη ενότητα, την πρώτη και τη δεύτερη κ.ο.κ. συνεισφέροντας τις αντίστοιχες μονάδες για την κάθε ενότητα. Δε μπορεί όμως να εκτελεστεί μία ενότητα χωρίς να έχει ορθά εκτελεστεί η προηγούμενή της.

Η εργασία στοχεύει στην υλοποίηση ενός κωδικοποιητή/αποκωδικοποιητή ήχου κατά το πρότυπο Advanced Audio Coding (AAC). Παραλλαγές του AAC χρησιμοποιούνται από πολλά διεθνή πρότυπα όπως τα MPEG-2, MPEG-4, H.264 κλπ. Η εκδοχή που παρουσιάζεται στην εργασία μοιάζει περισσότερο με τις προδιαγραφές 3GPP TS 26.403 όπου απουσιάζουν κάποια στάδια επεξεργασίας. Εξαίρεση αποτελεί το ψυχοακουστικό μοντέλο, που είναι μία λίγο απλουστευμένη εκδοχή του MPEG AAC. Παρόλες τις απλουστεύσεις, η συγκεκριμένη εκδοχή οδηγεί σε αρκετά καλά αποτελέσματα, όπως θα διαπιστώσετε και στην πράξη.

2 Απλοποιημένη κωδικοποίηση/αποκωδικοποίηση AAC

Η μορφή του κωδικοποιητή και του αποκωδικοποιητή φαίνεται στα Σχήματα 1 και 2 αντίστοιχα.

Η κωδικοποίηση/αποκωδικοποίηση AAC ανήκει στην κατηγορία waveform compression και επιχειρεί να αναπαραστήσει το αρχικό σήμα με τέτοια μορφή ώστε η αποκωδικοποιημένη εκδοχή του να ακούγεται όσο γίνεται πιο όμοια με το αρχικό σήμα. Σαν κριτήριο πιστότητας χρησιμοποιείται το ψυχοακουστικό μοντέλο που επιτρέπει την εισαγωγή παραμορφώσεων του σήματος (θορύβου λόγω κβαντισμού) ο οποίος είναι κάτω από το κατώφλι ακουστότητας. Για το λόγο αυτό κυρίαρχο ρόλο παίζει ο μηχανισμός Psychoacoustic Model που καθοδηγεί το μηχανισμό Quantizer. Για τη μείωση της περίσσειας πληροφορίας ο AAC χρησιμοποιεί κατά βάση την προσέγγιση κωδικοποίησης μετασχηματισμού που υλοποιείται με τη χρήση του λεγόμενου Modified Discrete Cosine Transform (MDCT) στη βαθμίδα Filterbank ενώ για κωδικοποίηση εντροπίας χρησιμοποιεί κωδικοποίηση Huffman που υλοποιείται στην ομώνυμη βαθμίδα.

Αναλυτικότερα, κατά την κωδικοποίηση το αρχικό σήμα ήχου (για μας stereo με δειγματοληψία 48000 samples/sec) χωρίζεται σε επικαλυπτόμενα κατά 50% τμήματα (frames) μήκους 2048 δειγμάτων. Στη συνέχεια κάθε frame κωδικοποιείται αυτόνομα και συνεπώς το τελικά κωδικοποιημένο bitstream αποτελείται από την παράθεση των ακολουθιών bits που αντιστοιχούν στα διαδοχικά frames.

2.1 Sequence Segmentation Control

Μετά το χωρισμό των δειγμάτων σε frames επιλέγεται από το μηχανισμό Sequence Segmentation Control ο τύπος του frame. Υπάρχουν δύο βασικοί και δύο συμπληρωματικοί τύποι:

ONLY_LONG_SEQUENCE	Βασικός τύπος που επιλέγεται όταν στη διάρκεια του frame το σήμα διατηρεί σχετικά αμετάβλητα φασματικά χαρακτηριστικά
EIGHT_SHORT_SEQUENCE	Βασικός τύπος που επιλέγεται όταν τα φασματικά χαρακτηριστικά του σήματος αλλάζουν έντονα στη διάρκεια του frame
LONG_START_SEQUENCE	Συμπληρωματικός τύπος που παρεμβάλλεται μεταξύ ενός ONLY_LONG_SEQUENCE και ενός EIGHT_SHORT_SEQUENCE
LONG_STOP_SEQUENCE	Συμπληρωματικός τύπος που παρεμβάλλεται μεταξύ ενός EIGHT_SHORT_SEQUENCE και ενός ONLY_LONG_SEQUENCE

Η επιλογή του τύπου για το frame i γίνεται ως ακολούθως:

1. Αν το προηγούμενο frame $(i-1)$ ήταν ONLY LONG SEQUENCE το τρέχον θα είναι υποχρεωτικά είτε πάλι ONLY LONG SEQUENCE είτε LONG START SEQUENCE. Μεταξύ των δύο αυτών επιλογών η απόφαση λαμβάνεται

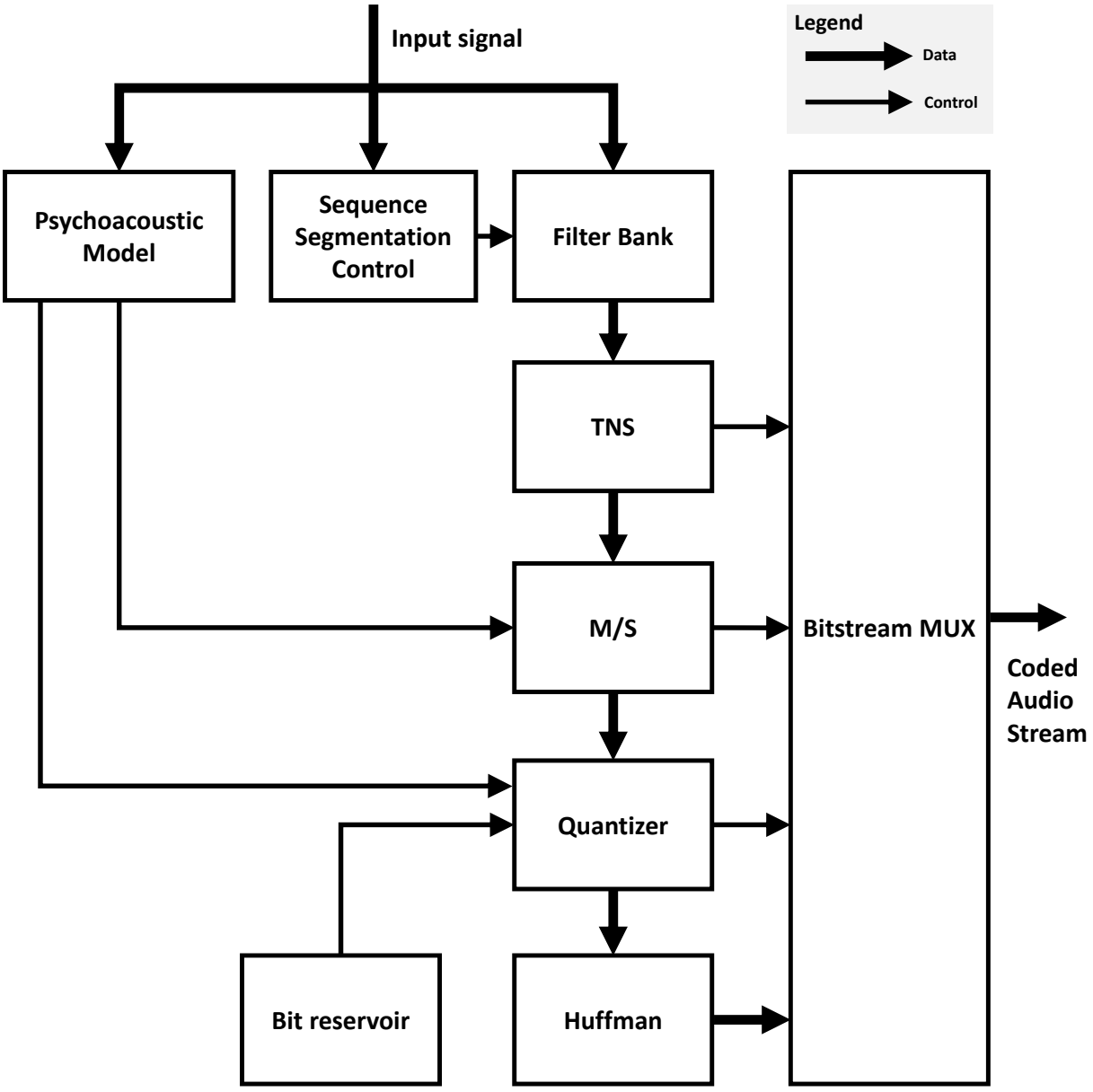

Figure 1: Απλοποιημένος κωδικοποιητής AAC

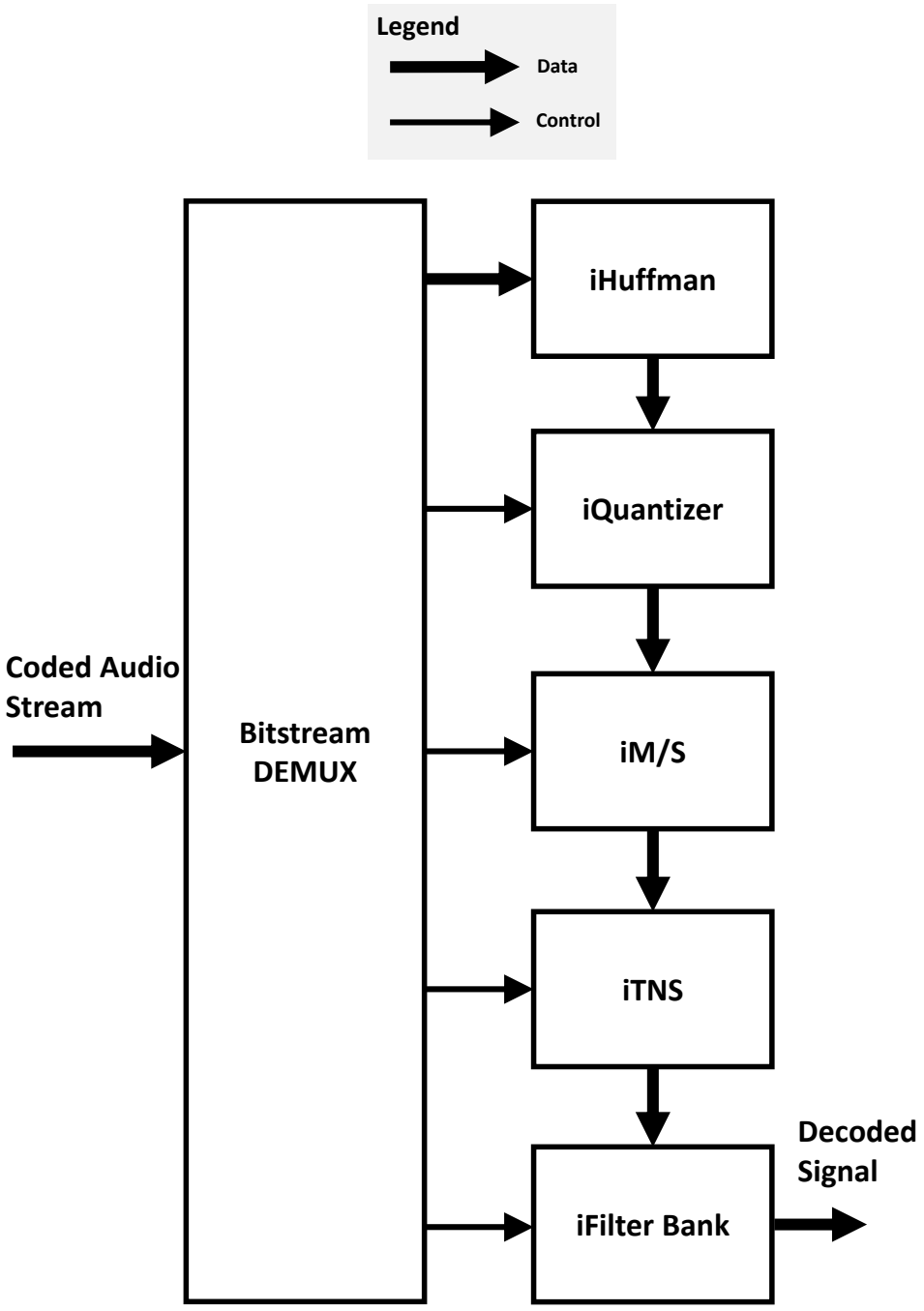

Figure 2: Απλοποιημένος αποκωδικοποιητής AAC

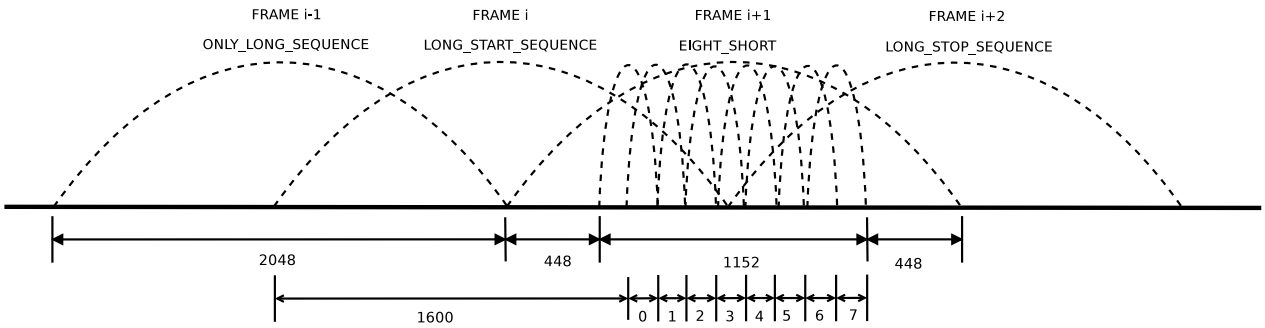

Figure 3: Ακολουθία από 4 frames

με βάση αν το επόμενο frame $(i+1)$ πρόκειται να είναι EIGHT SHORT SEQUENCE ή όχι. Αν το frame $(i+1)$ πρόκειται να είναι EIGHT SHORT SEQUENCE τότε το τρέχον frame i θα κατηγοριοποιηθεί ως μεταβατικό frame τύπου LONG START SEQUENCE.

2. Αντίστροφα, αν το προηγούμενο frame $(i-1)$ ήταν EIGHT SHORT SEQUENCE το τρέχον θα είναι υποχρεωτικά είτε πάλι EIGHT SHORT SEQUENCE είτε LONG STOP SEQUENCE. Μεταξύ των δύο αυτών επιλογών η απόφαση λαμβάνεται πάλι με βάση το αν το επόμενο frame $(i+1)$ πρόκειται να είναι EIGHT SHORT SEQUENCE ή όχι. Αν το frame $(i+1)$ πρόκειται να είναι EIGHT SHORT SEQUENCE τότε το τρέχον frame i θα κατηγοριοποιηθεί ως EIGHT SHORT SEQUENCE αλλιώς ως μεταβατικό frame τύπου LONG STOP SEQUENCE.

3. Τέλος, αν το προηγούμενο frame $(i-1)$ ήταν LONG START SEQUENCE ή LONG STOP SEQUENCE το τρέχον frame i θα κατηγοριοποιηθεί υποχρεωτικά ως EIGHT SHORT SEQUENCE και ONLY LONG SEQUENCE αντίστοιχα.

Είναι φανερό ότι στις παραπάνω περιπτώσεις 1 και 2 απαιτείται εκ των προτέρων έλεγχος του επόμενου frame $(i+1)$. Ο έλεγχος αυτός γίνεται ως εξής με αναφορά στο Σχήμα 3:

1. Τα δείγματα του frame $(i+1)$ φιλτράρονται με το υψιπερατό φίλτρο

$$H(z) = \frac{0.7548 - 0.7548z^{-1}}{1 - 0.5095z^{-1}} \tag{1}$$

Υπόδειξη: Μπορείτε να υλοποιήσετε το φιλτράρισμα σε Python χρησιμοποιώντας τη συνάρτηση `scipy.signal.lfilter` (Documentation)

2. Για κάθε μία από τις 8 περιοχές από 0 έως 7, μήκους 128 δειγμάτων που φαίνονται στο Σχήμα 3, υπολογίζεται, ως εκτίμηση της ενέργειάς τους, το άθροισμα $s_l^2, l = 0, \ldots, 7$ των τετραγώνων των δειγμάτων τους.

3. Υπολογίζονται τα λεγόμενα attack values:

$$ds_l^2 = \frac{s_l^2}{(1/l)\sum_{m=0}^{l-1} s_m^2}$$

που δηλώνουν το λόγο της ενέργειας ενός τμήματος σε σχέση με το μέσο όρο των προηγουμένων

4. Το frame $(i+1)$ αποφασίζεται να κατηγοριοποιηθεί ως EIGHT SHORT SEQUENCE αν συντρέχουν οι συνθήκες $s_l^2 > 10^{-3}$ και $ds_l^2 > 10$ έστω και για ένα από τα τμήματα $l = 1, \ldots, 7$.

Στην πραγματικότητα ολόκληρη η παραπάνω διαδικασία εφαρμόζεται χωριστά στα δύο κανάλια στερεοφωνικού ήχου. Η κοινή τελική κατηγοριοποίηση γίνεται σύμφωνα με τη λογική του Πίνακα 1.

2.2 Filterbanks

Η πληροφορία του τύπου κάθε frame διαβιβάζεται στο μηχανισμό Filterbanks που χρησιμοποιεί το μετασχηματισμό Modified Discrete Cosine Transform (MDCT) για να μειώσει τη συσχέτιση των δειγμάτων μεταβαίνοντας ταυτόχρονα από το πεδίο του χρόνου στο πεδίο της συχνότητας. Ο μετασχηματισμός MDCT αποτελεί παραλλαγή του DCT η οποία λειτουργεί σε επικαλυπτόμενα παράθυρα ώστε να αποφεύγει τα προβλήματα που εμφανίζονται στα όρια των παραθύρων. Έτσι, για N δείγματα του παραθύρου i (ολόκληρο frame ή sub-frame), ο MDCT οδηγεί σε $N/2$ συντελεστές, σύμφωνα με τη σχέση:

$$X_{i,k} = 2\sum_{n=0}^{N-1} s_{i,n} \cos\left(\frac{2\pi}{N}(n+n_0)(k+\frac{1}{2})\right),\ 0 \le k < N/2$$

Table 1: Πίνακας απόφασης του τελικού τύπου ενός frame από τις επιμέρους αποφάσεις για καθένα από τα δύο κανάλια στερεοφωνικού ήχου.

Τύπος καναλιού 0	**Τύπος καναλιού 1**	**Κοινός τελικός τύπος**
ONLY_LONG_SEQUENCE	ONLY_LONG_SEQUENCE	ONLY_LONG_SEQUENCE
ONLY_LONG_SEQUENCE	LONG_START_SEQUENCE	LONG_START_SEQUENCE
ONLY_LONG_SEQUENCE	EIGHT_SHORT_SEQUENCE	EIGHT_SHORT_SEQUENCE
ONLY_LONG_SEQUENCE	LONG_STOP_SEQUENCE	LONG_STOP_SEQUENCE
LONG_START_SEQUENCE	ONLY_LONG_SEQUENCE	LONG_START_SEQUENCE
LONG_START_SEQUENCE	LONG_START_SEQUENCE	LONG_START_SEQUENCE
LONG_START_SEQUENCE	EIGHT_SHORT_SEQUENCE	EIGHT_SHORT_SEQUENCE
LONG_START_SEQUENCE	LONG_STOP_SEQUENCE	EIGHT_SHORT_SEQUENCE
EIGHT_SHORT_SEQUENCE	ONLY_LONG_SEQUENCE	EIGHT_SHORT_SEQUENCE
EIGHT_SHORT_SEQUENCE	LONG_START_SEQUENCE	EIGHT_SHORT_SEQUENCE
EIGHT_SHORT_SEQUENCE	EIGHT_SHORT_SEQUENCE	EIGHT_SHORT_SEQUENCE
EIGHT_SHORT_SEQUENCE	LONG_STOP_SEQUENCE	EIGHT_SHORT_SEQUENCE
LONG_STOP_SEQUENCE	ONLY_LONG_SEQUENCE	LONG_STOP_SEQUENCE
LONG_STOP_SEQUENCE	LONG_START_SEQUENCE	EIGHT_SHORT_SEQUENCE
LONG_STOP_SEQUENCE	EIGHT_SHORT_SEQUENCE	EIGHT_SHORT_SEQUENCE
LONG_STOP_SEQUENCE	LONG_STOP_SEQUENCE	LONG_STOP_SEQUENCE

όπου $s_{i,n}$ είναι τα δείγματα του σήματος στο i-οστό παράθυρο, n είναι ο δείκτης δείγματος του σήματος, k ο δείκτης του συντελεστή MDCT, και $n_0 = (N/2+1)/2$.

Ο αντίστροφος μετασχηματισμός δίνεται από τη σχέση

$$s_{i,n} = \frac{2}{N}\sum_{k=0}^{\frac{N}{2}-1} X_{i,k}\cos\left(\frac{2\pi}{N}(n+n_0)\left(k+\frac{1}{2}\right)\right),\ 0 \le n < N$$

Πριν την εφαρμογή του MDCT, καθώς και μετά την εφαρμογή του αντίστροφου, IMDCT, τα δείγματα πολλαπλασιάζονται στο πεδίο του χρόνου με παράθυρο οι συντελεστές του οποίου ικανοποιούν συγκεκριμένες συνθήκες που επιτρέπουν την ανακατασκευή του σήματος στο αποκωδικοποιητή. Δύο είδη συναρτήσεων παραθύρων προδιαγράφονται στο πρότυπο: Τα παράθυρα Kaiser-Bessel-Derived (KBD) και sinusoid (SIN). Τα παράθυρα Kaiser-Bessel-Derived δίνονται από τη σχέση:

$$W_{KBD\_LEFT,N}(n) = \sqrt{\frac{\sum_{i=0}^{n} w(i,a)}{\sum_{i=0}^{N/2} w(i,a)}}, \qquad 0 \le n < \frac{N}{2}$$

$$W_{KBD\_RIGHT,N}(n) = \sqrt{\frac{\sum_{i=0}^{N-n} w(i,a)}{\sum_{i=0}^{N/2} w(i,a)}}, \qquad \frac{N}{2} \le n < N$$

όπου w η συνάρτηση παραθύρου Kaiser-Bessel που ορίζεται ως

$$w(n) = \frac{I_0\left[\pi a\sqrt{1-\left(\frac{n-N/4}{N/4})^2\right)}\right]}{I_0(\pi a)}, 0 \le n \le \frac{N}{2}$$

όπου I_0 η μηδενικής τάξης τροποποιημένη συνάρτηση Bessel του πρώτου είδους. Ο συντελεστής a για τα παράθυρα KBD λαμβάνει τιμή 6 για long windows και 4 για short windows.

Υποδειξη: Για τον υπολογισμό της $w(n)$ στην Python, μπορεί να χρησιμοποιηθεί η συνάρτηση `scipy.signal.windows.kaiser` του πακέτου `scipy` (documentation).

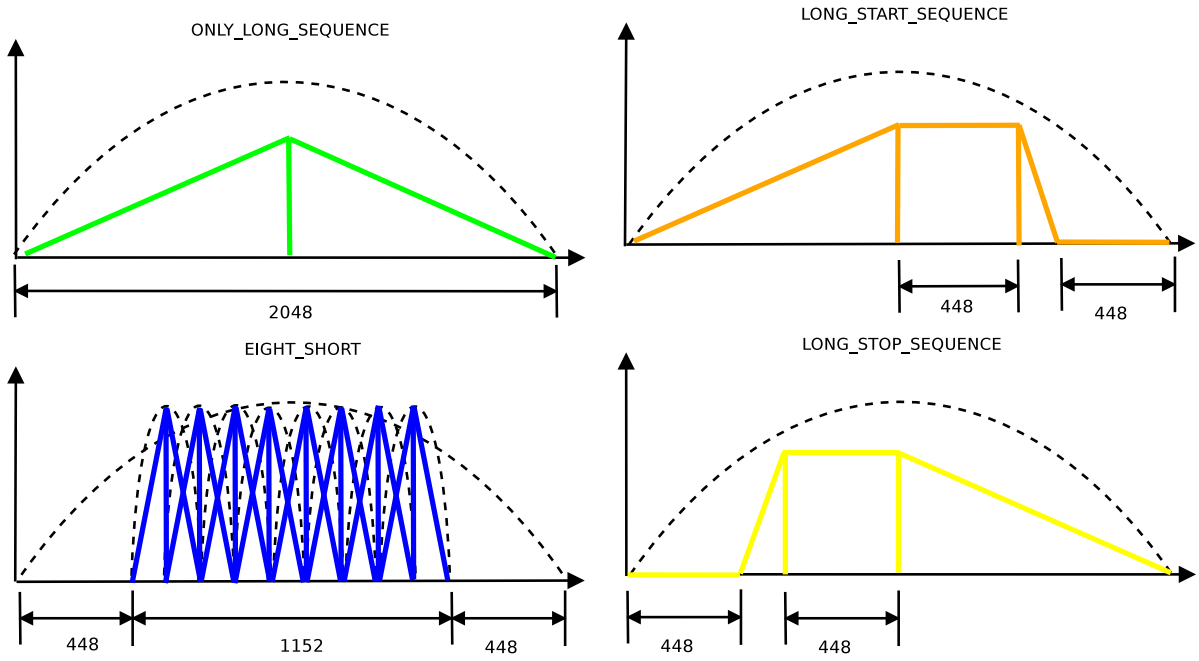

Figure 4: Οι 4 τύποι παραθύρων στον AAC και η τοποθέτησή τους σε ένα frame. Παρατηρήστε ότι στα frames τύπου EIGHT_SHORT_SEQUENCE χρησιμοποιούνται 8 παράθυρα, ένα για κάθε subframe.

Τα παράθυρα τύπου sinusoid δίνονται από τη σχέση:

$$W_{SIN\_LEFT,N}(n) = \sin\left(\frac{\pi}{N}\left(n+\frac{1}{2})\right)\right), \qquad 0 \le n < \frac{N}{2}$$

$$W_{SIN\_RIGHT,N}(n) = \sin\left(\frac{\pi}{N}\left(n+\frac{1}{2})\right)\right), \qquad \frac{N}{2} \le n < N$$

Η διαδικασία παραθύρωσης-μετασχηματισμού διαφοροποιείται ανάλογα με τον τύπο του frame (βλ. και Σχήμα 4).

1. Για τα frames τύπου ONLY LONG SEQUENCE, LONG START SEQUENCE και LONG STOP SEQUENCE χρησιμοποιείται ένα παράθυρο μήκους 2048 σημείων για ολόκληρο το frame και στη συνέχεια MDCT ίδιου μήκους. Σύμφωνα με τον ορισμό του MDCT το αποτέλεσμα είναι ένα διάνυσμα 1024 συντελεστών.

2. Αντίθετα για τα frames τύπου EIGHT SHORT SEQUENCE από τα 2048 δείγματα του frame επιλέγονται μόνο τα 1152 κεντρικά δείγματα και αγνοούνται τελείως τα 2x448 δείγματα που βρίσκονται εκατέρωθεν της κεντρικής περιοχής. Στη συνέχεια η κεντρική περιοχή χωρίζεται σε 8 επικαλυπτόμενα κατά 50% υπο-τμήματα (subframes) μήκους 256 δειγμάτων το καθένα. Η διαδικασία παραθύρωσης - μετασχηματισμού εφαρμόζεται σε καθένα από αυτά τα subframes χρησιμοποιώντας αντίστοιχου μήκους παράθυρα MDCT.

3. Για τα frames τύπου ONLY LONG SEQUENCE και τα subframes των frames τύπου EIGHT SHORT SEQUENCE χρησιμοποιούνται συμμετρικά παράθυρα αντίστοιχου μήκους, W_l και W_s αντίστοιχα. Το πρότυπο επιτρέπει και μη-συμμετρικά παράθυρα με την έννοια ότι πχ το αριστερό τους μέρος είναι KBD και το δεξί SIN αλλά στην απλοποιημένη μας εκδοχή θα το αποφύγουμε. Γενικότερα θα υποθέσουμε ότι καθόλη την κωδικοποίηση χρησιμοποιούμε είτε μόνο KBD είτε μόνο SIN.

4. Για τα frames τύπου LONG START SEQUENCE χρησιμοποιούνται μη συμμετρικά παράθυρα που κατασκευάζονται από την παράθεση (α) μισού αριστερού W_l (1024 συντελεστές), (β) 448 συντελεστών ίσων με τη μονάδα, (γ) μισού δεξιού W_s (128 συντελεστές) και τέλος (δ) 448 συντελεστών ίσων με το μηδέν.

5. Για τα frames τύπου LONG STOP SEQUENCE χρησιμοποιούνται μη συμμετρικά παράθυρα που κατασκευάζονται από την παράθεση (α) 448 συντελεστών ίσων με το μηδέν, (β) μισού αριστερού W_s (128 συντελεστές) (γ) 448 συντελεστών ίσων με τη μονάδα και τέλος μισού δεξιού W_l (1024 συντελεστές).

Η διαδικασία εφαρμογής των παραθύρων περιγράφεται αναλυτικά στις σελίδες 127-132 του μέρους ANNEX A του προτύπου (αρχείο `w2203tfa`). Φροντίστε ωστόσο να αγνοήσετε τις περιπτώσεις όπου έχουμε εναλλαγές παραθύρων (από KBD σε SIN και το αντίστροφο), όπως αναφέρθηκε παραπάνω. Τονίζουμε ότι εφαρμόζεται το ίδιο παράθυρο στο πεδίο του χρόνου τόσο πριν την εφαρμογή του MDCT όσο και μετά την εφαρμογή του IMDCT.

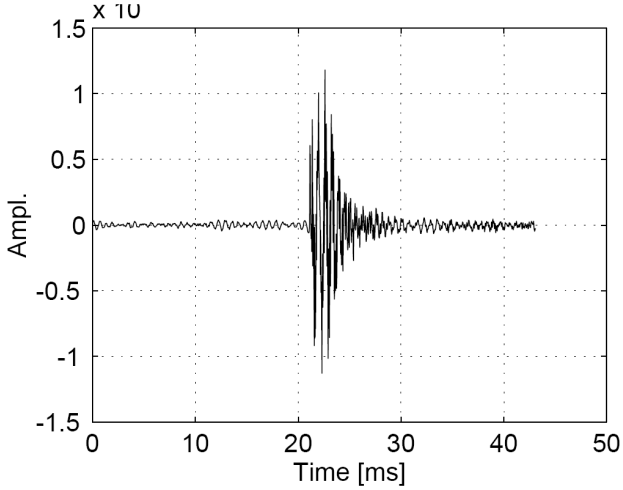

Figure 5: Παράδειγμα σήματος που παράγεται από καστανιέτες μέσα σε ένα παράθυρο. Παρατηρείτε την απότομη μετάβαση του σήματος στη μέση του παραθύρου. Εικόνα από το [1].

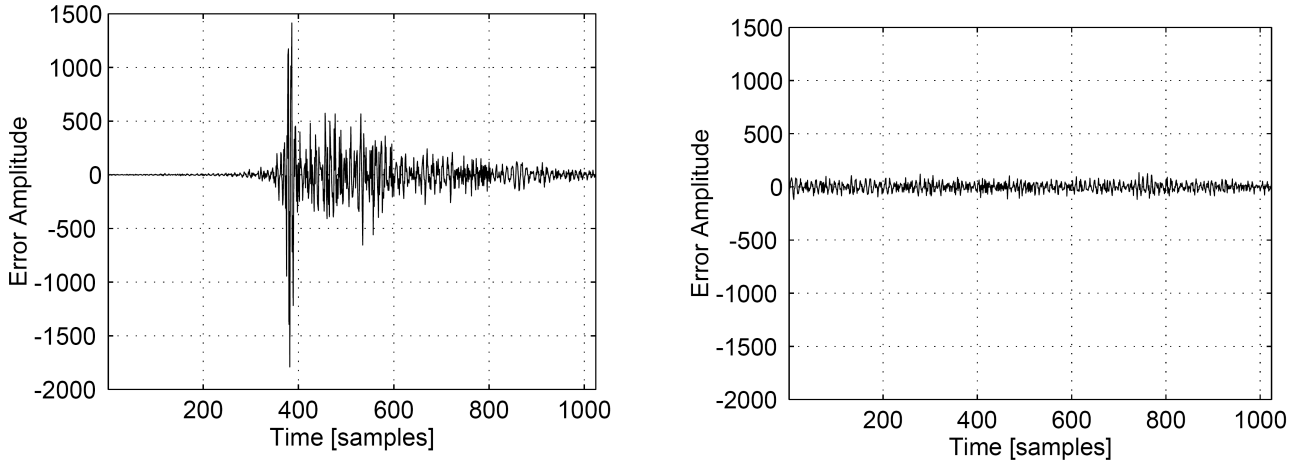

Figure 6: (a) Το σήμα σφάλματος στο χρόνο μετά από κωδικοποίηση μετασχηματισμού, κβαντισμό και αποκωδιοποίηση. Παρατηρείτε την αύξηση του σφάλματος στο μέσο του παραθύρου. (b) Το ίδιο σήμα σφάλματος με χρήση TNS. Εικόνα από το [1].

2.3 Temporal Noise Shaping (TNS)

Η αποτελεσματική εφαρμογή του ψυχοακουστικού μοντέλου στον AAC (βλ. Ενότητα 2.4) στηρίζεται στην παραδοχή ότι το παρατηρούμενο σήμα είναι στάσιμο κατά τη διάρκεια του παραθύρου εφαρμογής. Αυτή η παραδοχή συνεπάγεται ότι το σφάλμα κβαντισμού των συντελεστών MDCT κατανέμεται ομοιόμορφα στη διάρκεια του παραθύρου στο πεδίο του χρόνου. Αυτό δεν είναι πρόβλημα όταν τα στατιστικά του σήματος παραμένουν σταθερά κατά τη διάρκεια του παραθύρου, αλλά αποτελεί πρόβλημα όταν έχουμε "μεταβατικά" παράθυρα, στη διάρκεια των οποίων εμφανίζεται κάποιος έντονος ήχος (πχ ένας παλμός), ή κάποιος περιοδικός ήχος τύπου pitch φωνής. Σ' αυτή την περίπτωση το σφάλμα κβαντισμού θα οδηγήσει σε αντιληπτές παραμορφώσεις γύρω από τη χρονική περίοδο που εμφανίζεται ο παλμός.

Στη βιβλιογραφία προτείνεται μία λύση σ' αυτό το πρόβλημα βασισμένη στη δυαδικότητα χρόνου / συχνότητας: Είναι γνωστό ότι σήματα με μη-ομοιόμορφο φασματικό περιεχόμενο μπορούν να κωδικοποιηθούν αποτελεσματικά είτε απευθείας με μετασχηματισμό στο πεδίο της συχνότητας, ή με γραμμική πρόβλεψη στο πεδίο του χρόνου. Κατά συμμετρικό τρόπο, σήματα που είναι μη-ομοιόμορφα στο πεδίο του χρόνου μπορούν να κωδικοποιηθούν αποτελεσματικά είτε με απευθείας κωδικοποίηση των τιμών στο πεδίο του χρόνου, είτε εφαρμόζοντας γραμμική πρόβλεψη *στους συντελεστές στο πεδίο της συχνότητας*. Με βάση αυτές τις παρατηρήσεις η κεντρική ιδέα του Temporal Noise Shaping είναι η εφαρμογή ενός μοντέλου γραμμικής πρόβλεψης στους συντελεστές MDCT ώστε το σφάλμα κβαντισμού να *προσαρμοστεί στο σχήμα του σήματος στο πεδίο του χρόνου*. Το παράδειγμα της Εικόνας 5 επιδεικνύει αυτή την ιδέα ("δανεισμένο" από το [1], όπου μπορείτε να βρείτε περισσότερες πληροφορίες σχετικά με το TNS).

Ο μηχανισμός Temporal Noise Shaping (TNS) μετασχηματίζει τους συντελεστές MDCT σε ένα νέο σύνολο ισάριθμων

συντελεστών στο οποίο έχουν απαλειφθεί οι περιοδικότητες. Θεωρούμε τις μπάντες του ψυχοακουστικού μοντέλου που ορίζονται στους πίνακες Table B.2.1.9.a και Table B.2.1.9.b των σελίδων 117-119 του προτύπου (αρχείο `w2203tfa`). Κάθε μία από τις μπάντες αυτές αντιστοιχεί περίπου σε 1/3 ενός Bark. Στα παρακάτω, b_j είναι ο αύξων αριθμός του πρώτου συντελεστή της υπαριθμόν j μπάντας (στήλη `w_low` του πίνακα). Η βαθμίδα TNS εφαρμόζει την ακόλουθη διαδικασία:

1. Κανονικοποιεί τους συντελεστές MDCT $X(k)$ ως προς την ενέργεια της μπάντας στην οποία ανήκουν:

$$X_w(k) = \frac{X(k)}{S_w(k)} \tag{2}$$

Ο συντελεστής κανονικοποίησης $S_w(k)$ υπολογίζεται από την ενέργεια $P(j)$ της κάθε μπάντας j,

$$P(j) = \sum_{k=b_j}^{b_{j+1}-1} X(k)^2,\ j = 0, \ldots, N_B - 1 \tag{3}$$

Όπου N_B ο αριθμός από μπάντες. Έτσι,

$$S_w(k) = \sqrt{P(j)},\ b_j \leq k < b_{j+1} \tag{4}$$

Οι συντελεστές $S_w(k)$ που προκύπτουν με αυτό τον τρόπο έχουν σταθερή τιμή στο εσωτερικό κάθε μπάντας και αλλάζουν απότομα όταν μετακινούμαστε ανάμεσα σε μπάντες. Πριν λοιπόν εφαρμοστούν στην Εξ. (2) εξομαλύνονται ως ακολούθως:

for $k = 1022 : -1 : 0$, $S_w(k) = ((S_w(k) + S_w(k+1))/2$
for $k = 1 : +1 : 1023$, $S_w(k) = ((S_w(k) + S_w(k-1))/2$

2. Υπολογίζει τους συντελεστές γραμμικής πρόβλεψης $a_1, \ldots, a_p$ για κάθε frame (ή subframe, αν πρόκειται για EIGHT SHORT SEQUENCE frame). Στον απλοποιημένο κωδικοποιητή θα χρησιμοποιήσουμε φίλτρα γραμμικής πρόβλεψης σταθερής τάξης $p = 4$. Οι συντελεστές υπολογίζονται έτσι ώστε να ελαχιστοποιείται το τετράγωνο του σφάλματος πρόβλεψης:

$$e^2(k) = \left(X_w(k) - \sum_{l=1}^{p} a_l X_w(k-l) \right)^2 \tag{5}$$

Κατά τα γνωστά, αν $\mathbf{a} = [a_1, \ldots, a_p]^T$, οι βέλτιστοι συντελεστές είναι η λύση των κανονικών εξισώσεων

$$\mathbf{Ra} = \mathbf{r} \tag{6}$$

όπου $\mathbf{R}$, $\mathbf{a}$ το διάνυσμα των συντελεστών a_l και $\mathbf{r}$ ο $p \times p$ πίνακας και το $p \times 1$ διάνυσμα αυτοσυσχέτισης της ακολουθίας $X_w(k)$,

$$\mathbf{R} = \begin{bmatrix} r(0) & r(-1) & \ldots & r(-p+1) \\ r(1) & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ r(p-1) & \ldots & \ldots & r(0) \end{bmatrix} \qquad \mathbf{a} = \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} \qquad \mathbf{r} = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(p) \end{bmatrix} \tag{7}$$

3. Κβαντίζει τους παραπάνω συντελεστές με 4 bits χρησιμοποιώντας ομοιόμορφο συμμετρικό κβαντιστή βήματος 0.1.

4. Εφαρμόζει το FIR φίλτρο

$$H_{TNS}(z) = 1 - a_1 z^{-1} - a_2 z^{-2} - \ldots - a_p z^{-p} \tag{8}$$

στην αρχική ακολουθία συντελεστών $X(k)$. Οι συντελεστές a_i που χρησιμοποιούνται εδώ είναι κβαντισμένοι ώστε η διαδικασία να είναι αντιστρέψιμη. Κατά τον υπολογισμό τους, για την εργασία, κάντε έναν προληπτικό έλεγχο ευστάθειας του αντίστροφου φίλτρου H_{TNS}^{-1} που θα χρησιμοποιηθεί στην αποκωδικοποίηση. Φροντίστε οι συντελεστές να οδηγούν σε ευσταθές H_{TNS}^{-1}.

Υπόδειξη: Υπενθυμίζεται ότι ένα γραμμικό, χρονικά ανεξάρτητο φίλτρο είναι αιτιατό και ευσταθές όταν όλοι οι πόλοι του φίλτρου βρίσκονται εντός του μοναδιαίου δίσκου $|z| < 1$. Για το σκοπό αυτό, στην υλοποίηση σε Python μπορεί να χρησιμοποιηθεί κατάλληλα η συνάρτηση `numpy.polynomial.polynomial.Polynomial(...).roots()` (documentation).

Η αντιστροφή της διαδικασίας επιτυγχάνεται στην αποκωδικοποίηση με την εφαρμογή του αντίστροφου φίλτρου.

2.4 Psychoacoustic Model

Για τη μείωση των απαιτούμενων bit κατά την κβάντιση, χρησιμοποιείται το ανθρώπινο ψυχοακουστικό μοντέλο. Έτσι υπολογίζονται κατώφλια ακουστότητας για κάθε συχνότητα, που στην ουσία αποτελούν το επιτρεπόμενο περιθώριο σφάλματος του κβαντιστή. Οποιοσδήποτε ήχος βρίσκεται κάτω από το κατώφλι ακουστότητας στη συγκεκριμένη συχνότητα δεν κωδικοποιείται.

Για να το πετύχει αυτό το ψυχοακουστικό μοντέλο υπολογίζει πως μεταβάλλεται το κατώφλι ακουστότητας σε ησυχία χρησιμοποιώντας μία συνάρτηση κατανομής της μάσκας (spreading function) και λαμβάνοντας υπόψη αν οι maskers αντιστοιχούν σε τόνους (tone) ή σε θόρυβο στενού εύρους ζώνης (narrowband noise). Περισσότερες λεπτομέρειες σχετικά με τις αρχές λειτουργίας των ψυχοακουστικών μοντέλων μπορείτε να βρείτε στο [2].

Συγκεκριμένα, τα βήματα που ακολουθεί το πρότυπο περιγράφονται στις σελίδες 95-101 του αρχείου `w2203tfa` και επίσης χρησιμοποιούν τις μπάντες που περιγράφονται στους πίνακες Table B.2.1.9.a και Table B.2.1.9.b των σελίδων 117-119 του προτύπου. Για την υλοποίηση της εργασίας μας ενδιαφέρουν μόνο οι τιμές για συχνότητα δειγματοληψίας 48kHz. Οι τιμές αυτών των πινάκων σας δίνονται στο αρχείο `TableB219.mat`.

Υπόδειξη: μπορείτε στην Python να διαβάσετε το περιχόμενο αυτού του αρχείου χρησιμοποιώντας τη συνάρτηση `scipy.io.loadmat()`:

```
from scipy.io import loadmat

mat_data = loadmat("TableB219.mat")

b219a_array = mat_data['B219a']
b219b_array = mat_data['B219b']
```

Η βαθμίδα επεξεργασίας αυτή αντιστοιχεί σε παράθυρα 2048 δειγμάτων (frame) ή σε υπο-παράθυρα 256 δειγμάτων (sub-frame) αναλόγως με το αν το frame είναι EIGHT SHORT ή όχι. Στα πρώτα αντιστοιχεί ο Πίνακας B.2.1.9.a και στα δεύτερα ο Πίνακας B.2.1.9.b του προτύπου.

Για τον απλουστευμένο κωδικοποιητή μας τα βήματα που ακολουθούνται για τον υπολογισμό του ψυχοακουστικού μοντέλου είναι τα ακόλουθα:

1. Προϋπολογίζεται η spreading function ως εξής:

Υπολογισμός spreading function. i και j είναι οι τιμές του index της μπάντας που διασπείρει και της μπάντας στην οποία γίνεται η διασπορά, αντίστοιχα (πρώτη στήλη του πίνακα). Η έξοδος της συνάρτησης αυτής μπορεί να αποθηκευτεί για όλους τους συνδυασμούς από μπάντες που μας ενδιαφέρει και να χρησιμοποιείται στα επόμενα ως πίνακας (ένας πίνακας για τα long και ένας για τα short).

procedure x = SPREADINGFUN(i, j)
  if $i \geq j$ **then** ▷ bval(k) είναι η κεντρική συχνότητα της μπάντας με δείκτη k (στήλη bval του πίνακα)
    $tmpx = 3.0(\text{bval}(j) - \text{bval}(i))$
  else
    $tmpx = 1.5(\text{bval}(j) - \text{bval}(i))$
  end if
  $tmpz = 8 \min\left((tmpx - 0.5)^2 - 2(tmpx - 0.5), 0\right)$
  $tmpy = 15.811389 + 7.5(tmpx + 0.474) - 17.5(1.0 + (tmpx + 0.474)^2)^{0.5}$
  if then$(tmpy < -100)$
    $x = 0$
  else
    $x = 10^{\frac{tmpz+tmpy}{10}}$
  end if
end procedure

---

2. Κάθε frame ή sub-frame πολλαπλασιάζεται με ένα παράθυρο Hann στο πεδίο του χρόνου:

$$s_w(n) = s(n)\left(0.5 - 0.5\cos\left(\frac{\pi(n+0.5)}{N}\right)\right)$$

όπου $s(n)$ το σήμα στο πεδίο του χρόνου και N το μήκος του παραθύρου (frame ή sub-frame).

Έπειτα υπολογίζεται ο FFT του s_w και υπολογίζονται το μέτρο $r(w)$ και η φάση $f(w)$ του μετασχηματισμού σε κάθε διακριτή συχνότητα w. Πρακτικά ενδιαφερόμαστε για δείκτες συχνότητας από 0 έως 1023 (στους υπόλοιπους ο FFT έχει συμμετρικό πλάτος).

Η ίδια ακριβώς διαδικασία συμβαίνει και για κάθε subframe 256 δειγμάτων (απ' όπου διατηρούμε 128 συντελεστές) του κάθε παραθύρου (που αντιστοιχούν στα subframes όπως έχουν περιγραφεί για frames τύπου EIGHT SHORT SEQUENCE).

Κάθε στιγμή απαιτείται να συντηρούμε στη μνήμη το τωρινό παράθυρο και τα δύο προηγούμενά του, για να μπορούμε να υλοποιήσουμε το επόμενο βήμα (τόσο για τα frames όσο και για τα subframes). Στα επόμενα, όταν αναφερόμαστε σε "παράθυρο" εννοούμε τόσο τα frames των 2048 δειγμάτων όσο και τα sub-frames των 256 δειγμάτων.

3. Υπολογίζουμε προβλέψεις των $r(w)$ και $f(w)$ για κάθε παράθυρο ως εξής:

$$r_{\text{pred}}(w) = 2r_{-1}(w) - r_{-2}(w)$$
$$f_{\text{pred}}(w) = 2f_{-1}(w) - f_{-2}(w)$$

όπου $w = 0, \ldots, N/2$, οι δείκτες των συντελεστών του FFT (lines) και r_{-1} και r_{-2} οι αντίστοιχες τιμές του προηγούμενου και του προ-προηγούμενου παραθύρου υπολογισμού threshold αντίστοιχα. Για την επεξεργασία των frames μήκους $N = 2048$ χρειαζόμαστε τα 2 προηγούμενα frames επίσης μήκους N. Για την επεξεργασία των subframes μήκους $N = 256$ αντιστοίχως χρειαζόμαστε τα ακριβώς δύο προηγούμενα subframes. Έτσι, για το 1ο subframe χρειαζόμαστε το 7ο και 8ο subframe του προηγούμενου frame κλπ. Το ίδιο ακριβώς ισχύει και για τη φάση f.

4. Χρησιμοποιούμε τις προβλέψεις $r_{\text{pred}}(w)$ και $f_{\text{pred}}(w)$ για να υπολογίσουμε ένα μέτρο της "προβλεψιμότητας" (predictability) του καθε frame και subframe.

$$c(w) = \frac{\sqrt{\left(r(w)\cos(f(w)) - r_{pred}(w)\cos(f_{pred}(w))\right)^2 + \left(r(w)\sin(f(w)) - r_{pred}(w)\sin(f_{pred}(w))\right)^2}}{r(w) + |r_{pred}(w)|}$$

5. Υπολογίζουμε την ενέργεια και την (βεβαρυμένη) προβλεψιμότητα για όλες τις συχνότητες σε κάθε μπάντα με δείκτη b των Πινάκων.

$$e(b) = \sum_{w=w_{\text{low}}(b)}^{w_{\text{high}}(b)} r(w)^2$$
$$c(b) = \sum_{w=w_{\text{low}}(b)}^{w_{\text{high}}(b)} c(w)r(w)^2$$

Οι μπάντες περιγράφονται στους πίνακες Table B.2.1.9.a και Table B.2.1.9.b όπως αναφέρθηκε παραπάνω και αντιστοιχούν περίπου σε 1/3 των critical bands του ψυχοακουστικού μοντέλου. Στις εξισώσεις οι τιμές w_{low} και w_{high} αντιστοιχούν στις στήλες wlow και whigh των πινάκων. Οι δείκτες του FFT τόσο στο πρότυπο, όσο και στην παρούσα περιγραφή, ξεκινούν από το μηδέν.

6. Συνδυάζουμε την ενέργεια και την προβλεψιμότητα με τη spreading function

$$ecb(b) = \sum_{bb=0}^{N_B-1} e(bb)\text{spreading\_function}(bb, b)$$
$$ct(b) = \sum_{bb=0}^{N_B-1} c(bb)\text{spreading\_function}(bb, b)$$

όπου N_B ο αριθμός από μπάντες (δηλ το bb διατρέχει την πρώτη στήλη του πίνακα). Οι δύο αυτές τιμές κανονικοποιούνται ως εξής:

$$cb(b) = \frac{ct(b)}{ecb(b)}$$
$$en(b) = \frac{ecb(b)}{\sum_{bb=0}^{N_B-1} \text{spreading\_function}(bb, b)}$$

7. Από την τιμή $cb(b)$ μπορούμε να υπολογίσουμε τον "δείκτη τονικότητας" (tonality index) $tb(b)$ κάθε μπάντας

$$tb(b) = -0.299 - 0.43\ln(cb(b))$$

Το $tb(b)$ λαμβάνει τιμές στο διάστημα $(0, 1)$.

8. Υπολογίζουμε το απαιτούμενο SNR κάθε μπάντας. Θεωρούμε την τιμή "Noise Masking Tone", $NMT(b) = 6dB$ και την τιμη "Tone Masking Noise", $TMN(b) = 18dB$ για όλα τα b. Με βάση αυτές τις τιμές,

$$SNR(b) = tb(b)TMN(b) + (1 - tb(b))NMT(b)$$

Επομένως παρατηρούμε ότι όταν έχουμε υψηλό tonality index ($tb(b) \to 1$), τότε $SNR(b) \to TMN(b) = 6$, δηλαδή εύκολα θα ακούσουμε έναν ζωνοπερατό θόρυβο που καλύπτεται από τόνους. Στην αντίθετη περίπτωση $SNR(b) \to NMT(b) = 18$ και θα χρειαστεί πολύ υψηλότερο SNR για να ακουστεί ένας τόνος που καλύπτεται από θόρυβο. Οι παρατηρήσεις αυτές είναι συμβατές με τη θεωρία των ψυχοακουστικών μοντέλων, όπου όταν ο masker είναι θόρυβος έχουμε πολύ υψηλότερες απαιτήσεις σε SNR ώστε να μην έχουμε masking.

9. Μετατρέπουμε από dB σε λόγο ενέργειας

$$bc(b) = 10^{-\frac{SNR(b)}{10}}$$

10. Υπολογίζουμε το κατώφλι ενέργειας

$$nb(b) = en(b)bc(b)$$

11. Σ' αυτό το σημείο το πρότυπο περνά στις μπάντες που χρησιμοποιούνται για τον κβαντισμό, οι οποίες ονομάζονται scalefactor bands και δίνονται από τους πίνακες Table 2.3 (για long frames) και Table 2.4 για EIGHT SHORT SEQUENCE frames, στη σελίδα 67 του αρχείου `w2203tfs`. Πρακτικά οι scalefactor bands είναι λιγότερο αναλυτικές (κάθε μία αντιστοιχεί περίπου σε ένα critical band) έτσι ώστε να ελαχιστοποιηθεί η μετάδοση των scalefactors.

    Ωστόσο για λόγους απλότητας στη συγκεκριμένη υλοποίηση θεωρούμε ότι οι scalefactor bands ταυτίζονται με τις μπάντες του ψυχοακουστικού μοντέλου, δηλαδή είναι οι μπάντες των πινάκων Table B.2.1.9.a και B.2.1.9.b.

    Υπολογίζουμε το επίπεδο θορύβου ως:

$$npart(b) = \max\{nb(b), \hat{q}_{\text{thr}}(b)\} \tag{9}$$

    όπου $\hat{q}_{\text{thr}}$ προκύπτει από τη στήλη qsthr των πινάκων και αντιστοιχεί στο κατώφλι σε ησυχία. Για τον υπολογισμό του $\hat{q}_{\text{thr}}$ χρησιμοποιήστε τον ακόλουθο τύπο (μετατροπή σε ενέργεια, δείτε `w2203tfa` σελ. 99, βήμα 11):

$$\hat{q}_{\text{thr}} = \epsilon \frac{N}{2} 10^{\frac{\text{qsthr}}{10}} \tag{10}$$

    όπου το ϵ στην Python δίνεται από την εντολή `numpy.finfo('float').eps` και N είναι το μήκος του FFT (2048 για long και 256 για short). Επομένως με βάση την Εξ. (9) σε κάθε θέση το κατώφλι ακουστότητας του θορύβου κβαντισμού είναι το μέγιστο του κατωφλίου σε ησυχία και του κατωφλίου λόγω masking.

12. Υπολογίζουμε το Signal to Mask Ratio (SMR) της μπάντας με δείκτη b:

$$SMR(b) = \frac{e(b)}{npart(b)}$$

13. Τέλος υπολογίζουμε τα κατώφλια $T(b)$ ακουστότητας του κωδικοποιητή. Για να το πετύχουμε αυτό υπολογίζουμε την ενέργεια των συντελεστών MDCT $X(k)$ σε κάθε scalefactor band

$$P(b) = \sum_{k=w_{\text{low}}(b)}^{w_{\text{high}}(b)} X(k)^2$$

    και

$$T(b) = \frac{P(b)}{SMR(b)}$$

    Πλέον έχουμε όλη την απαραίτητη πληροφορία για τον κβαντιστή.

2.5 Mid/Side stereo

Η βαθμίδα αυτή μετατρέπει το δικαναλικό σήμα από στέρεο σε Mid/Side

$$M = \frac{L + R}{2},\ S = \frac{L - R}{2} \tag{11}$$

και χρησιμοποιεί διαφορετικά κατώφλια για κάθε διαφορετική κωδικοποίηση (Left/Right ή Mid/Side) και επιλέγεται αυτή με τα λιγότερα bits.

Για λόγους απλούστευσης η βαθμίδα αυτή και η αντίστροφή της ΔΕΝ θα υλοποιηθούν.

2.6 Quantization

Ο κβαντιστής παραλαμβάνει τους συντελεστές της βαθμίδας TNS και τα κατώφλια $T(b)$ της κάθε scalefactor band b και παράγει σύμβολα που στη συνέχεια θα κωδικοποιηθούν από τη βαθμίδα Huffman.

Αναγκαστικά οι παράμετροι κβαντισμού κωδικοποιούνται και ενσωματώνονται στο παραγόμενο bitstream και για το λόγο αυτό είναι απολύτως αναγκαίο οι παράμετροι να μην αφορούν μεμονωμένους συντελεστές αλλά ομάδες συντελεστών. Έτσι ο κβαντιστής προσαρμόζει τα επίπεδα κβαντισμού του ανάλογα με τα χαρακτηριστικά της κάθε μπάντας (τα Scale Factor Bands που αναφέραμε παραπάνω) και εφαρμόζει τις ίδιες παραμέτρους κβάντισης για όλους τους συντελεστές εντός της μπάντας.

Η ποιότητα του κβαντιστή (που έμμεσα αντιστοιχεί στον αριθμό bits που θα παραχθούν τελικά) επηρεάζεται από δύο παράγοντες: (α) Το κατώφλι ακουστότητας $T(b)$ του που έχει υπολογίσει το ψυχοακουστικό μοντέλο και (β) Το διαθέσιμο αριθμό bits προκειμένου συνολικά ο κωδικοποιητής να παράγει σταθερό μέσο bitrate.

Ως προς τις τιμές $X(k)$ των συντελεστών MDCT ο κβαντιστής είναι μη ομοιόμορφος και υλοποιείται μέσω της διαδικασίας ακέραιας στρογγυλοποίησης

$$S(k) = \text{sgn}(X(k))\text{int}\left[(|X(k)| \times 2^{-\frac{1}{4}\alpha})^{\frac{3}{4}} + \text{MagicNumber}\right] \tag{12}$$

όπου η σταθερά Magic Number = 0.4054 έχει επιλεγεί πειραματικά και ο συντελεστής α (scale factor gain) ρυθμίζει την ποιότητα του κβαντιστή. Όσο η τιμή του α μεγαλώνει τόσο πιο χονδροειδής γίνεται ο κβαντισμός και συνεπώς τόσο περισσότερο αυξάνει το προκαλούμενο σφάλμα κβαντισμού. Αντίθετα, εύκολα μπορεί να διαπιστώσει κανείς ότι όσο μικραίνει το α το σφάλμα κβαντισμού τείνει στο 0 (πχ διαισθητικά υποθέστε ότι παραλείπουμε τη διαδικασία στρογγυλοποίησης και θεωρούμε ότι η σταθερά MagicNumber είναι πολύ μικρή σε σχέση με τον κυρίως όρο). Ακριβώς η επιλογή αυτού του συντελεστή εξαρτάται από το κατώφλι ακουστότητας και το διαθέσιμο αρθιμό bits όπως προαναφέρθηκε. Η διαδικασία επιλογής του θα περιγραφεί στη συνέχεια. Σε κάθε περίπτωση, ο αποκβαντιστής υλοποιείται από τη σχέση

$$\hat{X}(k) = \text{sgn}(S(k))|S(k)|^{\frac{4}{3}} \times 2^{\frac{1}{4}\alpha} \tag{13}$$

Η βέλτιστη τιμή του Scalefactor Gain α υπολογίζεται με βάση τα κατώφλια ακουστότητας που έχουν υπολογιστεί στη βαθμίδα Psychoacoustic Model:

1. Ως πρώτη προσέγγιση χρησιμοποιείται η τιμή

$$\hat{\alpha}(b) = 16/3 * \log_2\left(\frac{\max_k(X(k))^{\frac{3}{4}}}{MQ}\right) \tag{14}$$

   για όλα τα b (το μέγιστο αφορά όλους τους συντελεστές MDCT, όχι μόνο τους συντελεστές της μπάντας b). Η παράμετρος MQ αντιστοιχεί στον μέγιστο αριθμό επιπέδων κβαντισμού ($2MQ + 1$). Βάσει προτύπου θέτουμε $MQ = 8191$.

2. Η πρώτη προσέγγιση θεωρείται μία πολύ καλής ποιότητας τιμή. Έπειτα, επαναληπτικά αυξάνουμε (κατά μία μονάδα κάθε φορά) την τιμή του scalefactor gain και ελέγχουμε την ισχύ του σφάλματος κβαντισμού

$$P_e(b) = \sum_{k=w_{\text{low}}(b)}^{w_{\text{high}}(b)} (X(k) - \hat{X}(k))^2$$

   Αν η ισχύς είναι κάτω από το κατώφλι ακουστότητας, τότε αυξάνουμε τον αντίστοιχο scale factor $\alpha(b)$ κατά ένα. Η διαδικασια επαναλαμβάνεται έως ότου φτάσουμε το κατώφλι $T(b)$. Επιπλέον, σταματάμε τη διαδικασία αύξησης του $\alpha(b)$ αν $\max_b(|\alpha(b+1) - \alpha(b)|) > 60$ (επομένως η μέγιστη διαφορά διαδοχικών scalefactors πρέπει να είναι μέχρι 60).

Μετά την οριστική επιλογή του Scalefactor Gain, οι συντελεστές Scalefactor Band πρέπει να κωδικοποιηθούν. Αντί για αυτούς κωδικοποιούνται οι ποσότητες:

1. Global gain του frame: $G = \alpha(0)$.
2. Scale factor της μπάντας b του frame, $\alpha(b)$, για $b > 0$. Μάλιστα για τα Scale Factors ενός frame χρησιμοποιείται DPCM μεταξύ των συντελεστών κάθε μπάντας. Δηλαδή κωδικοποιούνται τα

$$\{sfc(1), \ldots, sfc(b), \ldots\} \tag{15}$$

   όπου $sfc(b) = \Delta\alpha(b) = \alpha(b) - \alpha(b-1)$. Σημειώνεται ότι $sfc(0) = \alpha(0) = G$.

2.7 Κωδικοποίηση Huffman

Η βαθμίδα αυτή παραλαμβάνει τα σύμβολα S_k των κβαντισμένων συντελεστών MDCT και $sfc(b)$ των κβαντισμένων scalefactors κάθε μπάντας. Η κωδικοποίηση Huffman υλοποιείται όπως ακριβώς περιγράφεται στο πρότυπο.

2.8 Bit reservoir

Ο ρόλος της βαθμίδας αυτής είναι να παρακολουθεί τα "καταναλισκόμενα" bits με στόχο τη διατήρηση ενός σχεδόν σταθερού μέσου bitrate. Η βαθμίδα αυτή έχει τη δικαιοδοσία να τροποποιεί τα υποτιθέμενα κατώφλια ακουστότητας που αντιλαμβάνεται ο κβαντιστής. Αυξάνοντας τα κατώφλια κατ' αυτόν τον τρόπο στο σήμα προστίθεται και "ακουστός" θόρυβος προς χάριν της συμπίεσης.

Για λόγους απλούστευσης η βαθμίδα αυτή ΔΕΝ θα υλοποιηθεί.

Τούτο θα έχει ως αποτέλεσμα να έχουμε μία διαδικασία κωδικοποίησης σταθερής ποιότητας αλλά μεταβλητού και μάλιστα ανεξέλεγκτου bitrate.

3 Διάρθρωση και Παραδοτέα

Η εργασία θα περιλαμβάνει τις συναρτήσεις που περιγράφονται στη συνέχεια, υλοποιημένες σε Python.

Κάθε συνάρτηση θα πρέπει να συνοδεύεται από την αντίστροφή της που ανακατασκευάζει την είσοδο της πρώτης και που θα αποτελέσει μέρος του αποκωδικοποιητή.

Οι συναρτήσεις που αντιστοιχούν σε κάθε ένα από τα τρία επίπεδα ολοκλήρωσης (βλ. παρακάτω) θα πρέπει να τοποθετηθούν σε αντίστοιχα folders με ονόματα `level_1/, level_2/, level_3/` αντίστοιχα. Προσοχή, κάθε folder πρέπει να περιέχει όλες τις συναρτήσεις που χρειάζονται για το αντίστοιχο επίπεδο ακόμη και αν αυτό επιβάλει την επανάληψη κάποιων από αυτές.

Η συλλογή προγραμμάτων που θα παραδοθεί θα συνοδεύεται υποχρεωτικά και από γραπτή αναφορά η οποία θα περιγράφει τον τρόπο χρήσης των προγραμμάτων και θα επιδεικνύει ενδεικτικά αποτελέσματα.

Η υποβολή σας θα αποτελείται από ένα αρχείο zip, με όνομα `AEM1_AEM2.zip` που θα περιλαμβάνει όλα τα .py, .mat αρχεία (στην παραπάνω δομή) που είναι απαραίτητα για να εκτελεστεί ο κώδικας, καθώς και το αρχείο report.pdf της αναφοράς.

Για τον έλεγχο του κωδικοποιητή / αποκωδικοποιητή που θα κατασκευαστεί θα χρησιμοποιηθεί ενδεικτικό αρχείο ασυμπίεστου ήχου σε μορφή wav.

Υπόδειξη: για λειτουργικότητα που αφορά την ανάγνωση/εγγραφή αρχείων ήχου, μπορεί να χρησιμοποιηθεί το πακέτο `soundfile` της Python.

3.1 1ο Επίπεδο (1 μονάδα)

```
frame_type = SSC(frame_T, next_frame_T, prev_frame_type)
```

Υλοποιεί τη βαθμίδα Sequence Segmentation Control.

`frame_type`: Ο τύπος του frame που επιλέγεται:

- "`OLS`": για ONLY_LONG_SEQUENCE
- "`LSS`": για LONG_START_SEQUENCE
- "`ESH`": για EIGHT_SHORT_SEQUENCE
- "`LPS`": για LONG_STOP_SEQUENCE

`frame_T`: Το frame i στο πεδίο του χρόνου. Περιέχει 2 κανάλια ήχου. Πίνακας 2048×2.

`next_frame_T`: Το επόμενο frame, $i + 1$. Χρησιμοποιείται για την επιλογή παραθύρου. Πίνακας 2048×2.

`prev_frame_type`: Ο τύπος που είχε επιλεγεί για το frame $i - 1$.

```
frame_F = filter_bank(frame_T, frame_type, win_type)
```

Υλοποιεί τη βαθμίδα Filterbank.

`frame_F`: Το frame στο πεδίο της συχνότητας υπό τη μορφή συντελεστών MDCT. Πίνακας 1024×2 που περιέχει είτε (α) τους συντελεστές των δύο καναλιών όταν `frame_type = OLS | LSS | LPS` είτε (β) οκτώ υποπίνακες 128×2 έναν για κάθε subframe στην περίπτωση `frame_type = ESH` τοποθετημένους σε στήλες σύμφωνα με τη σειρά του subframe.

`frame_T`: Το frame στο πεδίο του χρόνου

`frame_type`: Ο τύπος που έχει επιλεγεί για το υπό κωδικοποίηση frame

`win_type`: Ο τύπος που έχει επιλεγεί για το παράθυρο βαρών του τρέχοντος frame. Επιτρεπτές τιμές οι "KBD" και "SIN"

```
frame_T = i_filter_bank(frame_F, frame_type, win_type)
```

Υλοποιεί την αντίστροφη της `filterbank`.

```
aac_seq_1 = aac_coder_1(filename_in)
```

`aac_seq_1`: Λίστα πλήθους K, όπου K το πλήθος των frames που έχουν κωδικοποιηθεί. Κάθε στοιχείο της παραπάνω λίστας είναι ένα dictionary που αποτελείται από τα παρακάτω πεδία:

- `aac_seq_1[i]["frame_type"]`: Όπως παραπάνω
- `aac_seq_1[i]["win_type"]`: Όπως παραπάνω
- `aac_seq_1[i]["chl"]["frame_F"]`: Οι συντελεστές MDCT για το αριστερό κανάλι διαστάσεων 128×8 για frames τύπου EIGHT SHORT SEQUENCE και 1024×1 για όλους τους άλλους τύπους
- `aac_seq_1[i]["chr"]["frame_F"]`: Οι συντελεστές MDCT για το δεξί κανάλι διαστάσεων 128×8 για frames τύπου EIGHT SHORT SEQUENCE και 1024×1 για όλους τους άλλους τύπους

`filename_in`: Το όνομα του αρχείου wav με το σήμα ήχου προς κωδικοποίηση. Υπόθεση: Το αρχείο περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48kHz.

```
x = i_aac_coder_1(aac_seq_1, filename_out)
```

Αντιστρέφει τη διαδικασία `aac_coder_1()`.

`x` : η αποκωδικοποιημένη ακολουθία δειγμάτων.

`aac_seq_1`: Όπως πραραπάνω.

`filename_out` : Το όνομα του αρχείου wav στο οποίο θα εγγραφεί το σήμα ήχου μετά την αποκωδικοποίηση. Υπόθεση: Το αρχείο θα περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48 kHz.

```
SNR = demo_aac_1(filename_in, filename_out)
```

Επιδεικνύει την κωδικοποίηση του 1ου επιπέδου.

`SNR`: Συνολικός σηματοθορυβικός λόγος (σε dB).

3.2 2ο Επίπεδο (+1 μονάδα)

```
[frame_F_out, tns_coeffs] = tns(frame_F_in, frame_type)
```

Υλοποιεί τη βαθμίδα Temporal Noise Shaping (TNS) για ένα κανάλι.

`frame_F_in`, `frame_F_out`: Οι συντελεστές MDCT πριν και μετά το Temporal Noise Shaping, διαστάσεων 128×8 για EIGHT SHORT SEQUENCE, 1024×1 αλλιώς

`tns_coeffs`: Οι κβαντισμένοι συντελεστές TNS, διαστάσεων 4×8 για EIGHT SHORT SEQUENCE, 4×1 αλλιώς

`frame_type`: Όπως παραπάνω

```
frame_F_out = i_tns(frame_F_in, frame_type, tns_coeffs)
```

Αντιστρέφει τη βαθμίδα Temporal Noise Shaping (TNS).

`frame_F_in`, `frame_F_out`: Όπως παραπάνω

`frame_type`, `tns_coeffs`: Όπως παραπάνω

```
aac_seq_2 = aac_coder_2(filename_in)
```

`aac_seq_2`: λίστα πλήθους K όπου K το πλήθος των frames που έχουν κωδικοποιηθεί. Κάθε στοιχείο αυτής, είναι ένα dictionary που αποτελείται από τα παρακάτω πεδία:

- `aac_seq_2[i]["frame_type"]`: Όπως παραπάνω
- `aac_seq_2[i]["win_type"]`: Όπως παραπάνω
- `aac_seq_2[i]["chl"]["tns_coeffs"]`: Οι κβαντισμένοι συντελεστές TNS του αριστερού καναλιού
- `aac_seq_2[i]["chr"]["tns_coeffs"]`: Οι κβαντισμένοι συντελεστές TNS του δεξιού καναλιού
- `aac_seq_2[i]["chl"]["frame_F"]`: Οι συντελεστές MDCT για το αριστερό κανάλι μετά το TNS
- `aac_seq_2[i]["chr"]["frame_F"]`: Οι συντελεστές MDCT για το δεξί κανάλι μετά το TNS

`filename_in`: Το όνομα του αρχείου wav με το σήμα ήχου προς κωδικοποίηση. Υπόθεση: Το αρχείο περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48kHz.

```
x = i_aac_coder_2(aac_seq_2, filename_out)
```

Αντιστρέφει τη διαδικασία `aac_coder_2()`.

`x`: η αποκωδικοποιημένη ακολουθία δειγμάτων

`aac_seq_2`: Όπως παραπάνω.

`filename_out`: Το όνομα του αρχείου wav στο οποίο θα εγγραφεί το σήμα ήχου μετά την αποκωδικοποίηση. Υπόθεση: Το αρχείο θα περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48 kHz.

```
SNR = demo_aac_2(filename_in, filename_out)
```

Επιδεικνύει την κωδικοποίηση του 2ου επιπέδου.

`SNR`: Συνολικός σηματοθορυβικός λόγος (σε dB).

3.3 3ο Επίπεδο (+2 μονάδες)

Υλοποιήστε τη συνάρτηση

```
SMR = psycho(frame_T, frame_type, frame_T_prev_1, frame_T_prev_2)
```

η οποία υλοποιεί τη βαθμίδα Psychoacoustic model για ένα κανάλι.

`SMR`: Signal to Mask Ratio, διάστασης 42×8 για frames τύπου EIGHT SHORT SEQUENCE και 69×1 για όλους τους άλλους τύπους

`frame_T`: Όπως παραπάνω

`frame_type`: Όπως παραπάνω

`frame_T_prev_1`: Το προηγούμενο frame του `frame_T` στο ίδιο κανάλι

`frame_T_prev_2`: Το προ-προηγούμενο frame του `frame_T` στο ίδιο κανάλι

```
S, sfc, G = aac_quantizer(frame_F, frame_type, SMR)
```

Υπολογίζει εσωτερικά το κατώφλι ακουστότητας $T(b)$ και υλοποιεί τη βαθμίδα Quantizer για ένα κανάλι.

`S`: Πίνακας 1024×1 με τα σύμβολα κβάντισης των συντελεστών MDCT του τρέχοντος frame (για όλους τους τύπους frame)

`sfc`: Πίνακας $N_B \times 8$ για τα frames τύπου EIGHT SHORT SEQUENCE (όπου ο αριθμός από μπάντες) και $N_B \times 1$ για όλους τους υπόλοιπους τύπους, με τις τιμές των συντελεστών Scalefactor για κάθε Scalefactor band

`G`: Το Global gain του τρέχοντος frame (1×8 για EIGHT SHORT SEQUENCE ή μία τιμή για όλες τις άλλες περιπτώσεις)

`frame_F`, `frame_type`: Όπως παραπάνω

`SMR`: Όπως παραπάνω

```
frame_F = i_aac_quantizer(S, sfc, G, frame_type)
```

Υλοποιεί τη βαθμίδα iQuantizer που αντιστρέφει τη συνάρτηση quantizer().

Το επόμενο και τελευταίο βήμα είναι η κωδικοποίηση των κβαντισμένων συντελεστών MDCT και των scalefactors sfc κατά Huffman. Για το σκοπό αυτό σας δίνονται στο αρχείο `huff_utils.py` οι ακόλουθες συναρτήσεις:

`huff_sec, huff_codebook = encode_huff(coeff_sec, huff_LUT_list, force_codebook)` Κωδικοποίηση εντροπίας (με Huffman) ενός συνόλου συντελεστών ή scalefactors.

`coeff_sec`: Οι κβαντισμένοι συντελεστές MDCT S ή οι κβαντισμένοι scale factors a_j για ένα frame.

`huff_LUT_list`: Η λίστα με τα Look-Up Tables που θα χρησιμοποιηθεί. Η μεταβλητή αυτή λαμβάνεται καλώντας τη συνάρτηση `load_LUT()` που επίσης σας δίνεται, και με τη σειρά της καλείται με όρισμα το path του αρχείου `huffCodebooks.mat`

`force_codebook`: Προαιρετικό όρισμα, αναγκάζει τη χρήση του codebook που δίνεται ως όρισμα force_codebook. Πρακτικά χρησιμοποιείται μόνο για την κωδικοποίηση των scale factors, όπου θέτουμε `force_codebook = 11`. Για την κωδικοποίση των MDCT το όρισμα αυτό παραλείπεται .

`huff_sec`: Μία ακολουθία από '1' και '0' που αντιστοιχούν στη δυαδική κωδικοποίηση κατά Huffman.

`huff_codebook`: Το Huffman codebook που χρησιμοποιήθηκε.

`dec_coeffs = decode_huff(huff_sec, huff_codebook, huff_LUT)`

Αποκωδικοποίηση ενός συνόλου συντελεστών ή scalefactors που έχουν κωδικοποιηθεί με Huffman.

`huff_sec`: Όπως παραπάνω

`huff_LUT`: Όπως παραπάνω

`huff_codebook`: Όπως παραπάνω

`dec_coeffs`: Οι αποκωδικοποιημένοι συντελεστές

`aac_seq_3 = aac_coder_3(filename_in, filename_aac_coded)`

`aac_seq_3` λίστα πλήθους K όπου K το πλήθος των frames που έχουν κωδικοποιηθεί. Κάθε στοιχείο αυτής είναι dictionary που αποτελείται από τα παρακάτω πεδία:

- `aac_seq_3[i]["frame_type"]`: Όπως παραπάνω
- `aac_seq_3[i]["win_type"]`: Όπως παραπάνω
- `aac_seq_3[i]["chl"]["tns_coeffs"]`: Όπως παραπάνω
- `aac_seq_3[i]["chr"]["tns_coeffs"]`: Όπως παραπάνω
- `aac_seq_3[i]["chl"]["T"]`: Τα κατώφλια του ψυχοακουστικού μοντέλου ($N_B \times 1$) για το αριστερό κανάλι (για λόγους οπτικοποίησης, δε χρειάζονται για στην κωδικοποίηση)
- `aac_seq_3[i]["chr"]["T"]`: Τα κατώφλια του ψυχοακουστικού μοντέλου ($N_B \times 1$) για το δεξί κανάλι (για λόγους οπτικοποίησης, δε χρειάζονται για στην κωδικοποίηση)
- `aac_seq_3[i]["chl"]["G"]`: Τα κβαντισμένα global gains (1 ή 8) για το αριστερό κανάλι
- `aac_seq_3[i]["chr"]["G"]`: Τα κβαντισμένα global gains (1 ή 8) για το δεξί κανάλι
- `aac_seq_3[i]["chl"]["sfc"]`: Η κωδικοποιημένη (από Huffman) ακολουθία των sfc για το αριστερό κανάλι
- `aac_seq_3[i]["chr"]["sfc"]`: Η κωδικοποιημένη (από Huffman) ακολουθία των sfc για το δεξί κανάλι
- `aac_seq_3[i]["chl"]["stream"]`: Η κωδικοποιημένη (από Huffman) ακολουθία των κβαντισμένων συντελεστών MDCT για το αριστερό κανάλι.
- `aac_seq_3[i]["chr"]["stream"]`: Η κωδικοποιημένη (από Huffman) ακολουθία των κβαντισμένων συντελεστών MDCT για το δεξί κανάλι.
- `aac_seq_3[i]["chl"]["codebook"]`: Το Huffman codebook που χρησιμοποιήθηκε για το αριστερό κανάλι
- `aac_seq_3[i]["chr"]["codebook"]`: Το Huffman codebook που χρησιμοποιήθηκε για το δεξί κανάλι

filename_in: Το όνομα του αρχείου wav με το σήμα ήχου προς κωδικοποίηση. Υπόθεση: Το αρχείο περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48kHz.

filename_aac_coded: Αρχείο .mat στο οποίο αποθηκεύεται η δομή `aac_seq_3`.

```
x = i_aac_coder_3(aac_seq_3, filename_out)
```

Αντιστρέφει τη διαδικασία `aac_coder_3()`.

x: η αποκωδικοποιημένη ακολουθία δειγμάτων

aac_seq_3: Όπως παραπάνω.

filename_out: Το όνομα του αρχείου wav στο οποίο θα εγγραφεί το σήμα ήχου μετά την αποκωδικοποίηση. Υπόθεση: Το αρχείο θα περιέχει ήχο δικαναλικό με συχνότητα δειγματοληψίας 48 kHz.

```
SNR, bitrate, compression = demo_aac_3(filename_in, filename_out, frame_aac_coded)
```

Επιδεικνύει την κωδικοποίηση του 3ου επιπέδου.

filename_in, filename_out: Όπως παραπάνω

filename_aac_coded: Αρχείο .mat στο οποίο αποθηκεύεται η δομή `aac_seq_3`

SNR: Όπως παραπάνω

bitrate: bits ανά sec

compression: bitrate πριν την κωδικοποίηση προς bitrate μετά την κωδικοποίηση

Καλή επιτυχία!

References

[1] Jürgen Herre and James D Johnston. "Enhancing the performance of perceptual audio coders by using temporal noise shaping (TNS)". In: *Audio Engineering Society Convention 101*. Audio Engineering Society. 1996.

[2] Yiqing Lin and Waleed H. Abdulla. "Principles of Psychoacoustics". In: *Audio Watermark: A Comprehensive Foundation Using MATLAB*. Cham: Springer International Publishing, 2015, pp. 15–49. isbn: 978-3-319-07974-5. doi: 10.1007/978-3-319-07974-5_2. url: http://dx.doi.org/10.1007/978-3-319-07974-5_2.