ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

Τμημα ΗΜΜΥ. Τομεας Ηλεκτρονικης

Συστηματα Πολυμεσων και Εικονικη Πραγματικοτητα

Εργασία Εξαμήνου

Απλοποιημένη κωδικοποίηση-αποκωδικοποίηση AAC

Συντάκτης:
Χρήστος Χουτουρίδης [8997]
cchoutou@ece.auth.gr

Διδάσκων:
Αναστάσιος Ντελόπουλος
antelopo@ece.auth.gr

22 Φεβρουαρίου 2026

1. Εισαγωγη

Η παρούσα εργασία αφορά την υλοποίηση ενός απλοποιημένου κωδικοποιητή και αποκωδικοποιητή ήχου κατά το πρότυπο **Advanced Audio Coding** (AAC). Το AAC αποτελεί μία μέθοδο κωδικοποίησης μετασχηματισμού (transform coding) η οποία συνδυάζει ανάλυση στο πεδίο της συχνότητας, ψυχοακουστικό μοντέλο και κωδικοποίηση εντροπίας, με στόχο την αποδοτική συμπίεση ηχητικών σημάτων υψηλής ποιότητας. Η βασική αρχή λειτουργίας του βασίζεται στη μείωση της πλεονάζουσας πληροφορίας μέσω του μετασχηματισμού **MDCT** και στην εκμετάλλευση των ιδιοτήτων της ανθρώπινης ακοής ώστε να επιτρέπεται ελεγχόμενη απώλεια πληροφορίας που δεν είναι αντιληπτή.

Στο πλαίσιο της εργασίας υλοποιούμε μία *απλοποιημένη εκδοχή* του προτύπου, όπως αυτή περιγράφεται στην εκφώνηση, παραλείποντας ορισμένες βαθμίδες όπως το Mid/Side stereo και το Bit Reservoir, διατηρώντας όμως τον βασικό κορμό της αλυσίδας κωδικοποίησης. Συγκεκριμένα, αναπτύσσουμε διαδοχικά τις βαθμίδες Sequence Segmentation Control, Filterbank (MDCT/IMDCT), Temporal Noise Shaping, Psychoacoustic Model, Quantization και Huffman coding, καθώς και τις αντίστροφες διαδικασίες αποκωδικοποίησης.

Στην παρούσα αναφορά περιγράφουμε τη θεωρητική βάση κάθε βαθμίδας, τον τρόπο υλοποίησής της και τα αποτελέσματα που προκύπτουν από την πειραματική αξιολόγηση του συστήματος.

1.1. Παραδοτέα

Τα παραδοτέα της εργασίας αποτελούνται από:

- Την παρούσα αναφορά.
- Τον κεντρικό κατάλογο `source` που περιέχει τους ζητούμενους από την εκφώνηση καταλόγους `level_<i>`, $i = 1, 2, 3$ με τον κώδικα της εφαρμογής. Ο καθένας περιέχει ένα αντίστοιχο script `level_<i>.py` που καλεί την demonstration συνάρτηση του αντίστοιχου level. Επίσης ο κατάλογος `source` περιέχει και τον κατάλογο `core`, όπου έγινε η κεντρική ανάπτυξη του κώδικά και των tests.
- Το σύνδεσμο με το αποθετήριο που περιέχει όλο το project με τον κώδικα της εφαρμογής και της παρούσας αναφοράς.

2. Υλοποιηση

Η υλοποίηση οργανώνεται **αρθρωτά**, με σαφή διαχωρισμό των επιμέρους λειτουργικών βαθμίδων, ώστε κάθε στάδιο της κωδικοποίησης και της αποκωδικοποίησης να μπορεί να ελεγχθεί και να επαληθευτεί ανεξάρτητα. Η δομή αυτή μας επιτρέπει να αναπτύξουμε σταδιακά τρία επίπεδα ολοκλήρωσης, από την πλήρως αναστρέψιμη μετασχηματιστική κωδικοποίηση έως την πλήρη απωλεστική κωδικοποίηση με ψυχοακουστικό έλεγχο και κωδικοποίηση εντροπίας.

Οι επιμέρους βαθμίδες του κωδικοποιητή και του αποκωδικοποιητή υλοποιήθηκαν ως **ανεξάρτητα modules**, τα οποία επαναχρησιμοποιούνται σε κάθε κατάλογο `level_<i>`. Αν και η δομή του παραδοτέου δίνει την εντύπωση ότι οι κατάλογοι των επιπέδων περιέχουν αντίγραφα των ίδιων αρχείων, στην πράξη αυτό δεν ισχύει. Η αρχική μας προσέγγιση ήταν η ύπαρξη ενός κεντρικού καταλόγου με κοινή υλοποίηση για όλα τα επίπεδα, ωστόσο οι περιορισμοί της εκφώνησης δεν επέτρεπαν τέτοια οργάνωση. Αντί της χρήσης υπο-αποθετηρίων, επιλέξαμε τη λύση των hard links, ώστε τα αρχεία του κοινού πυρήνα να εμφανίζονται σε κάθε κατάλογο επιπέδου χωρίς να υπάρχει πραγματικός διπλασιασμός του κώδικα. Με τον τρόπο αυτό διατηρούμε **ένα ενιαίο σημείο που συντηρείται ο κώδικας** και αποφεύγουμε αποκλίσεις μεταξύ των επιπέδων.

Κάθε module αναπτύχθηκε ανεξάρτητα, με εκτεταμένη χρήση αυτοματοποιημένων δοκιμών. Η ανάπτυξη ακολούθησε προσέγγιση **Test-Driven Development**, όπου τα tests σχεδιάστηκαν με βάση το δημόσιο interface κάθε module και περιγράφουν τη λειτουργική του συμπεριφορά. Τα tests έχουν λειτουργικό και συμβατικό χαρακτήρα (functional and contract-based) και ελέγχουν σχήματα δεδομένων, αριθμητική σταθερότητα, οριακές περιπτώσεις και ιδιότητες αναστρεψιμότητας. Η πρακτική αυτή μας επέτρεψε να εκτελούμε εύκολα **regression tests** μετά από κάθε τροποποίηση, διασφαλίζοντας ότι δεν εισάγονται ανεπιθύμητες παρενέργειες σε άλλα μέρη

του συστήματος.

Δημιουργήσαμε επίσης wrapper module για το δοθέν module κωδικοποίησης Huffman. Κατά την ενσωμάτωσή του **εντοπίστηκε σφάλμα τύπου off-by-one** στον μηχανισμό αποκωδικοποίησης escape τιμών του codebook 11. Το σφάλμα αφορούσε τον χειρισμό του bit διαχωρισμού (delimiter) στη μορφή κωδικοποίησης των escape τιμών. Συγκεκριμένα, κατά την αποκωδικοποίηση μετρούνταν σωστά τα N διαδοχικά bits τιμής '1', όμως το επόμενο bit '0', που λειτουργεί ως διαχωριστής, δεν παραλειπόταν πριν την ανάγνωση των N+4 bits του payload. Ως αποτέλεσμα, το bit διαχωρισμού συμπεριλαμβανόταν εσφαλμένα στα bits της escape τιμής, οδηγώντας σε μετατόπιση κατά ένα bit και σε λανθασμένη ανακατασκευή του μεγέθους. Το σφάλμα αυτό δεν προκαλούσε εξαίρεση εκτέλεσης, αλλά αλλοίωνε τις διαφορές των scalefactors. Η διόρθωση συνίσταται στη σωστή κατανάλωση του bit διαχωρισμού πριν την ανάγνωση των N+4 bits της escape τιμής. Η προτεινόμενη αλλαγή κοινοποιήθηκε στους διδάσκοντες και έγινε αποδεκτή.

Συνοπτικά, τα βασικά modules του συστήματος είναι τα ακόλουθα:

- **aac_ssc**: Υλοποίηση της βαθμίδας Sequence Segmentation Control.
- **aac_filterbank**: Υλοποίηση MDCT/IMDCT και διαχείριση παραθύρων.
- **aac_tns**: Υλοποίηση Temporal Noise Shaping.
- **aac_psycho**: Υλοποίηση ψυχοακουστικού μοντέλου και υπολογισμός SMR.
- **aac_quantizer**: Υλοποίηση μη ομοιόμορφου κβαντιστή και αντίστροφης κβάντισης.
- **aac_huffman**: Διαχείριση κωδικοποίησης και αποκωδικοποίησης Huffman.
- **aac_coder** / **aac_decoder**: Σύνθεση της πλήρους ροής κωδικοποίησης και αποκωδικοποίησης.
- **aac_utils**: Βοηθητικές συναρτήσεις και μετρικές αξιολόγησης.

Για την επιτυχή εκτέλεση του κώδικα απαιτείται η εγκατάσταση των εξαρτήσεων. Για το λόγο αυτό από τον κεντρικό κατάλογο πρέπει να εκτελεστεί:

```
pip install -r requirements.txt
```

Τα αυτοματοποιημένα tests δεν απαιτούνται από την εκφώνηση και δεν εκτελούνται από τα scripts των επιπέδων. Παρ' όλα αυτά, ο αναγνώστης μπορεί να τα εκτελέσει από τον κεντρικό κατάλογο του project:

```
pytest -v
# to run a specific module (for example filterbank)
pytest -v core/tests/test_filterbank.py
```

3. Level 1 – SSC και MDCT

3.1. Απαιτήσεις Level 1

Στο πρώτο επίπεδο ζητείται η υλοποίηση ενός πλήρως αναστρέψιμου συστήματος κωδικοποίησης–αποκωδικοποίησης βασισμένου στον μετασχηματισμό MDCT. Το επίπεδο αυτό δεν περιλαμβάνει ψυχοακουστικό μοντέλο, κβάντιση ή κωδικοποίηση εντροπίας. Στόχος είναι η σωστή υλοποίηση του Sequence Segmentation Control και του Filterbank, έτσι ώστε **η ανακατασκευή του σήματος να είναι αριθμητικά ταυτόσημη με το αρχικό**, μέχρι αριθμητική ακρίβεια κινητής υποδιαστολής.

Το Level 1 λειτουργεί ως θεμέλιο για τα επόμενα επίπεδα, καθώς οποιοδήποτε σφάλμα στο μετασχηματιστικό στάδιο θα μεταφερόταν και θα ενισχυόταν στα επόμενα στάδια.

3.2. Modules που χρησιμοποιούνται

Για το Level 1 χρησιμοποιούνται τα εξής modules:

- **aac_ssc** για την επιλογή τύπου frame.
- **aac_filterbank** για MDCT/IMDCT και OLA.
- **aac_coder_1** και **aac_decoder_1** για τη σύνθεση της ροής.
- **aac_utils** για βοηθητικές συναρτήσεις και υπολογισμό SNR.

3.3. Sequence Segmentation Control

Η βαθμίδα SSC υλοποιεί τον μηχανισμό επιλογής τύπου παραθύρου (OLS, LSS, ESH, LPS) με βάση την ανίχνευση spikes (attack detection). Η ανίχνευση βασίζεται στον υπολογισμό της ενέργειας σε υπο-τμήματα μήκους 128 δειγμάτων και στον λόγο διαδοχικών ενεργειών.

Η υλοποίηση είναι πλήρως συμμετρική για τα δύο κανάλια και ο τελικός τύπος frame προκύπτει από συγχώνευση των δύο καναλιών σύμφωνα με τον πίνακα μετάβασης. Επίσης, δόθηκε ιδιαίτερη προσοχή στη αριθμητική σταθερότητα, ώστε να αποφεύγονται διαιρέσεις με μηδενική ενέργεια.

3.4. Filterbank και MDCT

Το module `aac_filterbank` υλοποιεί τον MDCT και τον αντίστροφό του IMDCT σε διακριτή μορφή. Χρησιμοποιούνται παράθυρα τύπου SIN και KBD, σύμφωνα με τον τύπο frame. Η υλοποίηση διαχειρίζεται σωστά τις μεταβάσεις μεταξύ παραθύρων, εφαρμόζοντας overlap-add (OLA) ώστε να επιτυγχάνεται perfect reconstruction σε steady state.

Τόσο η ιδιότητα γραμμικότητας του μετασχηματισμού, όσο και η σχέση ενίσχυσης:

$$\mathrm{MDCT}(\mathrm{IMDCT}(X)) \approx 2X$$

ελέγχθηκαν αριθμητικά στο πλαίσιο των tests.

Ενδεικτικά να αναφαίρουμε το παράδειγμα στον έλεγχος της σχέσης ενίσχυσης:

```
X: MdctCoeffs     = rng.normal(size=K)
x: TimeSignal     = imdct(X)
X_hat: MdctCoeffs = mdct(x)
_assert_allclose(X_hat, 2.0 * X, rtol=tolerance, atol=tolerance)
```

3.5. Φιλοσοφία Ελέγχου Ορθότητας

Ως γνωστόν, στον προγραμματισμό και στο quality assurance, **δεν μπορεί να υπάρχει απόλυτη απόδειξη ορθότητας**. Υπάρχει μόνο μείωση της πιθανότητας σφάλματος μέσω στοχευμένων ελέγχων.

Για το Level 1 ελέγχθηκαν:

- Ιδιότητες γραμμικότητας MDCT/IMDCT.
- Απουσία NaN/Inf σε τυχαίες εισόδους.
- Σωστή διαχείριση μεταβάσεων παραθύρων.
- Ορθή συμπεριφορά SSC σε edge cases.
- End-to-end ταυτότητα σήματος με υψηλό SNR.

Τα tests μπορούν να εκτελεστούν με:

```
pytest -v core/tests/test_filterbank.py
pytest -v core/tests/test_ssc.py
pytest -v core/tests/test_utils.py
```

Η επιτυχής εκτέλεση των παραπάνω tests διασφαλίζει ότι το Level 1 αποτελεί αριθμητικά σταθερό θεμέλιο για τα επόμενα επίπεδα.

3.6. Demo και Αποτελέσματα

Η demonstration του Level 1 μπορεί να εκτελεστεί με:

```
cd level_1
python -m level_1 material/LicorDeCalandraca.wav material/LicorDeCalandraca_out_l1.wav
```

Η έξοδος που προέκυψε ήταν:

```
Encoding ..... done
Decoding ..... done
SNR = 257.437 dB
```

Η τιμή SNR άνω των 250 dB **υποδηλώνει πρακτικά αριθμητική ταυτότητα μεταξύ αρχικού και ανακατασκευασμένου σήματος**. Η απόκλιση οφείλεται αποκλειστικά σε **αριθμητική ακρίβεια κινητής υποδιαστολής** και όχι σε δομικό σφάλμα της υλοποίησης. Το αποτέλεσμα αυτό επιβεβαιώνει ότι η υλοποίηση του Filterbank και του SSC είναι συνεπής και πλήρως αναστρέψιμη.

4. Level 2 – Temporal Noise Shaping

4.1. Απαιτήσεις Level 2

Στο δεύτερο επίπεδο ζητείται η ενσωμάτωση της βαθμίδας **Temporal Noise Shaping (TNS)** στη ροή κωδικοποίησης. Το TNS εφαρμόζεται στο πεδίο των συντελεστών MDCT και βασίζεται σε γραμμική πρόβλεψη. Στόχος της βαθμίδας είναι η χρονική ανακατανομή του σφάλματος κβάντισης, ώστε το αντιληπτό σφάλμα να κατανέμεται χρονικά με πιο ευνοϊκό τρόπο.

Στο Level 2 η διαδικασία παραμένει πλήρως αναστρέψιμη. Η κβάντιση των συντελεστών πρόβλεψης γίνεται με συγκεκριμένο βήμα και σε περιορισμένο εύρος τιμών, ώστε να διασφαλίζεται η σταθερότητα του αντίστροφου φίλτρου.

4.2. Modules που χρησιμοποιούνται

Για το Level 2 χρησιμοποιούνται:

- **`aac_ssc`**.
- **`aac_filterbank`**.
- **`aac_tns`**.
- **`aac_coder_2`** και **`aac_decoder_2`**.
- **`aac_utils`**.

Το νέο στοιχείο σε σχέση με το Level 1 είναι το module `aac_tns`, το οποίο εφαρμόζεται μετά τον MDCT και πριν από την αντίστροφη διαδικασία στον αποκωδικοποιητή.

4.3. Υλοποίηση της βαθμίδας TNS

Για κάθε frame υπολογίζονται συντελεστές γραμμικής πρόβλεψης τάξης `PRED_ORDER` (4). Στην περίπτωση ESH, η διαδικασία εφαρμόζεται ανεξάρτητα σε κάθε ένα από τα 8 υπο-frames. Οι συντελεστές πρόβλεψης κβαντίζονται με σταθερό βήμα `QUANT_STEP` (0.1) και περιορίζονται στο διάστημα $[-\texttt{QUANT\_MAX}(-0.7), +\texttt{QUANT\_MAX}(+0.7)]$.

Το φίλτρο που ορίζεται από τους συντελεστές $\{a_k\}$ έχει μορφή:

$$H(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}.$$

Στον κωδικοποιητή το φίλτρο εφαρμόζεται σε μορφή **FIR**. Κάθε νέος συντελεστής MDCT προκύπτει αφαιρώντας γραμμικό συνδυασμό προηγούμενων συντελεστών. Η επιλογή FIR μορφής εξασφαλίζει αριθμητική σταθερότητα, καθώς δεν υπάρχει ανατροφοδότηση.

Στον αποκωδικοποιητή εφαρμόζεται το αντίστροφο φίλτρο:

$$H^{-1}(z) = \frac{1}{1 - \sum_{k=1}^{p} a_k z^{-k}},$$

το οποίο έχει μορφή **IIR**. Η αντιστροφή υλοποιείται αναδρομικά, επαναφέροντας τους αρχικούς συντελεστές MDCT. Η διαδικασία αυτή είναι πλήρως αναστρέψιμη υπό την προϋπόθεση ότι το IIR φίλτρο είναι σταθερό.

Η σταθερότητα απαιτεί όλες οι ρίζες του πολυωνύμου:

$$z^p - a_1 z^{p-1} - \cdots - a_p = 0$$

να βρίσκονται **εντός του μοναδιαίου κύκλου**. Ακόμη και μικρή υπέρβαση της μοναδιαίας ακτίνας θα οδηγούσε σε εκθετική ενίσχυση κατά την αναδρομική εφαρμογή του φίλτρου και σε αριθμητική υπερχείλιση.

4.4. Φιλοσοφία Ελέγχου Ορθότητας

Όπως και στο Level 1, δεν υπάρχει απόλυτη μαθηματική απόδειξη ορθότητας. Για τη βαθμίδα TNS ελέγχθηκαν:

- Σωστά σχήματα εξόδου για long και ESH frames.
- Κβάντιση πάνω στο επιθυμητό πλέγμα τιμών.
- Περιορισμός συντελεστών στο επιτρεπτό εύρος.
- Ρητός έλεγχος σταθερότητας μέσω υπολογισμού ριζών.
- Ιδιότητα round-trip: $\text{iTNS}(\text{TNS}(X)) \approx X$.
- Απουσία NaN/Inf σε τυχαίες και οριακές εισόδους.

Ο έλεγχος σταθερότητας υλοποιείται αριθμητικά με την εξής λογική:

```
poly = np.empty(p + 1)
poly[0] = 1.0
poly[1:] = -a_q
roots = np.roots(poly)
margin = 1e-12
assert np.all(np.abs(roots) < (1.0 - margin))
```

Η ιδιότητα αντιστρεψιμότητας ελέγχεται μέσω round-trip ελέγχου:

```
frame_F_tns, coeffs = aac_tns(frame_F_in, frame_type)
frame_F_hat = aac_i_tns(frame_F_tns, frame_type, coeffs)
np.testing.assert_allclose(frame_F_hat, frame_F_in)
```

Οι παραπάνω έλεγχοι δεν αποδεικνύουν μαθηματικά την ορθότητα, αλλά διασφαλίζουν ότι το ζεύγος FIR/IIR λειτουργεί ως ακριβές αντίστροφο εντός αριθμητικής ακρίβειας και ότι η κβάντιση δεν εισάγει αστάθεια.

Τα ακριβή tests μπορούν να εκτελεστούν με:

```
pytest -v core/tests/test_tns.py
```

4.5. Demo και Αποτελέσματα

Η demonstration του Level 2 εκτελείται με:

```
cd level_2
python -m level_2 material/LicorDeCalandraca.wav material/LicorDeCalandraca_out_l2.wav
```

Η έξοδος που προέκυψε ήταν:

```
    Encoding ..................... done
    Decoding ..................... done
    SNR = 257.437 dB
```

Η τιμή SNR είναι πρακτικά ταυτόσημη με εκείνη του Level 1. Το αποτέλεσμα αυτό επιβεβαιώνει ότι η ενσωμάτωση του TNS **δεν εισάγει απώλεια πληροφορίας στο συγκεκριμένο στάδιο**. Η υλοποίηση του FIR/IIR ζεύγους και ο έλεγχος σταθερότητας εξασφαλίζουν **πλήρη αναστρεψιμότητα** της διαδικασίας.

5. Level 3 – Πλήρης Απωλεστική Κωδικοποίηση

5.1. Απαιτήσεις Level 3

Στο τρίτο επίπεδο ζητείται η υλοποίηση της **πλήρους απωλεστικής αλυσίδας κωδικοποίησης**. Προστίθενται το ψυχοακουστικό μοντέλο, ο μη ομοιόμορφος κβαντιστής και η κωδικοποίηση Huffman. Σε αντίθεση με τα προηγούμενα επίπεδα, το Level 3 **δεν είναι πλέον πλήρως αναστρέψιμο**. Η απώλεια πληροφορίας είναι συνειδητή και ελεγχόμενη, με βάση τις αρχές της ψυχοακουστικής απόκρυψης.

Στόχος είναι η μείωση του bitrate διατηρώντας αποδεκτή αντιληπτή ποιότητα.

5.2. Modules που χρησιμοποιούνται

Για το Level 3 χρησιμοποιούνται όλα τα modules του συστήματος:

- **aac_ssc**.
- **aac_filterbank**.
- **aac_tns**.
- **aac_psycho**.
- **aac_quantizer**.
- **aac_huffman**.
- **aac_coder_3** και **aac_decoder_3**.
- **aac_utils**.

Το Level 3 αποτελεί σύνθεση όλων των προηγούμενων βαθμίδων με προσθήκη κβάντισης και κωδικοποίησης εντροπίας.

5.3. Ψυχοακουστικό Μοντέλο

Το module `aac_psycho` υπολογίζει το Signal-to-Mask Ratio (SMR) για κάθε Bark band. Η διαδικασία βασίζεται στους πίνακες B219a και B219b για long και short frames αντίστοιχα. Υπολογίζεται η ενέργεια ανά band, εφαρμόζεται spreading function και κατώφλι απόλυτης ακουστότητας. Το SMR καθορίζει το επιτρεπτό σφάλμα κβάντισης σε κάθε band.

Δόθηκε ιδιαίτερη προσοχή:

- Στην αποφυγή διαίρεσης με μηδέν μέσω χρήσης μικρού όρου `EPS`.
- Στον έλεγχο ότι όλες οι ενδιάμεσες ποσότητες παραμένουν πεπερασμένες.

5.4. Κβαντιστής και Scalefactors

Το module `aac_quantizer` εφαρμόζει μη ομοιόμορφη κβάντιση των συντελεστών MDCT. Για κάθε band επιλέγεται scalefactor που ικανοποιεί το αντίστοιχο SMR. Οι scalefactors κωδικοποιούνται διαφορικά (DPCM). Η ανακατασκευή γίνεται αθροιστικά και εφαρμόζεται εκθετικός παράγοντας της μορφής:

$$\hat{X} = \text{sign}(S)|S|^{4/3} \times 2^{\alpha/4}.$$

Η συγκεκριμένη μορφή καθιστά το σύστημα **ιδιαίτερα ευαίσθητο σε αριθμητικά σφάλματα**. Για τον λόγο αυτό χρησιμοποιήθηκαν:

- Έλεγχοι πεπερασμένων τιμών (finite checks).
- Έλεγχος αποφυγής overflow.
- Προστασία με `EPS` σε παρονομαστές.

5.5. Κωδικοποίηση Huffman

Οι κβαντισμένοι συντελεστές και τα DPCM scalefactors κωδικοποιούνται με Huffman. Το bitstream ανακατασκευάζεται πλήρως στον αποκωδικοποιητή. Ιδιαίτερη προσοχή δόθηκε:

- Στη σωστή **διαχείριση escape τιμών**.
- Στην ορθή **κατανάλωση bitstreams**.
- Στην **αποφυγή out-of-bounds** προσπελάσεων.

5.6. Φιλοσοφία Ελέγχου Ορθότητας

Το Level 3 **δεν μπορεί να ελεγχθεί με κριτήριο ταυτότητας**. Ελέγχθηκε με κριτήρια λειτουργικής ορθότητας και αριθμητικής σταθερότητας.

Συγκεκριμένα ελέγχθηκαν:

- Ορθότητα DPCM ανακατασκευής scalefactors.
- Συνεπής αντιστοίχιση ESH packing και unpacking.
- Απουσία NaN/Inf σε όλη τη ροή.
- Απουσία εκθετικής υπερχείλισης.
- Έλεγχος μηδενικού lag μεταξύ αρχικού και ανακατασκευασμένου σήματος.
- Έλεγχος μη ανταλλαγής καναλιών (L/R swap detection).
- Έλεγχος απουσίας clipping.
- Έλεγχος διατήρησης συνολικού gain.

Ο **έλεγχος lag** υλοποιείται μέσω εκτίμησης χρονικής μετατόπισης. Ο έλεγχος καναλιών διασφαλίζει ότι η **ενέργεια του αριστερού και δεξιού καναλιού δεν έχει ανταλλαγεί**. Οι έλεγχοι αυτοί διασφαλίζουν ότι η απώλεια ποιότητας οφείλεται αποκλειστικά στην κβάντιση και όχι σε δομικό σφάλμα.

Τα tests μπορούν να εκτελεστούν με:

```
pytest -v core/tests/test_psycho.py
pytest -v core/tests/test_quantizer.py
pytest -v core/tests/test_huffman.py
pytest -v core/tests/test_aac_coder_decoder.py
```

5.7. Demo και Αποτελέσματα

Η demonstration του Level 3 εκτελείται με:

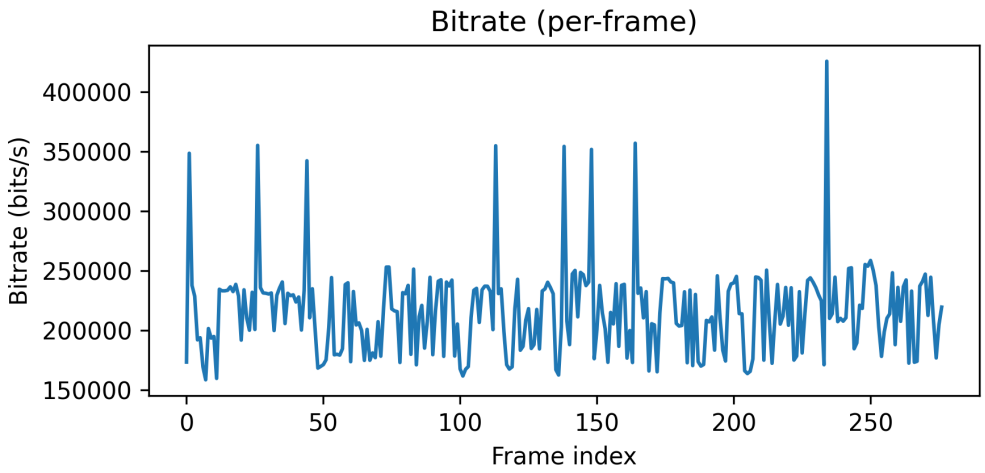
```
cd level_3
python -m level_3 material/LicorDeCalandraca.wav material/LicorDeCalandraca_out_l3.wav
↪  material/aac_seq_3.mat
```

Η έξοδος που προέκυψε ήταν:

```
Storing coded sequence to material/aac_seq_3.mat
Encoding ..................... done
Decoding ..................... done
SNR = 9.454 dB
Bitrate (coded) = 216710.22 bits/s
Compression ratio = 7.0881
```

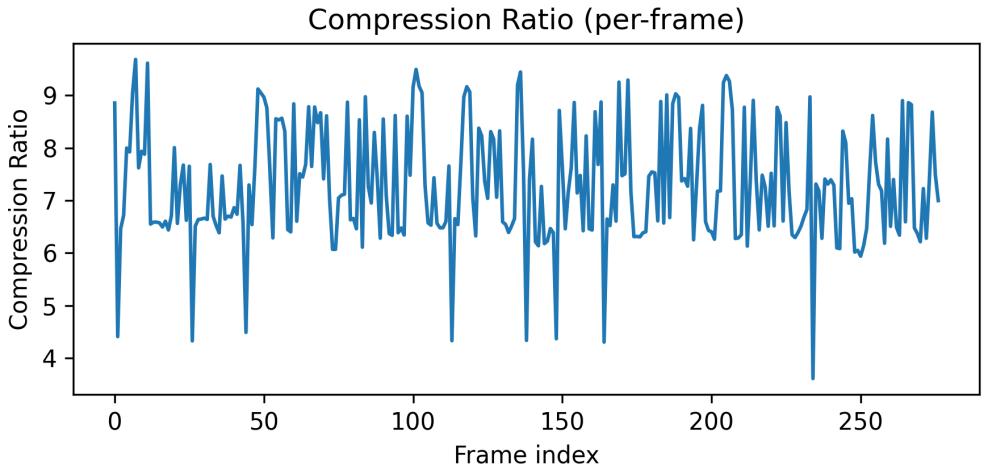
Παρατηρούμε πως η τιμή SNR είναι σημαντικά χαμηλότερη σε σχέση με τα προηγούμενα επίπεδα, γεγονός αναμενόμενο λόγω απωλεστικής κβάντισης. Παρά τη μείωση του SNR, ελέγχθηκε ότι:

- Δεν υπάρχει χρονική μετατόπιση (lag).
- Δεν υπάρχει ανταλλαγή καναλιών.
- Δεν εμφανίζεται clipping.
- Δεν υπάρχει μη ελεγχόμενη ενίσχυση (gain drift).



Εικόνα 1: *Κατανομή του bitrate ανά frame. Παρατηρείται σημαντική διακύμανση που αντανακλά την προσαρμοστική φύση της κωδικοποίησης.*

Η χαμηλή τιμή SNR δεν αντικατοπτρίζει πλήρως την αντιληπτή ποιότητα, καθώς το ψυχοακουστικό μοντέλο επιτρέπει μεγάλο ενεργειακό σφάλμα σε περιοχές όπου το σφάλμα καλύπτεται από το φαινόμενο απόκρυψης.



Εικόνα 2: *Λόγος συμπίεσης ανά frame. Η μεταβολή του compression ratio επιβεβαιώνει τη δυναμική κατανομή bit ανάλογα με το περιεχόμενο του σήματος.*

Η παρατηρούμενη απώλεια αφορά κυρίως λεπτομέρειες υψηλών συχνοτήτων, χωρίς δομική αλλοίωση του σήματος. Το αποτέλεσμα επιβεβαιώνει ότι η υλοποίηση είναι αριθμητικά σταθερή και ότι η απώλεια ποιότητας είναι αποτέλεσμα ελεγχόμενης κβάντισης και όχι υλοποιητικού σφάλματος.

Όπως φαίνεται στην Εικόνα 1, παρατηρείται σημαντική διακύμανση του bitrate μεταξύ διαδοχικών frames, με τιμές που κυμαίνονται περίπου **μεταξύ 160 kbps και 250 kbps** και σποραδικές κορυφές υψηλότερων τιμών. Η συμπεριφορά αυτή είναι αναμενόμενη, καθώς το ψυχοακουστικό μοντέλο κατανέμει περισσότερα bits σε χρονικές περιοχές με αυξημένη ενεργειακή ή φασματική πολυπλοκότητα. Αντίστοιχα, στην Εικόνα 2 παρατηρούμε ότι ο λόγος συμπίεσης μεταβάλλεται δυναμικά, με **τυπικές τιμές μεταξύ 6:1 και 9:1**.

Οι χαμηλότερες τιμές λόγου συμπίεσης αντιστοιχούν σε frames όπου απαιτούνται περισσότερα bits για την ικανοποίηση του SMR, ενώ οι υψηλότερες τιμές εμφανίζονται σε περιοχές μικρότερης φασματικής πυκνότητας. Η δυναμική αυτή συμπεριφορά επιβεβαιώνει ότι **η κωδικοποίηση δεν είναι σταθερού ρυθμού** (CBR), αλλά προσαρμοστική ως προς το περιεχόμενο του σήματος. Η συνολική μέση τιμή συμπίεσης περίπου **7:1** επιτυγχάνεται μέσω της συνδυασμένης λειτουργίας ψυχοακουστικής κβάντισης και κωδικοποίησης εντροπίας, χωρίς να παρατηρούνται δομικά σφάλματα όπως χρονική μετατόπιση, ανταλλαγή καναλιών ή ανεξέλεγκτη ενίσχυση.

6. Συμπερασματα

Στην παρούσα εργασία υλοποιήσαμε μία απλοποιημένη αλλά πλήρως λειτουργική αλυσίδα κωδικοποίησης και αποκωδικοποίησης τύπου AAC, ακολουθώντας σταδιακή προσέγγιση τριών επιπέδων. Ξεκινήσαμε από μία αυστηρά αναστρέψιμη μετασχηματιστική κωδικοποίηση με MDCT και Sequence Segmentation Control, επεκτείναμε το σύστημα με Temporal Noise Shaping διατηρώντας την αριθμητική αντιστρεψιμότητα και ολοκληρώσαμε με την ενσωμάτωση ψυχοακουστικού μοντέλου, μη ομοιόμορφης κβάντισης και κωδικοποίησης εντροπίας. Η **αρθρωτή σχεδίαση** και η **εκτεταμένη χρήση αυτοματοποιημένων δοκιμών μας επέτρεψαν να ελέγχουμε ανεξάρτητα κάθε βαθμίδα και να περιορίζουμε συστηματικά την πιθανότητα σφαλμάτων**.

Δώσαμε ιδιαίτερη έμφαση στη **αριθμητική σταθερότητα** του συστήματος. Εφαρμόσαμε ελέγχους σταθερότητας φίλτρων, προστασίες τύπου EPS σε παρονομαστές, ελέγχους πεπερασμένων τιμών, καθώς και ελέγχους απουσίας χρονικής μετατόπισης, ανταλλαγής καναλιών και clipping. Η φιλοσοφία αυτή αποδείχθηκε κρίσιμη στο Level 3, όπου μικρά αριθμητικά σφάλματα μπορούν να ενισχυθούν εκθετικά μέσω της απο-κβάντισης. Η συστηματική προσέγγιση testing και οι round-trip έλεγχοι διασφάλισαν ότι η παρατηρούμενη απώλεια ποιότητας οφείλεται αποκλειστικά στη σχεδιασμένη κβάντιση και όχι σε υλοποιητικά σφάλματα.

Τα αποτελέσματα έδειξαν ότι το σύστημα επιτυγχάνει **λόγο συμπίεσης περίπου 7:1**, με μέσο **bitrate περίπου 216 kbps**, διατηρώντας αποδεκτή αντιληπτή ποιότητα. Η ανάλυση ανά frame κατέδειξε ότι η κατανομή bit είναι δυναμική και εξαρτάται από τη φασματική πολυπλοκότητα του σήματος, γεγονός που επιβεβαιώνει τη σωστή λειτουργία του ψυχοακουστικού μοντέλου. Η σημαντική πτώση του SNR στο Level 3 είναι αναμενόμενη σε ενεργειακούς όρους και δεν ανταναχλά πλήρως την αντιληπτή ποιότητα, καθώς το σφάλμα κατανέμεται σύμφωνα με τα φαινόμενα απόκρυψης. Συνολικά, η υλοποίηση επιβεβαιώνει ότι ακόμη και ένα απλοποιημένο μοντέλο AAC μπορεί να επιτύχει ουσιαστική συμπίεση, εφόσον η σχεδίαση είναι προσεκτική και αριθμητικά συνεπής.