Ψηφιακή Επεξεργασία Σημάτων

1η Εργαστηριακή Άσκηση

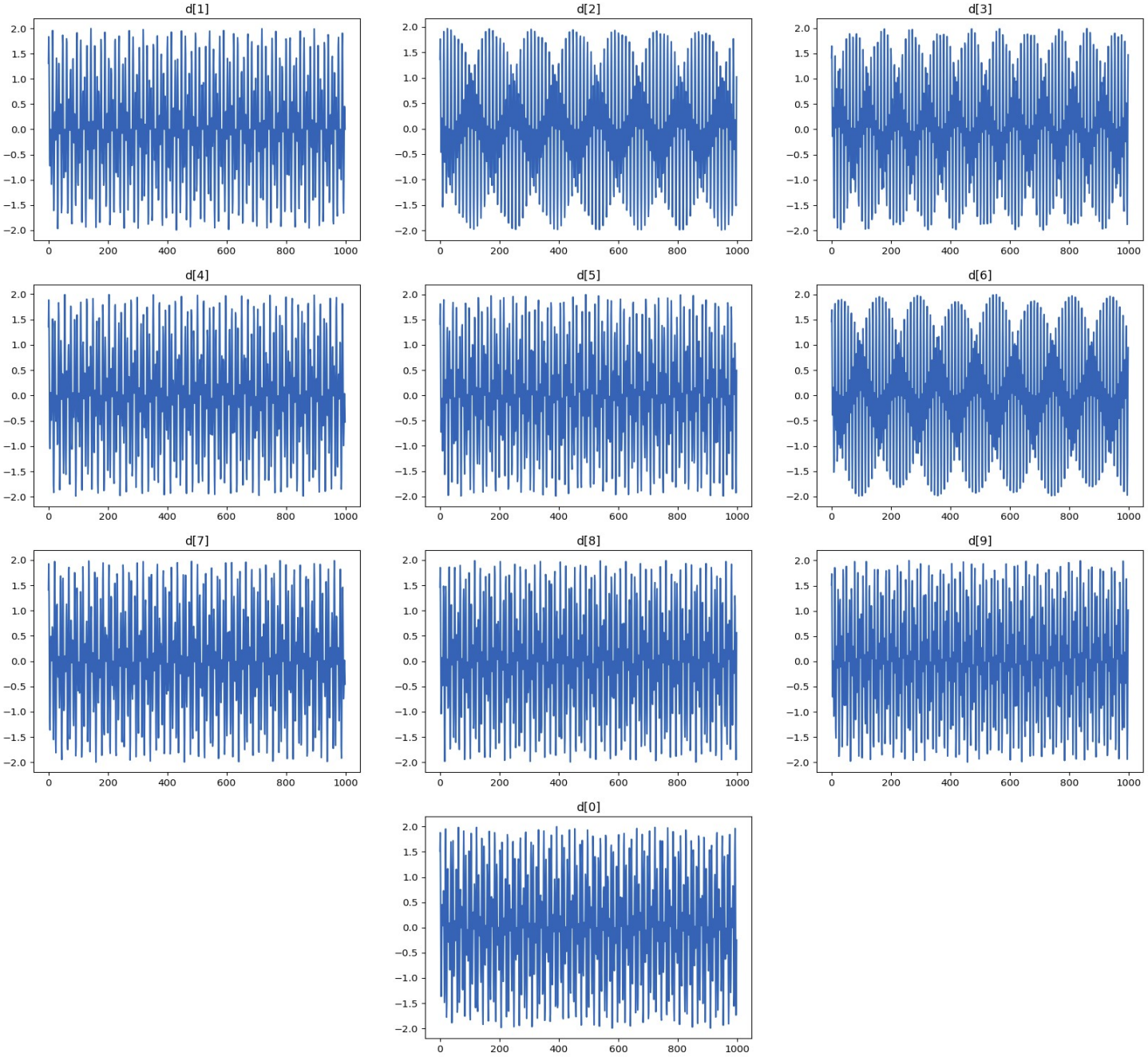
Παπαρρηγόπουλος Θοδωρής el18040 paparrigopoulosthodoris@gmail.com
Φιλιππόπουλος Ορφέας el18082 orfeasfil2000@gmail.com

Η εργασία συντάχθηκε σε python3.6.

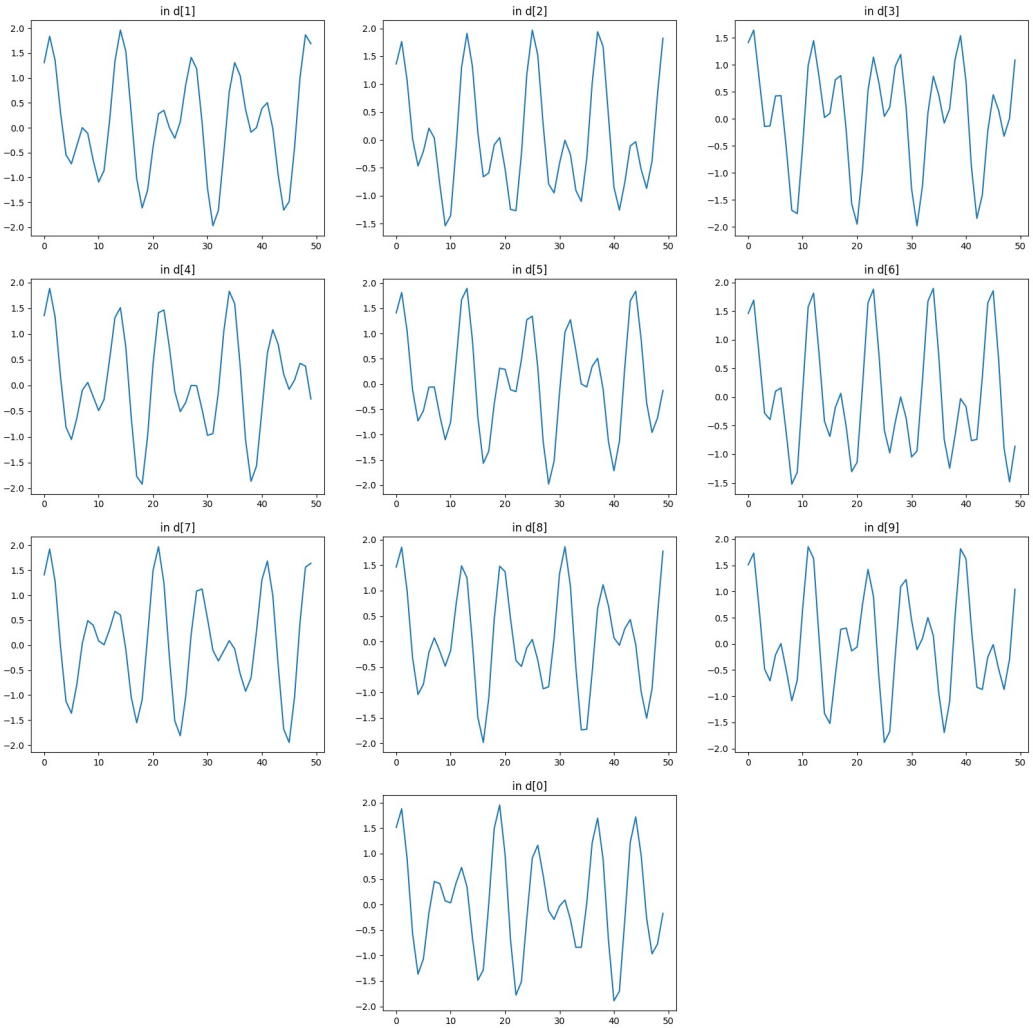
Επιπλέον να τονιστεί ότι για να τρέξουν οι κώδικες σε άλλον υπολογιστή πρέπει να υπάρχουν οι φάκελοι diagrams, ask2_1 και ask2_2 όπου και κάναμε save τα γραφήματα μας. Επίσης, όλα τα ηχητικά που διαβάζουμε βρίσκονται σε ένα επίπεδο πίσω (εξού και το ../ στα αντίστοιχα paths).

Μέρος 1

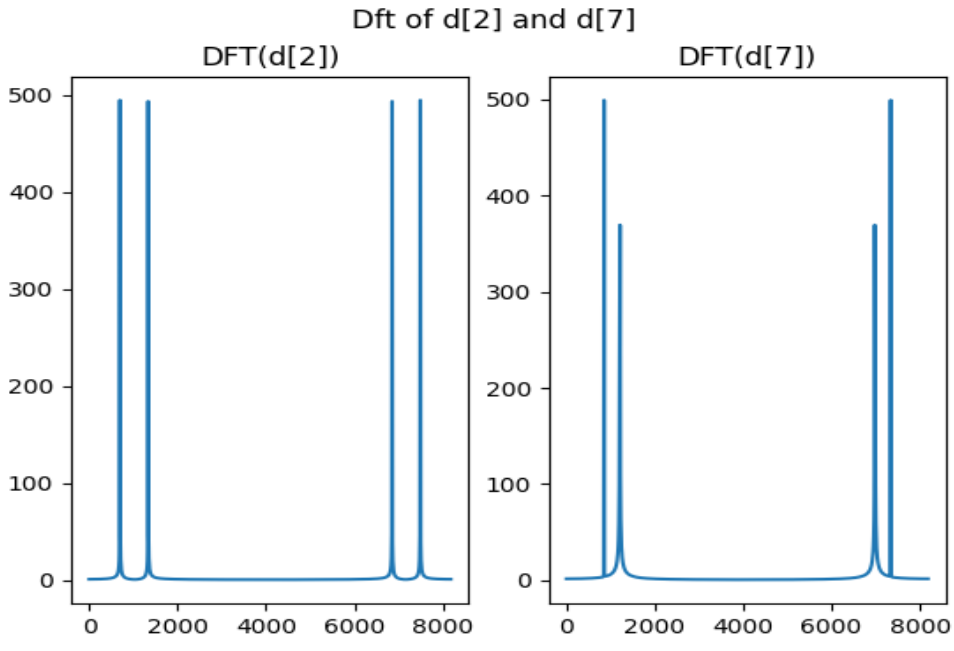
1.1) Αρχικά δημιουργήσαμε 10 διαφορετικούς τόνος με 1000 δείγματα ο καθένας, βασισμένοι στον πίνακα της άσκησης. Προέκυψαν τα παρακάτω διαγράμματα.

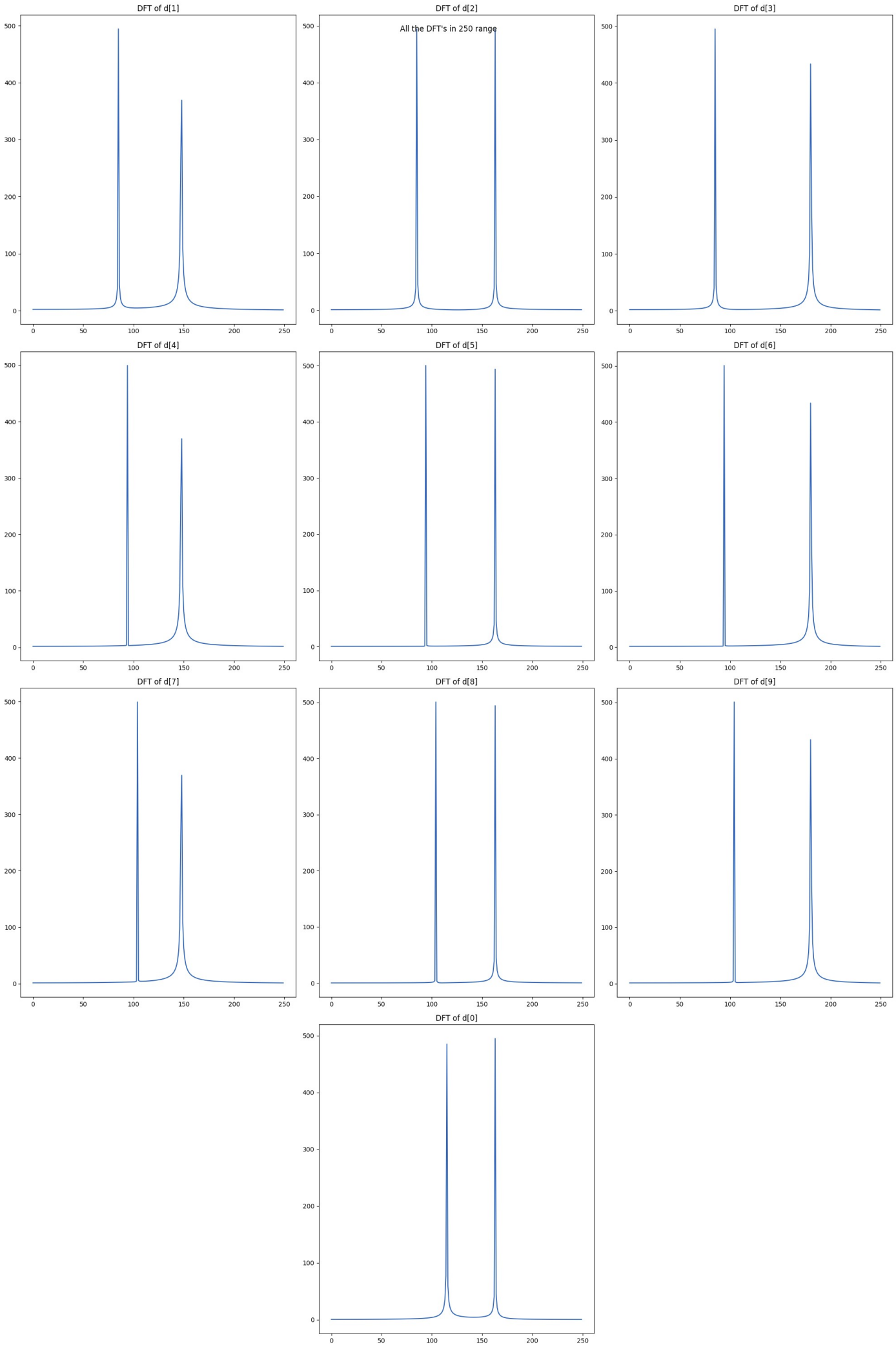


Βέβαια για να παρατηρήσουμε διαφορές πρέπει να κάνουμε zoom. Έτσι προκύπτουν :



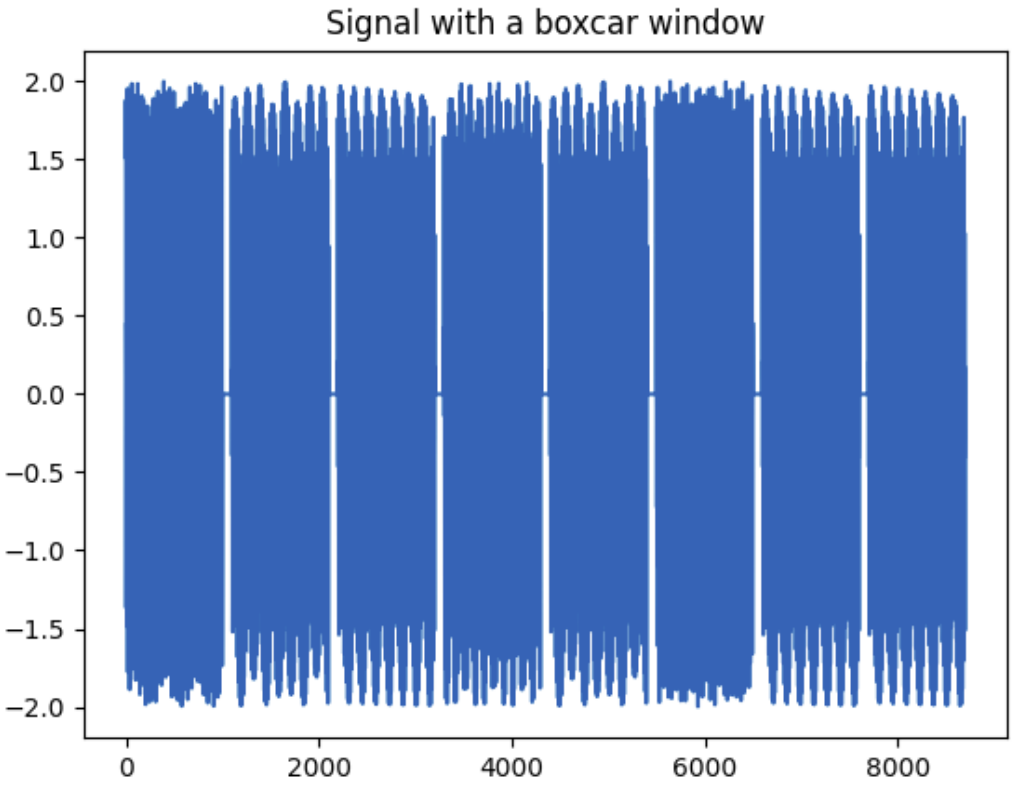
1.2) Στη συνέχεια, υπολογίσαμε τους DFT για τον αριθμό 2 και 7. Και επιπλέον, για πληρότητα υπολογίσαμε και για τους υπόλοιπους αριθμούς προκειμένου να τους παρατηρήσουμε. Από τα συνολικά 1000 δείγματα κρατήσαμε μόνο τα 250 προκειμένου να δούμε τις διαφορές στις συχνότητες.



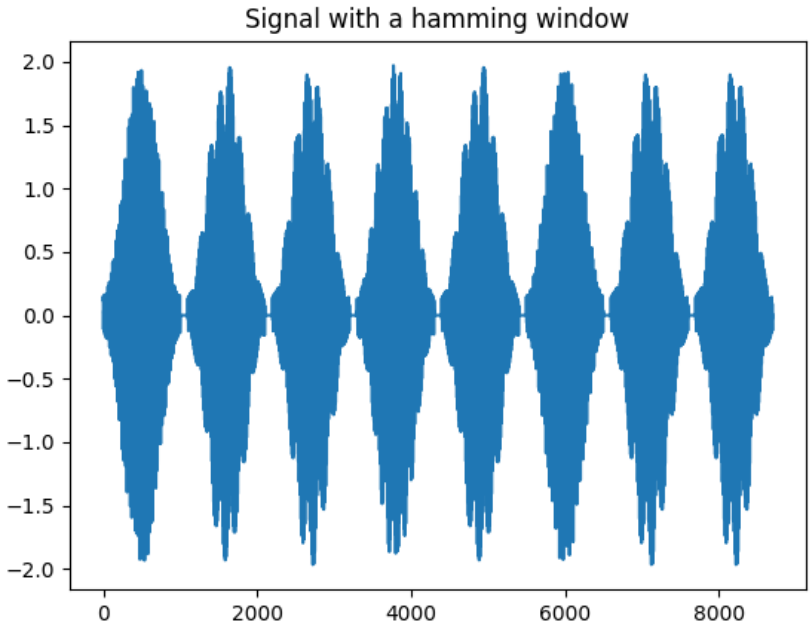

DFT of d[1]
DFT of d[2]
All the DFT's in 250 range
DFT of d[3]
DFT of d[4]
DFT of d[5]
DFT of d[6]
DFT of d[7]
DFT of d[8]
DFT of d[9]
DFT of d[0]

1.3) Έπειτα, ο αθροιστικός αριθμός των αμ μας είναι
03118040 + 03118082 = 06236122
Χρησιμοποιώντας των πίνακα που δημιουργήσαμε στην 1.1, δημιουργήσαμε το αρχείο tone_sequence.wav έχοντας 100 μηδενικά μεταξύ κάθε νότας.

1.4 i) Για να πάρουμε το τετραγωνικό μας παλμό χρησιμοποιήσαμε την get_window('boxcar', 1000). Έτσι πολλαπλασιάζοντας την με το σήμα μας προκύπτει το εξής διάγραμα.



ii) Ωστόσο, από αυτό το διάγραμμα δεν μπορούμε να βγάλουμε εύκολα συμπεράσματα. Για αυτό χρησιμοποιούμε και το παράθυρο hamming. Το οποίο βοηθάει στο να διακρίνουμε πιο έντονα τα μέγιστα τόσο στο πεδίο του χρόνου όσο και στης συχνότητας.



1.5) Προκειμένου να αναγνωρίσουμε έπειτα ποιον αριθμό έχουμε βάση της συχνότητας, δημιουργούμε μια λίστα που την ονομάζουμε frequency_peaks η οποία θα υπολογίσει τα 2 μέγιστα σημεία στην συχνότητα του κάθε τόνου. Για αυτό το ερώτημα χρησιμοποιήσαμε τον τύπο
$f = \Omega * 8192/ (2\pi)$.

Προκύπτουν
The frequency peaks are
[(941, 1336), (697, 1209), (697, 1336), (697, 1477), (770, 1209), (770, 1336), (770, 1477), (852, 1209), (852, 1336), (852, 1477)].

1.6) Προκειμένου να υλοποιήσουμε την ttdecode χρειάζεται πρώτα να υλοποιήσουμε 2 βοηθητικές συναρτήσεις.
Αρχικά, υλοποιήσαμε την findPeaksInInterval στην οποία του περνάμε ένα πίνακα και μας επιστρέφει μια λίστα με τα μέγιστα στην συχνότητα.
Στη συνέχεια, υλοποιήσαμε την mapToValues στην οποία του περνάμε τα peaks στη συχνότητα που έχουμε στο σήμα μας και τα peaks που υπολογίσαμε στην 1.5 και υπολογίζει σε ποιες συχνότητες βρίσκονται πιο κοντά και συνεπώς σε ποιον αριθμό αντιστοιχίζονται.
Για την υλοποίηση της ttdecode έχουμε ένα παράθυρο το οποίο είναι με 1000 δείγματα, υπολογίζουμε σε ποιον αριθμό αντιστοιχεί αυτό το διάστημα, και μετά μετακινούμε το παράθυρο μας ανάλογα με τα μηδενικά που βρίσκονται μετά από τον διάστημα αυτό. Και εν τέλη επιστρέφουμε τον πίνακα με τον αριθμό που υπολογίσαμε.

1.7) Κάνοντας load τους 2 πίνακες και χρησιμοποιώντας την ttdecode του προηγούμενου ερωτήματος υπολογίζουμε ότι οι αριθμοί που αντιστοιχούν στα δοθέν σήματα είναι :

Για το εύκολο σήμα: [8, 1, 0, 3, 9, 6, 3, 8]
Για το δύσκολο σήμα: [4, 8, 1, 9, 2, 1, 5, 3, 6, 3]

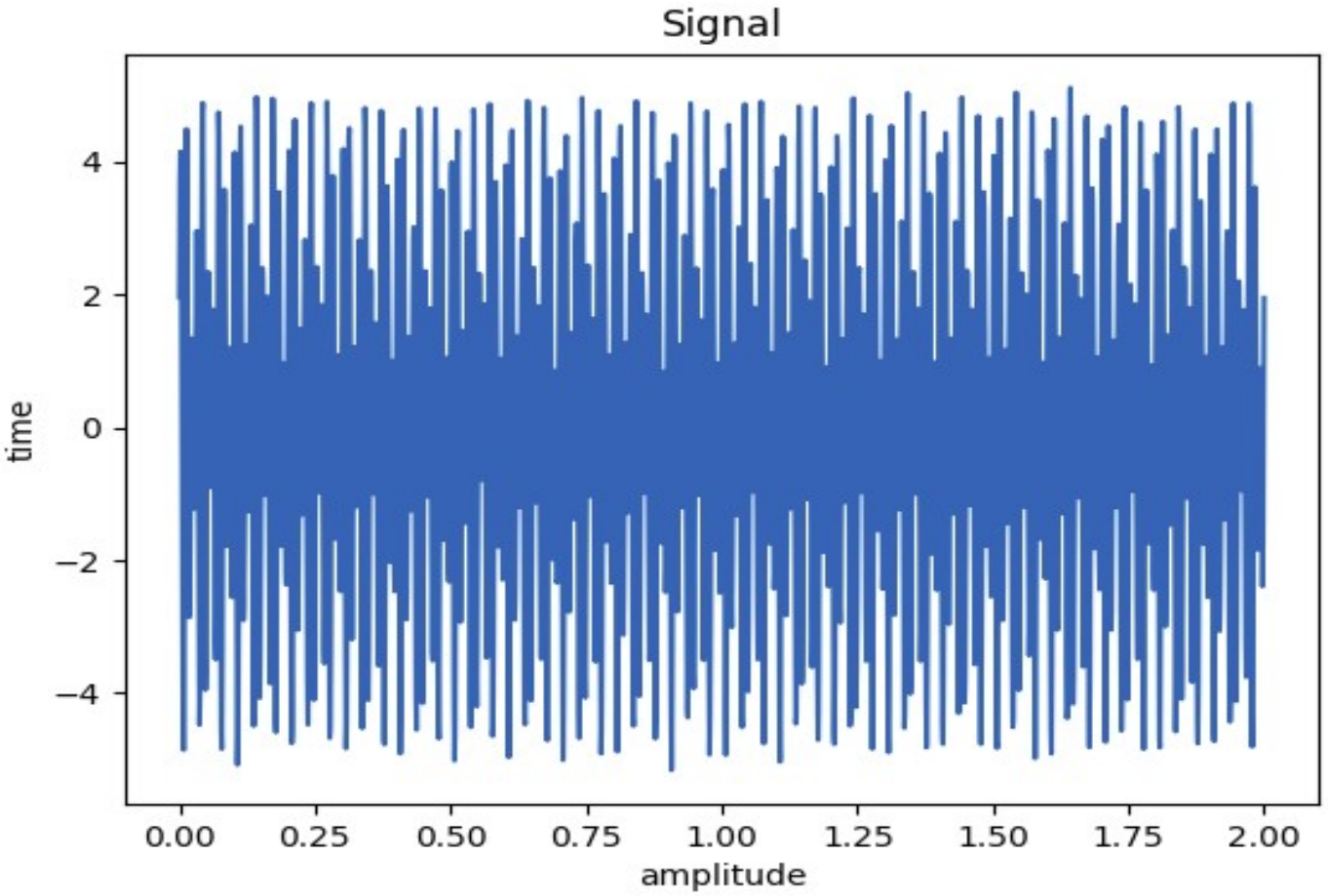
Μέρος 2

2.1)

Α) Αρχικά, δειγματοληπτούμε το παρακάτω σήμα:

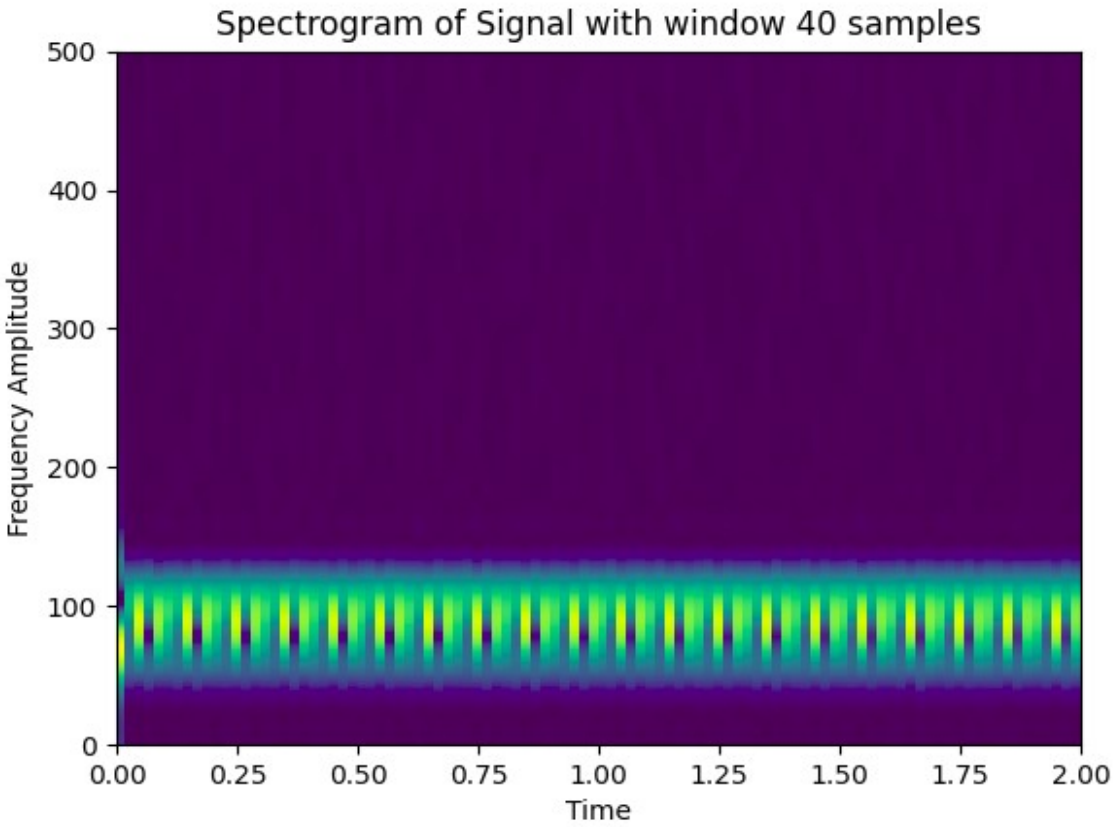
$$x(t) = 2\cos(2\pi 70t) + 3\sin(2\pi 100t) + 0.1v(t)$$

στο διάστημα [0, 2]s με συχνότητα δειγματοληψίας με αποτέλεσμα το παρακάτω σήμα:

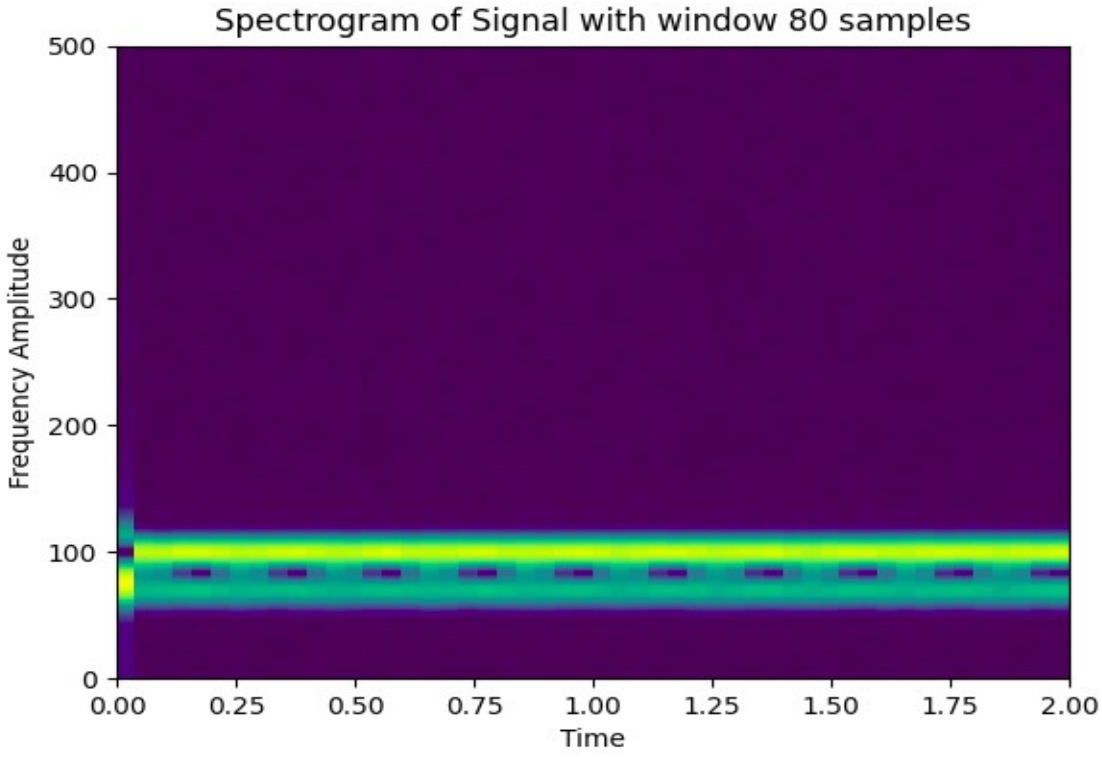


β,γ) Στη συνέχεια με χρήση της stft() της librosa υπολογίζουμε τον STFT του παραπάνω σήματος χρησιμοποιώντας παράθυρο μήκους 0.04s (που αντιστοιχεί σε 40 δείγματα) και overlap 20 δειγμάτων (το μισό μήκος του παραθύρου). Έπειτα με χρήση της pcolormesh() αναπαριστούμε τον STFT του παραπάνω σήματος. Επαναλαμβάνουμε τη παραπάνω διαδικασία με παράθυρα των 0.08s και 0.16s (που αντιστοιχούν σε 80 και 160 δείγματα αντίστοιχα) ενώ σε κάθε περίπτωση έχουμε overlap = window_length/2 (άρα overlap των 40 και 80 δειγμάτων αντίστοιχα). Κατά αυτό το τρόπο έχουμε τις παρακάτω απεικονίσεις:

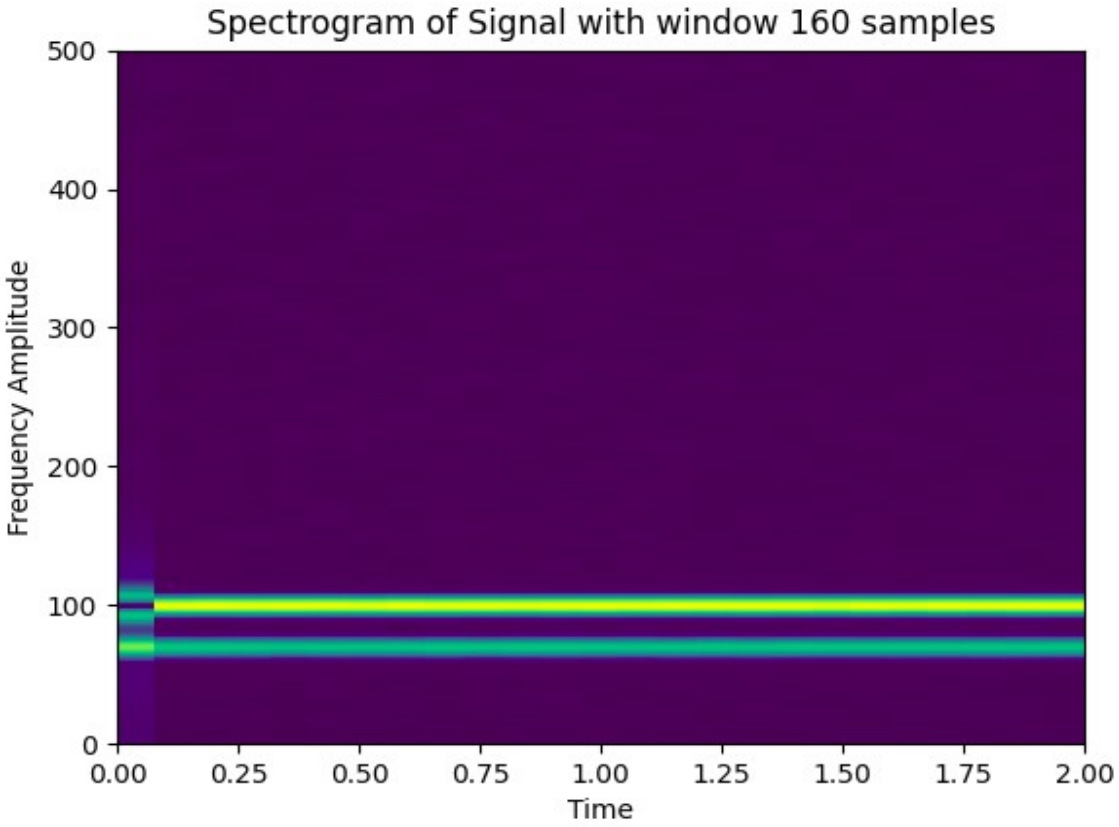
- Για window_length = 0.04s:



- Για window_length = 0.08s:



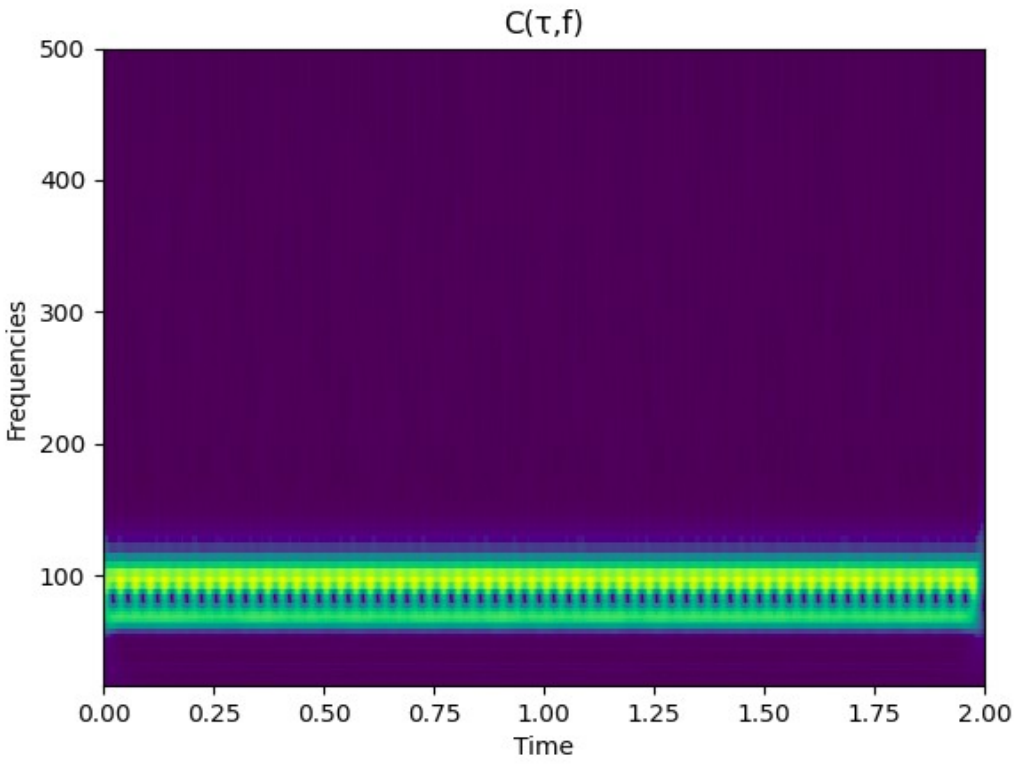
- Για window_length = 0.16s:



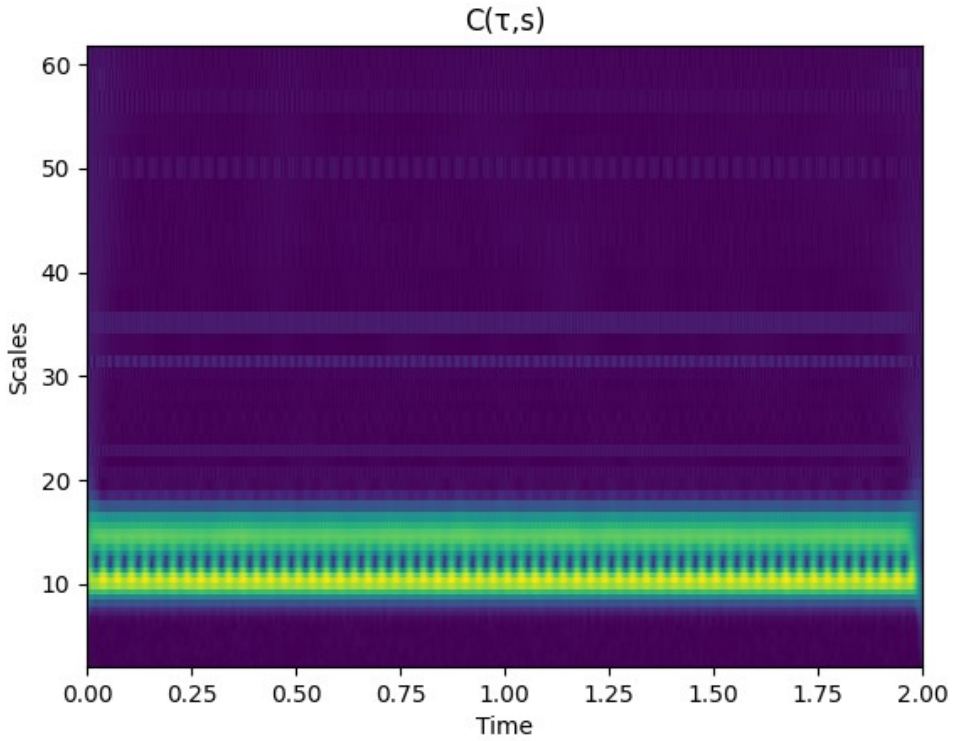
Είναι εμφανές το γεγονός ότι όσο αυξάνουμε το μέγεθος του παραθύρου αυξάνεται και η ποιότητα των συχνοτήτων του σήματος (στο 3ο διάγραμμα μπορούμε πλέον να πούμε πως έχουμε καθαρά 2 συχνότητες, μία στα 100 και μία στα 70Hz, στο 2ο διάγραμμα μπορούμε να πούμε πως έχουμε συχνοτικό περιεχόμενο στο σήμα μας που αντιστοιχεί στα 100Hz ενώ αυτό που αντιστοιχεί στα 70Hz δεν είναι τόσο καθαρό ενώ τέλος στο 1ο διάγραμμα μπορούμε να αποφασίσουμε για την περιοχή συχνοτήτων που έχει το σήμα αλλά όχι ακριβώς για τις συχνότητες που έχει!). Βέβαια αυξάνοντας το μέγεθος του παραθύρου , προκειμένου να έχουμε καλύτερη ακρίβεια στον άξονα της συχνότητας, θυσιάζουμε στο βωμό της ακρίβειας την χρονική (Αρχή της αβεβαιότητας).

δ) Ορίζοντας τις κατάλληλες κλίμακες (έχοντας κατά νου πως η κλίματα με τιμή 1 αντιστοιχεί στη συχνότητα δειγματοληψίας, η κλίμακα με τιμή 2 αντιστοιχεί στο μισό της κ.ο.κ.) υπολογίζουμε τον CWT του παραπάνω σήματος με χρήση της cwt() χρησιμοποιώντας το Complex Morlet Wavelet. Αναπαριστούμε τα πλάτη του cwt που προέκυψαν αξιοποιώντας και πάλι τη συνάρτηση pcolormesh με αποτέλεσμα να προκύψει η παρακάτω απεικόνιση:

- CWT(τ, f) αρχικού σήματος



- CWT(τ, s) αρχικού σήματος



Αξίζει να παρατηρηθεί ότι στην αναπαράσταση του CWT με scales έχουν αντιστραφεί τα μέγιστα. Αυτό οφείλεται στο γεγονός ότι όσο μεγαλώνει το scale τόσο μικραίνει η συχνότητα με αποτέλεσμα στο χαμηλότερο scale να αντιστοιχεί στα 100 Hz και το μικρότερο στα 70 Hz.

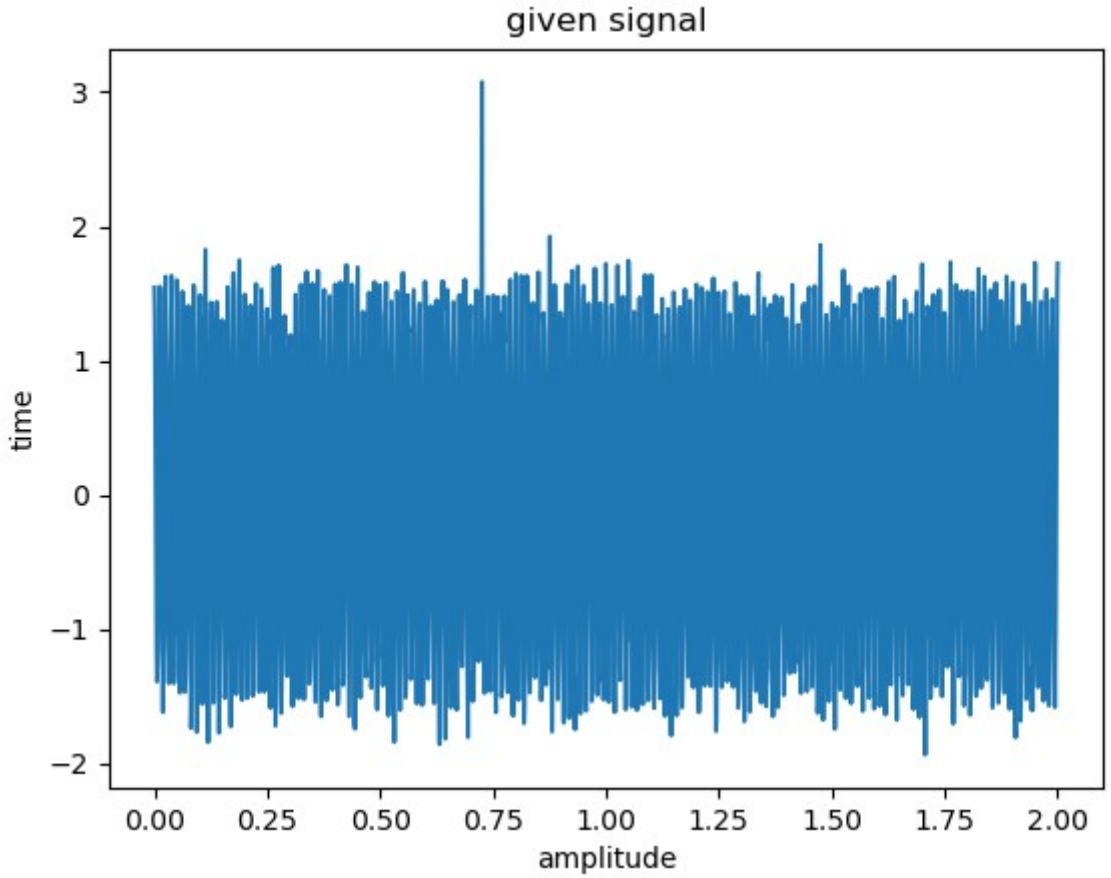
ε) Παρατηρούμε πως σε σύγκριση με τον STFT θα μπορούσε να χαρακτηριστεί θορυβώδης όμως μπορεί με αρκετά ικανοποιητική ακρίβεια να προσδιορίσει το συχνοτικό περιεχόμενο του σήματος.

2.2)

Α)Αρχικά, δειγματοληπτούμε το παρακάτω σήμα:

$$x(t) = 1.5\cos(2\pi 80t) + 0.15v(t) + 1.7(\delta(t - 0.725) - \delta(t - 0.900))$$

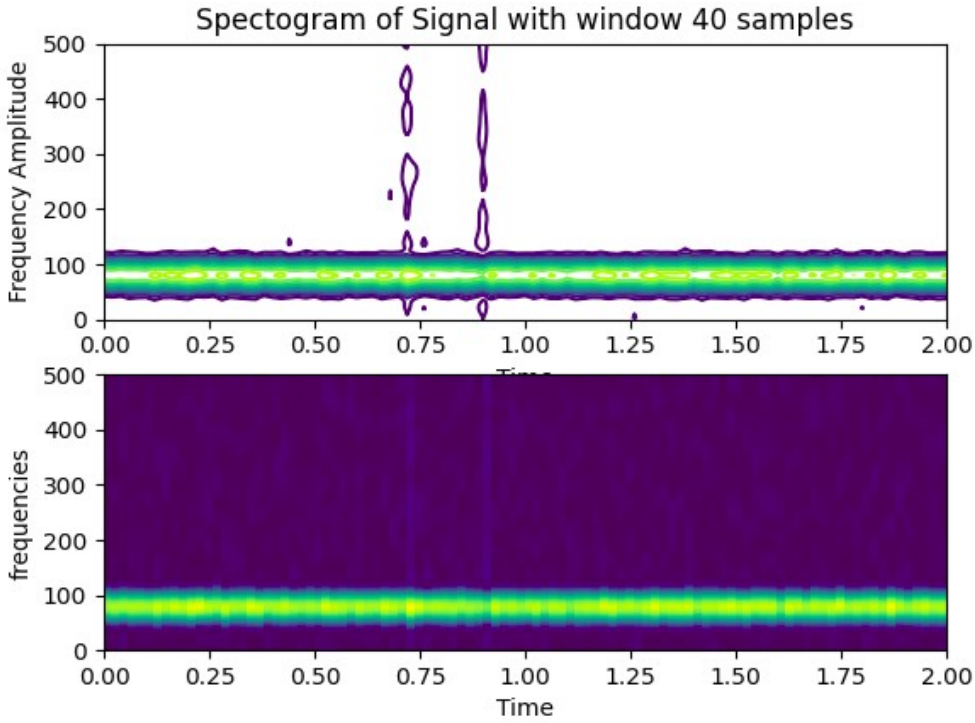
στο διάστημα [0, 2]s με συχνότητα δειγματοληψίας με αποτέλεσμα το παρακάτω σήμα:



Μπορεί εύκολα να διακρίνει κανείς πως την dirac στα 0.725s μπορούμε με το μάτι να τη διακρίνουμε πιο εύκολα.

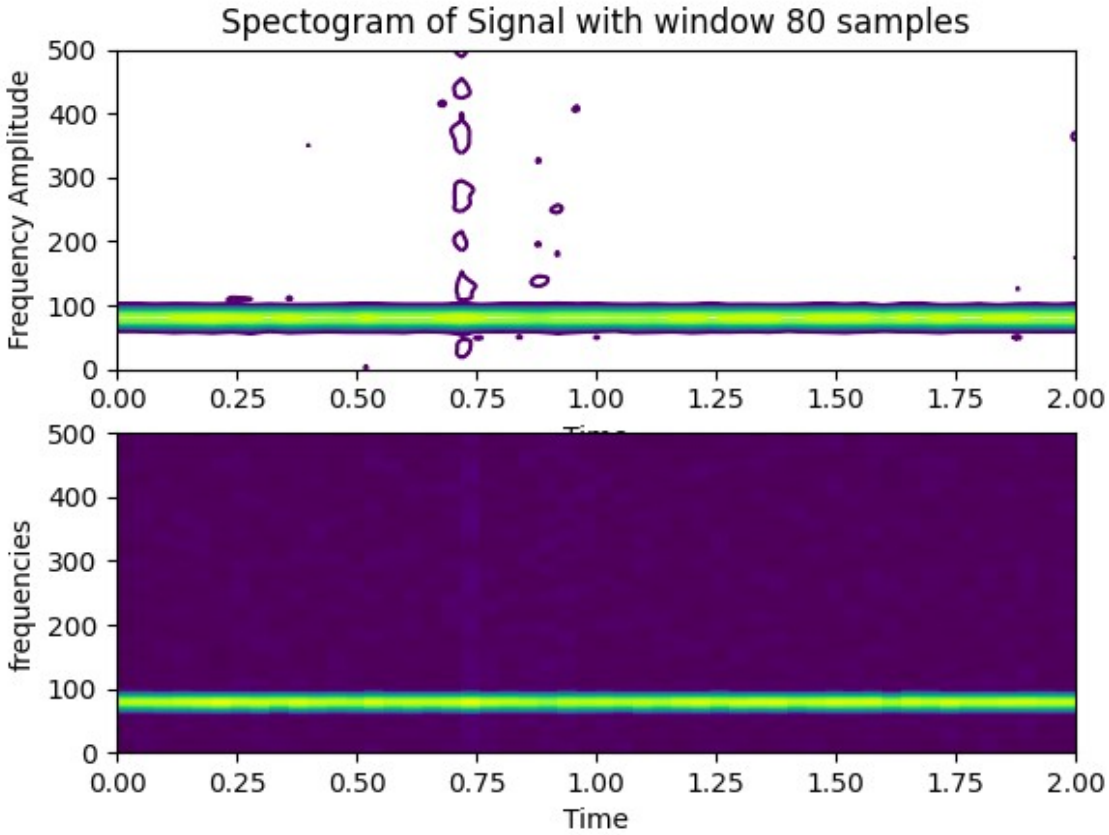
β, γ) Επαναλαμβάνοντας την ίδια διαδικασία με πριν και αναπαριστώντας με την contour και color mesh τον STFT του παραπάνω σήματος με τα αντίστοιχα παράθυρα και τα αντίστοιχα overlaps προκύπτουν τα παρακάτω:

- Για window_length = 0.04s:



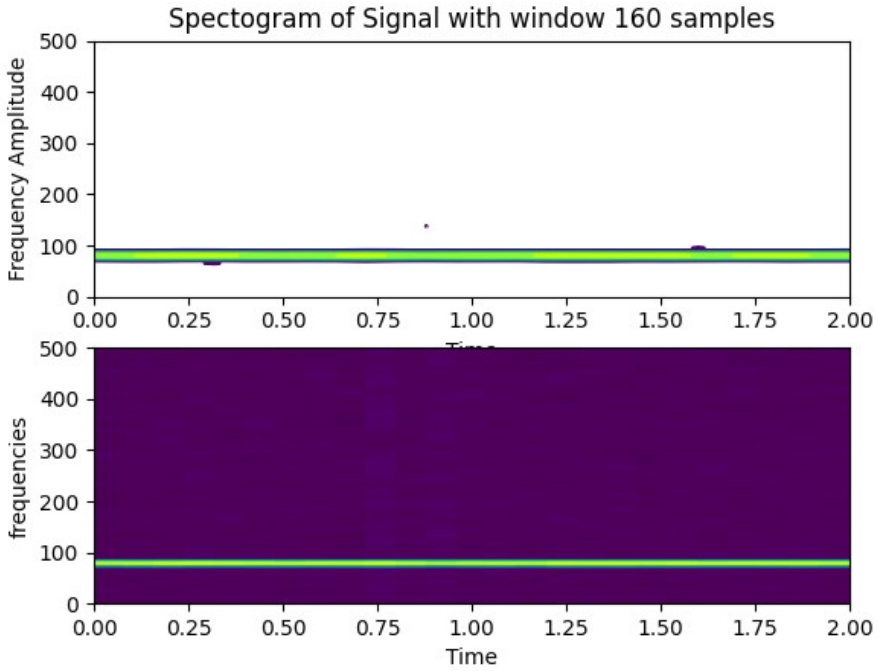
Έπειτα από προσθήκη της παραμέτρου περί ανάλυσης της contour σε 15 παρατηρούμε πως εμφανίζονται έντονα 2 αντίστοιχες dirac στην αναπαράσταση του stft. Από κάτω βάλαμε την αντίστοιχη αναπαράσταση του stft με χρήση της pcolormesh για πληρότητα. Εδώ, παρατηρούμε τις 2 dirac καθώς έχουμε μικρό παράθυρο και έτσι έχουμε καλή χρονική ακρίβεια.

- Για window_length = 0.08s:



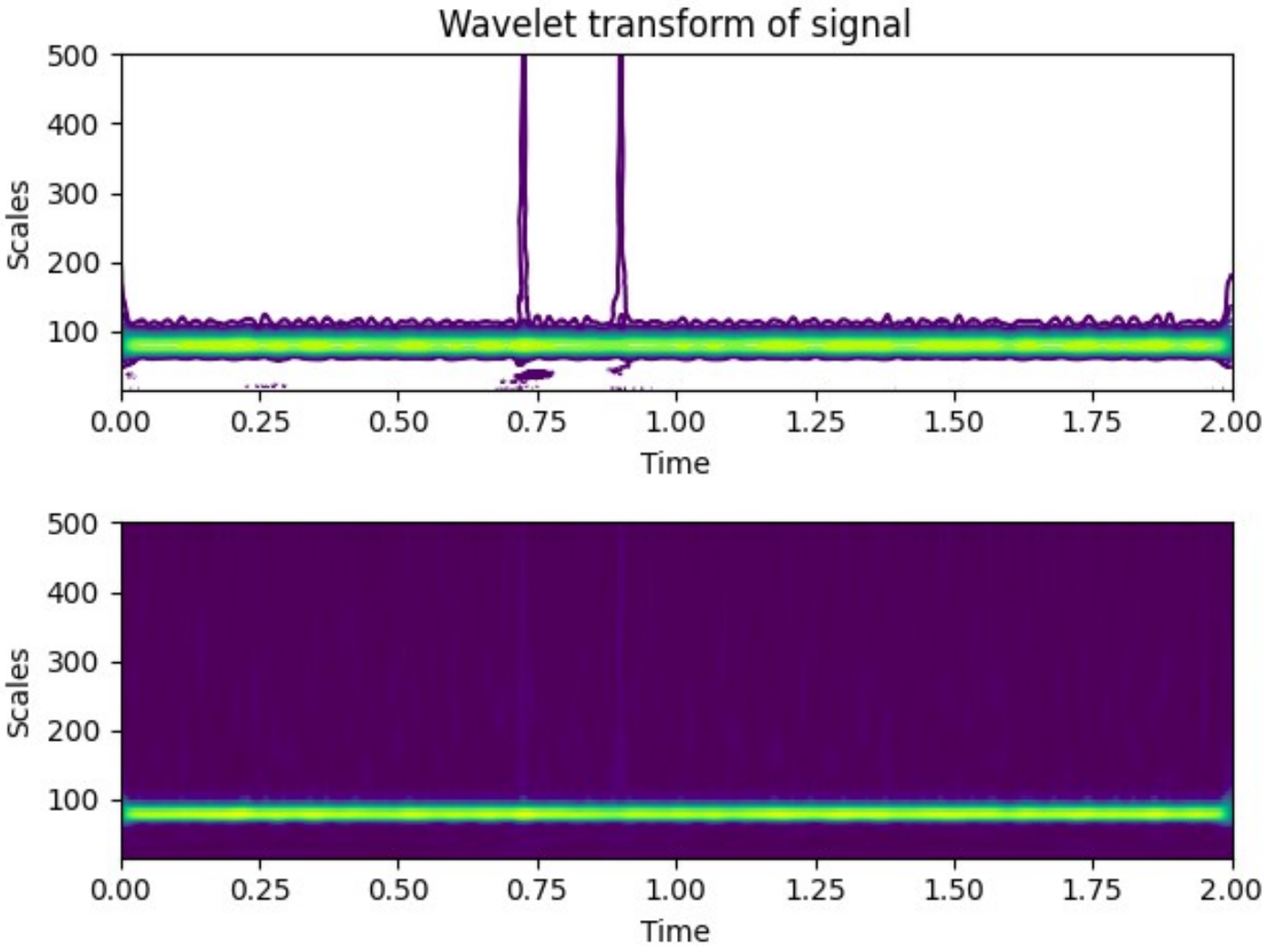
Εδώ, δεν είναι τόσο εμφανές τι συμβαίνει. Παρατηρούμε 2 ανωμαλίες στις χρονικές στιγμές 0.75 και 0.9s, ωστόσο δεν μπορούμε να ξέρουμε με σιγουριά ότι πρόκειται και τις dirac.

- Για window_length = 0.16s:



Εδώ παρατηρούμε ότι οι dirac εξαφανίζονται τελείως. Αυτό συμβαίνει καθώς χρησιμοποιήσαμε ακόμα μεγαλύτερο παράθυρο από πριν.

δ) Ακολουθώντας την ίδια λογική με το ερώτημα 2.1.ε και αναπαριστώντας το διάγραμμα με την contour (και colormesh) παίρνουμε ως αποτέλεσμα:



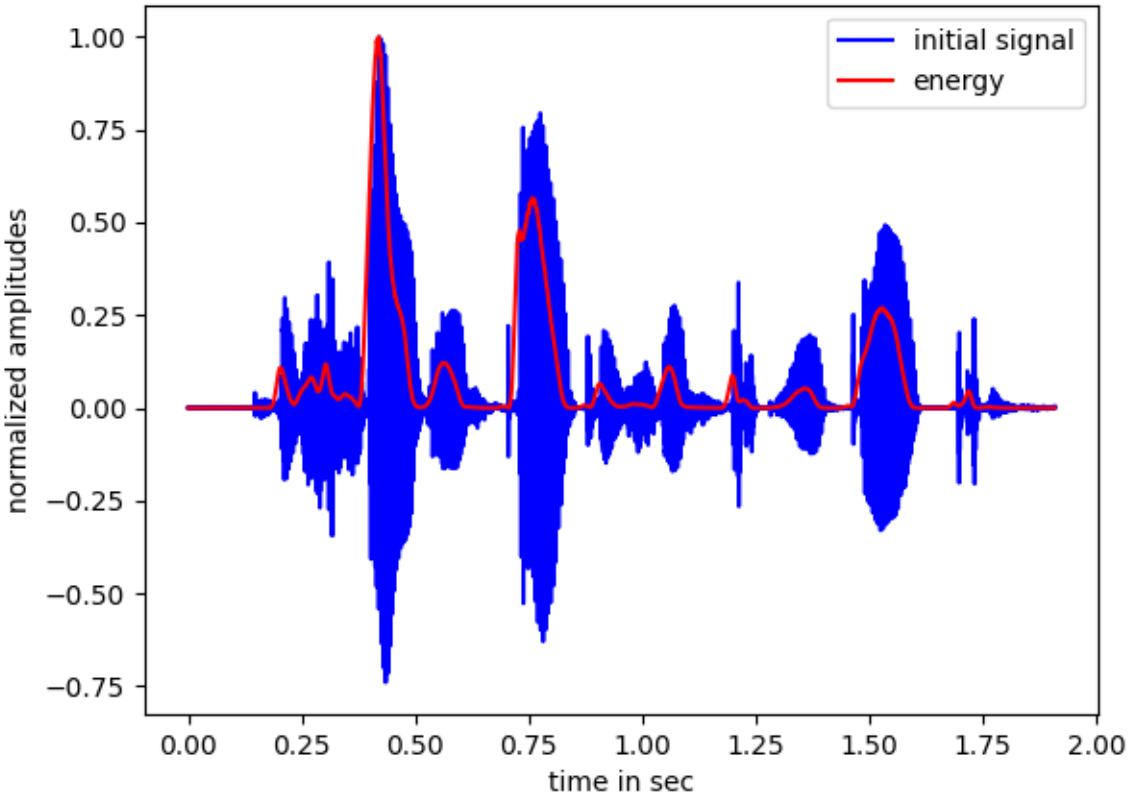
Εδώ, η ύπαρξη των 2 dirac είναι εμφανής στις 2 χρονικές στιγμές.

ε) Κλείνοντας, παρατηρούμε πως όσον αφορά τον stft χρειάζεται να επιλέξουμε τη χρυσή τομή στο μέγεθος του παραθύρου προκειμένου να αναγνωρίσουμε τις 2 dirac (χρονική ακρίβεια) και να μην χάσουμε την ακρίβεια στη συχνότητα. Αν επιλέξουμε μεγάλο παράθυρο τότε αυτό έχει ως αποτέλεσμα να χάνουμε σημαντικές πληροφορίες του σήματος μας στο πεδίο του χρόνου. Αντίθετα, ο cwt υπολογίζει δίχως πρόβλημα τις 2 dirac και κρατάει καλή ακρίβεια στη συχνότητα.

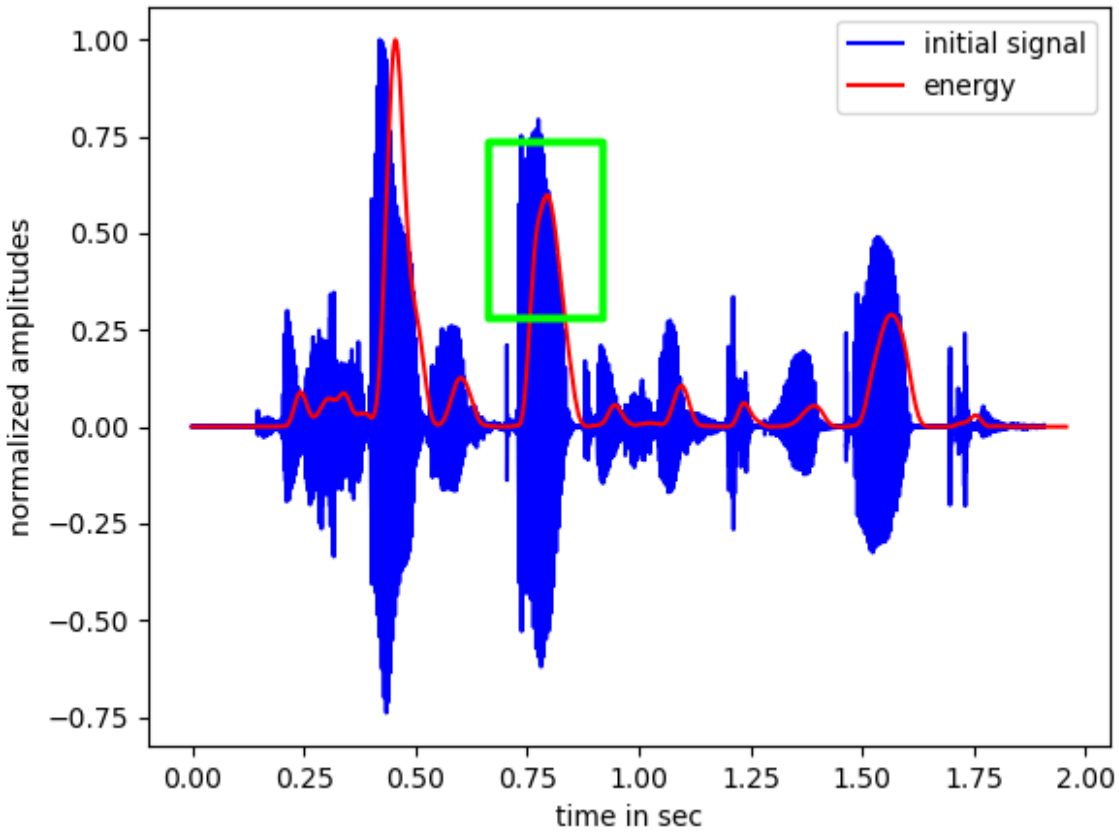
Μέρος 3

Αρχικά με χρήση της np.hamming() κατασκευάσαμε το παράθυρό μας.

3.1)
Όσον αφορά το speech utterance.wav η ενέργεια βραχέος χρόνου (Short Time Energy), σε συνδυασμό με το αρχικό μας σήμα, με μήκος παραθύρου 25 ms (400 δείγματα) προκύπτει :

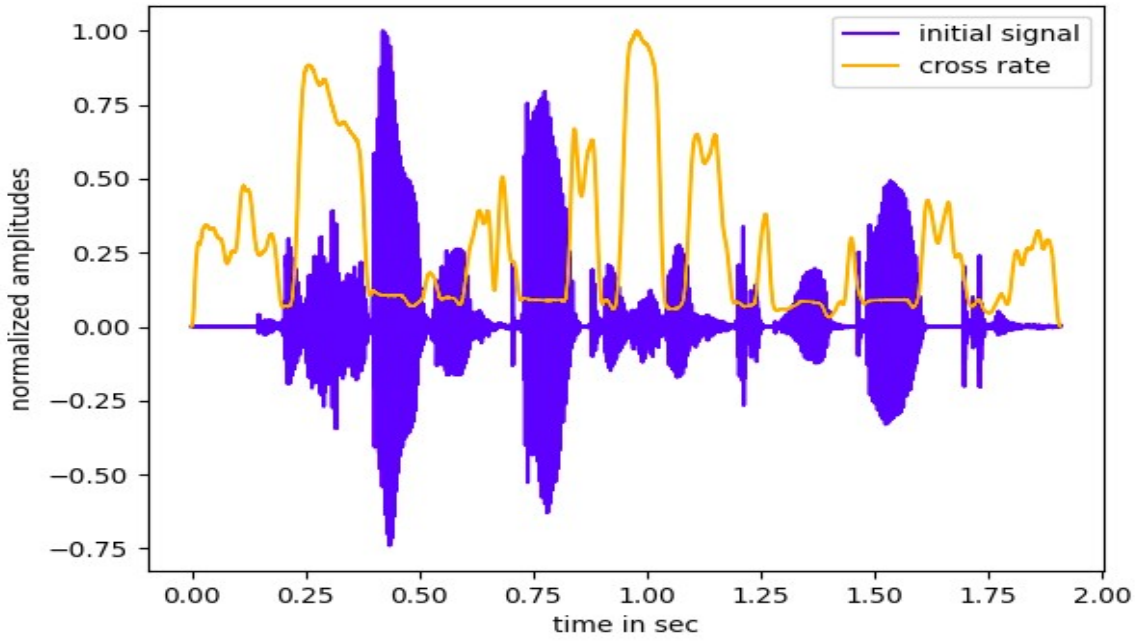


Μεγαλώνοντας το μήκος του παραθύρου στα 800 δείγματα παρατηρούμε ότι η ενέργεια βραχέως χρόνου εξομαλύνεται και χάνονται οι λεπτομέρειες γεγονός που το περιμέναμε (είναι εμφανής η εξομάλυνση στο πράσινο πλαίσιο):

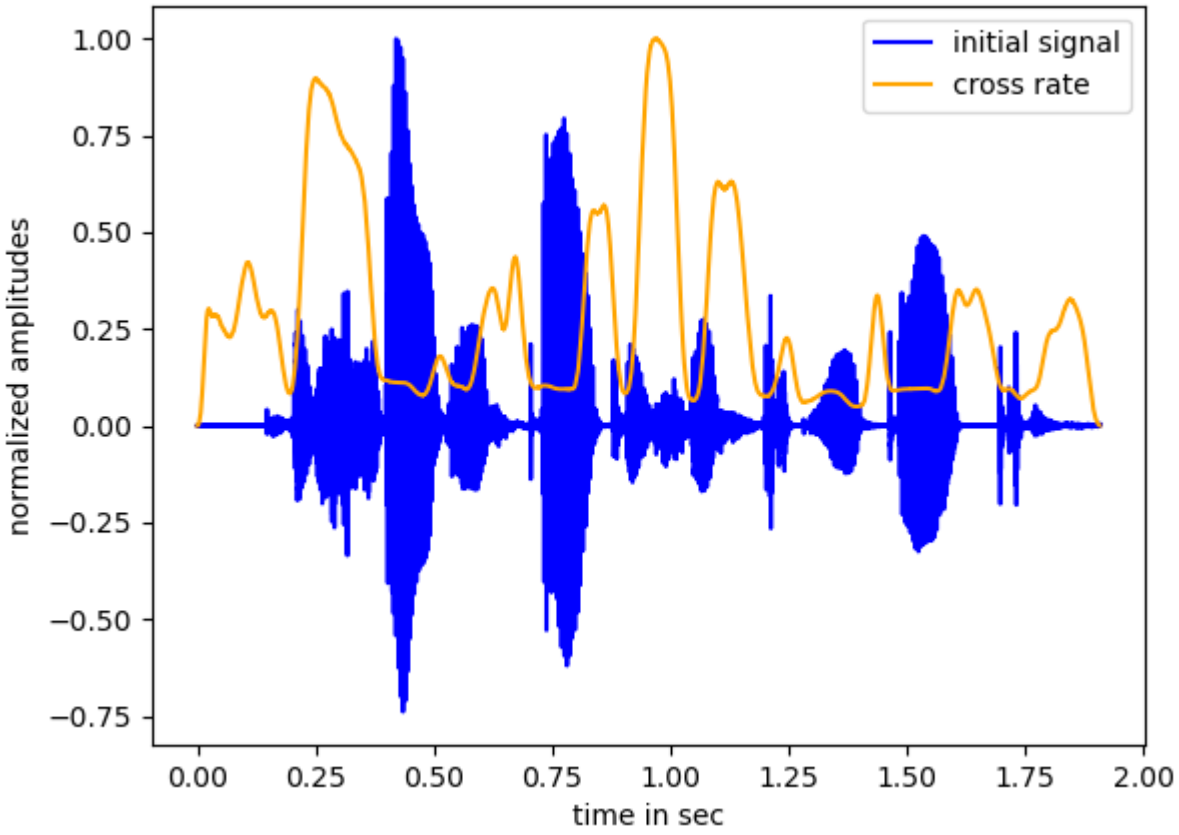


Εάν δεν κάναμε την κανονικοποίηση(που προφανώς όμως πρέπει να γίνει) η εξομάλυνση θα ήταν πολύ πιο έντονη και πολύ πιο εμφανής!

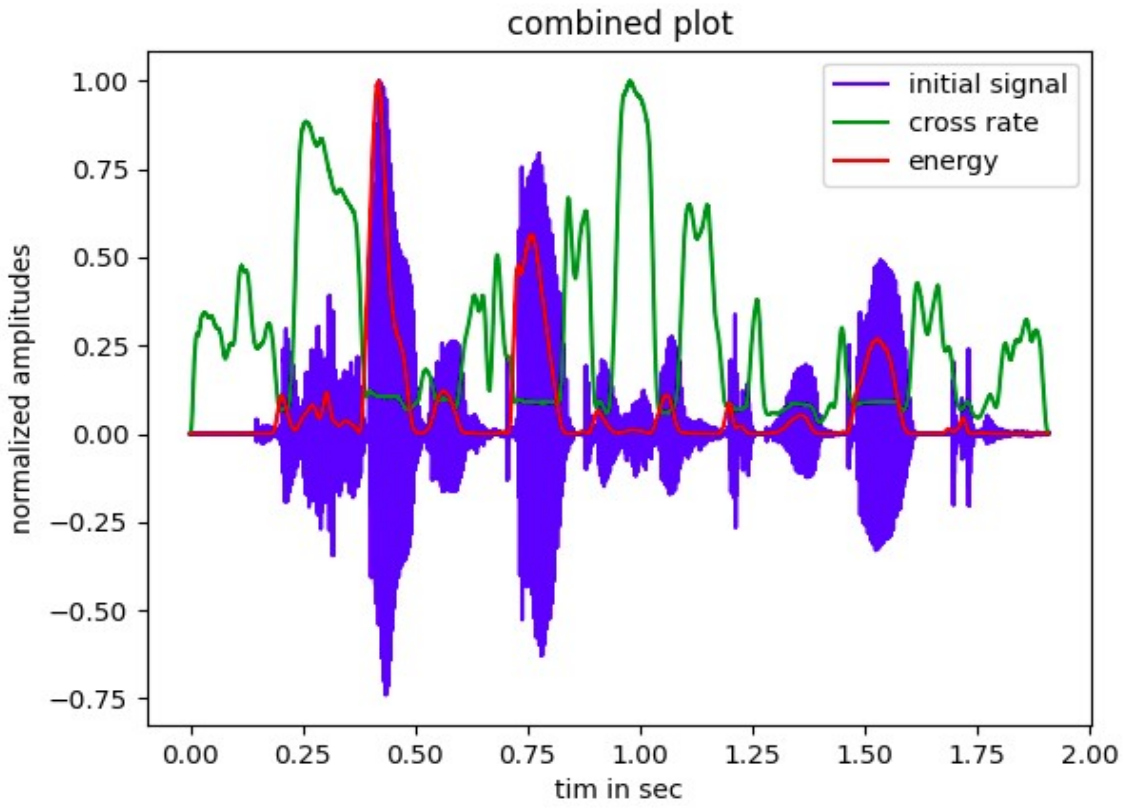
Όσον αφορά το Zero Cross Rate για παράθυρο 400 δειγμάτων προκύπτει:



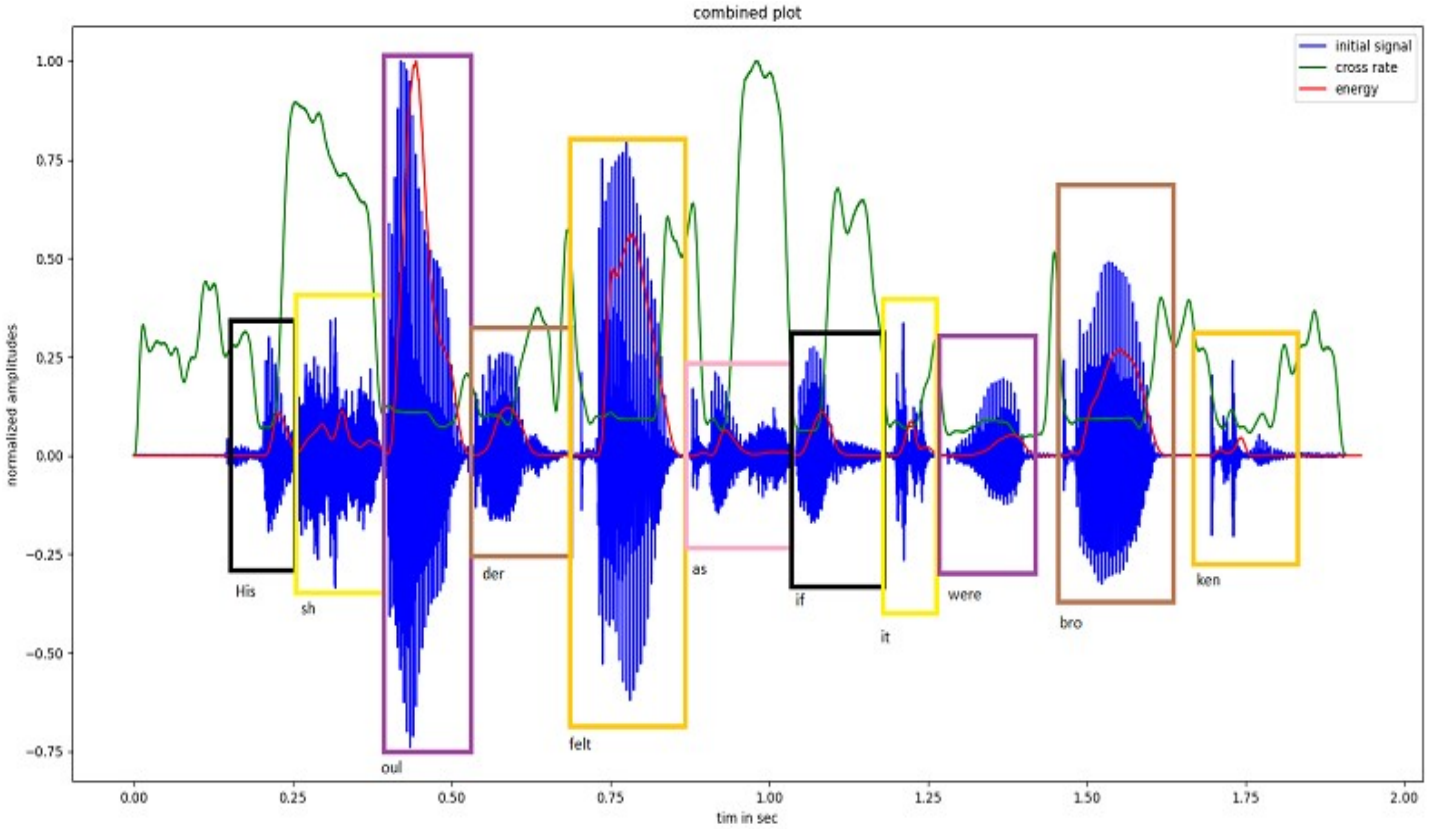
Αυξάνωντας το μέγεθος του παραθύρου στα 800 δείγματα παρατηρούμε ότι εξομαλύνεται το Zero Cross Rate και αυτό γίνεται φανερό στις κορυφές σε σύγκριση με το προηγούμενο διάγραμμα!



Signal and Short Time Energy and Zero Cross Rate:

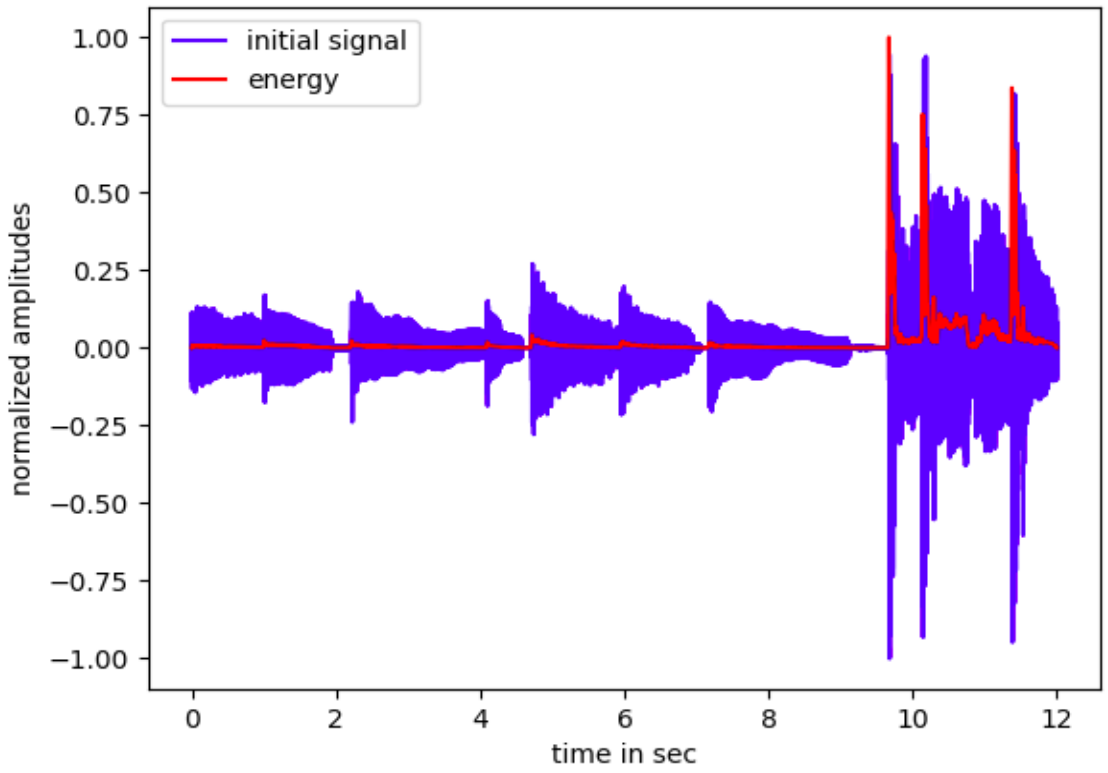


Χοντρικά, ισχύει πως όπου έχουμε μεγάλη ενέργεια έχουμε φωνήεν, ενώ όπου έχουμε μεγάλο zero crossing rate έχουμε σύμφωνο (μερικά σύμφωνα πλησιάζουν πολύ κοντά σαν ήχος το θόρυβο, όπως τα θ και σ).
Η φράση μας είναι, “His shoulder felt as if it were broken” και μπορούμε να την αποκωδικοποιήσουμε ως εξής: “hIs shOUldEr fElt As If It wEr(e) brOkEn”.
Με κεφαλαία έχουμε τα γράμματα που έχουν μεγάλη ενέργεια, με μικρά όσα έχουν μεγάλο crossing rate και με παρενθέσεις όσα δεν προφέρονται.

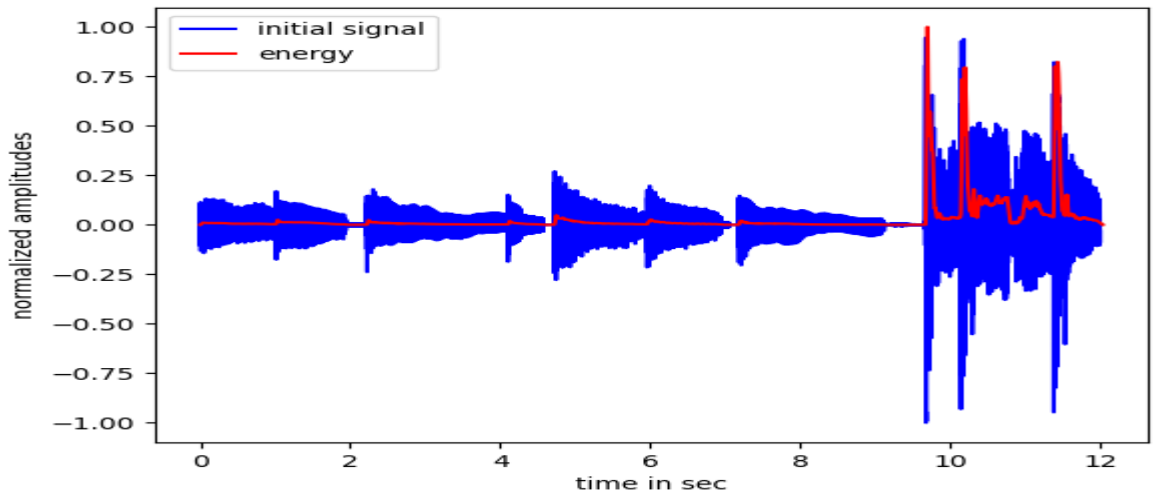


3.2)
Όσον αφορά το speech music.wav η ενέργεια βραχέους χρόνου (Short Time Energy) προκύπτει:

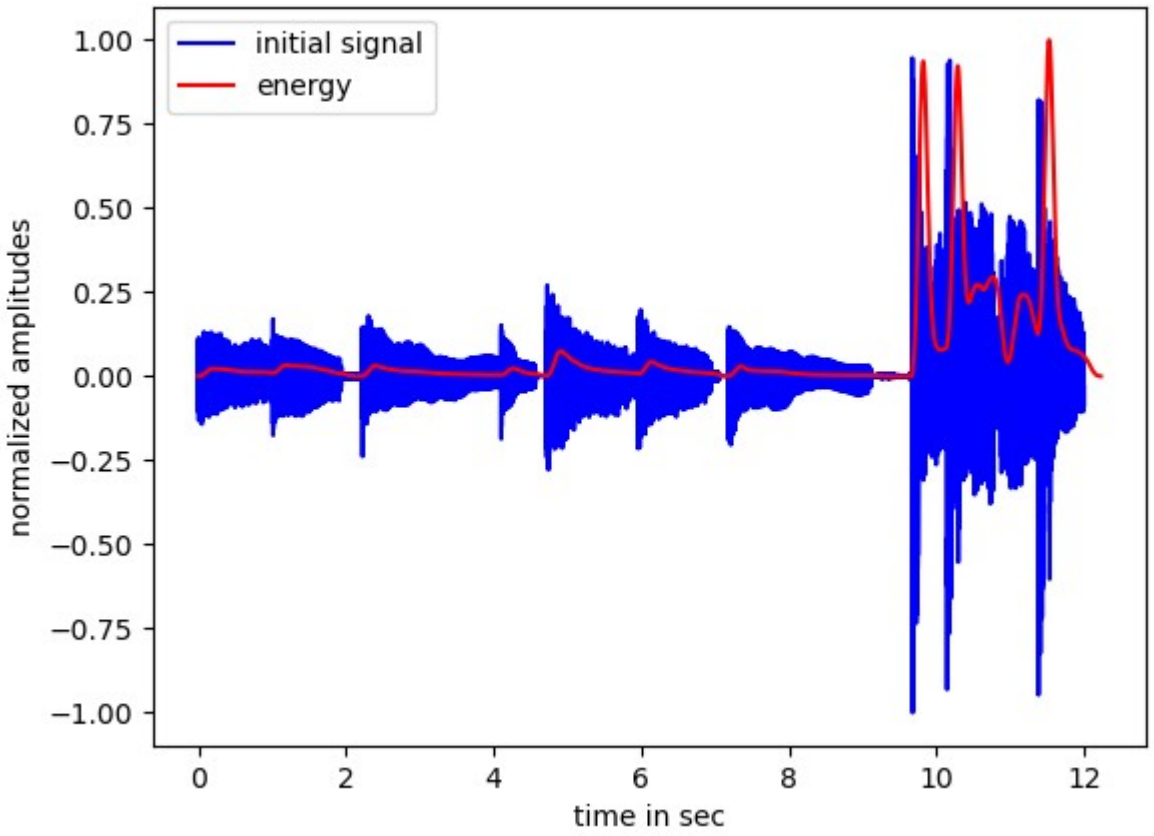
- Με μήκος παραθύρου 25ms (που αντιστοιχεί σε 400 δείγματα)



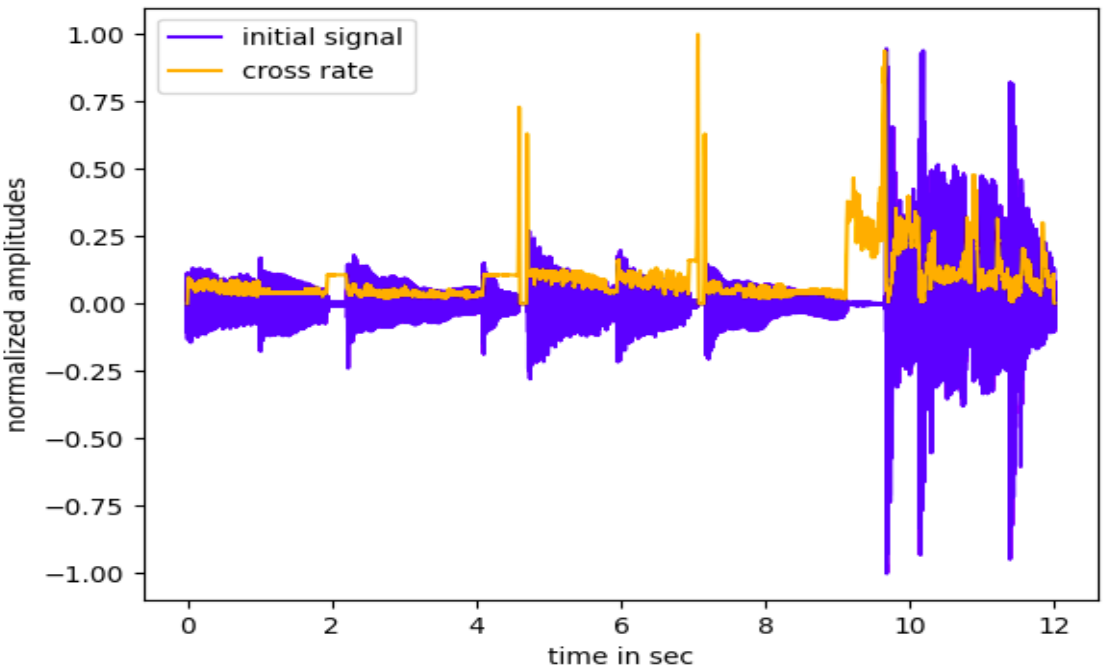
Μεγαλώνοντας το μήκος του παραθύρου στα 800 δείγματα παρατηρούμε ότι η ενέργεια βραχέως χρόνου εξομαλύνεται και χάνονται οι λεπτομέρειες γεγονός που το περιμέναμε (είναι εμφανής η εξομάλυνση στο πράσινο πλαίσιο) μόνο που σε αυτή τη περίπτωση δεν είναι έντονο λόγο της μορφής τους σήματος (έπρεπε να θέσω αρκετά μεγάλα μήκη παραθύρων της τάξης του 3000-6000 για να δω αρκετά αισθητές εξομαλύνσεις):



Για παράθυρο μήκους 5000 δειγμάτων (συμπληρωματικά και μόνο) είναι πλέον πολύ αισθητή η εξομάλυνση**:**

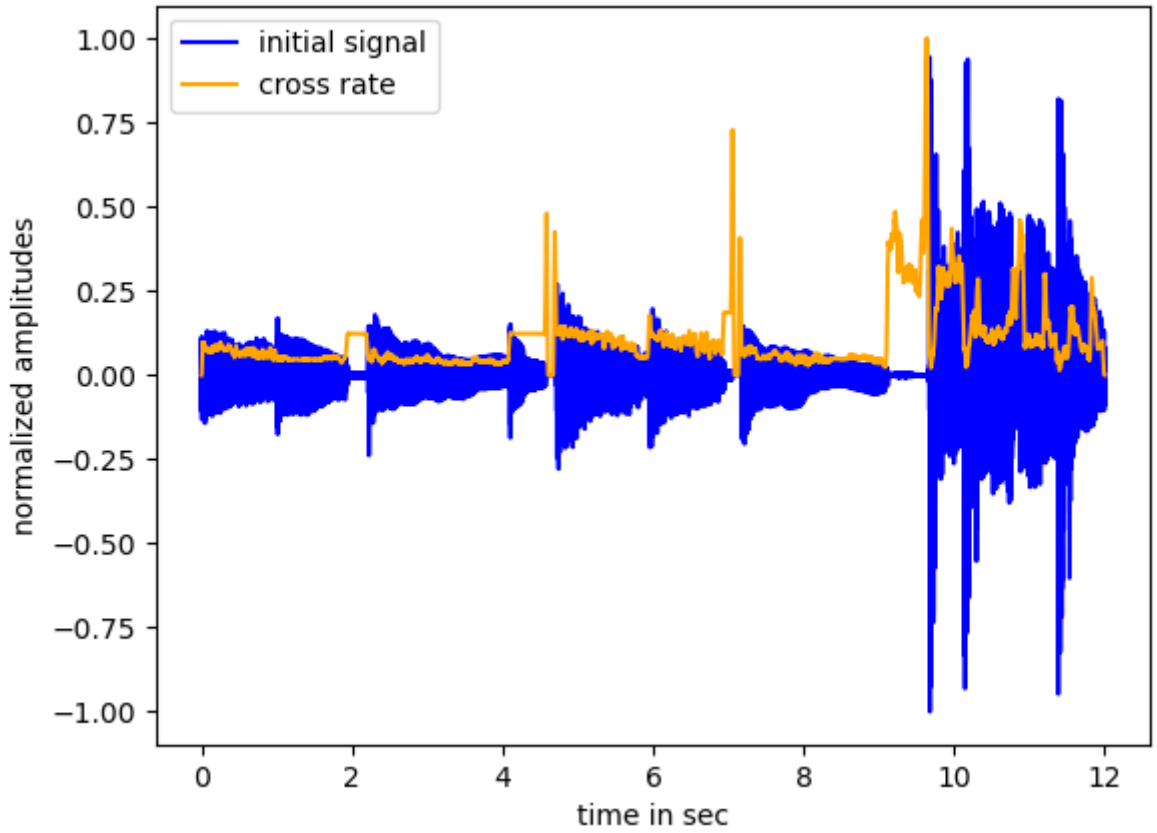


Όσον αφορά το Zero Cross Rate για παράθυρο 400 δειγμάτων προκύπτει:

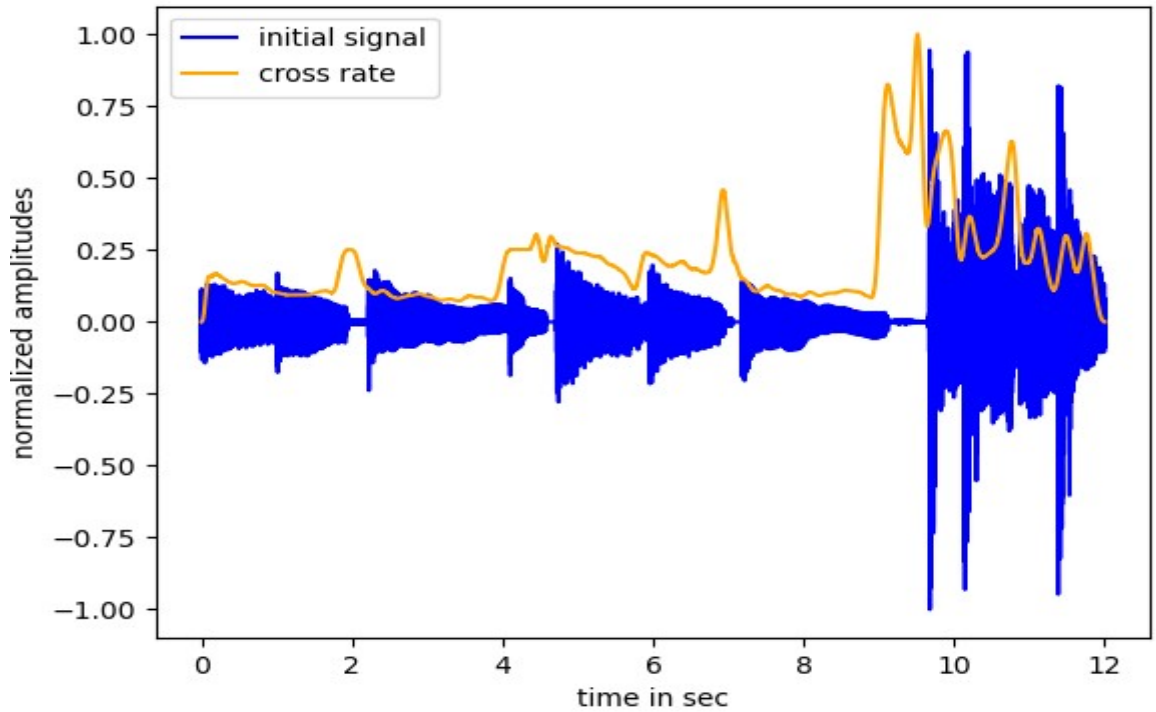


Όμοια με την ενέργεια για να παρατηρηθούν “με γυμνό μάτι” εξομαλύνσεις πρέπει να πάρουμε αρκετά μεγάλο παράθυρο!

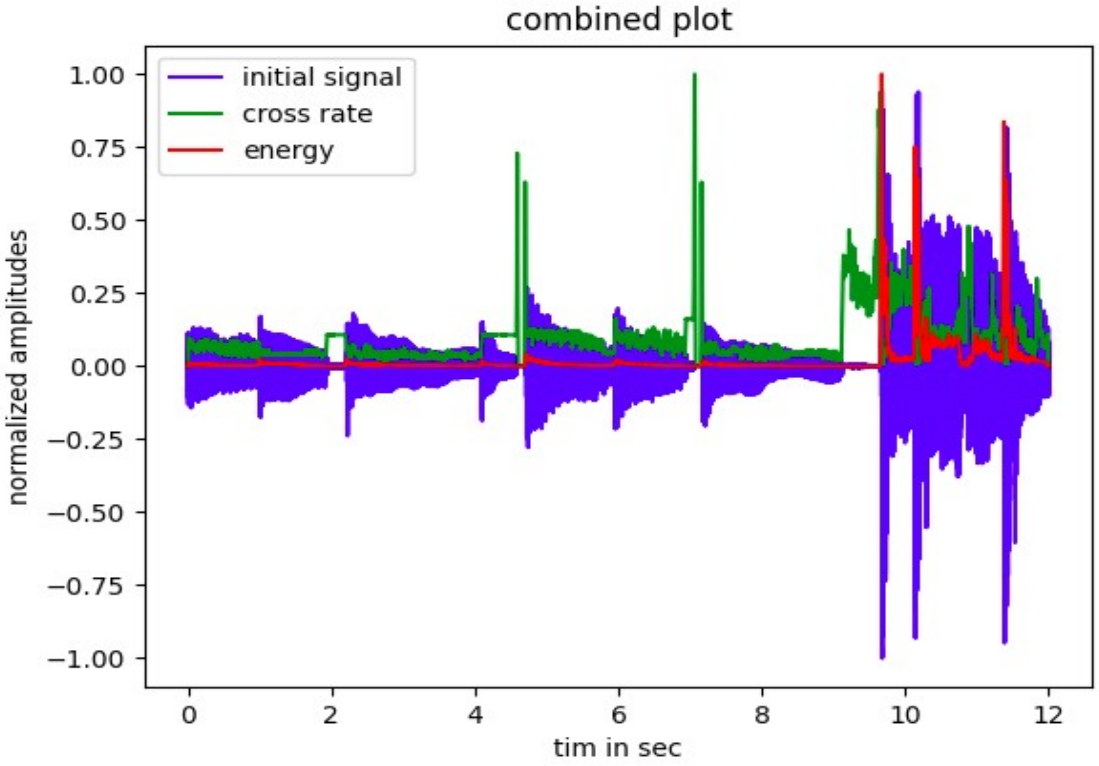
Για παράθυρο 800 δειγμάτων έχουμε:



Για παράθυρο μήκους 5000 δειγμάτων (συμπληρωματικά και μόνο) είναι πλέον πολύ αισθητή η εξομάλυνση:

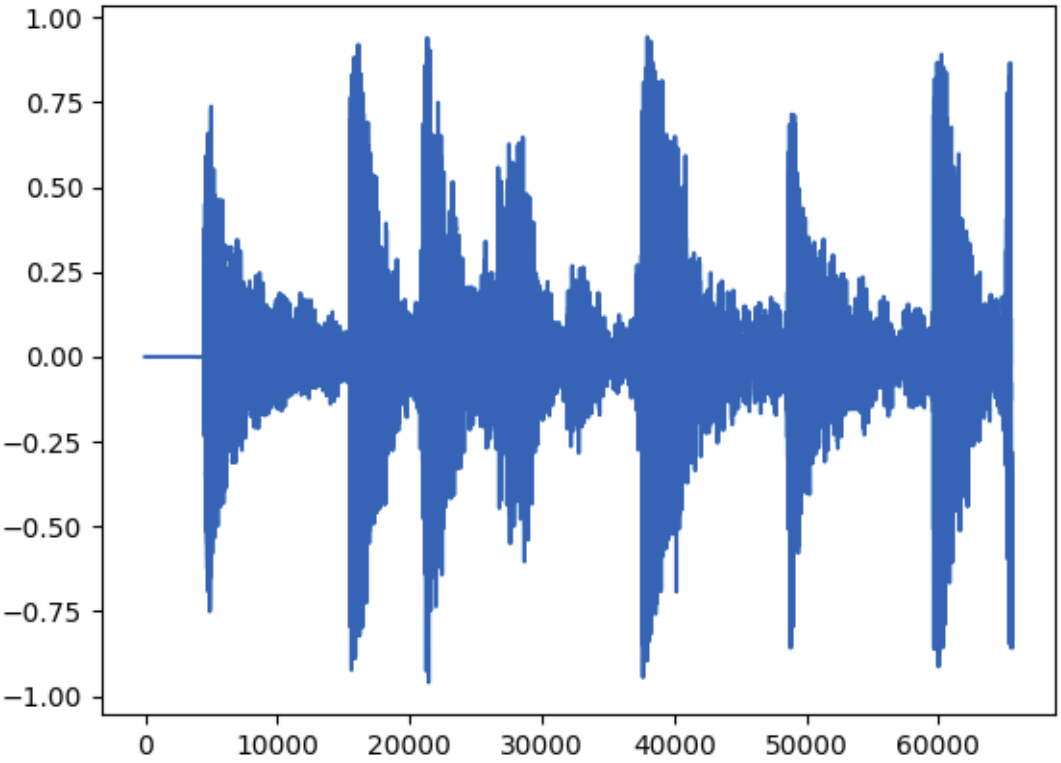


Signal and Short Time Energy and Zero Cross Rate:


combined plot
initial signal
cross rate
energy
normalized amplitudes
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10
12
tim in sec

Μέρος 4

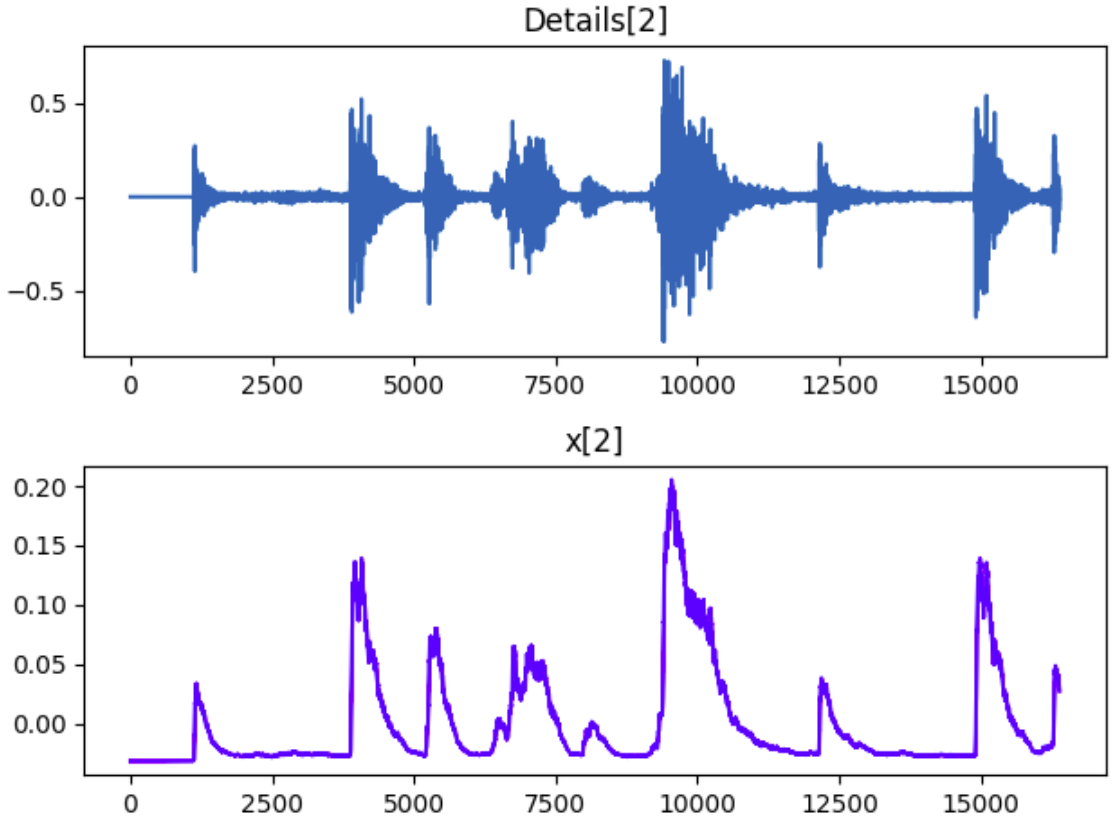
4.1) Αρχικά, από το όλο σήμα κρατούμε ένα παράθυρο μήκους 2^16 (μεταξύ 10000 και 2^16 + 10000) και το πλοτάρουμε.

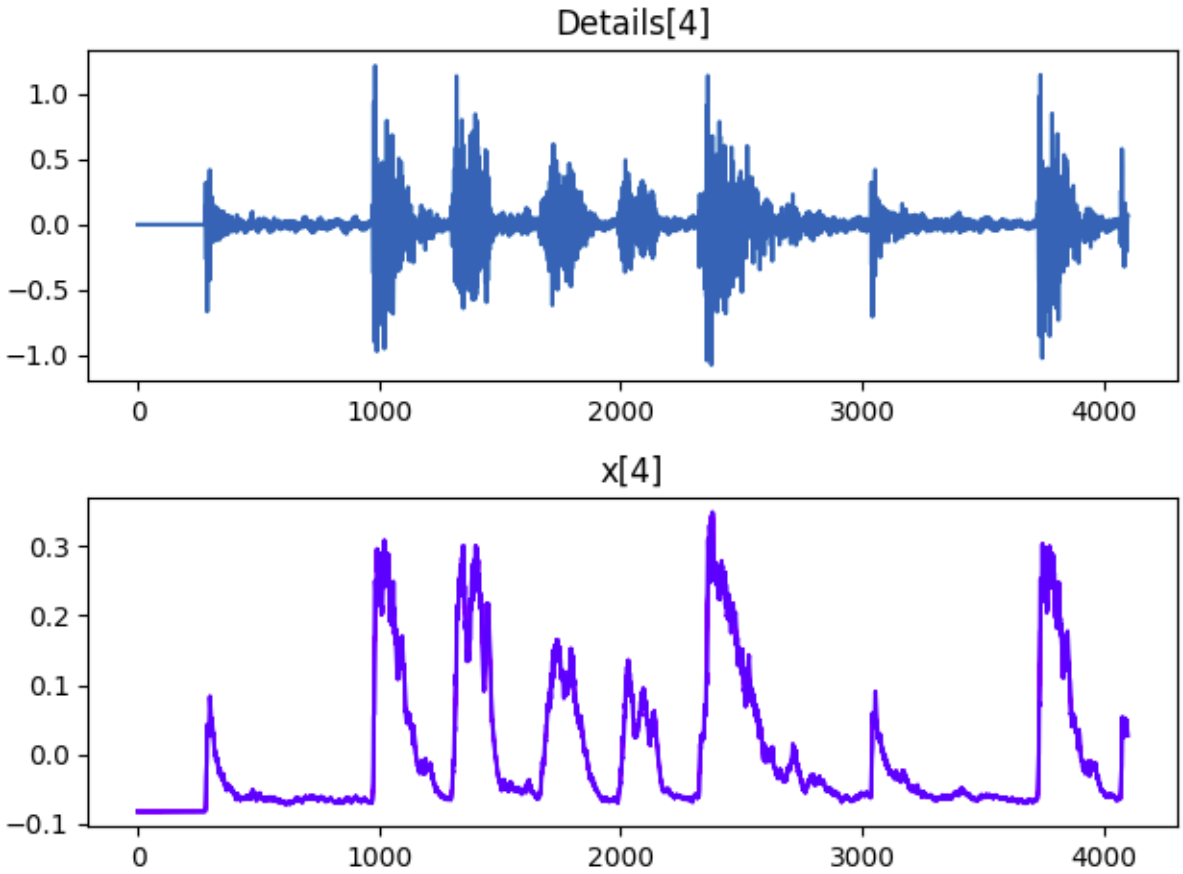


Η παλμική περιοδικότητα του φαίνεται να είναι κάθε (35000 – 10000 =) 25000 δείγματα επειδή κάθε 25000 δείγματα παρατηρούμε να έχουμε ολοκληρωμένο μουσικό μέτρο. Στο συγκεκριμένο κομμάτι είναι εύλογο να θεωρήσουμε 4 τέταρτα μέτρημα. Στο πρώτο μέτρο παρατηρούμε 2 διαδοχικούς δυνατούς μετρικούς χτύπους και 2 πιο χαμηλούς. Στο δεύτερο μέτρο παρατηρούμε έναν δυνατό (στα 35000) έναν αδύναμο (στα 40000), έναν δυνατό ξανά (στα 45000), και άλλον έναν αδύναμο κοντά στα 50000 δείγματα. Έτσι, δεδομένου πως σε ένα τραγούδι επικρατεί θεωρητικά το ίδιο tempo, στο τραγούδι αυτό έχουμε παλμική περιοδικότητα στα περίπου 25000 δείγματα.

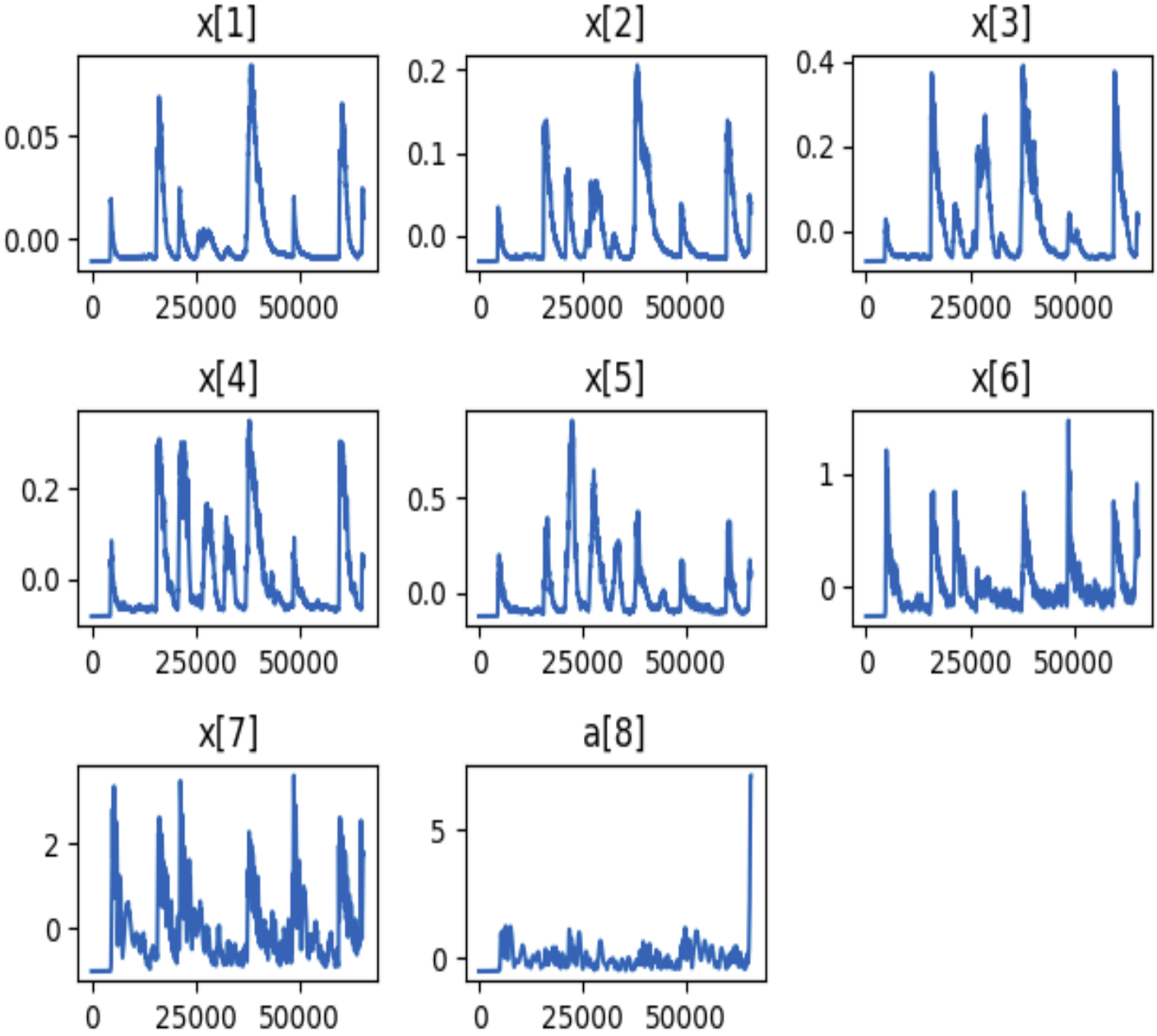
4.2) Σε αυτό το ερώτημα χρησιμοποιήσαμε την dwt() προκειμένου να αναλύσουμε το σήμα σε 7 λεπτομερείς συνιστώσες (details) και μια χαμηλόσυχνη συνιστώσα (approximation) αξιοποιώντας το Daubechies Wavelet.

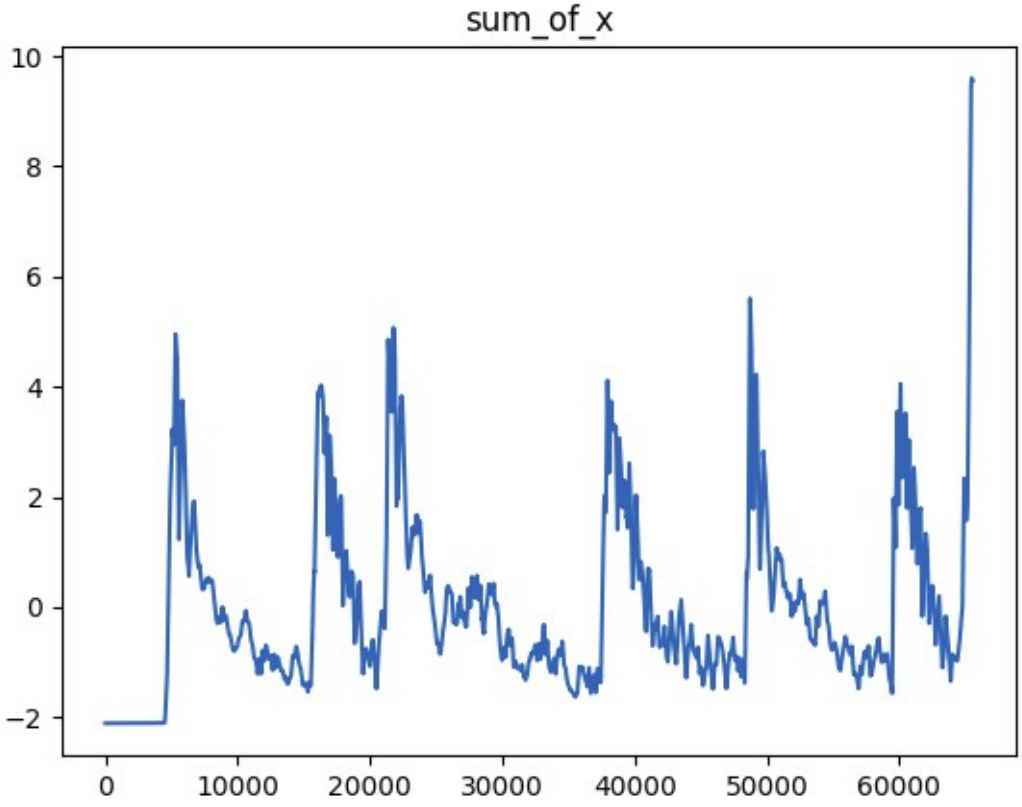
4.3 α) Αρχικά περάσαμε,

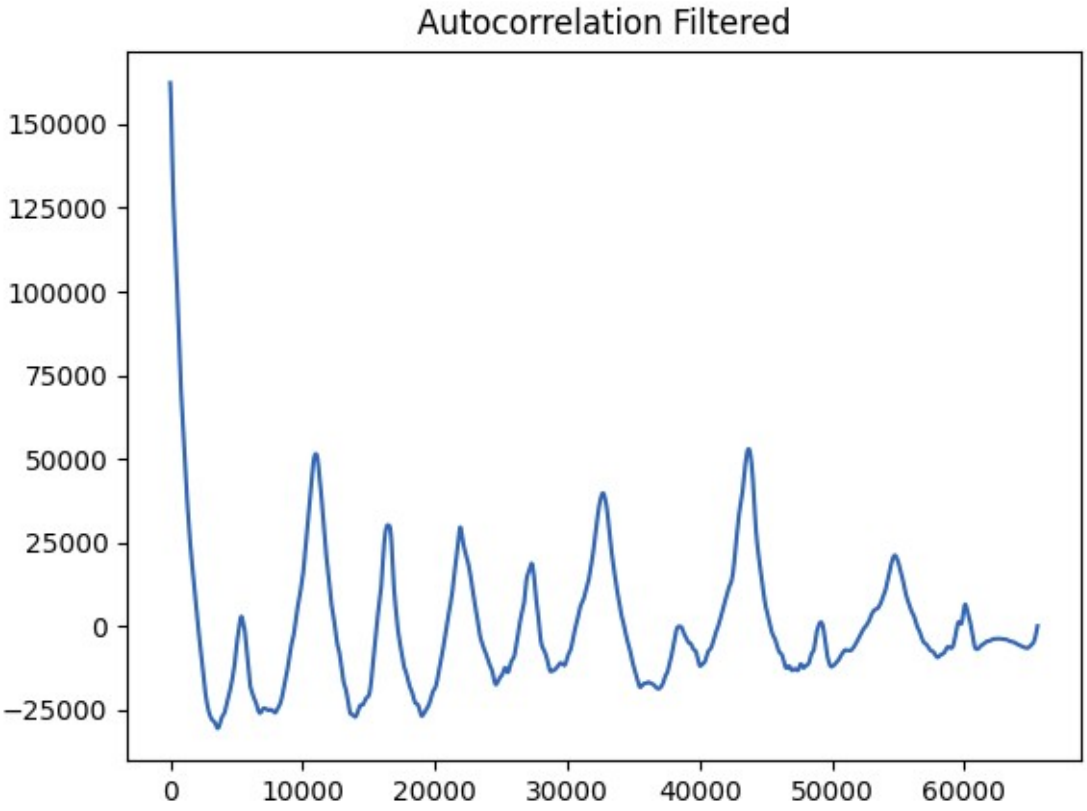




4.4) Αθροίζοντας τις περιβάλλουσες συνιστώσες του dwt αφού του εφαρμόσαμε γραμμική παρεμβολή προκειμένου οι πίνακες να έχουν ίδιο μέγεθος, και τέλος περνώντας από autocoreclation προκύπτει το παρακάτω διάγραμμα. Στο ίδιο διάγραμμα υπάρχει και η εξομάλυνση του με το φίλτρο που το περνάμε στο 4.5



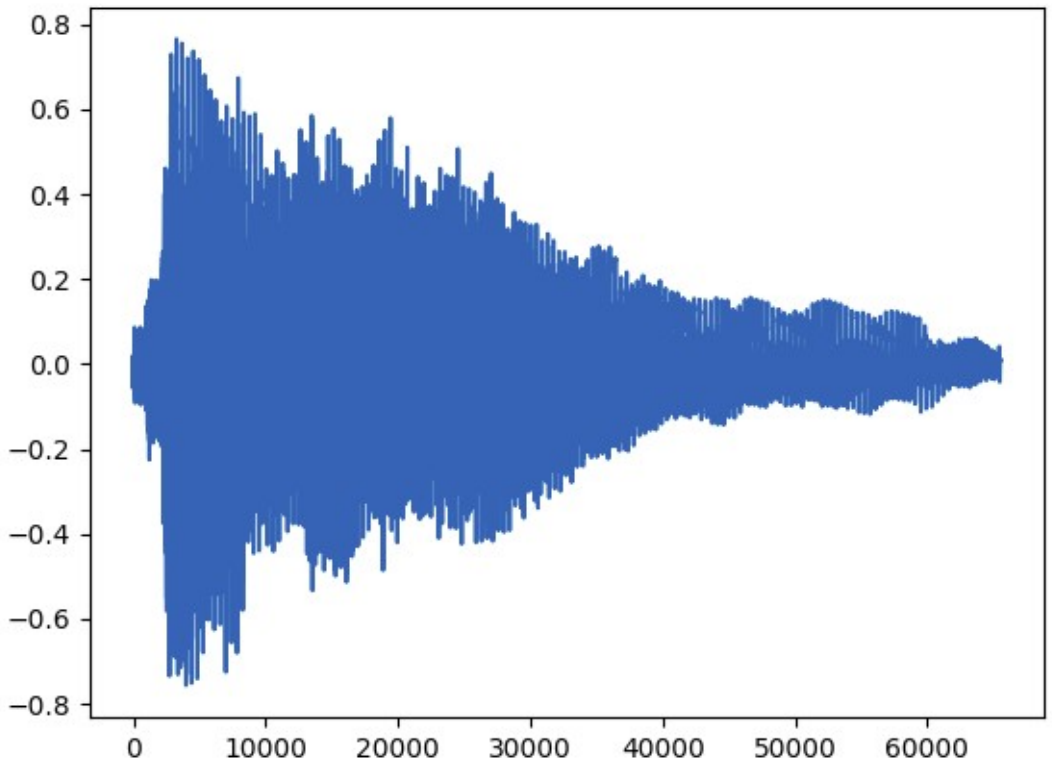

sum_of_x
10
8
6
4
2
0
−2
0
10000
20000
30000
40000
50000
60000

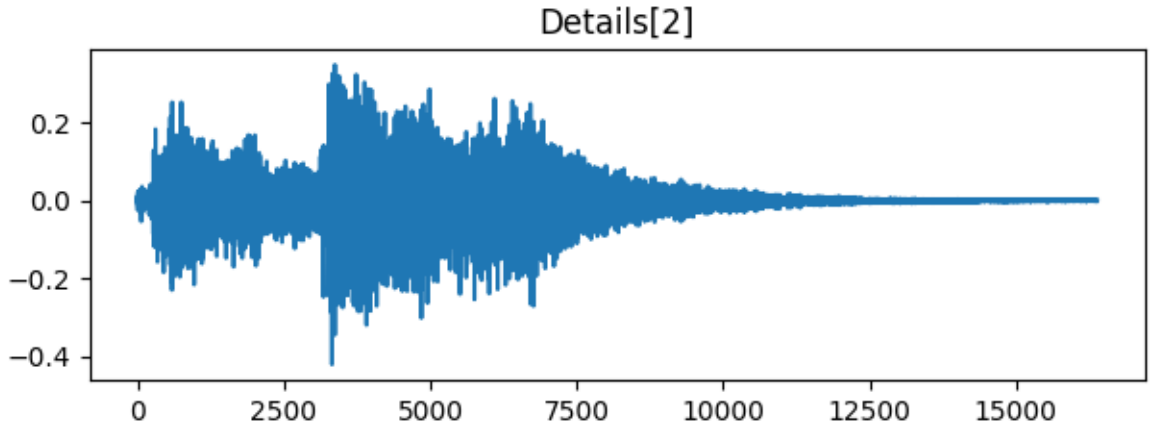

Autocorrelation Filtered
150000
125000
100000
75000
50000
25000
0
−25000
0
10000
20000
30000
40000
50000
60000

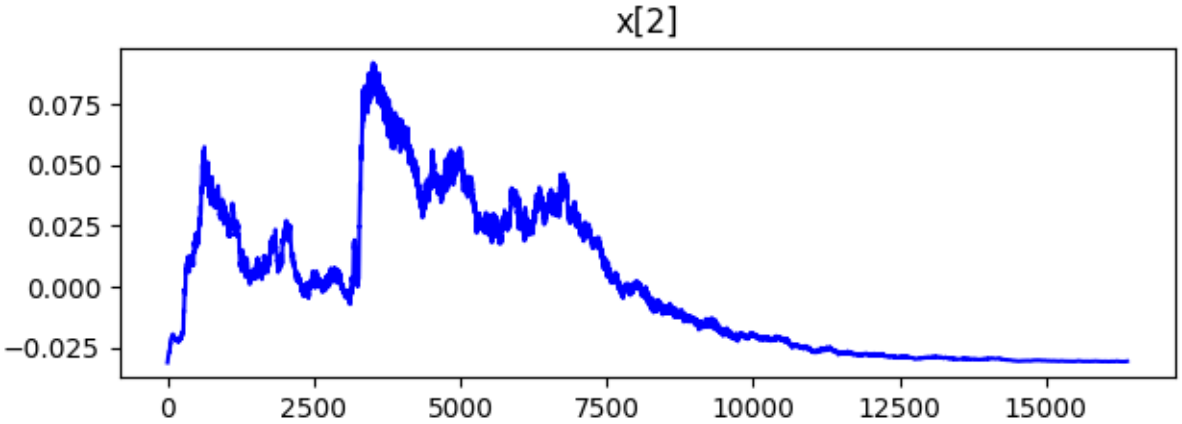
Μας εμφανίζει τα εξής (possible bpm, index, amplitude)

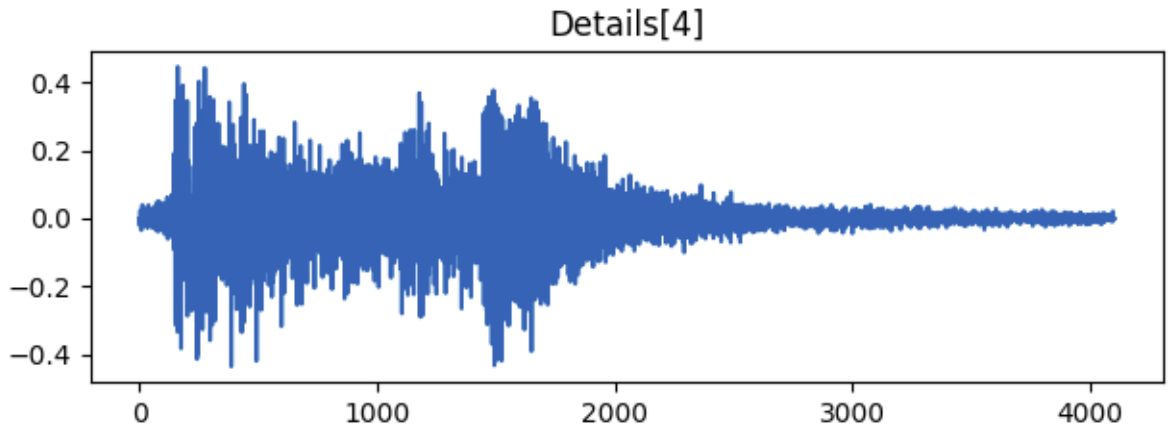
[186, 173, 120, 91, 80, 60]
[491 989 4394 7771 9827 15299]
[-24577.89154523822, -25011.462699714582, 51439.53736799714, -23692.561958106853, 30175.603756209915, 29530.891332785763]
Από αυτά, κρατάμε αυτό με το μέγιστο πλάτος το οποίο είναι το 120 bpm.
Επιπλέον, παρατηρήσαμε ότι αλλάζοντας το παράθυρο δειγματοληψίας αλλάζει και εν τέλη το τελικό bpm που υπολογίζεται. Ουσιαστικά δηλαδή αλλάζει το μέγιστο στο auto correlation diagram. Ουσιαστικά ο χτύπος μου μετράμε στην μουσική μπορεί να αλλάζει κρατώντας όμως ένα καθολικό tempo και εξαρτάται από το είδος νότας που χρησιμοποιείτε. Δηλαδή αν στην αρχή έχουμε μισά σε 60bpm και το γυρίσουμε σε τέταρτα θα μπορούμε να πούμε ότι μετράμε 120bpm. Αυτό οφείλεται στις τονικότητες. Συνήθως, οι μουσικοί προκειμένου να κρατούν σταθερό tempo χτυπάνε την πρώτη νότα στο μέτρο πιο δυνατά από τις υπόλοιπες (και μερικές φορές και ενδιάμεσες νότες) και για αυτό μεταβάλετε εν τέλη το αποτέλεσμα μας.

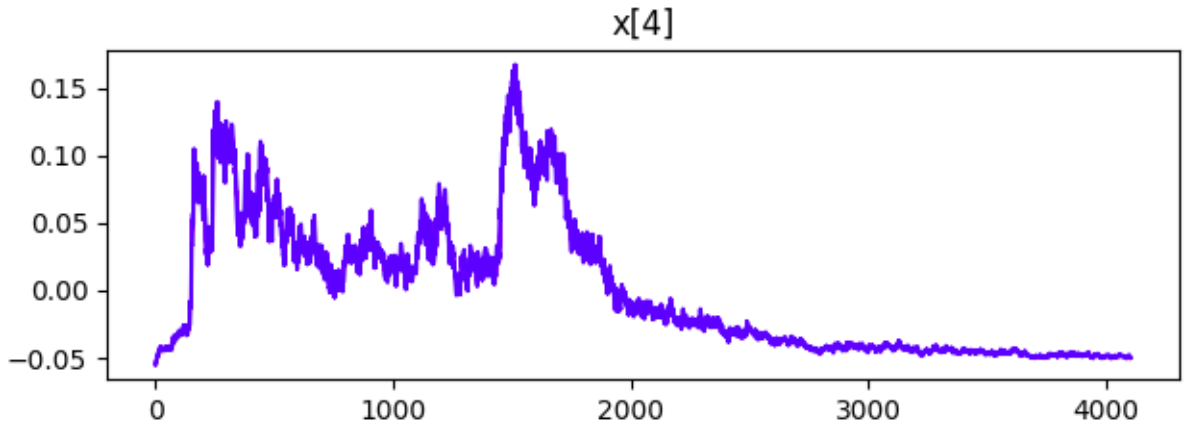
4.6) **Για το foxtrot_excerpt2.mp3**

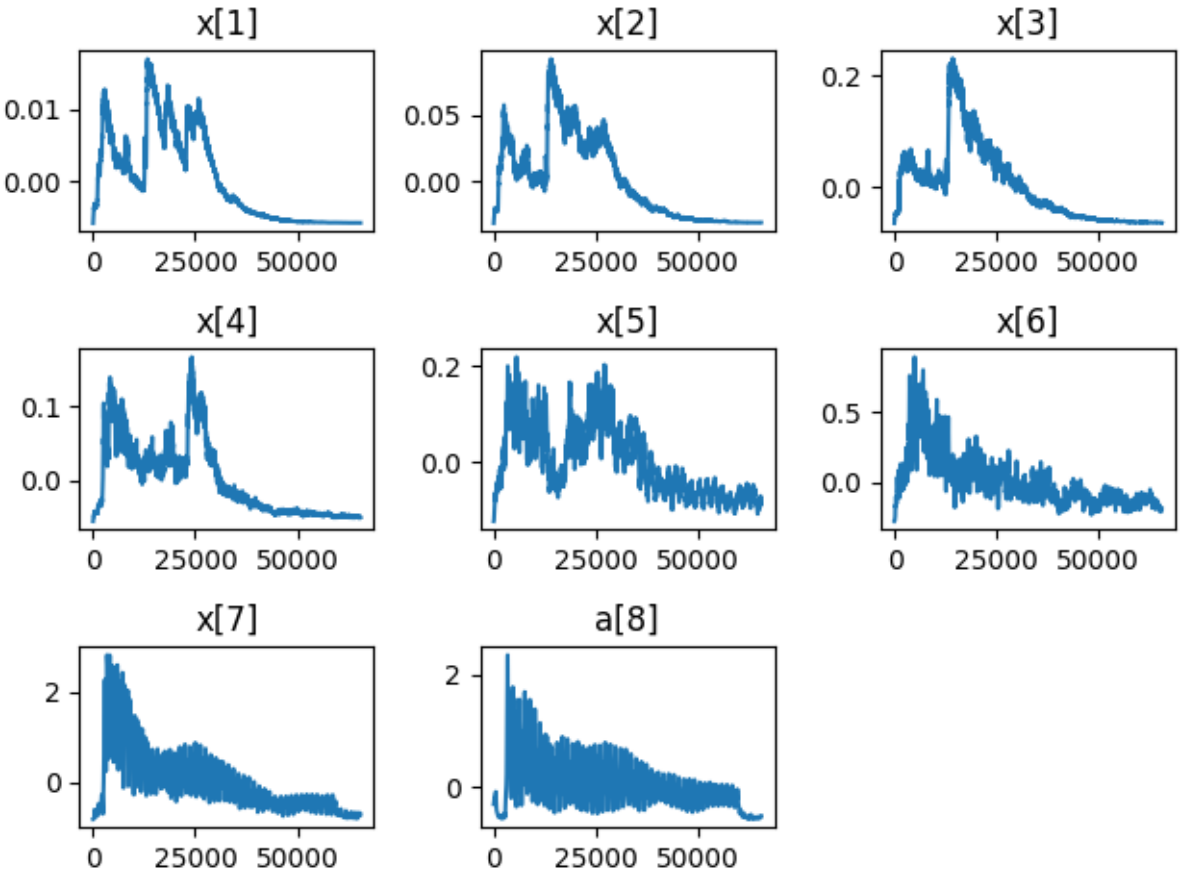


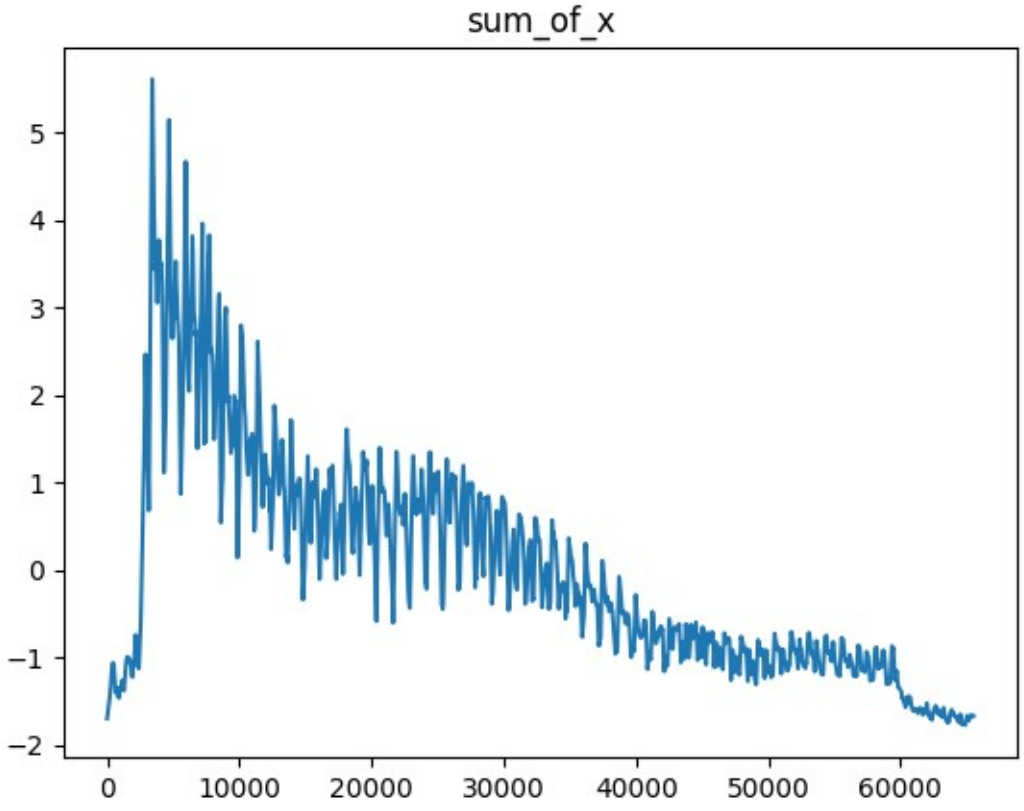

Details[2]
0.2
0.0
−0.2
−0.4
0
2500
5000
7500
10000
12500
15000

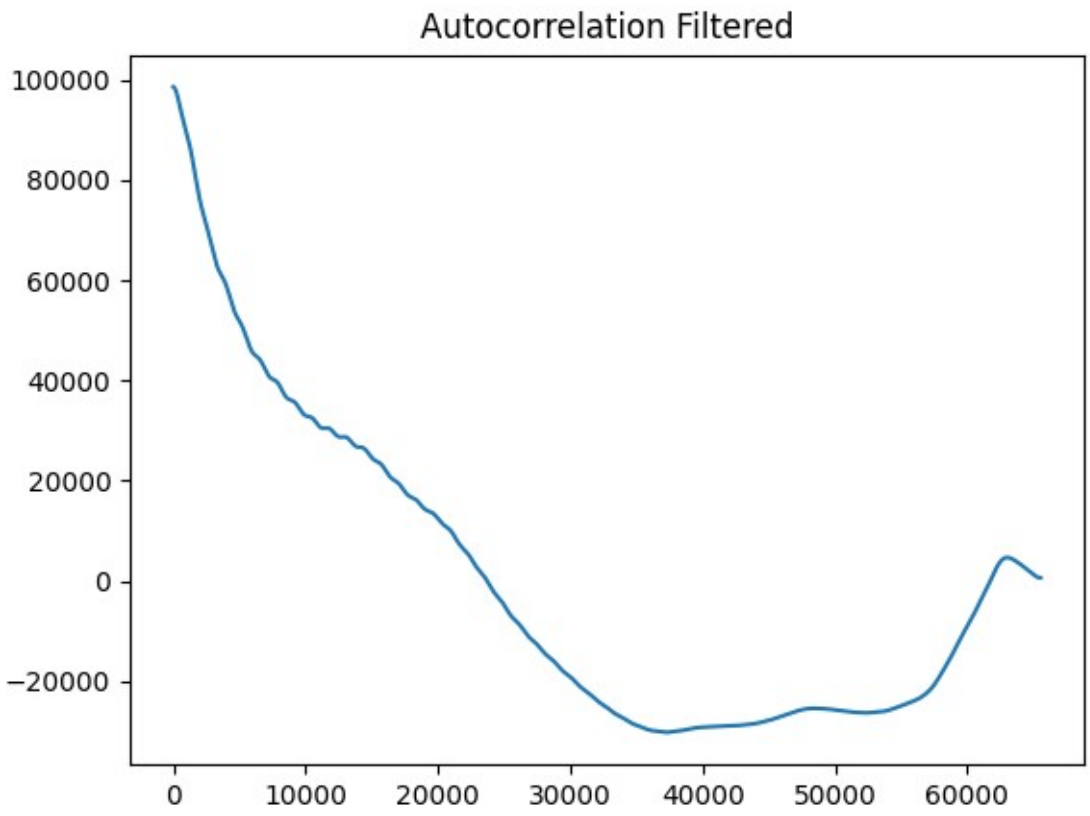

x[2]
0.075
0.050
0.025
0.000
−0.025
0
2500
5000
7500
10000
12500
15000


Details[4]
0.4
0.2
0.0
−0.2
−0.4
0
1000
2000
3000
4000


x[4]
0.15
0.10
0.05
0.00
−0.05
0
1000
2000
3000
4000


x[1]
x[2]
x[3]
x[4]
x[5]
x[6]
x[7]
a[8]
0
25000
50000


sum_of_x
0
10000
20000
30000
40000
50000
60000

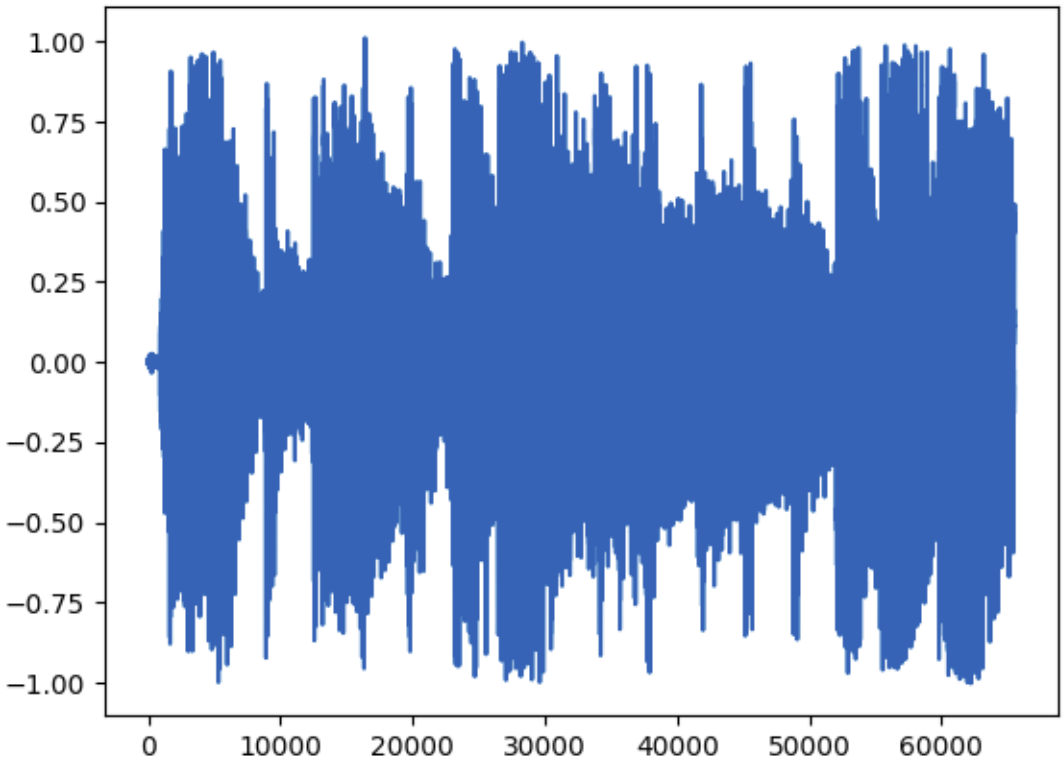


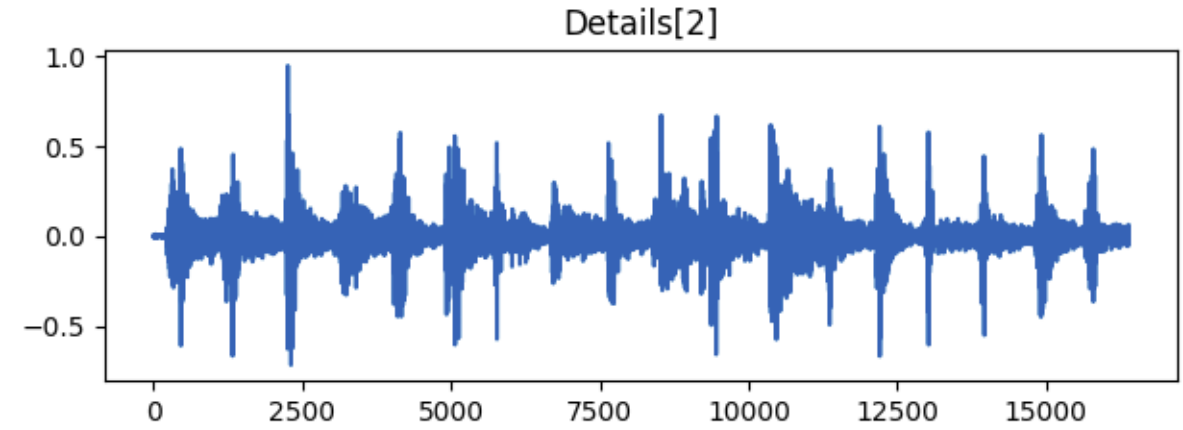
Μας εμφανίζει τα εξής (possible bpm, index, amplitude)

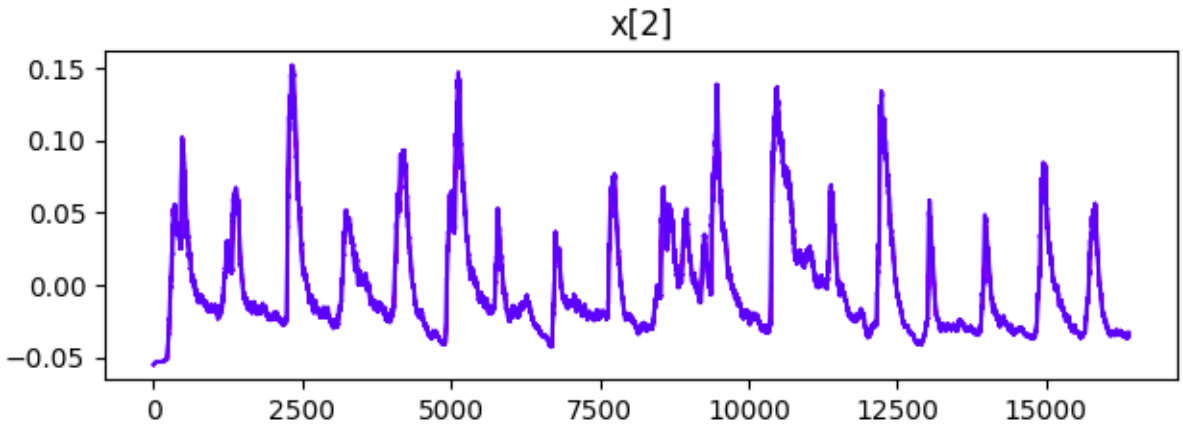
[113, 102]
[5017 6301]
[30545.42279815653, 28789.187571848197]
113

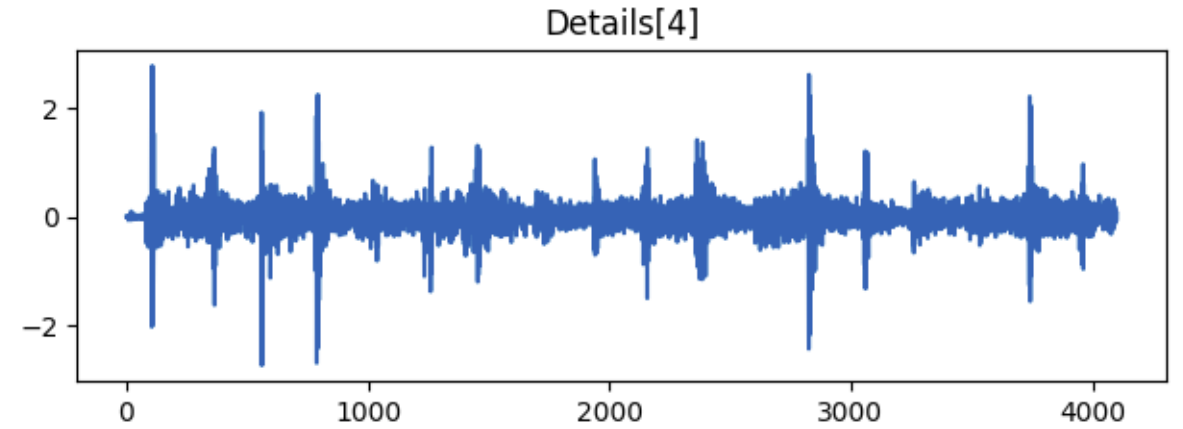
Από αυτά, κρατάμε αυτό με το μέγιστο πλάτος το οποίο είναι το 113.

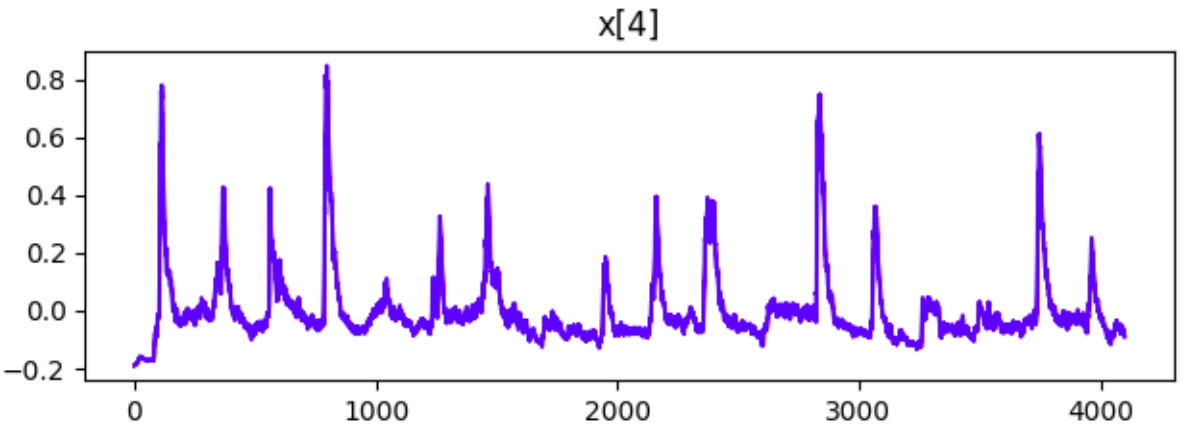
Για το salsa_excerpt.mp3

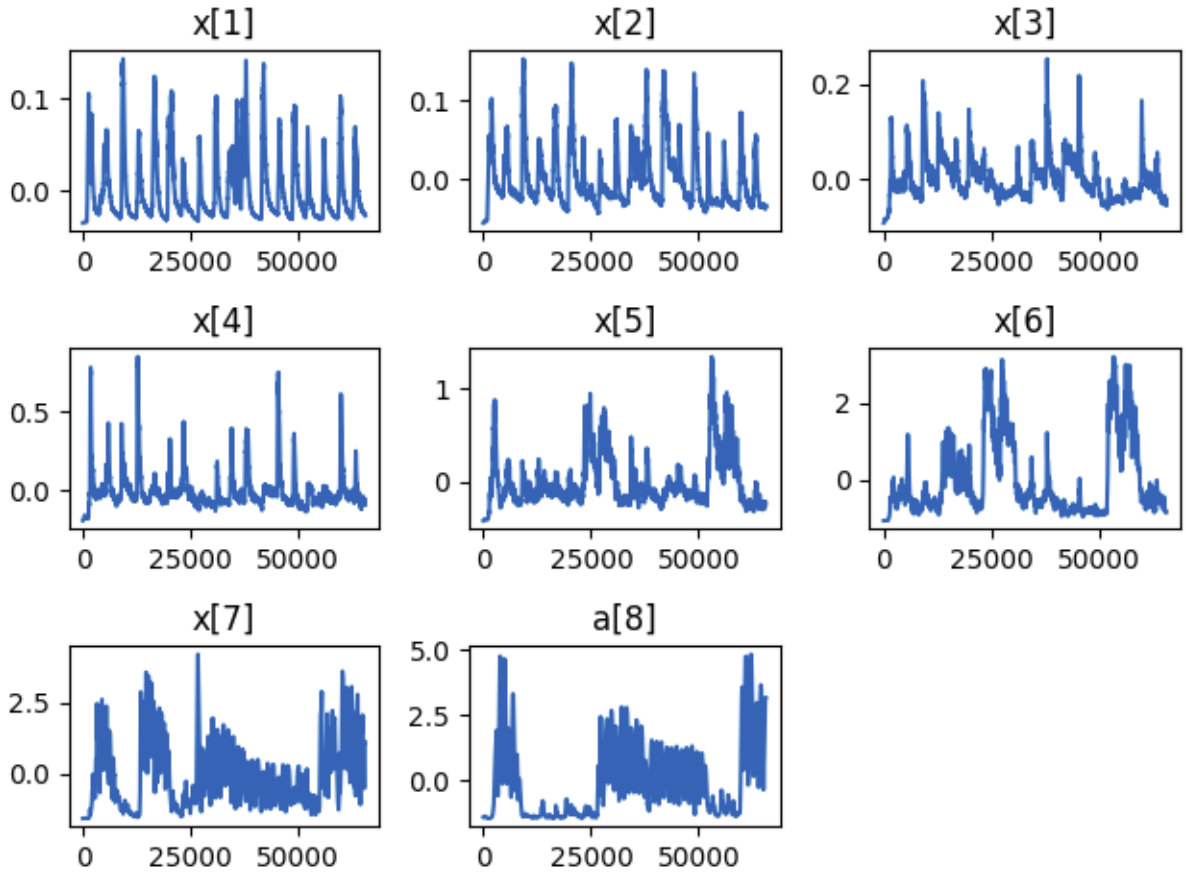


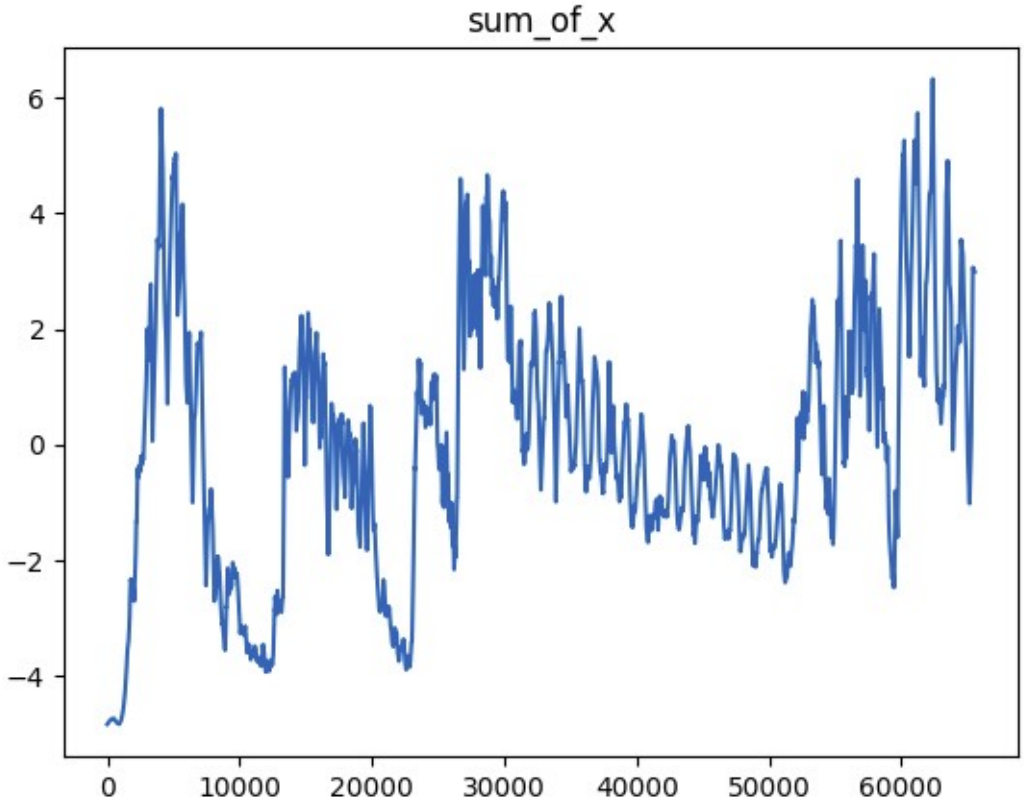

Details[2]
1.0
0.5
0.0
−0.5
0
2500
5000
7500
10000
12500
15000

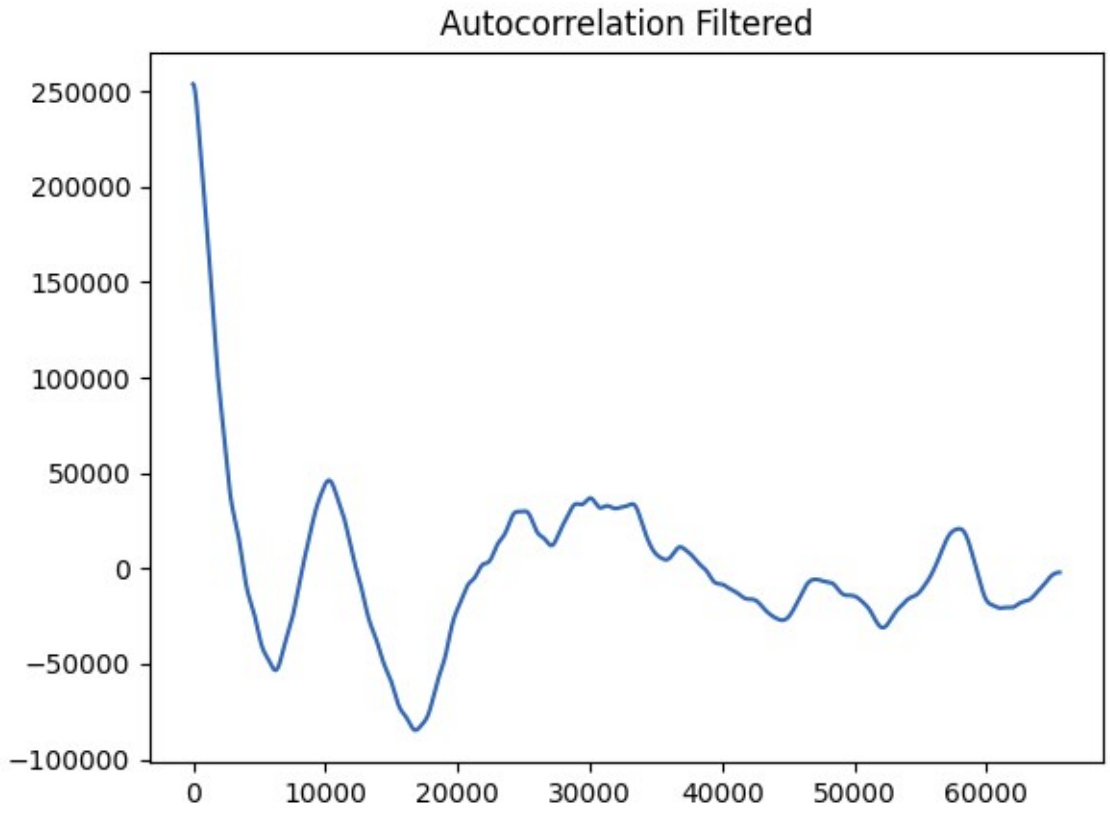

x[2]
0.15
0.10
0.05
0.00
−0.05
0
2500
5000
7500
10000
12500
15000


Details[4]
2
0
−2
0
1000
2000
3000
4000


x[4]
0.8
0.6
0.4
0.2
0.0
−0.2
0
1000
2000
3000
4000


x[1]
x[2]
x[3]
x[4]
x[5]
x[6]
x[7]
a[8]
0
25000
50000


sum_of_x
6
4
2
0
−2
−4
0
10000
20000
30000
40000
50000
60000



Μας εμφανίζει τα εξής (possible bpm, index, amplitude)
[128]
[3663]
[46171.418087589045]
128

Από αυτά, κρατάμε αυτό με το μέγιστο πλάτος το οποίο είναι το 128.