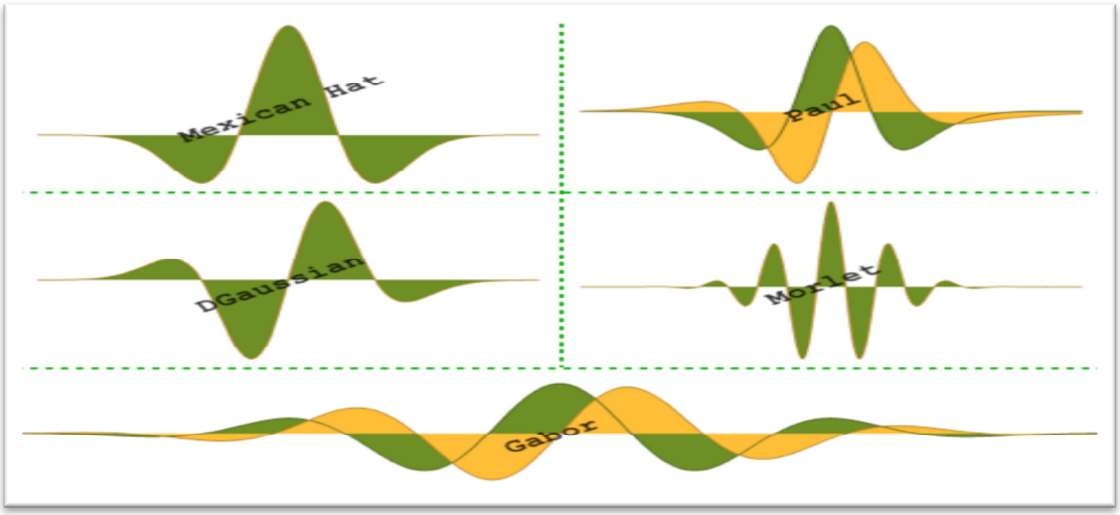
τμήμα μηχανικών πληροφορικής τ.ε.
Τ.Ε.Ι. Δυτικής Ελλάδας

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΦΑΡΜΟΓΕΣ ΤΩΝ WAVELETS ΣΤΗΝ ΨΗΦΙΑΚΗ ΕΠΕΞΕΡΓΑΣΙΑ ΣΗΜΑΤΟΣ

ΣΤΑΜΑΤΑΚΗ ΑΘΗΝΑ (ΑΜ: 2055)



ΕΠΟΠΤΕΥΩΝ ΚΑΘΗΓΗΤΗΣ: ΑΘΑΝΑΣΙΟΣ ΚΟΥΤΡΑΣ

ΝΑΥΠΑΚΤΟΣ, 2018

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΜΗ ΛΟΓΟΚΛΟΠΗΣ

Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Ακόμα δηλώνω ότι αυτή η γραπτή εργασία προετοιμάστηκε από εμένα προσωπικά και αποκλειστικά και ειδικά για την συγκεκριμένη πτυχιακή εργασία και ότι θα αναλάβω πλήρως τις συνέπειες εάν η εργασία αυτή αποδειχθεί ότι δεν μου ανήκει.

ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 1	**ΑΜ**	**ΥΠΟΓΡΑΦΗ**
ΣΤΑΜΑΤΑΚΗ ΑΘΗΝΑ	2055	
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 2 *(σε περίπτωση που είναι απαραίτητο)*	**ΑΜ**	**ΥΠΟΓΡΑΦΗ**
.....	.....	.....
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 3 *(σε περίπτωση που είναι απαραίτητο)*	**ΑΜ**	**ΥΠΟΓΡΑΦΗ**
.....	.....	.....

ΕΥΧΑΡΙΣΤΙΕΣ

Καταρχάς, θα ήθελα να ευχαριστήσω θερμά τον Καθηγητή Αθανάσιο Κούτρα, που μου έδωσε την ευκαιρία να ασχοληθώ με τον κλάδο των κυμάτων(wavelet) στην Ψηφιακή Επεξεργασία Σήματος και για την βοήθεια και την κατανόηση του. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου για την στήριξη και την συμπαράσταση της σε όλη τη διάρκεια των σπουδών μου.

ΠΕΡΙΛΗΨΗ

Τα κυματίδια αποτελούν ένα σύγχρονο εργαλείο στην επεξεργασία σήματος και συνήθως απεικονίζονται ως μια σύντομη ταλάντωση. Τα κυματίδια χρησιμοποιούνται σκόπιμα με συγκεκριμένες ιδιότητες που τα καθιστούν χρήσιμα στην επεξεργασία σήματος. Ιστορικά άρχισαν να εμφανίζονται στις αρχές της δεκαετίας του 1980.Οι δυνατότητες που προσφέρουν τα κυματίδια αποδεικνύουν και τον λόγο της χρησιμότητας τους καθώς βοηθούν να εξετάσουμε ένα σήμα τόσο στο πεδίο του χρόνου όσο και στο πεδίο της συχνότητας τους.

Λέξεις Κλειδιά: Κυματίδια, Μετασχηματισμός Fourier, Διακριτός Μετασχηματισμός, Συνεχής Μετασχηματισμός Κυματιδίων, Οικογένειες Κυματιδίων, Συμπίεση, Αποθορυβοποίηση.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Η θεωρία των κυματιδίων αναπτύχθηκε τις τελευταίες δεκαετίες ως ένα εργαλείο για την ανάλυση σημάτων για διαφορετικές συχνοτικές συνιστώσες. Μέσα από τα κυματάκια η ανάλυση σημάτων πέρασε από τη συχνοτική ανάλυση, στην ανάλυση κλίμακας, δίνοντας τη δυνατότητα να μελετήσουμε συγκεκριμένα χαρακτηριστικά μιας συνάρτησης, τα οποία δεν ανιχνεύονται με άλλες τεχνικές ανάλυσης. Στο πεδίο της επεξεργασίας σήματος τα κυματίδια εφαρμόστηκαν ως ένα σύγχρονο εργαλείο και σε πολλές άλλες επιστήμες.

Η παρούσα πτυχιακή εργασία έχει ως στόχο την απόκτηση βασικών γνώσεων για τα κυματίδια wavelet, την αναφορά σε βασικά χαρακτηριστικά του μετασχηματισμού κυματιδίων που αφορούν τόσο την εφαρμογή τους σε διάφορους τομείς όσο και την εκμάθηση τους ως σύγχρονο εργαλείο επεξεργασίας σήματος. Εμπεριέχει σημαντικά στοιχεία που αφορούν τη χρησιμότητα των wavelets στην ανάλυση σήματος και τα αποτελέσματα που έχει στη διαδικασία ένταξη τους.

Τέλος με τη βοήθεια εφαρμογής που κατασκεύασα στο πρόγραμμα της MATLAB μπορεί ο κάθε χρήστης να πειραματιστεί και να αλληλεπιδράσει σε πρακτική εξάσκηση μέσα από επιλογές κυματιδίων που εμπεριέχονται στο πρόγραμμα και τίθεται να επιλέξει και να τα εφαρμόσει σε εικόνες, ήχους και μουσική με στόχο την κατανόηση των κυματιδίων και την εξαγωγή αποτελεσμάτων που θα τον βοηθήσουν να κατανοήσει τη χρησιμότητα τους. Η εφαρμογή περιέχει και ένα εγχειρίδιο χρήστη που καταγράφει τις οδηγίες που θα ακολουθήσει ο χρήστης ώστε να χρησιμοποιήσει σωστά τις εντολές του προγράμματος και να καταλάβει πλήρως τις επιλογές που έχει να κάνει ανάμεσα σε πολλά είδη wavelet.Το πρόγραμμα περιέχει τη μέθοδο συμπίεσης και αποσυμπίεσης δεδομένων, την επιλογή διαφορετικών οικογενειών κυματιδίων όπως για παράδειγμα Haar ή Symlet. Έπειτα εμφανίζει το αντίστοιχο αποτέλεσμα εφαρμογής του κυματιδίου εμφανίζοντας τα νέα στοιχεία που προέκυψαν.

Από μαθηματική άποψη, τα wavelets είναι μια εξέλιξη του μετασχηματισμού Fourier. Μέσω της χρήσης τους οι επιστήμονες κατάφεραν να ξεπεράσουν το πρόβλημα του περιορισμένου εύρους των χρονολογικών αναλύσεων που μέχρι τώρα

μπορούσαν να αναλυθούν με τον μετασχηματισμό Fourier, δίνοντας δυνατότητας ανάπτυξης και εξέλιξης της ανάλυσης σημάτων. Το γεγονός ανάπτυξης νέων μαθηματικών εργαλείων και υπολογιστικών μεθόδων ,εγκαθίδρυσε τη λειτουργία των wavelets ως το πιο εξελιγμένο μαθηματικό εργαλείο που χρησιμεύει σε πολλές επιστήμες και διευκολύνει την επεξεργασία και ανάλυση δεδομένων. Τα κύματα αναπτύχθηκαν ανεξάρτητα στους τομείς των μαθηματικών, της κβαντικής φυσικής, της ηλεκτρολογία και της σεισμικής γεωλογίας. Οι ανταλλαγές μεταξύ αυτών των πεδίων τα τελευταία δέκα χρόνια έχουν οδηγήσει σε πολλές νέες εφαρμογές wavelet όπως συμπίεση εικόνας, αναταράξεις, ανθρώπινη όραση, ραντάρ και σεισμική πρόβλεψη.

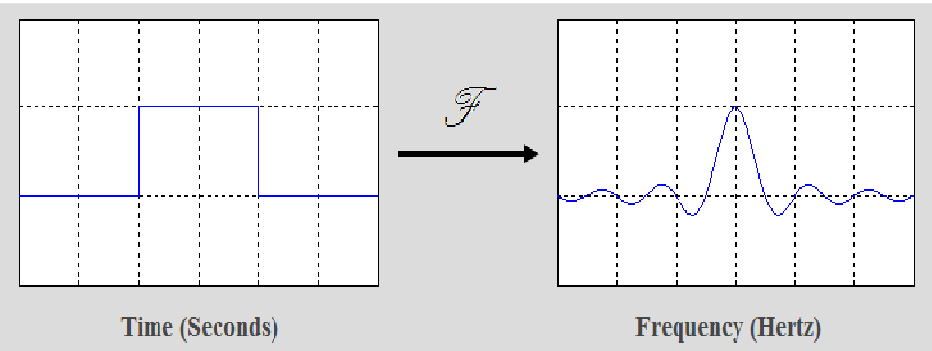
1. ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ FOURIER (FT)

Η ανάλυση Fourier είναι ένα από τα παλαιότερα αντικείμενα στην εφαρμοσμένη μαθηματική ανάλυση. Στην πράξη, η ανάλυση Fourier υλοποιείται είτε με τον μετασχηματισμό Fourier στην ολοκληρωτική μορφή του είτε με τις σειρές Fourier.Στην ανάλυση Fourier ο στόχος είναι η ανακάλυψη σημαντικών πληροφοριών στον χώρο των συχνοτήτων η οποία δεν είναι εμφανής στην χρονική αναπαράσταση του σήματος. Επομένως παρέχει αναπαράσταση συχνοτήτων που εμφανίζουν το πλάτος σήματος, χωρίς όμως να δίνει χρονικές πληροφορίες.

Τα σήματα που χαρακτηρίζονται ως σήματα απλής συχνότητας διέρχονται αναλλοίωτα από ένα γραμμικό χρονικά αμετάβλητο σύστημα. Ωστόσο υπάρχουν κάποια μαθηματικά εργαλεία που επιτρέπουν την ανάλυση ενός σήματος σε σήματα απλών συχνοτήτων. Ο Γάλλος φυσικός-μαθηματικός Joseph Fourier υιοθέτησε για πρώτη φορά, την ανάλυση μιας σύνθετης συνάρτησης σε άθροισμα συναρτήσεων απλών συχνοτήτων για να μελετηθούν φαινόμενα διάδοσης της θερμότητας.

Ο μετασχηματισμός του Fourier(FT) αποσυνθέτει μια συνάρτηση του χρόνου σε ένα σήμα, με τρόπο παρόμοιο, με τον οποίο μια μουσική χορδή μπορεί να εκφραστεί ως οι συχνότητες των συστατικών της σημείων. Ο μετασχηματισμός Fourier μιας συνάρτησης χρόνου είναι ο ίδιος μια πολύπλοκη συνάρτηση της συχνότητας, όπου η απόλυτη τιμή της αντιπροσωπεύει το ποσό εκείνης της συχνότητας, που υπάρχει στην αρχική συνάρτηση του σήματος και του οποίου το περίπλοκο επιχείρημα είναι η μετατόπιση φάσης του βασικού ημιτονοειδούς σε αυτή τη συχνότητα. Ο Fourier Transform ονομάζεται εκπροσώπηση πεδίου συχνότητας του αρχικού σήματος.

Ο μετασχηματισμός Fourier είναι ένα εργαλείο που σπάζει μια κυματομορφή(μια συνάρτηση ή ένα σήμα) σε μια εναλλακτική αναπαράσταση που χαρακτηρίζεται από ημιτονοειδή. Συγκεκριμένα δείχνει ότι οποιαδήποτε κυματομορφή μπορεί να αποτυπωθεί ως άθροισμα ημιτονοειδών λειτουργιών.Ο Fourier μετασχηματισμός, αποτελεί ένα μαγικό μαθηματικό εργαλείο διότι δίνει έναν μοναδικό τρόπο να δούμε οποιαδήποτε λειτουργία ως το άθροισμα απλών ημιτονοειδών.



Εικόνα 1: Fourier Transform(FT)

Γενικά ο μετασχηματισμός Fourier χρησιμοποιείται συχνά με σκοπό την μετατροπή οποιασδήποτε κυματομορφής από το πεδίο του χρόνου στην αντίστοιχη μορφή της

στο πεδίο της συχνότητας. Επίσης υπάρχει και ο αντίστροφος μετασχηματισμός Fourier ο οποίος εκτελεί την αντίστροφη μετατροπή, δηλαδή από το πεδίο της συχνότητας στο πεδίο του χρόνου.

1.1 Ιδιότητες Μετασχηματισμού Fourier

Αρχικά ο μετασχηματισμός Fourier είναι ένας γραμμικός μετασχηματισμός.Για παράδειγμα, ας πούμε ότι έχουμε δύο λειτουργίες g(t) και h(t), με μετασχηματισμούς Fourier που δίνονται από $G(f)$ και $H(f)$ αντίστοιχα.Στη συνέχεια, ο μετασχηματισμός Fourier οποιουδήποτε γραμμικού συνδυασμού g και h μπορεί να βρεθεί με τον τύπο:

$$\mathcal{F}\{c_1 g(t) + c_2 h(t)\} = c_1 G(f) + c_2 H(f) \tag{1.1.1}$$

Στην εξίσωση [1.1.1] τα c1 και c2 είναι οποιαδήποτε σταθερά (πραγματικοί ή σύνθετοί αριθμοί).Για να αποδειχθεί ότι είναι αληθής χρησιμοποιώντας τον ορισμό μετασχηματισμού Fourier:

$$\begin{aligned}\mathcal{F}\{c_1 g(t) + c_2 h(t)\} &= \int_{-\infty}^{\infty} c_1 g(t) e^{-i2\pi ft}\,dt + \int_{-\infty}^{\infty} c_2 h(t) e^{-i2\pi ft}\,dt \\ &= c_1 \int_{-\infty}^{\infty} g(t) e^{-i2\pi ft}\,dt + c_2 \int_{-\infty}^{\infty} h(t) e^{-i2\pi ft}\,dt \\ &= c_1 G(f) + c_2 H(f)\end{aligned} \tag{1.1.2}$$

Ο Fourier έχει ορισμένες ιδιότητες τις οποίες θα αναφέρουμε παρακάτω και αποτελούν βασικές ιδιότητες του μετασχηματισμού γιατί παρέχουν ορισμένες δυνατότητες. Το κίνητρο για το μετασχηματισμό Fourier προέρχεται από τη μελέτη της σειράς Fourier .

Στη μελέτη των σειρών Fourier, οι περίπλοκες αλλά περιοδικές συναρτήσεις έχουν γραφτεί ως το άθροισμα των απλών κυμάτων που στα μαθηματικά αντιπροσωπεύεται από ημίτονα και συνημίτονα.

Κλίμακα χρόνου και συχνότητας: Ο FT μας επιτρέπει να μετακινούμαστε από το πεδίο του χρόνου στο πεδίο της συχνότητας και αντίστροφα, όμως μία από αυτές τις απεικονίσεις είναι διαθέσιμη σε μία μόνο χρονική στιγμή. Δε μπορούμε δηλαδή να εξάγουμε πληροφορίες για τη συχνότητα από τη χρονοσειρά, ούτε πληροφορίες για το πότε υπάρχουν οι συνιστώσες συχνότητες. Οι πληροφορίες αυτές δε χρειάζονται όταν το σήμα είναι στάσιμο. Στην κατηγορία στάσιμων σημάτων περιλαμβάνονται τα σήματα από τα οποία το συχνοτικό τους περιεχόμενο δεν αλλάζει στον χρόνο.

Συμμετρία: $F\{\hat{f}(t)\} = 2\pi f(-\omega)$. (1.1.3)

Γραμμικότητα: Ο μετασχηματισμός ενός γραμμικού συνδυασμού δύο σημάτων ισούται με τον ίδιο γραμμικό συνδυασμό των αντίστοιχων μετασχηματισμών Fourier.

Χρονική Μετατόπιση: Εάν F(ω) είναι ο μετασχηματισμός του $f(t)$ τότε ο μετασχηματισμός του σήματος $f(t - t_0)$ είναι $e^{-i\omega t_0} f(\omega)$.

Αντιστροφή του χρόνου: Εάν F(ω) είναι ο μετασχηματισμός του σήματος $f(t)$ τότε ο μετασχηματισμός του σήματος $f(-t)$ είναι F(-ω).

Σχέση Συζυγίας: Για πραγματικά σήματα $f(t)$ ισχύει F(-ω) = F(-ω).Εάν το σήμα είναι επιπλέον άρτιο,τότε ο μετασχηματισμός Fourier θα είναι πραγματικός,ενώ εάν το σήμα είναι περιττό,τότε ο μετασχηματισμός Fourier θα είναι καθαρά φανταστικός.

Αλλαγή της κλίμακας του χρόνου: Ο μετασχηματισμός Fourier του σήματος $f(\alpha t)$ θα είναι:

$$\frac{1}{|\alpha|} F\left(\frac{\omega}{\alpha}\right) \quad (1.1.4)$$

Μετατόπιση Συχνότητας: Ο πολλαπλασιασμός ενός σήματος $f(t)$ με τη φανταστική εκθετική συνάρτηση $e^{i\omega_0 t}$ συνεπάγεται τη μετατόπιση στη συχνότητα F(ω-ω_0).

Παραγώγιση: Ο μετασχηματισμός Fourier της παραγώγου ενός σήματος ισούται με iωF(ω).

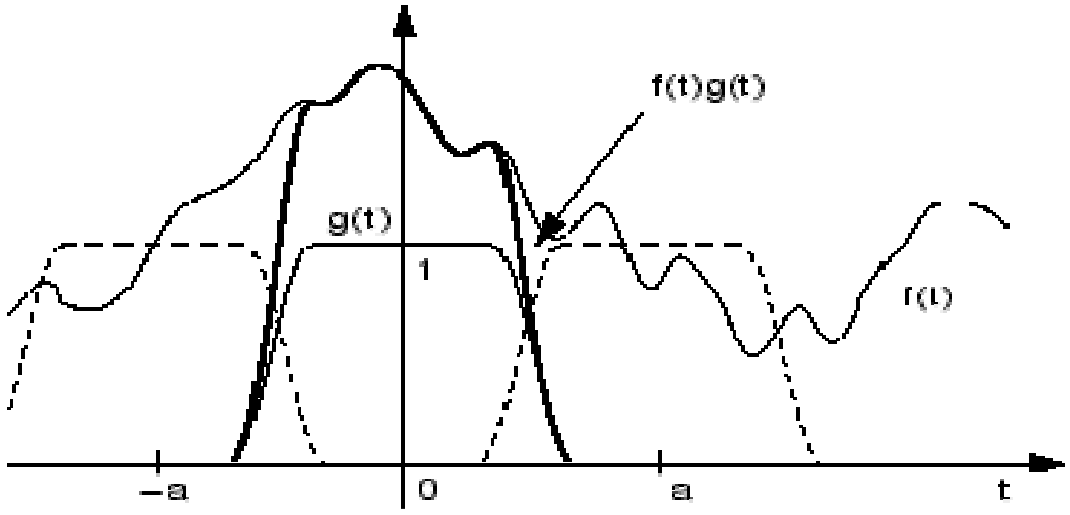
Συνέλιξη: Έστω f, g : R → C τοπικά ολοκληρώσιμες συναρτήσεις και $\hat{f}$= F{f} και $\hat{g}$= F{g} οι μετασχηματισμοί Fourier των f και g. Τότε, ισχύουν οι ιδιότητες:

Συνέλιξη στο χρόνο

$$F\{(f * g)(t)\} = \hat{f}(\omega)\,\hat{g}(\omega),$$

δηλαδή ο μετασχηματισμός Fourier απεικονίζει τη συνέλιξη συναρτήσεων στο πεδίο του χρόνου και στο γινόμενο των συναρτήσεων στο πεδίο της συχνότητας.

1.2 Μετασχηματισμός Fourier με παράθυρο



Εικόνα 2: Μετασχηματισμός Fourier με παράθυρο

Για την εύρεση του φάσματος συχνοτήτων του σήματος κατά τόπους, θα πρέπει να απομονώσουμε το σήμα σε διαστήματα όπου είναι στάσιμο και στη συνέχεια να εκτελέσουμε το μετασχηματισμό Fourier στα τμήματα αυτά. Με τον υπολογισμό αυτό καταλήγουμε σε ένα είδος ανάλυσης γνωστό ως Short Time Fourier Transform(STFT) ή Windowed Fourier Transform(WFT).Κατά το μετασχηματισμό STFT , το σήμα διαιρείται σε τομείς αρκετά μικρούς σε μήκος ώστε η υπόθεση ότι το σήμα είναι στάσιμο σε αυτούς να στέκει με μικρή πιθανότητα σφάλματος. Για το σκοπό αυτό, επιλέγουμε μια συνάρτηση παραθύρου w.Έστω 2Δw το μήκος του πεδίου ορισμού της συνάρτησης w , με τιμή 0 εκτός του 2Δw. Το παράθυρο εφαρμόζεται πρώτα στην αρχή του σήματος, πολλαπλασιάζεται με αυτό και στη συνέχεια υπολογίζονται τα γινόμενα FT.Η διαδικασία αυτή επαναλαμβάνεται μέχρι να σαρωθεί όλο το σήμα, με το ίδιο πάντα παράθυρο. Ο μαθηματικός ορισμός του STFT είναι:

$$X^{(w)}(t,f)=\int_{-\infty}^{+\infty}[x(t)w*(t-t')]e^{-j2\pi}\, f \in dt. \quad (1.2.1)$$

Η συνάρτηση παραθύρου w μπορεί να είναι μιγαδική αρκεί να ικανοποιεί τη συνθήκη:

$$\mathrm{W}(0) = \int_{-\infty}^{+\infty} w(t)dt \neq 0 \quad (1.2.2)$$

Η συνάρτηση παραθύρου συμπεριφέρεται σαν χαμηλοπερατό φίλτρο αφού έχει μη μηδενικό DC φάσμα. Πιο περιεκτικά δίνει μια προσέγγιση του φάσματος κοντά στο σημείο t=t΄. Ο STFT στην εφαρμογή του έχει και κάποια αρνητικά στοιχεία καθώς όταν χωρίζει το σήμα σε φέτες(κομμάτια) πολλές φορές φαίνεται σαν να υπάρχουν απώλειες ή απότομες μεταβολές στο σήμα.

2. ΟΡΙΣΜΟΣ WAVELET

Το Wavelet είναι μια κυματοειδή ταλάντωση με πλάτος που ξεκινάει από το μηδέν, αυξάνεται και στη συνέχεια μειώνεται στο μηδέν. Μπορεί συνήθως να απεικονιστεί ως μια "σύντομη ταλάντωση" όπως αυτή που καταγράφεται από ένα σεισμογράφο η καρδιογράφημα. Γενικά τα wavelets δημιουργούνται σκόπιμα ώστε να έχουν συγκεκριμένες ιδιότητες που τα καθιστούν χρήσιμα για την επεξεργασία σήματος. Χρησιμοποιώντας μια τεχνική αντιστροφής, μετατόπισης, πολλαπλασιασμού και ενσωμάτωσης που ονομάζεται συνέλιξη. Επιπλέον μπορούν να συνδυαστούν με τα γνωστά τμήματα ενός κατεστραμμένου σήματος για την εξαγωγή πληροφοριών από άγνωστα τμήματα. Ο μετασχηματισμός κυματιδίου γράφεται και ως WT ως συντομογραφία του όρου Wavelet Transform και χρησιμεύει στην ανάλυση σημάτων που περιέχουν κάποιο θόρυβο ή απότομες αλλαγές. Οι δυνατότητες που προσφέρουν τα κυματίδια, τα καθιστούν πολύ χρήσιμα γιατί μπορούν να εξετάσουν ένα σήμα τόσο στο πεδίο του χρόνου όσο και στην συχνότητα του.

Για να αναλύσουμε δομές σήματος διαφορετικών μεγεθών είναι απαραίτητο να χρησιμοποιήσουμε άτομα χρονικής συχνότητας με διαφορετικά χρονικά υποστηρίγματα (Debnath and Shah, 2010, p.338). Ο μετασχηματισμό κύματος αποσυνθέτει τα σήματα πάνω από τα διαστελλόμενα και μετατοπισμένα κυματίδια. Ένα κυμάτιο είναι μια συνάρτηση $\Psi \in L^2$ (R) με μηδενικό μέσο όρο :

$$\int_{-\infty}^{+\infty} \psi(t)\, dt = 0 \quad (2.1.1)$$

Αυτό κανονικοποιείται $\| \Psi \| = 1$, και επικεντρώνεται στη γειτονιά του t = 0. Μια οικογένεια ατόμων χρονικής συχνότητας με κλιμάκωση ψ από το s και μετατοπίζοντας το με το u:

$$\Psi_{u,s}(\mathrm{t}) = \frac{1}{\sqrt{s}}\, \psi\left(\frac{t-u}{s}\right). \quad (2.1.2)$$

Αυτά τα άτομα παραμένουν κανονικοποιημένα : $\| \Psi_{u,s} \| = 1$. Ο μετασχηματισμός wavelet του $f \in L^2(R)$ στον χρόνο u και κλίμακας s είναι

$$\mathrm{W}f\ (\mathrm{u},\ \mathrm{s}) = \langle f\ , \Psi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t)\ \frac{1}{\sqrt{s}}\ \psi\ ^{*}\left(\frac{t-u}{s}\right)\ dt. \quad (2.1.3)$$

Γραμμικό Φιλτράρισμα Ο μετασχηματισμός wavelet μπορεί να ξαναγραφεί ως ένα αντικείμενο συνέλιξης:

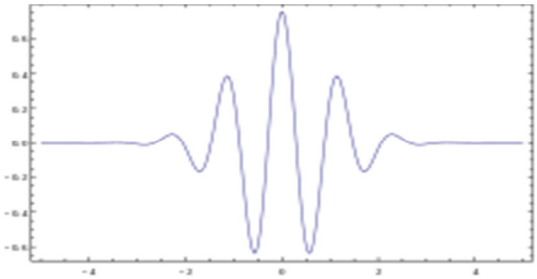
$$\mathrm{W}f(\mathrm{u},\ \mathrm{s}) = \int_{-\infty}^{+\infty} f(t)\ \frac{1}{\sqrt{s}}\ \psi\ ^{*}\left(\frac{t-u}{s}\right)\ \mathrm{dt} = f\ *\ \bar{\Psi}\mathrm{s}(u) \quad (2.1.4)$$

με

$$\bar{\Psi}\mathrm{s}(t) = \frac{1}{\sqrt{s}}\psi\ ^{*}\left(\frac{-t}{s}\right). \quad (2.1.5)$$

2.1 Ιστορική Αναδρομή των Wavelets

Ιστορικά η έννοια ondelettes ή wavelets άρχισε να εμφανίζεται στις αρχές του 1980.Αυτή η έννοια μπορεί να θεωρηθεί ως μια σύνθεση διαφόρων ιδεών που προέρχονται από διαφορετικούς κλάδους, συμπεριλαμβανομένων των μαθηματικών(Calderon-Zygmund και τη θεωρία Littlewood-Palley), της Φυσικής(όπου δηλώνει τον φορμαλισμό στη κβαντική μηχανική και την ομάδα επανακανονικοποίησης), στη μηχανική(με τα τετραγωνικά φίλτρα καθρέφτη(QMF)),στην κωδικοποίηση πλευρικής ζώνης, στην επεξεργασία σήματος και των πυραμιδικών αλγόριθμων στην επεξεργασία εικόνων. Το 1982, ο Jean Morlet, ένας Γάλλος Γεωφυσικός ανακαλύψεων τον όρο wavelet σε μια εργασία του, στον τομέα των κυματιδίων, παρέχοντας ένα νέο μαθηματικό εργαλείο για την ανάλυση σεισμικών κυμάτων.



Εικόνα 3: Wavelet

Στην ανάλυση του Morlet το σήμα αποτελείται από διαφορετικά χαρακτηριστικά στον χρόνο και στη συχνότητα και τα εξαρτήματα τους, στις υψηλές συχνότητες έχουν μικρότερη διάρκεια από τα αντίστοιχα χαμηλών συχνοτήτων. Για πρώτη φορά η ιδέα των wavelets ως οικογένεια λειτουργιών που κατασκευάζονται από

μετατοπίσεις και διαστολές, μιας μόνο συνάρτησης που αποκαλείται «Mother Wavelet» $\psi(t)$. Τα παραπάνω ορίζονται από τον τύπο:

$$\Psi_{a,b}(\mathrm{t}) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right), \mathrm{a}, \mathrm{b} \in R, a \neq 0, \quad (2.2.1)$$

όπου το α ονομάζεται παράμετρος κλιμάκωσης που μετράει το βαθμό συμπίεσης ή κλίμακας και το b την παράμετρο μετατόπισης που καθορίζει την χρονική θέση του κύματος. Εάν το $|\alpha|$ είναι μικρότερο του 1 το wavelet είναι η συμπιεσμένη έκδοση του κύματος της μητέρας και αντιστοιχεί κυρίως σε υψηλότερες συχνότητες. Αντίθετα όταν το $|\alpha|$ είναι μεγαλύτερο του 1,Ψ,α,b(t) έχει μεγαλύτερο χρονικό πλάτος και αντιστοιχεί σε χαμηλότερες συχνότητες. Έτσι τα wavelets έχουν προσαρμοσμένες συχνότητες στο πλάτος τους. Αυτός είναι ο κύριος λόγος για την επιτυχία των κυμάτων του Morlet στο σήμα και στην επεξεργασία και ανάλυση χρονικής συχνότητας του σήματος. Η ανάλυση των wavelets σε διαφορετικές κλίμακες ποικίλλει στις περιοχές χρόνου και συχνότητας όπως καθορίζονται από την αρχή της αβεβαιότητας του Heisenberg.Ο Morlet ανέπτυξε μια πρωτοποριακή ανάλυση σήματος στον χρόνο και στη συχνότητα, χρησιμοποιώντας μια μέθοδο που την αποκάλεσε "Κύματα Σταθερού Σχήματος" για να τα αντιπαραβάλλουμε με τις λειτουργίες ανάλυσης του σύντομου μετασχηματισμού Fourier που δεν έχει σταθερή μορφή. Ο Γάλλος θεωρητικός φυσικός Alex Grossmann αναγνώρισε γρήγορα τη σημασία του μετασχηματισμού κύματος του Morlet που είναι κάπως παρόμοια με τον φορμαλισμό για τις συνεκτικές καταστάσεις στην κβαντική μηχανική και ανέπτυξε μια μέθοδο αναστροφής από τον μετασχηματισμό wavelet. Η ονομασία των συνεκτικών καταστάσεων προέρχεται συχνά επειδή συνδέονται με μια συγγενική ομάδα (η ομάδα "ax+b"). Τα wavelets Ψα,b(x) είναι στην πραγματικότητα αποτέλεσμα της δράσης των φορέων U (α,b) σχετικά με τη λειτουργία Ψ έτσι ώστε:

$$\left[U_{(a,b)}\Psi\right](x) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{x-b}{a}\right). \quad (2.2.2)$$

Οι συγκεκριμένοι χειριστές είναι όλοι ενιαίοι στον χώρο του Hilbert L^2(R) ο οποίος αποτελεί ένα αφηρημένο διανυσματικό χώρο που διαθέτει τη δομή ενός εσωτερικού γινομένου που επιτρέπει το μήκος και η γωνία που πρέπει να μετρηθεί. Επιπλέον αποτελούν αναπαράσταση της ομάδα "αx+b" :

$$\mathrm{U}(\alpha, \mathrm{b})\ \mathrm{U}(\mathrm{c},\mathrm{d}) = \mathrm{U}(\mathrm{ac},\mathrm{b} + \mathrm{ad}). \quad (2.2.3)$$

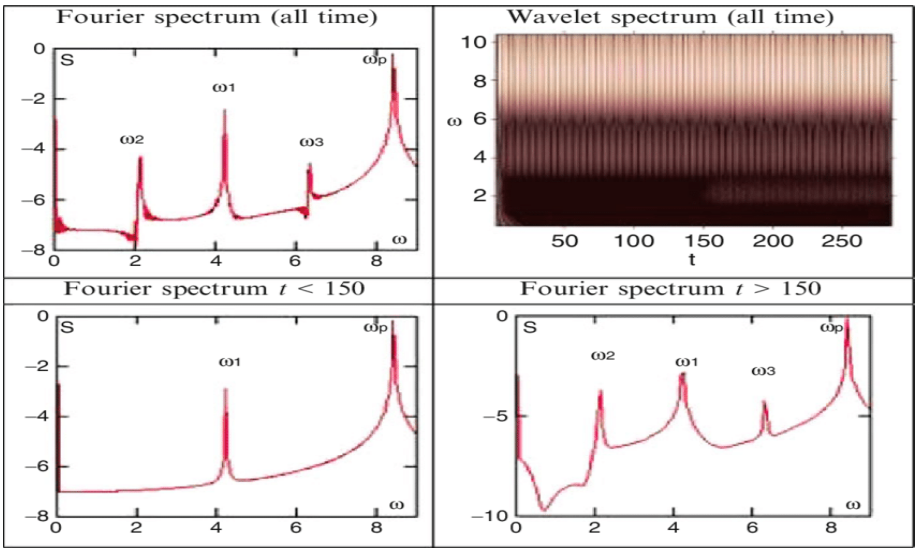
Η παραπάνω ομάδα αναπαράστασης είναι αναλλοίωτη που σημαίνει ότι για οποιοδήποτε μη μηδενικό f ∈ L^2 (R), υπάρχει το ορθογωνικό g που δεν είναι ίσο με μηδέν για όλα τα U(α,b) f. Επομένως τα U(α,b) f καλύπτουν ολόκληρο το διάστημα.

Η συνεκτική κατάσταση για τη συγγενική ομάδα (ax+b) η οποία είναι γνωστή ως wavelets και διατυπώθηκε για πρώτη φορά από τους Aslaksen και Klauder (1968,1969) στο πλαίσιο γενικότερων εκδηλώσεων ομάδων. Η επιτυχία των αριθμητικών αλγορίθμων του Morlet ώθησε τον Grossmann να κάνει μια εκτενέστερη μελέτη για τον μετασχηματισμό του κύματος, του Morlet που οδήγησε στην αναγνώριση ότι τα wavelets αντιστοιχούν σε μια ενσωματωμένη τετραγωνική αναπαράσταση της συγγενικής ομάδας.

Ο Morlet στον τομέα της σεισμολογίας γνώριζε ότι οι διαμορφωμένοι παλμοί που αποστέλλονται υπόγεια έχουν διάρκεια πολύ μεγαλύτερη στις υψηλές συχνότητες για να διαχωρίζουν τις επιστροφές των λεπτών, στενά διαχωρισμένων στρωμμάτων. Επομένως αντί να εκπέμπει παλμούς ίσης διάρκειας σκέφτηκε να στέλνει βραχύτερες κυματομορφές σε υψηλές συχνότητες. Τέτοιες κυματομορφές αποκτούνται απλά με κλιμάκωση που ονομάζεται κυματίδιο(wavelet).

2.2 Σύγκριση με Μετασχηματισμό Fourier

Για την κατανόηση των βασικών διαφορών ανάμεσα στον μετασχηματισμό μέσω Wavelets και στον μετασχηματισμό Fourier θα πρέπει να γίνει μια αναφορά στην εξέλιξη και στην αρχική εμφάνιση του Fourier.Ο μετασχηματισμός Fourier εισήχθηκε σαν μαθηματικό εργαλείο που θα βοηθούσε στην ανάλυση σημάτων, με κύριο στόχο την αποκάλυψη πληροφοριών στο συχνοτικό χώρο, οι οποίες δεν ήταν εμφανείς στην χρονική αναπαράσταση του σήματος εισόδου. Η παραπάνω ιδέα επιτεύχθηκε αλλά είχε ένα βασικό μειονέκτημα ότι κατά την αναπαράσταση πλάτους χάνεται οποιαδήποτε χρονική πληροφορία.



Εικόνα 4: Φασματογράφημα Fourier και Wavelet

Επιπλέον στον μετασχηματισμό Fourier σαν συναρτήσεις βάσεων χρησιμοποιούνται ημιτονοειδής συναρτήσεις διαφορετικών συχνοτήτων, που εκτείνονται στο άπειρο. Επομένως ο αντίστροφος μετασχηματισμός Fourier μπορεί να εφαρμοστεί μόνο σε περιπτώσεις όπου υπάρχουν περιοδικά, άπειρα ή στατικά σήματα εισόδου, γεγονός που αποτελεί βασικό πρόβλημα για την ανάλυση μεταβατικών φαινομένων που μπορεί να εμφανίζονται σε κάποιο σήμα. Με βάση τα παραπάνω αποτελέσματα αποδεικνύεται ότι με τον μετασχηματισμό Fourier τα τοπικά χαρακτηριστικά ενός σήματος γίνονται στο τέλος ολικά χαρακτηριστικά του Fourier.

Ο STFT δημιουργήθηκε για την επίλυση της εξαγωγής τοπικής πληροφορίας από το αρχικό σήμα όπου βασική του ιδέα ήταν μια παροδική συνάρτηση βάσης, η οποία μετατοπίζεται σε όλο το χρονικό πεδίο, με αποτέλεσμα το «παράθυρο» αυτό να χωρίζει σε επιμέρους μικρότερα τμήματα το αρχικό σήμα. Το μέγεθος που χρησιμοποιείται στο «παράθυρο» για τον επιμερισμό του σήματος εισόδου βασίζεται στην αρχή της αβεβαιότητας του Heisenberg.Επομένως μπορούμε να γνωρίζουμε μόνο τα χρονικά διαστήματα όπου υπάρχουν συγκεκριμένες συχνότητες.

Το παράθυρο STFT Που χρησιμοποιείται έχει ένα συγκεκριμένο μέγεθος, γεγονός που το καθιστά αδύναμο στην ανίχνευση σημάτων με υψηλές συχνότητες ή με χαμηλές αντίστοιχα. Όλα αυτά τα μειονεκτήματα οδήγησαν στην εξέλιξη των κυματιδίων. Η κύρια διαφορά ανάμεσα στα wavelets και τον μετασχηματισμό Fourier εντοπίζεται τόσο σε χρόνο όσο και σε συχνότητα ενώ ο τυπικός μετασχηματισμός Fourier εντοπίζεται μόνο στη συχνότητα.

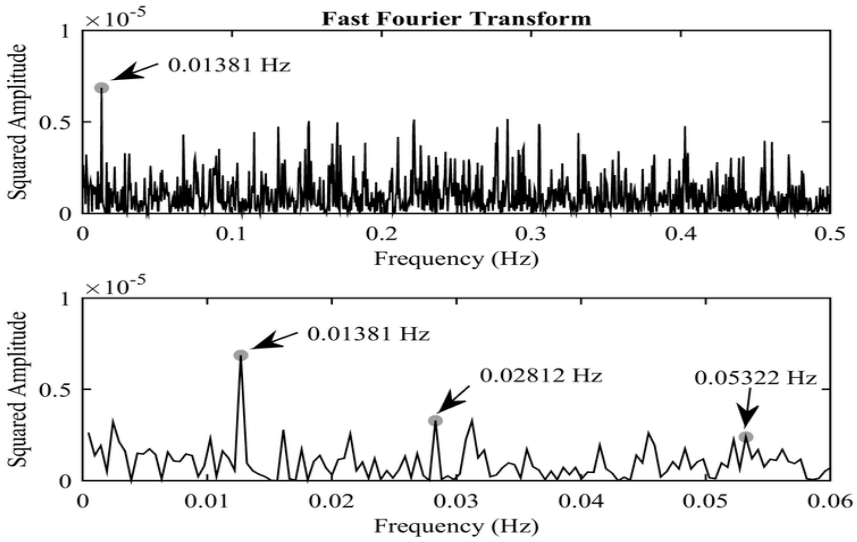
Εικόνα 5: Daubechies 4 tap wavelet

Για παράδειγμα ένας μετασχηματισμός Fourier(FT) μας λέει ποιες συχνότητες υπάρχουν στο σήμα μας αντίθετα ένας μετασχηματισμός Wavelet(WT) δηλώνει ποιες συχνότητες υπάρχουν και σε ποια κλίμακα. Εάν είχαμε ένα σήμα που έχει χρονικές μεταβολές ο Fourier Transform δεν θα μπορούσε να ανιχνεύσει το πότε συνέβη αυτό στα δεδομένα του. Το παραπάνω αποτέλεσμα συμβαίνει γιατί ένα wavelet μπορεί να οριστεί μέσα σε ένα συγκεκριμένο χρονικό διάστημα ξεκινώντας από το $F(t-0) = 0$ και τελειώνοντας στο F(t-end).Αυτό δίνει επομένως τον εντοπισμό του χρόνου.

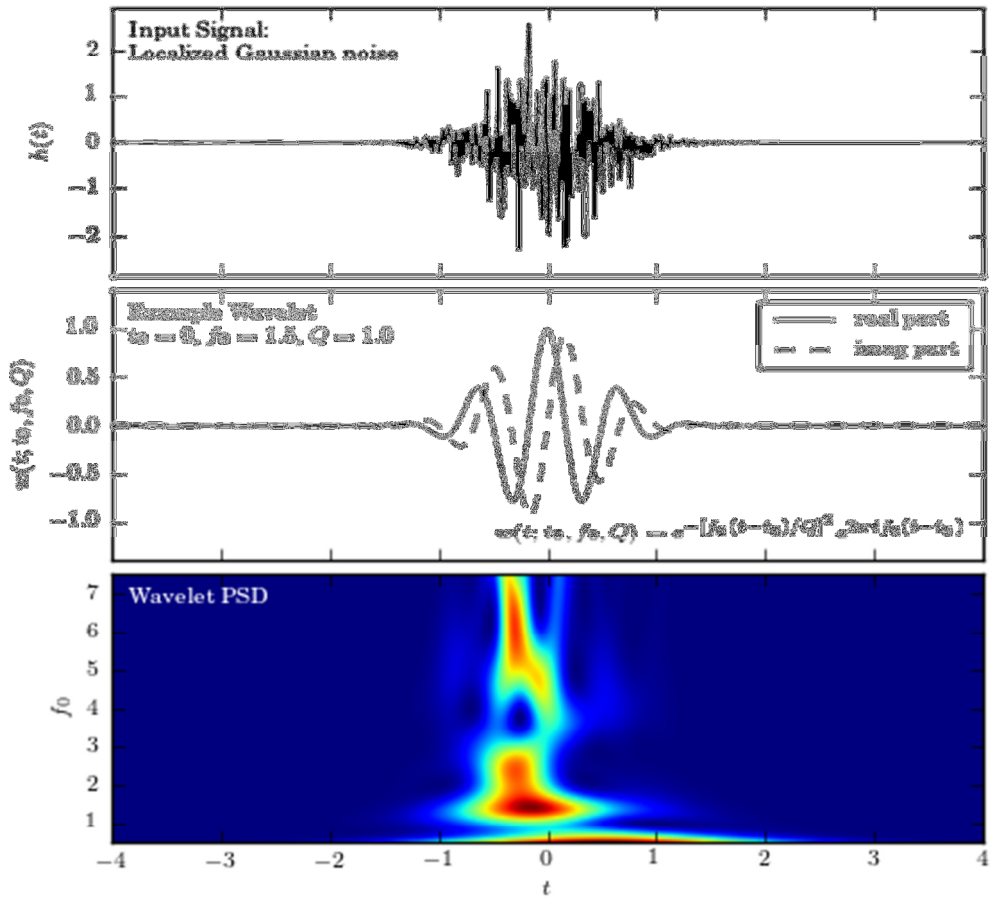
Κατά τη χρήση του STFT υπάρχουν προβλήματα γιατί εντοπίζεται δυσκολία επιλογής της χρονικής διάρκειας w(t) και της σταθερής ανάλυσης χρόνου-συχνότητας. Τα wavelets δίνουν λύση στα προηγούμενα προβλήματα γιατί παρέχουν συναρτήσεις με ολοένα αυξανόμενη συχνότητα στις χαμηλές συχνότητες και υψηλή ανάλυση χρόνου στις υψηλές συχνότητες

Η κλασική πράξη μετασχηματισμού Fourier της συνάρτησης επιτρέπει να κάνουμε μια μέτρηση με 0 εύρος ζώνης: η αξιολόγηση f(k) λέει ακριβώς το μέγεθος του στοιχείου της συχνότητας k . Αλλά με αυτόν τον τρόπο χάνεται όλος ο έλεγχος της χωρικής διάρκειας και δεν ξέρουμε πότε ακούγεται το σήμα. Αυτή είναι η περιοριστική περίπτωση της Αρχής Αβεβαιότητας: απόλυτη ακρίβεια στη συχνότητα και μηδενικό έλεγχο της χρονικής εξάπλωσης. Ο μετασχηματισμός wavelet εκμεταλλεύεται τις ενδιάμεσες περιπτώσεις της Αρχής της Αβεβαιότητας. Κάθε μέτρηση wavelet (ο μετασχηματισμός κύματος που αντιστοιχεί σε μια σταθερή παράμετρο) λέει κάτι σχετικά με τη χρονική έκταση του σήματος, καθώς και κάτι σχετικά με το φάσμα συχνοτήτων του σήματος. Δηλαδή, από την παράμετρο w (που είναι το ανάλογο της παραμέτρου συχνότητας k για το μετασχηματισμό Fourier), μπορούμε να αντλήσουμε μια χαρακτηριστική συχνότητα k(w) και ένα χαρακτηριστικό χρόνο t(w) και να πούμε ότι η αρχική μας λειτουργία περιλαμβάνει ένα σήμα της χονδρικής συχνότητα k(w) που συνέβη στον περίπου χρόνο t(w). Από την άλλη πλευρά, ένας μετασχηματισμός Fourier είναι ένα ενιαίο από t = - άπειρο σε t = + άπειρο. Επομένως, δεν υπάρχει χρονικός εντοπισμός. Ακόμα κι αν θα μετασχηματίσουμε ένα wavelet στο πεδίο συχνοτήτων του, η σχετική σχέση φάσης των διαφορετικών συνεισφερόμενων συχνοτήτων καθορίζει τη χρονική θέση του μετασχηματισμένου κύματος.

Ο μετασχηματισμός του Fourier είναι συχνά τόσο επικρατέστερος στην επιστήμη και την τεχνολογία (πιθανώς χάρη στην εφεύρεση της FFT, όταν η ψηφιακή τεχνολογία άρχισε να μεγαλώνει), έτσι συχνά θεωρούμε τη συχνότητα σαν sines και cosines χωρίς να το γνωρίζουμε. Είναι αλήθεια ότι στις συχνότητες Fourier (cos και sin βασικές λειτουργίες), ο μετασχηματισμός Fourier είναι αυτός που περιέχει μόνο συχνότητα και καμία πληροφορία "localization". Τουλάχιστον υπό την έννοια ότι ένα μόνο δείγμα θα μεταδίδει ίση ενέργεια σε όλο το φάσμα Fourier ανεξάρτητα από τη θέση ή το χρόνο του. Παράδειγμα σημάτων που είναι καλό να μοντελοποιηθούν με FFT είναι οι καθαροί μουσικοί τόνοι στον ήχο και τη μουσική. Ένας γρήγορος μετασχηματισμός Fourier (FFT) είναι ένας αλγόριθμος που δειγματοληπτεί ένα σήμα σε μια χρονική περίοδο (ή χώρο) και το διαιρεί σε συνιστώσες συχνότητας. Αυτές οι συνιστώσες είναι μοναδικές ημιτονοειδείς ταλαντώσεις σε ξεχωριστές συχνότητες, κάθε μία με το δικό της εύρος και φάση.



Εικόνα 6: Γρήγορος Μετασχηματισμός Fourier



Εικόνα 7: Φασματική Πυκνότητα σήματος που περιέχει Gaussian θόρυβο

2.3 Αναλυτικά Wavelets

Για να πραγματοποιηθεί η ανάλυση της χρονικής εξέλιξης των τόνων συχνότητας είναι απαραίτητο να χρησιμοποιηθεί ένα αναλυτικό κυματίδιο για τον διαχωρισμό των πληροφοριών που αφορούν τη φάση και το πλάτος του σήματος(Debnath and Shah,2010,p.340).Στο αναλυτικό σήμα μια συνάρτηση $f \in L^2$ (R) λέγεται ότι είναι αναλυτική εάν ο μετασχηματισμός Fourier είναι 0 για τις αρνητικές συχνότητες:

$$\hat{f}\alpha\,(w) = 0 \text{ εάν } w < 0. \quad (2.3.1)$$

Ο αναλυτικός μετασχηματισμός κύματος υπολογίζεται με ένα αναλυτικό κύμα ψ :

$$Wf(u, s) = \langle f\,, \Psi_{u,s} \rangle = \int_{-\infty}^{+\infty} f(t)\, \frac{1}{\sqrt{s}}\, \psi^* \left(\frac{t-u}{s}\right) dt. \quad (2.3.2)$$

Η ανάλυση της συχνότητας εξαρτάται από την εξάπλωση της συχνότητας των ατόμων wavelet $\Psi_{u,s}$. Υποθέτουμε ότι το ψ είναι κεντραρισμένο στο 0, το οποίο σημαίνει ότι το $\Psi_{u,s}$ του t = u. Με την αλλαγή της μεταβλητής u = $\frac{t-u}{s}$ επιβεβαιώνεται ότι

$$\int_{-\infty}^{+\infty} (t-u)^2 \,|\, \Psi_{u,s(t)}|^2 \;\mathrm{dt} = s^2\, \sigma_t^2, \quad (2.3.3)$$

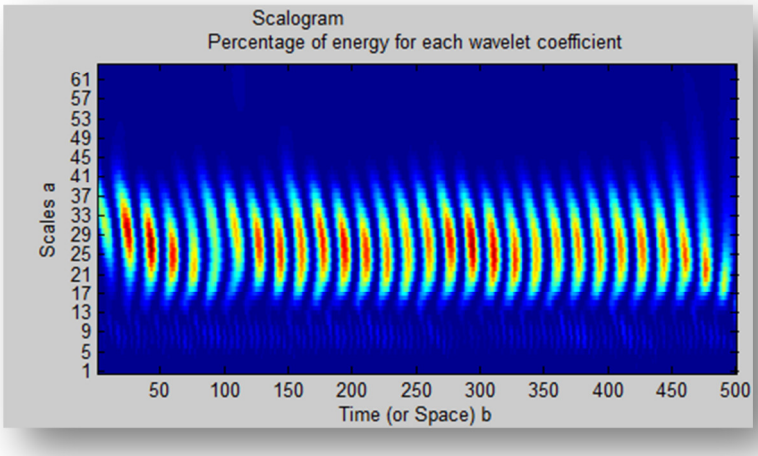
με $\sigma_t^2 = \int_{-\infty}^{+\infty} t^2 \,|\, \Psi_{(t)}|^2$ dt . Αφού $\widehat{\Psi}$ (w) είναι μηδέν στις αρνητικές συχνότητες, η κεντρική συχνότητα η της $\widehat{\Psi}$ είναι

$$\eta = \frac{1}{2\pi} \int_0^{+\infty} w \,|\, \Psi_{(w)}|^2 \,\mathrm{dw}. \quad (2.3.4)$$

Ένας αναλυτικός μετασχηματισμός wavelet ορίζει μια τοπική ενεργειακή πυκνότητα συχνότητας P wf, η οποία μετράει την ενέργεια του f στο κουτί Heisenberg κάθε κυματομορφής $\Psi_{u,s}$ με κέντρο στο (u,ξ = η/s) :

$$\mathrm{P\ w}\, f(u,\xi) = \mathrm{W}f(u,s)^2 = |\mathrm{W}f(u, \tfrac{\eta}{\xi})|^2 \,. \quad (2.3.5)$$

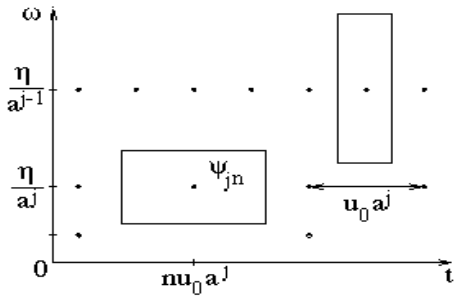
Η ενεργειακή αυτή πυκνότητα αποκαλείται σκαλόγραμμα (scalogram).



Εικόνα 8 : Σκαλόγραμμα κυματιδίων

Τα σκαλογράμματα των wavelet απεικονίζουν την ιδιότητα εντοπισμού της χρονικής συχνότητας του διακριτού μετασχηματισμού wavelet. Σε αυτή τη γραφική παράσταση, κάθε συντελεστής λεπτομέρειας εντοπίζεται ως γεμάτο ορθογώνιο του οποίου το χρώμα αντιστοιχεί στο μέγεθος του συντελεστή. Η θέση και το μέγεθος του ορθογωνίου σχετίζονται με το χρονικό διάστημα και την περιοχή συχνοτήτων για αυτόν τον συντελεστή. Οι συντελεστές σε χαμηλά επίπεδα παρουσιάζονται ως ευρείς και κοντές ορθογώνιες παραστάσεις για να δείξουν ότι εντοπίζουν ένα ευρύ χρονικό διάστημα αλλά ένα στενό εύρος συχνοτήτων στα δεδομένα.

2.4 Wavelet Frames



Εικόνα 9: Πλαίσια wavelet

Τα πλαίσια των wavelet κατασκευάζονται με δειγματοληψία του χρόνου και της κλίμακας παραμέτρων ενός μετασχηματισμού συνεχούς κύματος. Ένας πραγματικός μετασχηματισμός wavelet $f \in L^2(\mathrm{R})$ ορίζεται από

$$W f(u,s) = \langle f, \Psi_{u,s} \rangle, \quad (2.4.1)$$

όπως ψ είναι ένα πραγματικό wavelet και

$$\Psi_{u,s}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t-u}{s}\right) . \quad (2.4.2)$$

Η επιβολή $\|\Psi\| = 1$ συνεπάγεται ότι $\|\Psi_{u,s}\| = 1$. Για να κατασκευάσουμε ένα πλαίσιο πρέπει να καλύψουμε το επίπεδο χρονικής στιγμής με τα κουτιά Heisenberg της αντίστοιχης διακριτής οικογένειας κυμάτων. Ένα κυμάτιο $\Psi_{u,s}$ έχει μια ενέργεια στο χρόνο που είναι κεντραρισμένη στο u σε ένα πεδίο ανάλογο του s. Πέρα από τις θετικές συχνότητες, ο μετασχηματισμός Fourier $\Psi_{u,s}$ έχει υποστήριξη που επικεντρώνεται σε συχνότητα η/s, με μια εξάπλωση ανάλογη προς $1/s$. Για την απόκτηση πλήρους καλύμματος, έχουν δειγματοληψία του s κατά μήκος μιας εκθετικής ακολουθίας $\{a^j\}$ $j \in Z$, με ένα αρκετά μικρό βήμα διαστολής $\propto > 1$.Η χρονική μετατόπιση u δειγματίζεται ομοιόμορφα κατά διαστήματα ανάλογα με τη κλίμακα α^j.

Εφόσον

$$\Psi_{j,n}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t - nu_0 a^j}{a^j}\right). \quad (2.4.3)$$

Επομένως το

$\{\Psi_{j,n}\}(j,n) \in Z^2$ είναι ένα πλαίσιο του $L^2(R)$.

Η θεωρία πλαισίου αναλύει τη πληρότητα, τη σταθερότητα και τον πλεονασμό γραμμικών διακριτών αναπαραστάσεων του σήματος. Ένα πλαίσιο αποτελεί μια οικογένεια διανυσμάτων $\{\Phi_n\}\, n \in \Gamma$ που χαρακτηρίζει ένα σήμα f από τα εσωτερικά στοιχεία του $\{\langle f, \Phi_n \rangle\}\, n \in \Gamma$. Παράδειγμα εφαρμογής των Frames είναι η ανακατασκευή σημάτων κανονικοποιημένων ή μη κανονικοποιημένων

δειγματοληψιών. Επομένως τα πλαίσια χρησιμοποιούνται για ανίχνευση και διόρθωση σφαλμάτων ώστε να μη δημιουργηθούν σφάλματα στην παράδοση ψηφιακών δεδομένων καθώς και για τον σχεδιασμό και την ανάλυση τραπεζικών φίλτρων. Ένα πλαίσιο Wavelet:

$$\Psi_{j,n}(\mathrm{t}) = \frac{1}{\sqrt{s}}\,\psi\left(\frac{t - na^j u_0}{a^j}\right). \quad (2.4.4)$$

Αποδίδει εσωτερικά γινόμενα που δοκιμάζουν το συνεχή μετασχηματισμό κύματος σε χρονικά διαστήματα $\mathrm{a^j u_0}$:

$$\langle f, \Psi_{j,n}\rangle = f * \overline{\psi_\alpha^J}\,(\mathrm{n}a^j\, u_0) = \mathrm{W}f(na^j u_0, a^j). \quad (2.4.5)$$

Μετατοπίζοντας το f δίνεται:

$$\langle f_\tau, \Psi_{j,n}\rangle = f * \overline{\psi_\alpha^J}\,(\mathrm{n}a^j\, u_0 - \tau) = \mathrm{W}f(na^j u_0 - \tau, a^j). \quad (2.4.6)$$

Εάν το διάστημα δειγματοληψίας $a^j u_0$ είναι μεγαλύτερο σε σχέση με το ρυθμό μεταβολής του $f * \overline{\psi_\alpha^J}$, τότε οι συντελεστές $\langle f_\tau, \Psi_{j,n}\rangle$ μπορούν να πάρουν πολύ διαφορετικές τιμές που δεν μετατοπίζονται όσον αφορά το ένα στο άλλο.Το πρόβλημα αυτό είναι ιδιαίτερα οξύ για τις ορθογώνιες βάσεις κυμάτων όπου το u είναι το μέγιστο. Οι ορθογώνιοι συντελεστές κύματος του f_τ, μπορεί να είναι πολύ διαφορετικοί από τους συντελεστές του $f(Mallat, 1998,\ p.202)$.

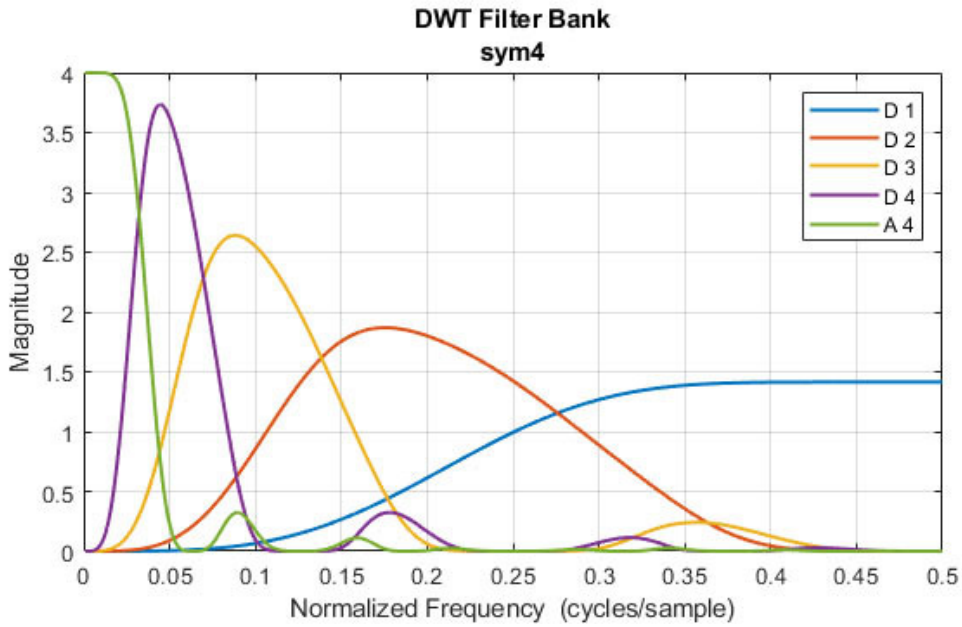
2.5 Ορθογώνιες Βάσεις Κυματιδίων

Η αναζήτηση ορθογωνίων κυματιδίων ξεκίνησε με μια προσέγγιση πολυανάλυσης.Για το $\boldsymbol{f} \in \boldsymbol{L^2}(\mathrm{R})$, το μερικό άθροισμα του συντελεστή $\sum_{\boldsymbol{n}}^{+\infty} = \langle \boldsymbol{f}, \boldsymbol{\Psi_{j,n}}\rangle \boldsymbol{\Psi_{j,n}}$ μπορεί να ερμηνευτεί ως η διαφορά δύο προσεγγίσεων του $\boldsymbol{f}$ στην ανάλυση $\boldsymbol{2^{-j+1}}$ και $\boldsymbol{2^{-j}}$. Οι προσεγγίσεις πολλαπλών αναλύσεων υπολογίζουν την προσέγγιση των σημάτων σε διάφορες αναλύσεις με ορθογώνιες προβολές σε διαφορετικά διαστήματα $\{\boldsymbol{V_j}\}$ $\mathrm{j} \in \mathrm{z}$. Οι προσεγγίσεις πολλαπλών αναλύσεων χαρακτηρίζονται από ένα συγκεκριμένο διακριτό φίλτρο που ρυθμίζει την απώλεια πληροφοριών σε όλες τις αναλύσεις. Μπορεί κάποιος να κατασκευάσει κυματίδια ψ τέτοια έτσι ώστε η διεσταλμένη οικογένεια wavelet να είναι μια

ορθοκανονική βάση του $\boldsymbol{L^2}$ **(R)**. Πίσω από αυτή τη δήλωση ανοίγεται μια ανταλλαγή μεταξύ της αρμονικής ανάλυσης και της διακριτής επεξεργασίας σήματος. Ορθογώνια κύματα που διαστέλλονται με $\boldsymbol{2^j}$ μεταφέρουν τις παραλλαγές σήματος στην ανάλυση του $\boldsymbol{2^{-j}}$.

2.6 Τράπεζα Φίλτρων

Τα φίλτρα τράπεζας προήλθαν το 1976 με τη συμπίεση ομιλίας του Esteban και Galand όπου χρησιμοποιήθηκε μια αντιστρεπτή τράπεζα φίλτρου η οποία κάνει αποσύνθεση ενός διακριτού σήματος $f[n]$ σε δύο σήματα μισού μεγέθους χρησιμοποιώντας μια διαδικασία υποδειγματοληψίας(Downsampling) και εφαρμογής φίλτρων. Επομένως το $f[n]$ έχει δυνατότητα ανάκτησης από την υποδειγματοληψία αυτών των σημάτων με την ακύρωση της προσωρινής αλλοίωσης(aliasing) λόγω δειγματοληψίας χαμηλής ανάλυσης, με τη χρήση μιας συγκεκριμένης κατηγορίας φίλτρων που ονομάζονται συζευγμένα φίλτρα καθρέπτη. Αυτή η ανακάλυψη οδήγησε σε μια δεκαετή ερευνητική προσπάθεια για την οικοδόμηση μιας ολοκληρωμένης θεωρίας τράπεζας φίλτρων. Μέσα από τη θεωρία πολλαπλών αναλύσεων των ορθογωνίων κυμάτων αποδεικνύεται ότι οποιοδήποτε συζευγμένο φίλτρο καθρέπτη χαρακτηρίζει ένα wavelet που παράγει μια ορθοκανονική βάση του L^2 (R).Επιπλέον επιτυγχάνεται και ένας γρήγορος διακριτός μετασχηματισμός κύματος μέσω της συσσώρευσης των συζευγμένων φίλτρων καθρέπτη. Οι τράπεζες φίλτρων διαχωρίζουν το σήμα εισόδου σε πολλαπλά συστατικά καθένα από τα οποία φέρει μια υποζώνη μιας συχνότητας του αρχικού σήματος.



Εικόνα 10: Filter Bank

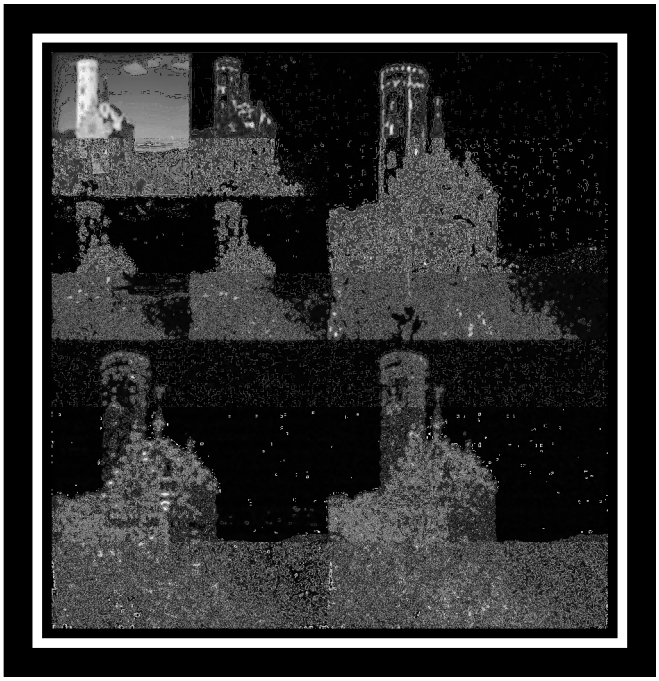
Τα φίλτρα τράπεζας αποτελούν στην ουσία,μια ανασύσταση ενός πλήρους σήματος που προκύπτει από συγκεκριμένες διαδικασίες.Ο γρήγορος διακριτός μετασχηματισμός κύματος αποσυνθέτει σήματα χαμηλής και υψηλής διέλευσης που έχουν υποδειγματοληπτεί από 2.Ο αντίστροφος μετασχηματισμός εκτελεί την ανακατασκευή. Στην επεξεργασία σήματος η μελέτη τέτοιων κλασικών τραπεζών πολλαπλών φίλτρων έγινε ένα σημαντικό θέμα επεξεργασίας καθώς ανακαλύφθηκε ότι με τετραγωνικά φίλτρα καθρέπτη είναι δυνατή η αποσύνθεση και ανακατασκευή σημάτων.

Οι ορθογώνιες και βιομορφογωνικές τράπεζες φίλτρων είναι διατάξεις των χαμηλοπερατών, υψιπερατών και ζωνοπερατών φίλτρων που διαιρούν τα δεδομένα σε υποζώνες. Εάν οι υποζώνες δεν τροποποιηθούν, αυτά τα φίλτρα επιτρέπουν την τέλεια ανακατασκευή δεδομένων. Οι ορθογώνιες τράπεζες φίλτρων δεν έχουν γραμμική φάση σε αντίθεση με τις βιομορφογωνικές. Η τεχνική lifting(ανύψωσης) επιτρέπει τον σχεδιασμό τέλειων τραπεζικών φίλτρων (Mallat,1998,p.344).

3. ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΙ WAVELET

Ο μετασχηματισμός κυματιδίων αποτελεί ένα εργαλείο το οποίο διασπά δεδομένα, συναρτήσεις ή τελεστές σε συστατικά διαφορετικών συχνοτήτων και στη συνέχεια εξετάζει κάθε συστατικό τους με ανάλυση που ταιριάζει στην αντίστοιχη κλίμακα του.Πιο συγκεκριμένα ο μετασχηματισμός κυματιδίων (wavelet transform) ενός σήματος ορίζεται ως η διαδικασία διάσπασης του σε ένα σύνολο βασικών συναρτήσεων(κυματιδίων),οι οποίες αποτελούν μετατοπισμένες και κλιμακωμένες εκδοχές του μητρικού κυματιδίου. Ο WT(Wavelet Transform) είναι σαν ένα φίλτρο ανάλυσης και εξετάζει(αναλύει) ή λειτουργεί στα αποσυντιθεμένα κομμάτια αντί στο αρχικό σήμα.

Η αντίστροφη διαδικασία της τοποθέτησης όλων αυτών των συστατικών μαζί, για την ανακατασκευή της αρχικής πληροφορίας είναι γνωστή ως αντίστροφος μετασχηματισμός κυματιδίων(Inverse Wavelet Transform). Ο τύπος ανακατασκευής είναι ένα φίλτρο σύνθεσης.Όμως κάποια απώλεια πληροφορίας ενδέχεται να συμβεί κατά την αντίστροφη διαδικασία της ανακατασκευής του αρχικού σήματος.



Εικόνα 11: Wavelet Transform

Υπάρχουν ουσιαστικά δύο μεγάλες κατηγορίες μετασχηματισμών κυματιδίων,ο Συνεχής Μετασχηματισμός Κυματιδίων (Continuous Wavelet Transform) και ο Διακριτός Μετασχηματισμός Κυματιδίων (Discrete Wavelet Transform).Στην περίπτωση του CWT η ανεξάρτητη μεταβλητή(π.χ ο χρόνος ή ο χώρος) είναι συνεχής. Στις σειρές κυματιδίων που προκύπτουν με δειγματοληψία DWT και CWT,οι παράμετροι του μετασχηματισμού δηλαδή η κλίμακα και η μετατόπιση είναι διακριτές. Σε περίπτωση ανακατασκευής του σήματος, οι πληροφορίες που παρέχει ο CWT είναι περιττές σε μεγάλο βαθμό και απαιτούν σημαντικό χρόνο υπολογισμού αλλά και πολλούς διαθέσιμους πόρους. Σε αντίθεση με τον DWT , ο οποίος βασίζεται στην κωδικοποίηση υποπεριοχών (subband coding),που παρέχει επαρκείς πληροφορίες τόσο για την ανάλυση όσο και την ανακατασκευή του αρχικού σήματος και έχει διαπιστωθεί ότι ο χρόνος υπολογισμού και οι διαθέσιμοι πόροι που απαιτούνται είναι λιγότεροι.

Στον Discrete Wavelet Transform η αναπαράσταση σήματος προκύπτει με διάφορες τεχνικές φιλτραρίσματος σε αντίθεση με την περίπτωση του Continuous Wavelet Transform που τα σήματα αναλύονται χρησιμοποιώντας ένα σύνολο συναρτήσεων βάσης, οι οποίες σχετίζονται μεταξύ τους μέσα από την μετατόπιση και την κλιμάκωση.

Στον Διακριτό Μετασχηματισμό κυματιδίων το σήμα ακολουθεί μια διαδικασία στην οποία περνάει μέσα από υψιπερατά και χαμηλοπερατά φίλτρα με σκοπό την ανάλυση των υψηλών και χαμηλών συχνοτήτων και χρησιμοποιούν φίλτρα με διαφορετικές συχνότητες αποκοπής για να αναλυθεί το σήμα σε διαφορετική κλίμακα.Ο συνεχής μετασχηματισμός κυματιδίων υπολογίζεται μεταβάλλοντας την κλίμακα τους προς ανάλυση παραθύρου, μετατοπίζοντας το παράθυρο στον χρόνο, πολλαπλασιάζοντας με το σήμα και εν συνεχεία ολοκλήρωση της διαδικασίας για ολόκληρο τον χρόνο.

3.1 Διακριτοί Μετασχηματισμοί Wavelet (DWT)

Ο Διακριτός μετασχηματισμός κυματιδίων προσφέρει την απαιτούμενη πληροφορία για ανάλυση και ανακατασκευή ενός σήματος με σημαντική μείωση των απαιτούμενων πόρων. Επομένως είναι ευκολότερος και πιο εύχρηστος στην υλοποίηση του. Στα σήματα διακριτού μετασχηματισμού, χρησιμοποιούμε φίλτρα με διαφορετικές συχνότητες αποκοπής ώστε το σήμα να περνάει από ένα σύνολο χαμηλοπερατών(low-pass) και υψιπερατών(high-pass).Η χρήση των φίλτρων βοηθάει στην ανάλυση του συχνοτικού περιεχομένου του σήματος.Οι μετασχηματισμοί μπορούν να υπολογιστούν μόνο σε κλίμακες από $N^{-1} < \mathrm{s} < 1$. Στον διακριτό υπολογισμό είναι ευκολότερο να ομαλοποιηθεί η απόσταση δειγματοληψίας σε 1 και συνεπώς να εξεταστεί το διεσταλμένο σήμα $f(t) = f(N^{-1}t)$.Μια μεταβολή της μεταβλητής στον ολοκληρωμένο μετασχηματισμό wavelet αποδεικνύει ότι:

$$\mathrm{W}f(u,s) = N^{-1/2}\ \mathrm{W}f(\mathrm{Nu},\mathrm{Ns}). \quad (3.1.1)$$

Ο Διακριτός μετασχηματισμός wavelet υπολογίζεται σε κλίμακα $\mathrm{s} = a^j$ με $\alpha = 2^{1/u}$ οι οποίες παρέχουν u ενδιάμεσες κλίμακες σε κάθε οκτάβα $[2^j, 2^{j+1}]$. Έστω Ψ(t) ένα κυματίδιο του οποίου η στήριξη περιλαμβάνεται στο $[-k/2, k/2]$. Για $2 \leq \alpha^j \leq Nk^{-1}$, το διακριτό wavelet που έχει κλίμακα από a^j ορίζεται από:

$$\Psi j[n] = \frac{1}{\sqrt{a^j}} \psi \left(\frac{n}{a^j}\right). \quad (3.1.2)$$

Το διακριτό κύμα έχει για Ka^j μη μηδενικές τιμές για $[-N/2, N/2]$. Η κλίμακα a^jείναι μεγαλύτερη από 2,διαφορετικά το διάστημα δειγματοληψίας ίσως είναι μεγαλύτερο από το στήριγμα των κυματοειδών.

Γρήγορος Μετασχηματισμός: Για την αποφυγή προβλημάτων συνόρων,αντιμετωπίζουμε το $f[n]$ και τα κυματίδια ψ_j[n] ως περιοδικά σήματα της περιόδου N. Ο διακεκριμένος μετασχηματισμός wavelet μπορεί να γραφτεί ως κυκλική συνέλιξη $\overline{\Psi}_j$ [n] = $\psi_j * [-n]$:

$$Wf\left[n, a^j\right] = \sum_{m=0}^{N-1} f[m]\psi_{j*}[m-n] = f * \overline{\Psi}_j [n] \ . \quad (3.1.3)$$

Αυτή η κυκλική συνέλιξη υπολογίζεται με το γρήγορο αλγόριθμο μετασχηματισμού Fourier που απαιτεί πράξεις $0(N \log_2 N)$. Εάν $\alpha = 2^{1/u}$ εκεί υπάρχουν u log(N/(2k)) κλίμακες τύπου $a^j \in [2N^{-1}, K^{-1}]$.Ο συνολικός αριθμός των πράξεων για τον υπολογισμό του wavelet μετασχηματισμού σε όλες τις κλίμακες είναι συνεπώς $0(uN(\log_2 N)^2)$.

Για να υπολογίσουμε το σκαλογράφημα στο Pw[n,ξ]=$\left|Wf\left[n, \frac{n}{\xi}\right]\right|^2$ δηλαδή τους τρεις άξονες(χρόνου,κλίμακας,συντελεστή),υπολογίζουμε το $Wf[n,s]$ $\sigma\varepsilon$ $\kappa\acute{\alpha}\theta\varepsilon$ $\kappa\lambda\acute{\iota}\mu\alpha\kappa\alpha$ s.Στα διακριτά φίλτρα κλιμάκωσης ο μετασχηματισμός wavelet που υπολογίζεται μέχρι μια κλίμακα α,δεν είναι μια πλήρη αναπαράσταση σήματος για αυτό είναι απαραίτητο να προσθέσουμε τις χαμηλές συχνότητες Lf[n, a^j] που αντιστοιχούν σε κλίμακες μεγαλύτερες του a^j.

Με τις παραπάνω πράξεις αλλάζουμε την ανάλυση που αποτελεί το μέτρο των συχνοτικών πληροφοριών του σήματος γιατί η μελέτη επικεντρώνεται σε ένα μέρος του περιεχομένου συχνοτήτων του σήματος και αλλάζει η πληροφορία που φέρνει αρχικά.Το σήμα πλέον θα έχει λιγότερες συχνότητες αλλά ίδιο αριθμό δειγμάτων,επομένως περιέχει μη απαραίτητη πληροφορία όσον αφορά τον αριθμό δειγμάτων.Για να αποφύγουμε επανάληψη πληροφοριών αλλάζουμε το ρυθμό δειγματοληψίας στο σήμα.Ο διακριτός μετασχηματισμός κύματος αντιπροσωπεύει μια συνάρτηση από ένα μετρήσιμο σύνολο συντελεστών wavelet,που αντιστοιχούν σε σημεία ενός δισδιάστατου πλέγματος ή σε πλέγμα διακεκριμένων σημείων στην περιοχή κλίμακας χρόνου που αποδεικνύεται από το m και n.

Εάν το {Ψm,n(t)} ορίζεται από :

$$\Psi_{m,n(x)} = a_0^{-m/2} \psi(\alpha_0^{-m} x - n b_0) \quad (3.1.4)$$

Είναι πλήρης στο L^2(R) για κάποια επιλογή από τα ψ,α και b τότε το σύνολο ονομάζεται μετασχηματισμός συγγενούς κύματος. Στη συνέχεια μπορούμε να εκφράσουμε οποιαδήποτε $f(t) \in L^2$(R) ως την υπέρθεση

$$f(t) = \sum_{m,n=-\infty}^{\infty} \langle f, \Psi_{m,n} \rangle \, \Psi_{m,n}(t). \text{ (3.1.5)}$$

Τέτοια σύνολα αποκαλούνται και πλαίσια. Ως διακριτό μετασχηματισμό κύματος χαρακτηρίζουμε το μετασχηματισμό που αφορά συναρτήσεις συνεχούς χρόνου αλλά υπολογισμένο σε διακριτό πλέγμα. Ο διακριτός μετασχηματισμός κυματιδίου επινοήθηκε για την εκμετάλλευση των αδυναμιών που είχε η εφαρμογή του CWT και έχουν σκοπό τον περιορισμό της πλεονάζουσας πληροφορίας. Οι ορθοκανονικές βάσεις κυματιδίων θεωρούνται DWT (Debnath and Shah,2010,p.354).

3.2 Ορθοκανονικές Βάσεις Κυματιδίων

Για την ανάλυση μιας συνάρτησης αλλά και την ανακατασκευή της θα πρέπει να πληρούνται ορισμένες συνθήκες. Για τον λόγο αυτό θα πρέπει για συγκεκριμένες ειδικές τιμές των ψ,α_0 και b_0, να είναι απαραίτητο οι συναρτήσεις $\psi_{j,k}$ να συνιστούν ορθοκανονική βάση του χώρου L^2(R).Ένα ζευγάρι βολικών τιμών για τις παραμέτρους κλίμακας και μετάθεσης είναι: $\alpha_0 = 2 \; και \; b_0 = 1$. Τότε η συνάρτηση ψ υπάρχει με καλές ιδιότητες όσον αφορά τον χωροχρονικό εντοπισμό έτσι ώστε οι $\Psi_{j,k(t)=2^{-j/2}\psi(2^{-j}t-K)}$ να αποτελούν ορθοκανονική βάση του L^2(R).Η μητρική συνάρτηση κλιμακώνεται κατά δυνάμεις του δύο και μετατίθεται με ακέραιες τιμές. Επομένως το πλέγμα είναι πλέον δυαδικό. Ένα ορθοκανονικό σύστημα s σε ένα εσωτερικό χώρο αντικειμένου X ονομάζεται ορθοκανονική βάση εάν κάθε x $\in X$ έχει μια μοναδική αναπαράσταση x.

$$\text{X} = \sum_{n=1}^{\infty} a_n \, X_n \text{ (3.2.1)}$$

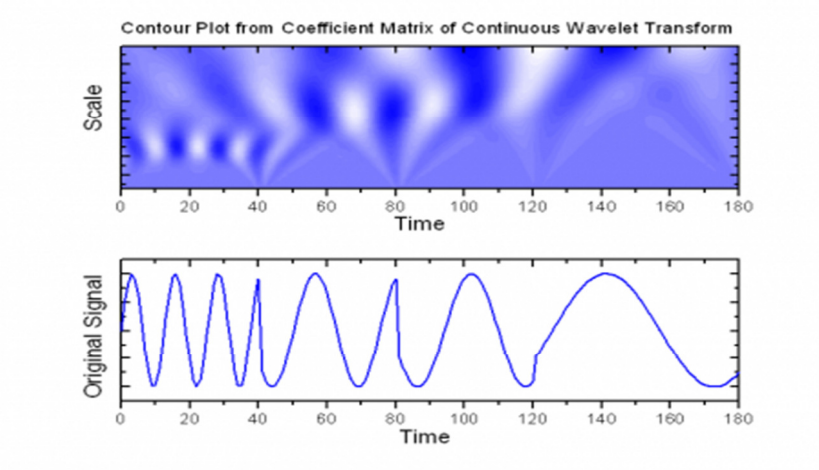
Όπου $a_n \in C$ και x_n,s είναι διακριτά στοιχεία του s.Σημειώνεται επίσης ότι μια ορθοκανονική ακολουθία $\{ x_n \}$ σε ένα εσωτερικό χώρο X είναι μια ορθοκανονική βάση στο X. Αρκεί να εκφράσουμε τη μοναδικότητα. (Debnath and Shah,2010,p.340) Τότε αν

$$\text{X} = \sum_{n=1}^{\infty} a_n \, X_n \qquad \text{και} \quad \text{X} = \sum_{n=1}^{\infty} B_n \; x_n \text{ , (3.2.2)}$$

Αυτό σημαίνει ότι $\alpha_{n = B_n}$ για όλα τα n $\in N$. Αποδεικνύεται με αυτό τον τρόπο η μοναδικότητα. Μια ορθοκανονική ακολουθία $\{ x_n \}$ σε ένα χώρο του Hilbert είναι πλήρης αν και μόνο αν $\langle x, x_n \rangle = 0$ για όλα τα n $\in N$ υποδηλώνει ότι x = 0.

3.3 Συνεχής Μετασχηματισμός Κυματιδίων (CWT)

Ο Συνεχής Μετασχηματισμός κυματιδίου(Continuous Wavelet Transform) αναπτύχθηκε ως μια εναλλακτική προσέγγιση του σύντομου μετασχηματισμού Fourier (STFT).



Εικόνα 12: Συνεχής Μετασχηματισμός Wavelet

Στην πράξη υπολογίζεται διαιρώντας ένα μεγαλύτερο χρονικό σήμα σε μικρότερα τμήματα ίσης διάρκειας και στη συνέχεια υπολογίζει χωριστά τον μετασχηματισμό Fourier σε κάθε μικρότερο τμήμα. Στον χώρο $L^2(R)$ δεδομένου ότι είναι ένας Hilbert χώρος, δηλαδή ένας αφηρημένος διανυσματικός χώρος που διαθέτει δομή εσωτερικού γινομένου που επιτρέπει το μήκος και η γωνία που πρέπει να μετρηθεί και με βάση τα παραπάνω, ισχύει ότι το εσωτερικό γινόμενο f και g ορίζεται το ολοκλήρωμα:

$$\langle f, \mathrm{g} \rangle = \int_{-\infty}^{+\infty} f(t)\mathrm{g}^*(\mathrm{t})\, dt. \quad (3.3.1)$$

Στα πλαίσια αυτά του ολοκληρώματος ορίζουμε τον συνεχή μετασχηματισμό κύματος (cwt) μιας συνάρτησης $f(t) \in L^2(R)$ ως:

$$\mathrm{T}f(\mathrm{a,b}) = \langle f, \psi_{a,b} \rangle = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\psi * \left(\frac{t-b}{a}\right) dt \quad (3.3.2)$$

Οι συναρτήσεις $\psi_{a,b}$ αποτελούν συναρτήσεις βάσης του μετασχηματισμού στις οποίες αποσυντίθεται η συνάρτηση f.Ο παραπάνω μετασχηματισμός λαμβάνοντας υπόψη το θεώρημα Parseval μπορεί να γραφεί ως:

$$T f(a,b)=\frac{1}{2\pi}\int_{-\infty}^{+\infty} f(w)\hat{\psi}*\alpha,b(w)dw \quad (3.3.3)$$

Όπου το $f(w)$ είναι ο μετασχηματισμός Fourier της f και η $\hat{\psi}\alpha,b(w)$ αντίστοιχα της ψ.Πέρα από αυτές τις βασικές εξισώσεις χρειάζονται και κάποιες επιπλέον ανάλογα με τον τύπο ανάλυσης που θέλουμε να πετύχουμε. Επομένως η χρήση των wavelets θα μπορούσαμε να πούμε ότι στην πράξη αποτελούν πρόσθετες συνθήκες ανάλογα με τον σκοπό που θέλουμε να πετύχουμε.Ο συνεχής μετασχηματισμός κύματος(cwt) είναι ένα επίσημο εργαλείο που παρέχει μια υπερβολική ανάλυση σε επίπεδο αναπαράστασης ενός σήματος επιτρέποντας τη μεταβολή των παραμέτρων, των μετατοπίσεων και της κλιμάκωσης των συνεχών κυμάτων(wavelet).Ο συνεχής μετασχηματισμός κύματος μιας συνάρτησης x(t) σε κλίμακα (α>0) $\alpha \in R$ και της μετατόπισης b∈ R εκφράζεται από το ακόλουθο ολοκλήρωμα:

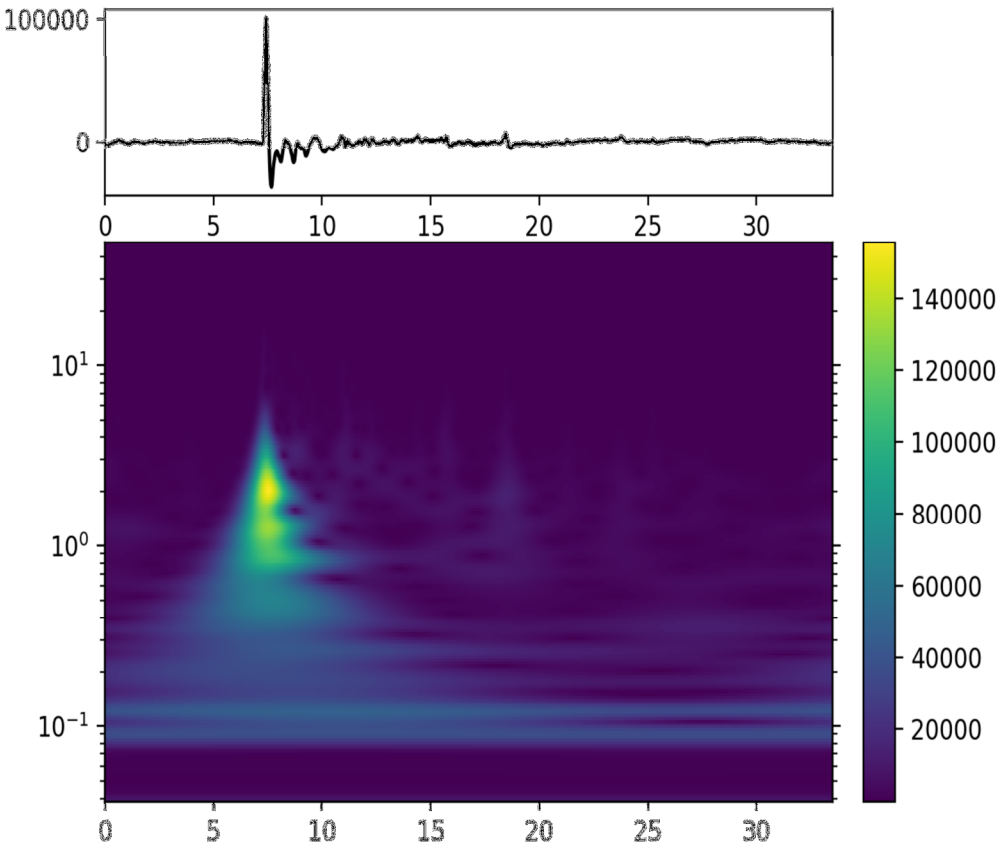
$$X_{w\,(a,b)}=\frac{1}{|a|1/2}\int_{-\infty}^{\infty} x(t)\bar{\psi}\left(\frac{t-b}{a}\right)dt \quad (3.3.4)$$

Ο συντελεστής κλίμακας είτε διαστέλλει ένα σήμα είτε εφαρμόζει τη συμπίεση του. Όταν ο συντελεστής κλίμακας είναι σχετικά χαμηλός, το σήμα είναι πιο συρρικνωμένο και με τη σειρά του οδηγεί σε λεπτομερή γραφική παράσταση. Υπάρχουν όμως και κάποια μειονεκτήματα της διαδικασίας καθώς ο συντελεστής χαμηλής κλίμακας δεν διαρκεί για ολόκληρη τη διάρκεια του σήματος. Από την άλλη πλευρά, όταν ο συντελεστής κλίμακας είναι υψηλός το σήμα είναι τεντωμένο, γεγονός που δείχνει ότι το γράφημα που θα προκύψει θα παρουσιάσει λιγότερες λεπτομέρειες αλλά θα έχει διάρκεια σε ολόκληρο το σήμα.

Πολλές εφαρμογές κυματιδίων έχουν ως ανεξάρτητη μεταβλητή το χώρο παρά τον χρόνο.Τα κυματίδια έχουν τη μορφή μικρών κυμάτων τοποθετημένων στον άξονα του χρόνου και στην πράξη υπάρχει ένας τεράστιος αριθμός κυματιδίων που χρησιμοποιούνται στην ανάλυση δεδομένων.Ο συνεχής μετασχηματισμός κυματιδίων ορίζεται ως το άθροισμα καθ'όλο το χρόνο του σήματος που έχει πολλαπλασιαστεί με μετατοπισμένες, κλιμακωμένες εκδόσεις της συνάρτησης κυματιδίων ψ. Το αποτέλεσμα του CWT είναι πολλοί συντελεστές κυματιδίων C,που είναι μια συνάρτηση κλίμακας και θέσης.Τα συστατικά κυματίδια του αυθεντικού κύματος παράγονται πολλαπλασιάζοντας κάθε συντελεστή με το κατάλληλα μετατοπισμένο και κλιμακωμένο κυματίδιο.Η κλιμάκωση ενός σήματος είναι όταν διατείνουμε ή συμπιέζουμε ένα σήμα και συμβολίζεται με το γράμμα α. Στην ανάλυση κυματιδίων,η κλίμακα σχετίζεται με τη συχνότητα του κύματος.Για τον συνεχή μετασχηματισμό κυματιδίων ακολουθούμε και ορισμένα βήματα. Τα βήματα για την εφαρμογή του συνεχή μετασχηματισμού κυμάτων δίνονται παρακάτω:

1. Παίρνουμε ένα κυματίδιο και το συγκρίνουμε με ένα τμήμα του αυθεντικού σήματος που θέλουμε να αναλύσουμε.
2. Υπολογίζουμε έναν αριθμό, C, ο οποίος εκφράζει πόσο συσχετίζεται το κυματίδιο με αυτό το τμήμα. Επιπλέον όσο μεγαλύτερο είναι το C τόσο μεγαλύτερη θα είναι και η ομοιότητα.Τα αποτελέσματα εξαρτώνται από τη μορφή του κυματιδίου που θα επιλέξουμε.
3. Μετατοπίζουμε το κυματίδιο προς τα δεξιά και επαναλαμβάνουμε τα βήματα ένα και 2 ώσπου να καλύψουμε όλο το σήμα.
4. Εκτείνουμε δηλαδή ή κλιμακώνουμε το κυματίδιο και επαναλαμβάνουμε τα βήματα 1 μέχρι 3.
5. Όταν ολοκληρώσουμε τα παραπάνω βήματα θα εμφανιστούν οι συντελεστές που παράχθηκαν σε διαφορετικές κλίμακες για διαφορετικά τμήματα του σήματος. Οι συντελεστές απαρτίζουν τα αποτελέσματα μιας οπισθοδρόμησης του αυθεντικού σήματος που εκτελέστηκε στα κυματίδια.

Επιπρόσθετα έχουμε τη δυνατότητα αν επιθυμούμε να δημιουργήσουμε ένα γράφημα όπου ο άξονας x θα αντιπροσωπεύει τη θέση στο σήμα δηλαδή τον χρόνο και ο άξονας y την κλίμακα και τα χρώματα σε κάθε σημείο και το μέγεθος του συντελεστή του κυματιδίου.Έτσι με αυτό τον τρόπο θα μπορέσουμε να καταλάβουμε όλους αυτούς τους συντελεστές του αποτελέσματος.



Εικόνα 13: Continuous Wavelet Transform

4. ΒΑΣΙΚΕΣ ΟΙΚΟΓΕΝΕΙΕΣ ΚΥΜΑΤΙΔΙΩΝ

Οι οικογένειες wavelets ποικίλλουν ως προς πολλές σημαντικές ιδιότητες ανάλογα με το αποτέλεσμα που θέλουμε να πετύχουμε. Εάν κατανοήσουμε τις ιδιότητες ανάλυσης και σύνθεσης των κυματομορφών οι οικογένειες των wavelets μπορούν να υποστηρίξουν το κυματίδιο σε χρόνο, συχνότητα και ρυθμό αποσύνθεσης. Υπάρχουν διαφορετικοί τύποι οικογενειών wavelet που χρησιμοποιούνται με διάφορα κριτήρια.

Τα βασικά κριτήρια είναι τα εξής:

- Η υποστήριξη του ψ,$\hat{\psi}$ (και φ,$\hat{\varphi}$) και η ταχύτητα σύγκλισης προς 0 αυτών των λειτουργιών ($\psi(t)$ ή $\hat{\psi}(w)$) όταν ό χρόνος t ή η συχνότητα w πηγαίνει στο άπειρο, ο οποίος ποσοτικοποιεί τόσο τον εντοπισμό του χρόνου όσο και της συχνότητας.
- Η συμμετρία η οποία είναι χρήσιμη για την αποφυγή της θορυβοποίησης στην επεξεργασία εικόνων.
- Ο αριθμός στιγμών εξαφάνισης για ψ ή για φ(αν υπάρχει), ο οποίος είναι χρήσιμος για σκοπούς συμπίεσης.
- Η κανονικότητα, η οποία είναι χρήσιμη για τη λήψη ωραίων χαρακτηριστικών, όπως η ομαλότητα του ανακατασκευασμένου σήματος ή εικόνας και για την εκτιμώμενη λειτουργία στην ανάλυση μη γραμμικής παλινδρόμησης.

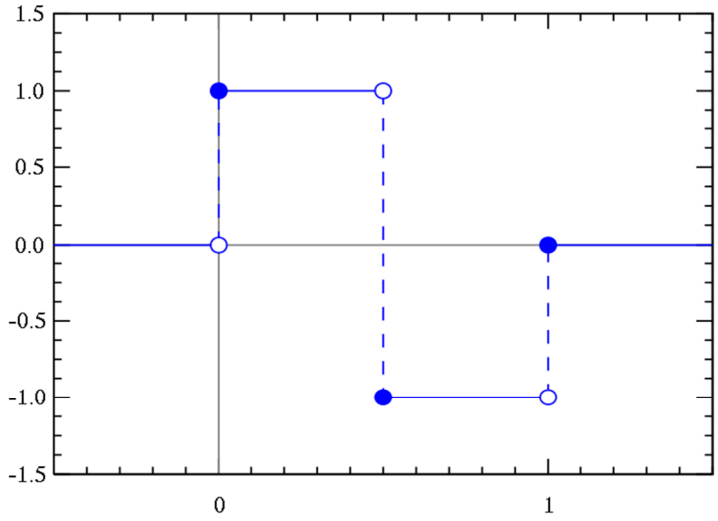
Τα παραπάνω συνδέονται με δύο ιδιότητες που επιτρέπουν γρήγορο αλγόριθμο και εξοικονόμηση χώρου κωδικοποίησης:

1. Η ύπαρξη μιας λειτουργίας κλιμάκωσης φ.

2. Η ορθογωνικότητα ή η βιομορφογωνικότητα της προκύπτουσας ανάλυσης.

Μπορούν επίσης να συσχετιστούν με αυτές τις λιγότερο σημαντικές ιδιότητες όπως είναι η ευκολία του πίνακα και η εξοικίωση με τη χρήση. Σε περίπτωση που θέλουμε να δούμε μια σύντομη επισκόπηση των κύριων ιδιοτήτων όλων των οικογενειών wavelet που είναι διαθέσιμες στην εργαλειοθήκη της Matlab πληκτρολογούμε την εντολή waveinfo. Οι λειτουργίες φ και ψ μπορούν να υπολογιστούν με την εντολή wavefun και τα φίλτρα δημιουργούνται χρησιμοποιώντας την εντολή wfilters.

4.1 Haar Wavelet



Εικόνα 14 : Κυματίδιο Haar

Το κυματίδιο Haar είναι μια ακολουθία αναβαθμισμένων λειτουργιών τετραγωνικού σχήματος,οι οποίες μαζί σχηματίζουν μια οικογένεια ή βάση wavelet.Η ακολουθία Haar αναγνωρίζεται ως η πρώτη βάση wavelet που προτάθηκε το 1909 όπου ο Alfred Haar ,χρησιμοποίησε αυτές τις λειτουργίες για να δώσει ένα παράδειγμα του ορθογωνικού συστήματος για τον χώρο των τετραγωνικών ολοκληρώσιμων λειτουργιών στο διάστημα μονάδων [0,1]. Το τεχνικό μειονέκτημα του Haar, είναι ότι δεν είναι συνεχής και επομένως δεν μπορεί να διαφοροποιηθεί.Το κυματίδιο Haar είναι επίσης γνωστό και ως Db1 από το κύμα Daubechies.Η συνάρτηση κύματος, κυματιδίων του Haar wavelet ψ(t) μπορεί να περιγραφεί και ως:

$$\psi(t) = \begin{cases} 1 & 0 \le t \le \frac{1}{2} \\ -1 & \frac{1}{2} \le t \le 1 \\ 0 & \sigma\varepsilon\ διαφορετική\ περίπτωση \end{cases}$$

Η λειτουργία κλιμάκωσης $\varphi(t)$ μπορεί να περιγραφεί ως :

$$\varphi(t) = \begin{cases} 1 & 0 \le t \le 1 \\ 0 & \sigma\varepsilon\ διαφορετική\ περίπτωση \end{cases}$$

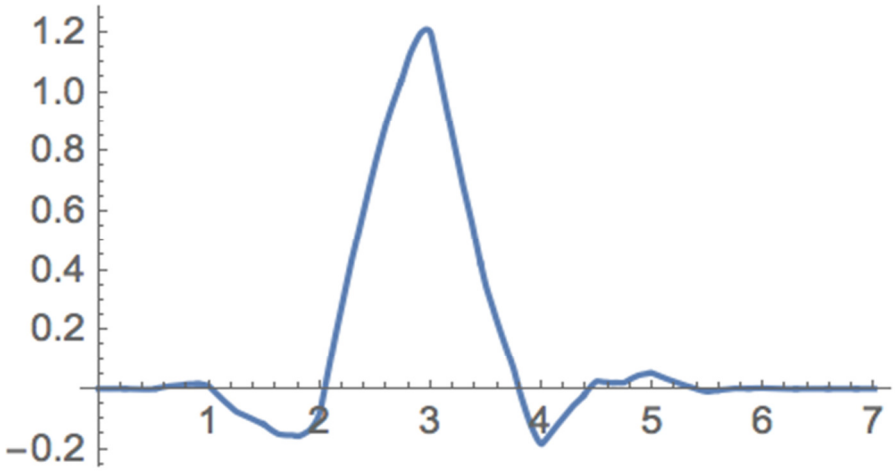
Το κύμα Haar έχει πολλές αξιοσημείωτες ιδιότητες. Κάθε συνεχής πραγματική συνάρτηση με συμπαγή μορφή μπορεί να προσεγγιστεί ομοιόμορφα από γραμμικούς συνδυασμούς των φ(t), φ(2t), φ(4t),… $\varphi(2^n$,t) και τις μετατοπισμένες λειτουργίες τους. Αυτό επεκτείνεται σε εκείνους τους χώρους λειτουργίας, όπου οποιαδήποτε λειτουργία σε αυτά μπορεί να προσεγγιστεί με συνεχείς λειτουργίες. Κάθε συνεχής πραγματική συνάρτηση στο [0,1] μπορεί να προσεγγιστεί ομοιόμορφα στο [0,1] με γραμμικούς συνδυασμούς της σταθερής συνάρτησης 1, ψ(t), ψ(2t), ψ(4t), ψ(2^n,t) και τις μετατοπισμένες λειτουργίες τους.

Ορθογώνια με τη μορφή:

$$\int_{-\infty}^{\infty} 2^{(n+n1)/2}\ \psi(2^n t - k)\psi(2^{n1} t - k1)dt = \delta n, n1\ \delta k, k1. \text{ (4.1.1)}$$

Ο μετασχηματισμός Haar είναι ο απλούστερος από τους μετασχηματισμούς wavelet.Ο μετασχηματισμός Haar πολλαπλασιάζει μια συνάρτηση ενάντια στο κύμα Haar με διάφορες μετατοπίσεις και εκτάσεις, όπως και η διασταύρωση του μετασχηματισμού Fourier πολλαπλασιάζει μια λειτουργία έναντι ενός ημιτονοειδούς κύματος με δύο φάσεις και πολλές εκτάσεις.

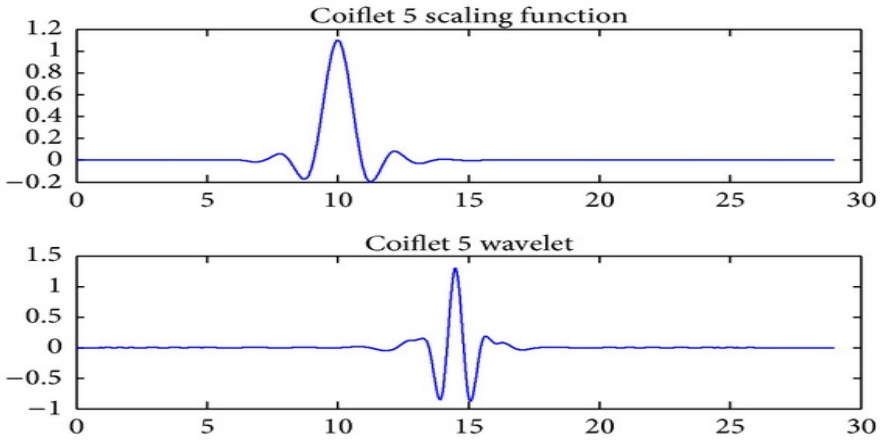
4.1 Symlet Wavelet



Εικόνα 15 : Symlet Wavelet

Τα symlet wavelet είναι επίσης γνωστά ως ελάχιστα ασύμμετρα κυματίδια Daubechies.Τα συμμετρικά σύμβολα είναι πιο συμμετρικά από τα κύματα ακραίων φάσεων. Στο Symlet το N είναι ο αριθμός των στιγμών εξαφάνισης. Επιπλέον έχουν υψηλότερο αριθμό στιγμών εξαφάνισης για ένα δεδομένο πλάτος στήριξης. Τα συσχετισμένα φίλτρα κλιμάκωσης βρίσκονται κοντά σε φίλτρα γραμμικής φάσης. Το σύντομο σύμβολο ονόματος τους είναι sym.Μερικά από αυτά χρησιμοποιούνται ως 2N αντί N.Πρόκειται για μια τροποποιημένη έκδοση των κυμάτων Daubechies. Η οικογένεια κυμάτων Daubechies είναι η πιο δημοφιλής οικογένεια wavelet που χρησιμοποιείται για την ανάλυση χαρακτηριστικών υφής, λόγω ορθογωνικών και συμπαγών δυνατοτήτων υποστήριξης. Το κύμα Daubechies χρησιμοποιεί επικαλυπτόμενα παράθυρα έτσι ώστε τα αποτελέσματα να αντανακλούν όλες τις αλλαγές μεταξύ των εντάσεων των εικονοστοιχείων. Δεδομένου ότι τα Daubechies έχουν μέσες τιμές για περισσότερα pixel, είναι πιο ομαλά από το κύμα Haar. Ο μετασχηματισμός Daubechies D4 έχει τέσσερις συντελεστές wavelet και κλίμακας. Το άθροισμα των συντελεστών της συνάρτησης κλιμάκωσης είναι επίσης ένα οπότε ο υπολογισμός είναι μέσο όρο σε τέσσερα γειτονικά εικονοστοιχεία.

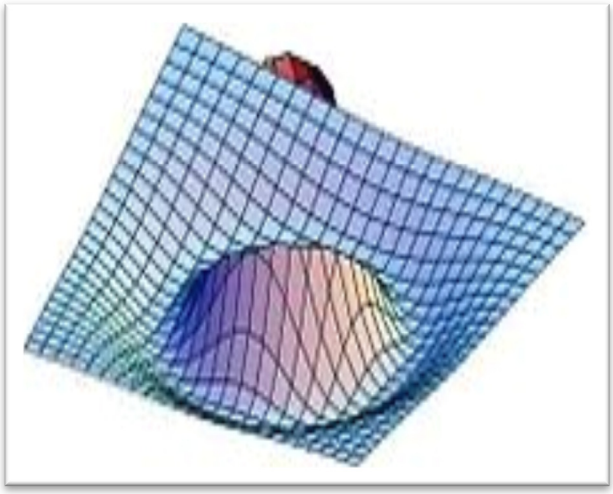
4.2 Coiflet Wavelet



Εικόνα 16: Coiflet 5 wavelet

Τα coiflets προήλθαν αρχικά από το κύμα Daubechies. Έχουν ένα ακόμα υψηλότερο υπολογιστικό κόστος και χρησιμοποιούν παράθυρα που επικαλύπτονται περισσότερο. Χρησιμοποιούν έξι συντελεστές συνάρτησης κλιμάκωσης και κύματος, έτσι η αύξηση του μέσου όρου των pixel και η διαφοροποίηση οδηγεί σε ένα ομαλότερο wavelet και σε αυξημένες δυνατότητες στις διάφορες τεχνικές επεξεργασίας εικόνας (απεικόνιση εικόνων κλπ.). Το φίλτρο ακολουθεί την ίδια δομή με το Haar και το Daubechies. Υπολογίζει και τους μέσους όρους και τις διαφορές χρησιμοποιώντας την ίδια μορφή, μόνο με έξι γειτονικά εικονοστοιχεία. Το κύμα Coiflet ακολουθεί επίσης την τεχνική του καθρέφτη. (Από την ταξινόμηση υφής των ιστών σε υπολογιστική τομογραφία με βάση το Wavelet).Η λειτουργία έχει στιγμές 2N ίσες με 0.Οι δύο λειτουργίες έχουν στήριξη μήκους 6N-1.Είναι συμπαγώς υποστηριζόμενα wavelets με τον υψηλότερο αριθμό στιγμών εξαφάνισης τόσο για το phi και για το psi λαμβάνοντας υπόψη το πλάτος υποστήριξης.

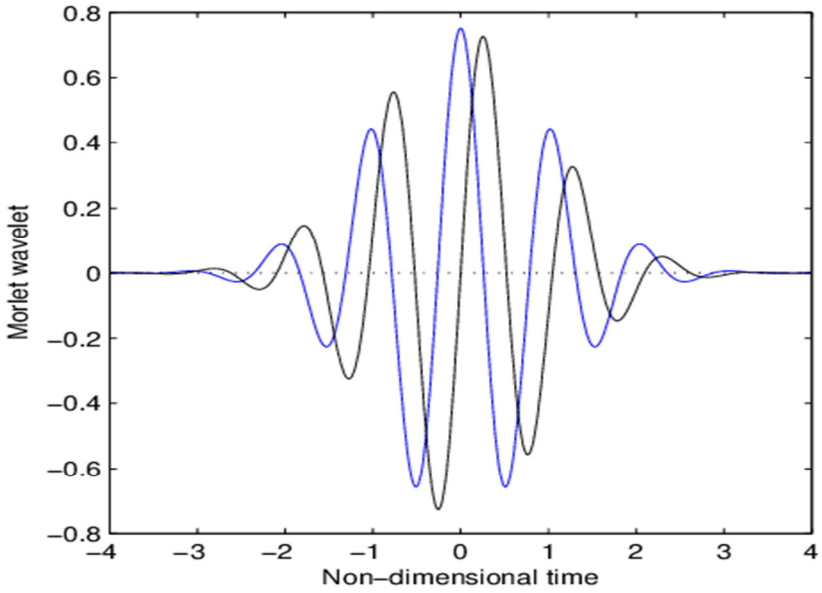
4.3 Mexican Hat Wavelet



Εικόνα 17: Mexican hat wavelet

Το κυματίδιο Mexican Hat είναι μια ειδική περίπτωση μιας μεγαλύτερης οικογένειας παραγώγων Gaussian και είναι γνωστή ως κύμα Ricker.Δεν υπάρχει συνάρτηση κλιμάκωσης που σχετίζεται με αυτό το κυμάτιο. Το κυματίδιο Ricker χρησιμοποιείται συχνά για να μοντελοποιήσει σεισμικά δεδομένα.Η ονομασία του ως κυματοειδές καπέλο του Μεξικού, στην Αμερική προέρχεται από τη μορφή του που μοιάζει με sombrero όπου χρησιμοποιείται ως πυρήνας επεξεργασίας 2D εικόνας.Η κλίμακα του χρησιμοποιείται συχνά ως ανιχνευτής κηλίδων και για την αυτόματη επιλογή κλίμακας σε εφαρμογές ηλεκτρονικών οραμάτων. Η πολυδιάστατη γενίκευση αυτού του κύματος ονομάζεται Laplacian της Gaussian λειτουργίας. Στην πράξη, αυτό το κύμα είναι μερικές φορές προσεγγισμένο από τη διαφορά της λειτουργίας Gaussians , διότι το DoG είναι διαχωρίσιμο και συνεπώς μπορεί να εξοικονομήσει σημαντικό χρόνο υπολογισμού σε δύο ή περισσότερες διαστάσεις.

4.4 Morlet Wavelet



Εικόνα 18: Morlet Wavelet

Η μέθοδος μετασχηματισμού κύματος Morlet εφαρμόζεται πολύ και στη μεταγραφή μουσικής καθώς παράγει πολύ ακριβή αποτελέσματα που δεν είναι δυνατά με τη χρήση τεχνικών μετασχηματισμού Fourier.Ο μετασχηματισμός κύματος Morlet είναι ικανός να καταγράφει σύντομες εκρήξεις επαναλαμβανόμενων και εναλασσόμενων μουσικών σημειώσεων με μια ξεκάθαρη ώρα έναρξης και λήξης για κάθε νότα. Επιπλέον έχει μεγάλη χρησιμότητα και στην ιατρική γιατί διακρίνει τη μη φυσιολογική συμπεριφορά του καρδιακού παλμού στο ηλεκτροκαρδιογράφημα. Αυτό συμβαίνει γιατί οι μεταβολές του παθολογικού καρδιακού παλμού είναι μη στατικό σήμα. Το κύμα Morlet είναι ένα κυματίδιο που αποτελείται από ένα σύνθετο εκθετικό πολλαπλασιασμένο με ένα Gaussian παράθυρο. Το κύμα αυτό είναι στενά συνδεδεμένο με την ανθρώπινη αντίληψη. Η παράμετρος σ στο κύμα Morlet επιτρέπει τις συναλλαγές μεταξύ αναλύσεων χρόνου και συχνότητας. Το κύριο χαρακτηριστικό του κύματος Morlet είναι ότι οι παράμετροι κλίμακας του είναι πολύ στενά συνδεδεμένες με το μήκος κύματος Fourier και ο μετασχηματισμός τους παρέχει πληροφορίες φάσης.

5. ΔΙΣΔΙΑΣΤΑΤΟΣ ΣΥΝΕΧΗΣ ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΣ WAVELET

Ο CWT μιας δισδιάστατης συνάρτησης $f(t)$ είναι μια συνάρτηση που περιέχει δύο μεταβλητές. Με αυτόν τον τρόπο αυξάνει κατά μια μεταβλητή την αρχική συνάρτηση. Ο CWT παρουσιάζει μια μεγάλη αύξηση στις πληροφορίες και στον απαιτούμενο όγκο αποθήκευσης και δεδομένων. Για τη συνάρτηση $f(x,y)$ ο Cwt θα είναι:

$$\mathrm{T}\, f(x,y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} f(x,y)\Psi_{\alpha}, b_{x,}b_{y}\text{από}\,(x,y)dxdy \quad (5.1.1)$$

όπου τα b_x και b_y δηλώνουν την μετάθεση στις δύο διαστάσεις.Ο αντίστροφος Cwt τότε θα είναι:

$$f(x,y) = C_{\psi}^{-1}\int_{0}^{+\infty}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} Tf\left(\alpha, b_x, b_y\right)\Psi_{\alpha}, b_x, b_y(x,y)\frac{d_{\alpha}}{\alpha^3}db_x, db_y \quad (5.1.2)$$

Όπου $\Psi_{\alpha}, b_x, b_y(x,y)$= $|(\alpha)|^1$ $\psi\left(\frac{x-b_x}{a}, \frac{x-b_y}{a}\right)$ και ψ(x,y) ένα δισδιάστατο βασικό κυματίδιο.

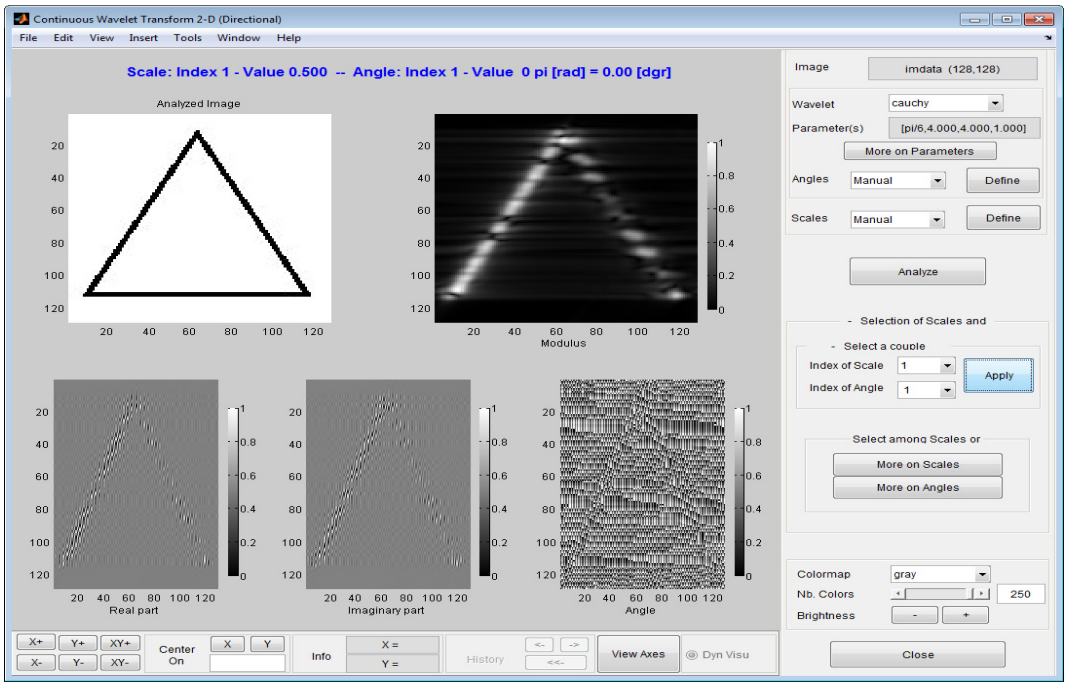
Η γενίκευση για συναρτήσεις μεγαλύτερες από δύο διαστάσεις μπορεί να επεκταθεί κάνοντας τον ίδιο τρόπο. Σε σύγκριση με τις παραδοσιακές μεθόδους φασματικής ανάλυσης, το 2D CWT παρέχει τοπικές φασματικές πληροφορίες του συνόλου δεδομένων που αναλύεται. Έχει επίσης το πλεονέκτημα έναντι του 2D διακριτού μετασχηματισμού κύματος (DWT), επειδή καλύπτει την περιοχή των δεδομένων που έχουν αναλυθεί με μια συνεχή ανάλυση από την οποία μπορούν να ληφθούν λεπτομερείς, αμετάβλητες φασματικές πληροφορίες διαφορετικών θέσεων και προσανατολισμών.

Το δισδιάστατο CWT είναι μια φυσική προέκταση του μονοδιάστατου CWT. Μπορεί γενικά να ορίζεται ως εξής:

$$(Wf)(a, \mathbf{b}, \theta) = \langle f, \psi_{a,\mathbf{b},\theta}\rangle \quad \text{and} \qquad (1)$$

$$\psi_{a,\mathbf{b},\theta}(\mathbf{x}) = a^{-1}\psi\left[\mathbf{R}_{\theta}^{-1}\left(\frac{\mathbf{x}-\mathbf{b}}{a}\right)\right], \qquad (2) \quad (5.1.3)$$

όπου $f \in L^2(R^2)$ είναι μια δισδιάστατη λειτουργία σήματος.



Εικόνα 19: Two-Dimensional CWT

Βασικό παράδειγμα σημάτων δύο διαστάσεων(two dimensional) αποτελούν οι εικόνες και μπορούν να αναλυθούν στα πλαίσια του δισδιάστατου μετασχηματισμού με την εξής διαδικασία. Το σήμα αναλύεται σε τέσσερα σήματα, με μέγεθος το καθένα ένα τέταρτο του αρχικού σε κάθε στάδιο του μετασχηματισμού. Κάθε ένα από τα τέσσερα αυτά σήματα, σχηματίζονται από το εσωτερικό γινόμενο του αρχικού σήματος με ένα από τα κυματιδιακά σήματα βάσης ακολουθούμενα από υποδειγματοληψία κατά 2 και ως προς τις δύο διαστάσεις. Τα εσωτερικά γινόμενα μπορούν να γραφούν και ως συνελίξεις.Σε περίπτωση που τα φίλτρα και η υποδειγματοληψία είναι διαχωρίσιμα τότε ο υπολογισμός έχει μεγαλύτερη απόδοση και ταχύτητα καθώς εφαρμόζεται ξεχωριστά στις γραμμές και στις στήλες της εικόνας. Αξίζει να σημειωθεί ότι η ανάλυση δισδιάστατων σημείων με τη χρήση κυματιδίων, θεωρείται από τις πιο ολοκληρωμένες προσεγγίσεις για τη μετάδοση, ανάλυση, συμπίεση και επεξεργασίας πληροφοριών ψηφιακών εικόνων, ήχων και μουσικής.

6. MATLAB

Στην ενότητα αυτή γίνεται αναφορά στο εργαλείο Matlab και παρουσιάζονται συνοπτικά το περιβάλλον δημιουργίας γραφικών διεπαφών GUIDE και η εργαλειοθήκη επεξεργασίας wavelets (Wavelet Toolbox) του Matlab. Τα εργαλεία αυτά χρησιμοποιήθηκαν για τη σχεδίαση και την ανάπτυξη των εφαρμογών συμπίεσης και αποθορυβοποίησης με χρήση wavelets, η οποία παρουσιάζεται στην επόμενη ενότητα.

8.1 Βασικά Χαρακτηριστικά

Το MATLAB είναι μία γλώσσα υψηλού επιπέδου, που χρησιμοποιείται για τεχνικούς υπολογισμούς. Το όνομα Matlab προέρχεται από τα αρχικά των λέξεων Matrix Laboratory και η λειτουργία του βασίζεται κυρίως στην χρήση πινάκων, τα στοιχεία των οποίων μπορούν να είναι πραγματικοί ή μιγαδικοί αριθμοί. Σχεδιάστηκε με αρχικό σκοπό την υπολογιστική επίλυση απλών και σύνθετων μαθηματικών προβλημάτων, ωστόσο σήμερα χρησιμοποιείται σχεδόν σε όλους τους επιστημονικούς κλάδους. Τα προγράμματα που είναι γραμμένα σε Matlab είναι γενικά σαφή, ευανάγνωστα και εύκολα στην κατανόηση. Η γλώσσα είναι αυστηρά δομημένη στη σύνταξη των εντολών και διαθέτει όλα τα χαρακτηριστικά μίας σύγχρονης γλώσσας προγραμματισμού, η οποία υποστηρίζεται από ένα εξελιγμένο προγραμματιστικό περιβάλλον με πληθώρα βοηθητικών εργαλείων. Τα προγράμματα σε Matlab είναι ανεξάρτητα της πλατφόρμας στην οποία εκτελούνται, ενώ ένα από τα βασικά χαρακτηριστικά τους είναι το γεγονός ότι ο προγραμματιστής δεν απαιτείται να ορίζει τον τύπο των μεταβλητών. Επιπλέον, η δυνατότητα του Matlab να κάνει συμβολικές πράξεις και ο μεγάλος αριθμός έτοιμων βιβλιοθηκών που διαθέτει, το κατατάσσει στα καλύτερα λογισμικά της κατηγορίας του (Marchand & Holland, 2003).

8.2 Το Περιβάλλον GUIDE

Το MATLAB διαθέτει ένα ενσωματωμένο περιβάλλον – εργαλειοθήκη για τη σχεδίαση φιλικών στον χρήστη γραφικών διεπαφών. Το περιβάλλον αυτό, που καλείται και ως GUIDE (Graphical User Interface Design Environment), διαθέτει μία πληθώρα έτοιμων στοιχείων ελέγχου, όπως πλήκτρα, πλαίσια, μενού, άξονες, πίνακες κοκ.

Μία γραφική διεπαφή αποτελείται από διάφορα παράθυρα, που μπορούν να διαθέτουν ποικίλα στοιχεία ελέγχου, όπως πεδία κειμένου, πλήκτρα, μπάρες κύλισης κοκ. Τα παράθυρα μπορούν να καλούν το ένα το άλλο, να δέχονται είσοδο από τον χρήστη, να επικοινωνούν μεταβιβάζοντας δεδομένα μεταξύ τους και γενικότερα να επιτελούν διάφορες λειτουργίες (Lent, 2013). Προκειμένου το GUIDE να το πετύχει αυτό, για κάθε ένα παράθυρο δημιουργεί δύο αρχεία, ένα Fig - file και ένα M - file. Πιο συγκεκριμένα:

- Το Fig – file είναι το παράθυρο (figure), όπου το MATLAB αποθηκεύει τα στοιχεία ελέγχου και την ακριβή τους θέση. Ουσιαστικά, στο αρχείο αυτό ο προγραμματιστής σχεδιάζει την εμφάνιση του παραθύρου.
- Το M – file είναι το αρχείο στο οποίο ο προγραμματιστής συγγράφει τον κώδικα που αφορά τη συμπεριφορά των στοιχείων ελέγχου του παραθύρου, προκειμένου το παράθυρο να επιτελεί τις επιθυμητές λειτουργίες.

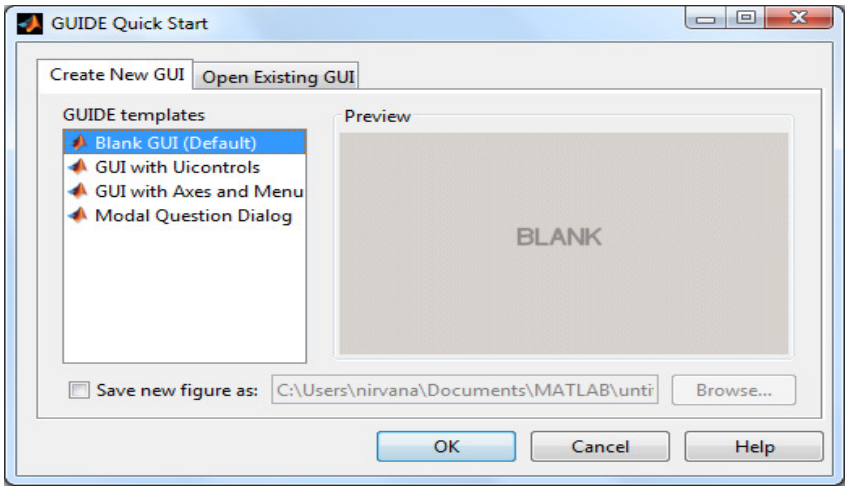
Επισημαίνεται ότι κάθε fig - file πρέπει να συνοδεύεται από ένα ομώνυμο αρχείο m – file. Αν χαθεί ή καταστραφεί το M – file, τότε το παράθυρο δε θα είναι λειτουργικό. Όποτε ο χρήστης δημιουργεί ένα νέο παράθυρο, το GUIDE δημιουργεί αυτομάτως και τους δύο προαναφερθέντες τύπους αρχείων (Βαρσάμης, 2006).

Η εκκίνηση του GUIDE γίνεται με δύο τρόπους:

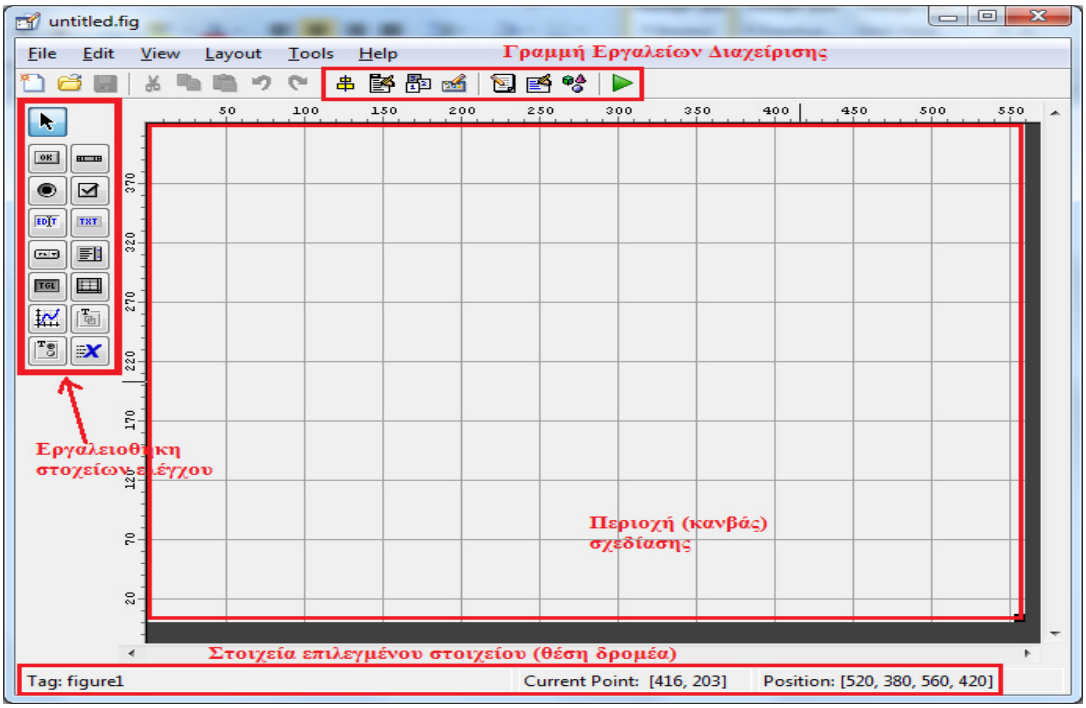
1. Γράφοντας την εντολή «guide» στη γραμμή εντολών του Matlab
2. Από το μενού του Matlab «File» → «New» → «GUI»

Αμέσως θα ξεκινήσει ένας οδηγός που θα καθοδηγήσει τον χρήστη στη δημιουργία ενός παραθύρου. Αρχικά ο χρήστης θα πρέπει να επιλέξει αν θέλει να δημιουργήσει ένα κενό παράθυρο (Blank GUI) ή ένα παράθυρο βασισμένο σε κάποια πρότυπα (πχ με άξονες και μενού) ή αν θέλει να ανοίξει ένα έτοιμο παράθυρο. Μόλις ο χρήστης επιλέξει το είδος του παραθύρου που επιθυμεί, τότε θα εμφανιστεί το νέο παράθυρο στο περιβάλλον σχεδίασης του GUIDE.

Στις ακόλουθες εικόνες (Εικόνα 20 και Εικόνα 21) φαίνονται ο οδηγός για τη δημιουργία ενός παραθύρου του GUIDE και το περιβάλλον σχεδίασης του νέου παραθύρου στο GUIDE. Τα βασικά τμήματα του περιβάλλοντος σχεδίασης του GUIDE εμφανίζονται με κόκκινη σήμανση.

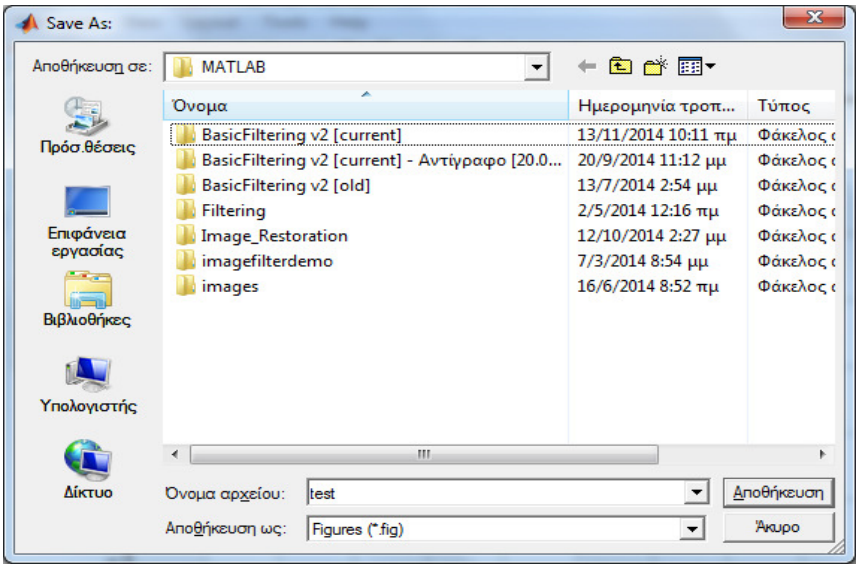


Εικόνα 20:Οδηγός δημιουργίας παραθύρου



Εικόνα 11: Περιβάλλον σχεδίασης νέου παραθύρου του GUIDE

Για την αποθήκευση του παραθύρου, ο χρήστης πρέπει να επιλέξει «File» → «Save as...», να επιλέξει τη θέση και το όνομα που θέλει να δώσει στο παράθυρο και να πατήσει το πλήκτρο «Αποθήκευση». Το Matlab θα δημιουργήσει το fig – file και το M – file και θα εμφανίσει σε ένα νέο παράθυρο τον κώδικα του M-file που αφορά το παράθυρο. Στην Εικόνα 22 φαίνεται η αποθήκευση του παραθύρου GUIDE.



Εικόνα 22: Αποθήκευση παραθύρου GUIDE

Στο παράθυρο σχεδίασης του GUIDE ο προγραμματιστής μπορεί να σχεδιάσει τη γραφική διεπαφή που θα εξυπηρετεί την εφαρμογή του. Για το σκοπό αυτόν, το GUIDE διαθέτει μία εργαλειοθήκη στοιχείων ελέγχου, όπου ο προγραμματιστής μπορεί να επιλέξει και να σύρει πάνω στον καμβά σχεδίασης το επιθυμητό στοιχείο.

Στον πίνακα που ακολουθεί (

Πίνακας 1) περιγράφεται η λειτουργία καθενός στοιχείου ελέγχου της εργαλειοθήκης του GUIDE. Επισημαίνεται ότι ο προγραμματιστής μπορεί να γράψει κώδικα για οποιοδήποτε στοιχείο ελέγχου, προκειμένου να καθορίσει τη συμπεριφορά του κατά την εκτέλεση της εφαρμογής.

Πίνακας 1: Στοιχεία ελέγχου της εργαλειοθήκης του GUIDE

Εικονίδιο	**Λεκτικό**	**Περιγραφή**
	Select	Εργαλείο επιλογής και μετακίνησης αντικειμένων.
	Push Button	Δημιουργεί ένα πλήκτρο, το οποίο εκτελεί μια διαδικασία (Callback) όταν πατηθεί από τον χρήστη κατά την εκτέλεση της εφαρμογής.
	Slider	Δημιουργεί μία μπάρα κύλισης, η οποία δέχεται μία αριθμητική είσοδο εντός ενός πεδίου τιμών που ορίζει ο προγραμματιστής.
	Radio Button	Δημιουργεί ένα πλήκτρο τύπου radio. Ο προγραμματιστής πρέπει να φροντίζει ώστε όταν κατά την εκτέλεση της εφαρμογής ο χρήστης επιλέξει κάποιο radio button, τότε να απενεργοποιούνται τα υπόλοιπα radio buttons. Χρησιμοποιείται όταν θέλουμε να δώσουμε τη δυνατότητα στον χρήστη να επιλέξει μόνο μία επιλογή από ένα πλήθος διαθέσιμων επιλογών.
	Check Box	Δημιουργεί ένα πλήκτρο τύπου check. Μοιάζει με ένα radio button, όμως όταν επιλέγεται (τσεκάρεται) από τον χρήστη κατά την εκτέλεση της εφαρμογής, δεν χρειάζεται να απενεργοποιούνται τα υπόλοιπα.
	Edit Text	Δημιουργεί ένα πλαίσιο μέσα στο οποίο ο χρήστης μπορεί να εισάγει δεδομένα μέσω του πληκτρολογίου κατά την εκτέλεση της εφαρμογής. Στη συνέχεια, τα δεδομένα αυτά μπορεί να τα αξιοποιήσει η εφαρμογή.
	Static Text	Επιτρέπει τη δημιουργία στατικών κειμένων, για την παροχή πληροφοριών προς τον χρήστη κατά την εκτέλεση της εφαρμογής.
	Pop-Up Menu	Δημιουργεί μία αναπτυσσόμενη λίστα επιλογών.
	List Box	Δημιουργεί μία απλή λίστα επιλογών.
	Toggle Button	Δημιουργεί ένα πλήκτρο τύπου «toggle». Μοιάζει με ένα απλό πλήκτρο (push button) με τη διαφορά ότι μπορεί να έχει δύο τιμές, τις «On» και «Off».

	Table	Χρησιμοποιείται για την οργάνωση και την παρουσίαση δεδομένων σε μορφή πίνακα.
	Axes	Επιτρέπει την τοποθέτηση γραφικών στοιχείων, όπως εικόνων και γραφημάτων.
	Panel	Χρησιμοποιείται για την ομαδοποίηση πολλών εργαλείων.
	Button Group	Χρησιμοποιείται για την ομαδοποίηση πολλών radio buttons ή toggle buttons.
	ActiveX Control	Επιτρέπει την προσθήκη και εμφάνιση ActiveX χειριστηρίων. Χρησιμοποιείται μόνο στα Windows.

Επιπλέον, το περιβάλλον του GUIDE διαθέτει μία γραμμή εργαλείων διαχείρισης, η οποία παρέχει άμεση πρόσβαση σε χρήσιμα εργαλεία. Η γραμμή των εργαλείων διαχείρισης φαίνεται στην Εικόνα 23.

Εικόνα 23: Γραμμή εργαλείων διαχείρισης του GUIDE

Στον πίνακα που ακολουθεί (

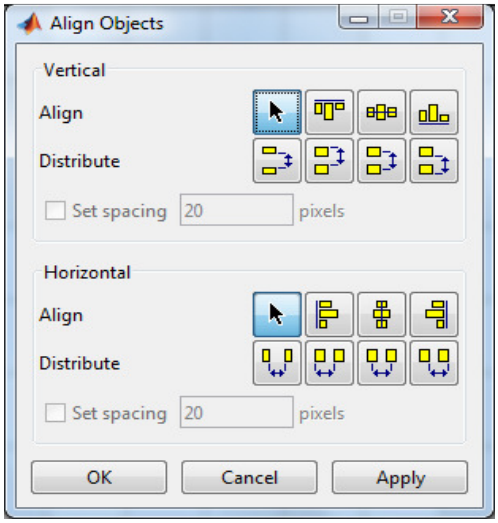
Πίνακας 2) περιγράφεται η λειτουργία καθενός στοιχείου της γραμμής εργαλείων διαχείρισης του GUIDE.

Πίνακας 2: Στοιχεία γραμμής εργαλείων διαχείρισης του GUIDE

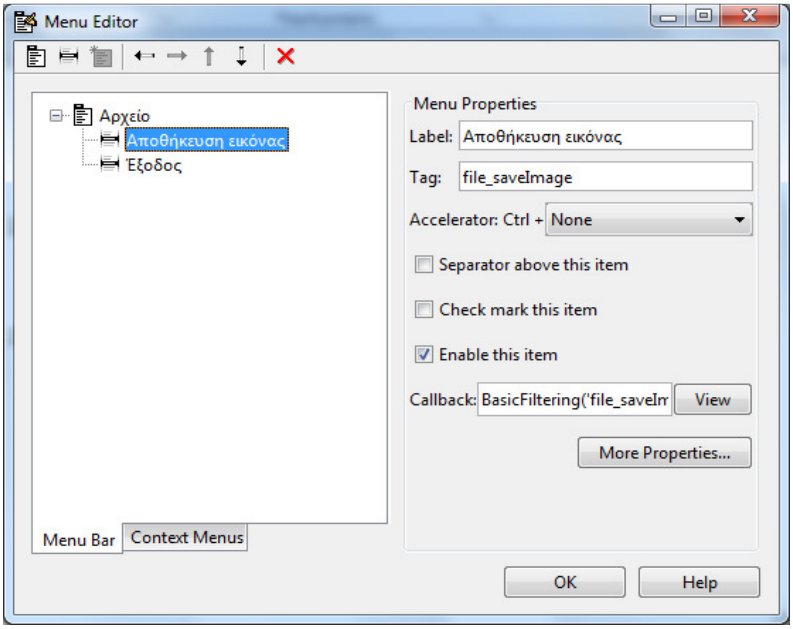
Εικονίδιο	**Λεκτικό**	**Περιγραφή**
	Align Objects	Εργαλείο για τη στοίχιση των αντικειμένων – στοιχείων ελέγχου στο παράθυρο της εφαρμογής.
	Menu Editor	Επιτρέπει τη δημιουργία και διαχείριση ενός βασικού μενού για το παράθυρο της εφαρμογής. Επιπλέον, επιτρέπει και τη δημιουργία αναπτυσσόμενων μενού, που ενεργοποιούνται όταν ο χρήστης κάνει δεξί κλικ κατά την εκτέλεση της εφαρμογής.
	Tab Order Editor	Χρησιμεύει για τον ορισμό της σειράς με την οποία τα στοιχεία ελέγχου του παραθύρου αποκτούν την εστίαση, όταν ο χρήστης πατάει το πλήκτρο «Tab» κατά την εκτέλεση της εφαρμογής.
	Toolbar Editor	Επιτρέπει τη δημιουργία γραμμής εργαλείων στο παράθυρο της εφαρμογής.
	Editor	Ανοίγει το παράθυρο επεξεργασίας του κώδικα του παραθύρου, ο οποίος βρίσκεται στο ομώνυμο M – file.

	Property Inspector	Πολύ χρήσιμο εργαλείο για τον προγραμματιστή, αφού παρέχει άμεση πρόσβαση στις ιδιότητες του επιλεγμένου στοιχείου ελέγχου. Οι ιδιότητες αυτές εμφανίζονται σε ένα παράθυρο.
	Object Browser	Επίσης πολύ χρήσιμο εργαλείο για τον προγραμματιστή, αφού εμφανίζει την ιεραρχία των στοιχείων ελέγχου που έχει τοποθετήσει στο παράθυρο της εφαρμογής.
	Run Figure	Εκτελεί το παράθυρο της εφαρμογής.

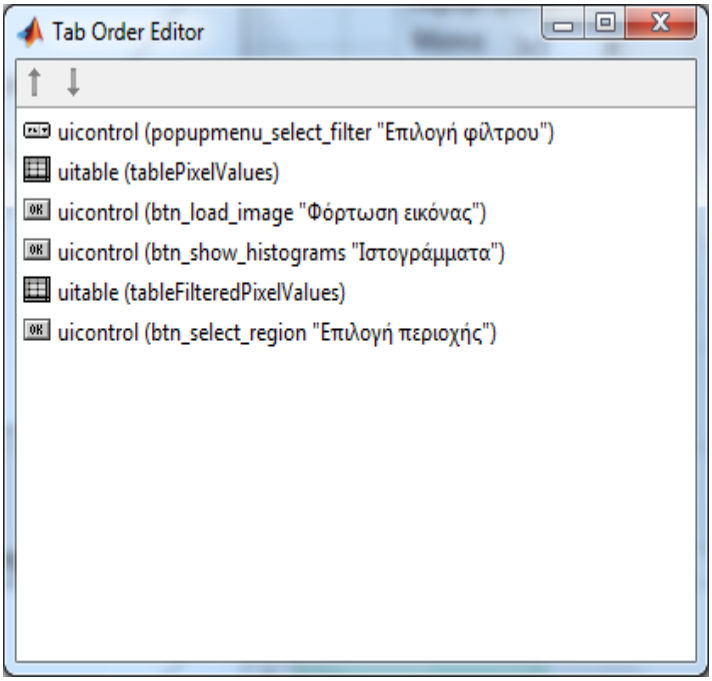
Επίσης, στις παρακάτω εικόνες (Εικόνα 23 ως Εικόνα 29) παρουσιάζονται τα παράθυρα που αντιστοιχούν στα στοιχεία της γραμμής εργαλείων διαχείρισης του GUIDE.



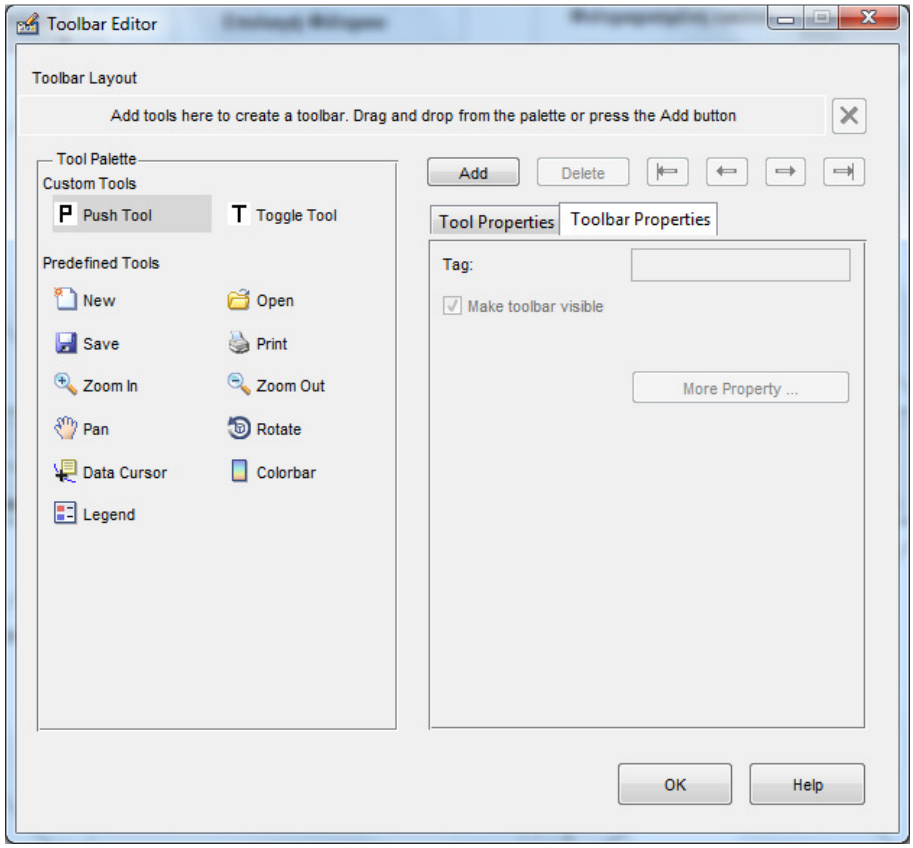
Εικόνα 23: Παράθυρο Align Objects του GUIDE



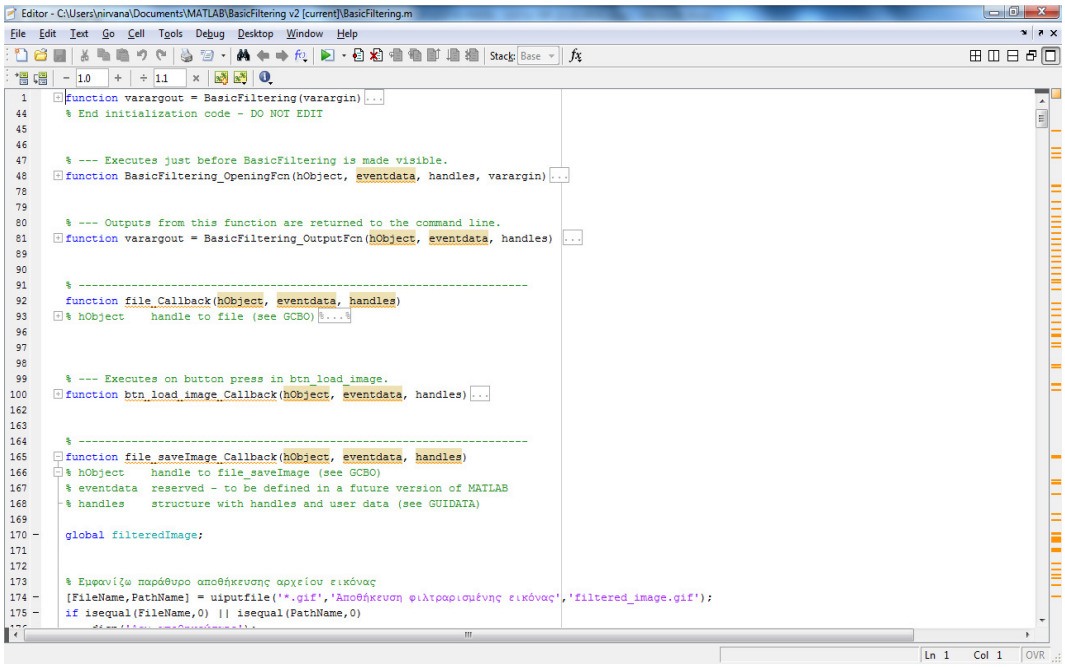
Εικόνα 24: Παράθυρο Menu Editor του GUIDE



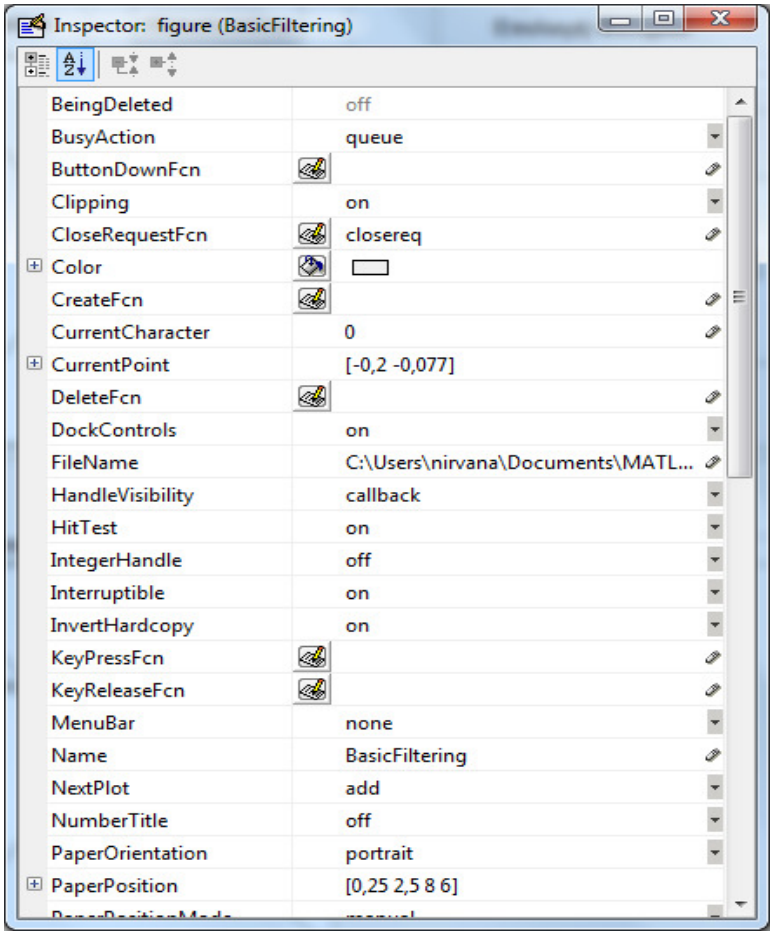
Εικόνα 25: Παράθυρο Tab Order Editor του GUIDE



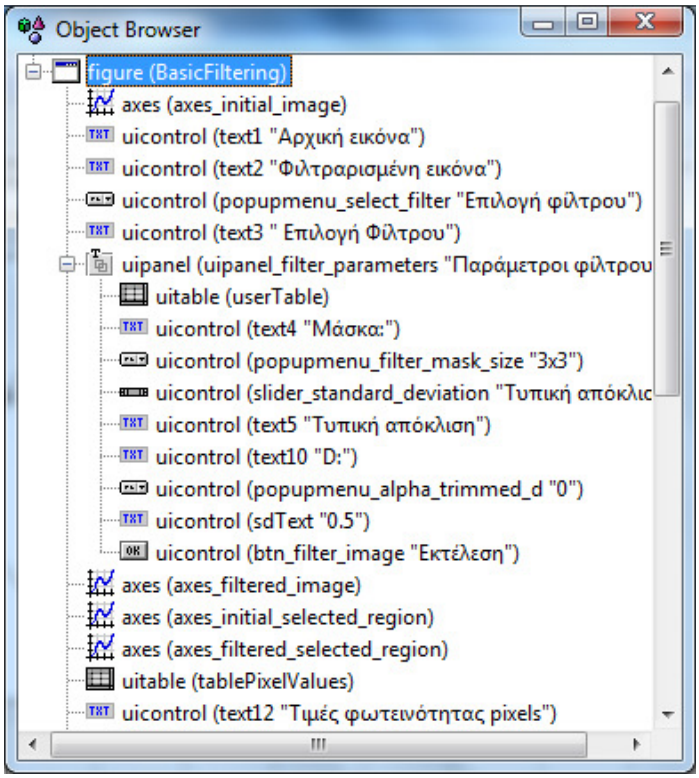
Εικόνα 26: Παράθυρο Toolbar Editor του GUIDE



Εικόνα 27: Παράθυρο Editor του GUIDE



Εικόνα 2: Παράθυρο Property Inspector του GUIDE



Εικόνα 29: Παράθυρο Object Browser του GUIDE

8.3 Εργαλειοθήκη Wavelet (Wavelet Toolbox)

Το Wavelet Toolbox παρέχει λειτουργίες και εφαρμογές για την ανάλυση και τη σύνθεση σημάτων και εικόνων. Η εργαλειοθήκη περιλαμβάνει αλγόριθμους για συνεχή ανάλυση wavelets, συσχέτιση wavelets, συγχρονιστική ανάλυση και ανάλυση χρόνου-συχνότητας προσαρμοσμένης στα δεδομένα. Η εργαλειοθήκη περιλαμβάνει επίσης εφαρμογές και λειτουργίες για την διακριτή ανάλυση wavelets σημάτων και εικόνων, συμπεριλαμβανομένων πακέτων wavelet και των μετασχηματισμών dual-tree.

Χρησιμοποιώντας τη συνεχή ανάλυση wavelet, δίνεται η δυνατότητα να μελετηθεί ο τρόπος με τον οποίο εξελίσσονται χρονικά τα φασματικά χαρακτηριστικά, να εντοπιστούν κοινά χρονικά μεταβαλλόμενα πρότυπα σε δύο σήματα και να πραγματοποιηθεί φιλτράρισμα με βάση το χρόνο. Χρησιμοποιώντας τη διακριτή ανάλυση κύματος, δίνεται η δυνατότητα να αναλυθούν σήματα και εικόνες σε διαφορετικές αναλύσεις για να ανιχνευθούν σημεία αλλαγής, ασυνέχειες και άλλα συμβάντα που δεν είναι άμεσα ορατά στα πρωτογενή δεδομένα. Επίσης, είναι δυνατό να συγκριθούν τα στατιστικά μέτρα στων σημάτων σε πολλαπλές κλίμακες και να διενεργηθεί ανάλυση fractals σε δεδομένα για να αποκαλυφθούν κρυμμένα μοτίβα.

Με το εργαλείο Wavelet Toolbox μπορούμε να αποκτήσουμε μια αραιή αναπαράσταση των δεδομένων μας, η οποία είναι χρήσιμη για την αποθορυβοποίηση ή τη συμπίεση των δεδομένων διατηρώντας ταυτόχρονα τα σημαντικά χαρακτηριστικά τους. Πολλές λειτουργίες – συναρτήσεις της εργαλειοθήκης wavelets υποστηρίζουν την παραγωγή κώδικα σε C/C++ για πρωτότυπα desktop εφαρμογών και την ανάπτυξη ενσωματωμένων συστημάτων (MathWorks.com, 2012).

Στον παρακάτω πίνακα (Πίνακας 3) φαίνονται οι βασικότερες συναρτήσεις της εργαλειοθήκης wavelet και μία σύντομη περιγραφή τους.

Πίνακας 3: Βασικές συναρτήσεις της εργαλειοθήκης επεξεργασίας εικόνας

Συνάρτηση	**Περιγραφή**
wavemngr	Εμφανίζει πληροφορίες για τις διαθέσιμες οικογένειες wavelets
waveinfo	Εμφανίζει πληροφορίες για το είδος ανάλυσης που υποστηρίζεται από το κάθε wavelet
cwt	Συνεχής 1-D μετασχηματισμός wavelet
cwtft2	Συνεχής 2-D μετασχηματισμός wavelet
wavedec	1-D wavelet αποσύνθεση
waverec	1-D wavelet ανασύνθεση
appcoef	1-D συντελεστές προσέγγισης
detcoef	1-D συντελεστές λεπτομερειών
wavedec2	2-D wavelet αποσύνθεση
waverec2	2-D wavelet ανασύνθεση
appcoef2	2-D συντελεστές προσέγγισης
detcoef2	2-D συντελεστές λεπτομερειών
wdenoise	Αποθορυβοποίηση σήματος με wavelets
ddencmp	Προκαθορισμένες τιμές για αποθορυβοποίηση ή συμπίεση
measerr	Μετρικές ποιότητας για προσέγγιση σήματος ή εικόνας
wdencmp	Αποθορυβοποίηση ή συμπίεση
wcompress	Πραγματική συμπίεση εικόνων με χρήση wavelets

9. Σχεδίαση και Υλοποίηση Εφαρμογών

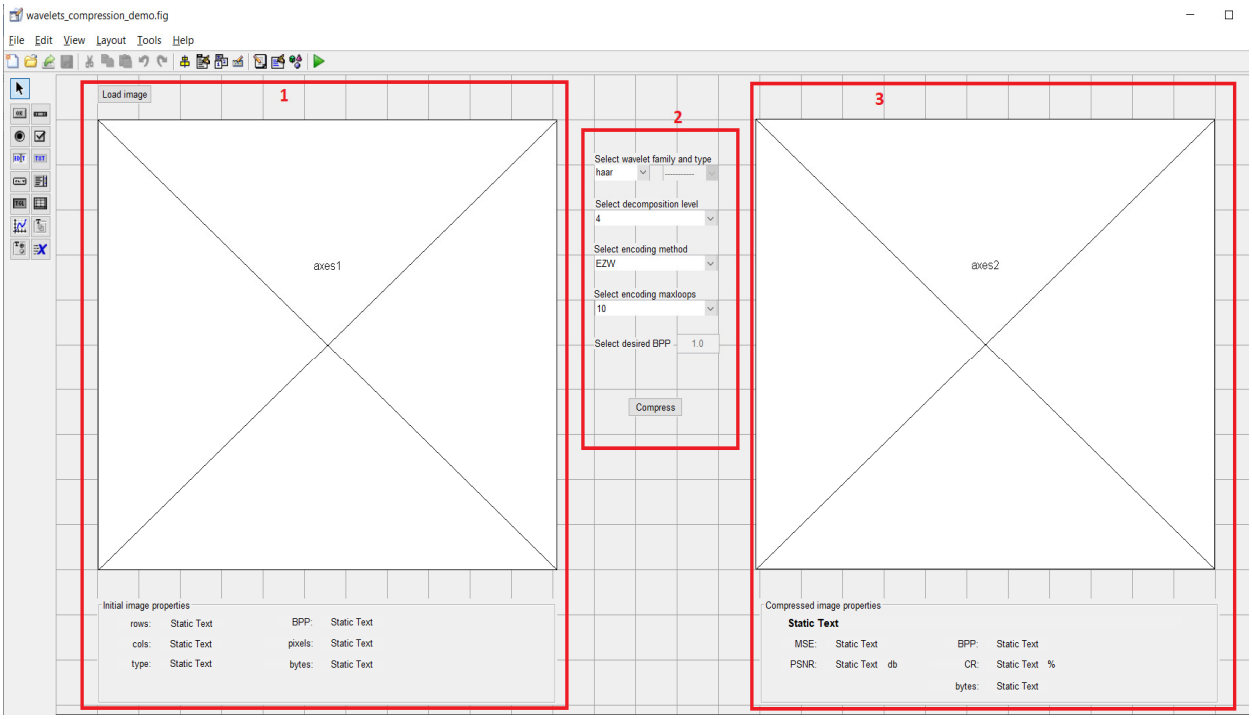
Στην ενότητα αυτή παρουσιάζεται η σχεδίαση και η υλοποίηση δύο εφαρμογών που αξιοποιούν τις δυνατότητες συμπίεσης εικόνων και αποθορυβοποίησης σημάτων με χρήση των μετασχηματισμών wavelets, με το περιβάλλον GUIDE και την εργαλειοθήκη wavelet (Wavelet Toolbox) του Matlab έκδοσης 2012a.

10.1 Εφαρμογή Συμπίεσης Εικόνας με Χρήση Wavelets

Το παράθυρο της εφαρμογής ονομάζεται **wavelets_compression_demo.fig** και αποτελείται από τρία (3) τμήματα με τα εξής στοιχεία:

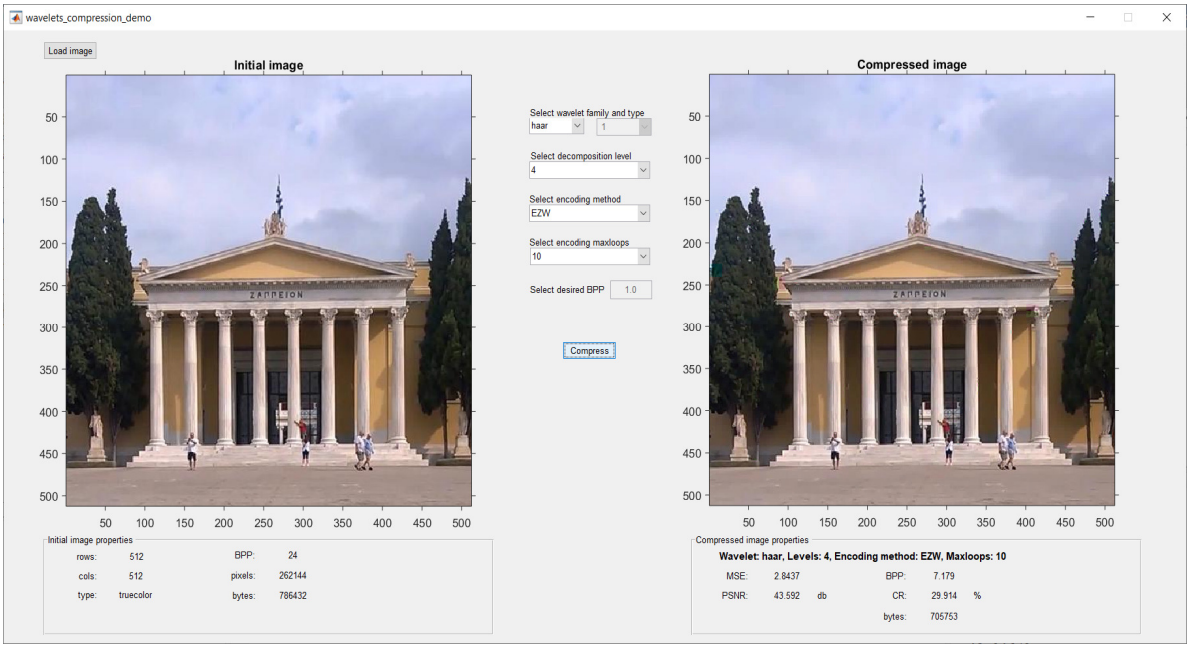
1. Πλήκτρο φόρτωσης της εικόνας που θα συμπιεστεί, περιοχή εμφάνισης της εικόνας και ιδιότητες της εικόνας, όπως: διαστάσεις, τύπος, BPP (bits ανά pixel), συνολικά pixels και bytes που απαιτούνται για την αποθήκευσή της.
2. Επιλογή παραμέτρων του μετασχηματισμού wavelet για τη συμπίεση της εικόνας, όπως: επιλογή wavelet (οικογένειας και τύπου), αριθμός επιπέδων αποσύνθεσης, μέθοδος κωδικοποίησης, μέγιστο πλήθος κύκλων κωδικοποίησης, επιθυμητός αριθμός BPP για τη συμπιεσμένη εικόνα και πλήκτρο εκτέλεσης συμπίεσης.
3. Περιοχή εμφάνισης της τελικής – συμπιεσμένης εικόνας και ιδιότητές της, όπως: MSE (μέσο τετραγωνικό σφάλμα), PSNR (λόγος σήματος προς θόρυβο), BPP, CR (λόγος συμπίεσης) και bytes που απαιτούνται για την αποθήκευση της συμπιεσμένης εικόνας.

Στην εικόνα που ακολουθεί (**Εικόνα**) παρουσιάζεται το παράθυρο της εφαρμογής και τα χειριστήριά του, όπως φαίνονται στο περιβάλλον GUIDE. Οι διακριτές κόκκινες περιοχές αντιπροσωπεύουν τα τμήματα του παραθύρου.



Εικόνα 30: Το παράθυρο εφαρμογής, όπως φαίνεται στο GUIDE του Matlab. Τα τμήματα του παραθύρου εμφανίζονται με κόκκινη σήμανση

Στην εικόνα που ακολουθεί (**Εικόνα**) παρουσιάζεται το παράθυρο και τα χειριστήρια της εφαρμογής κατά τον χρόνο εκτέλεσης.



Εικόνα 31: Το παράθυρο (wavelets_compression_demo.fig) κατά τον χρόνο εκτέλεσης

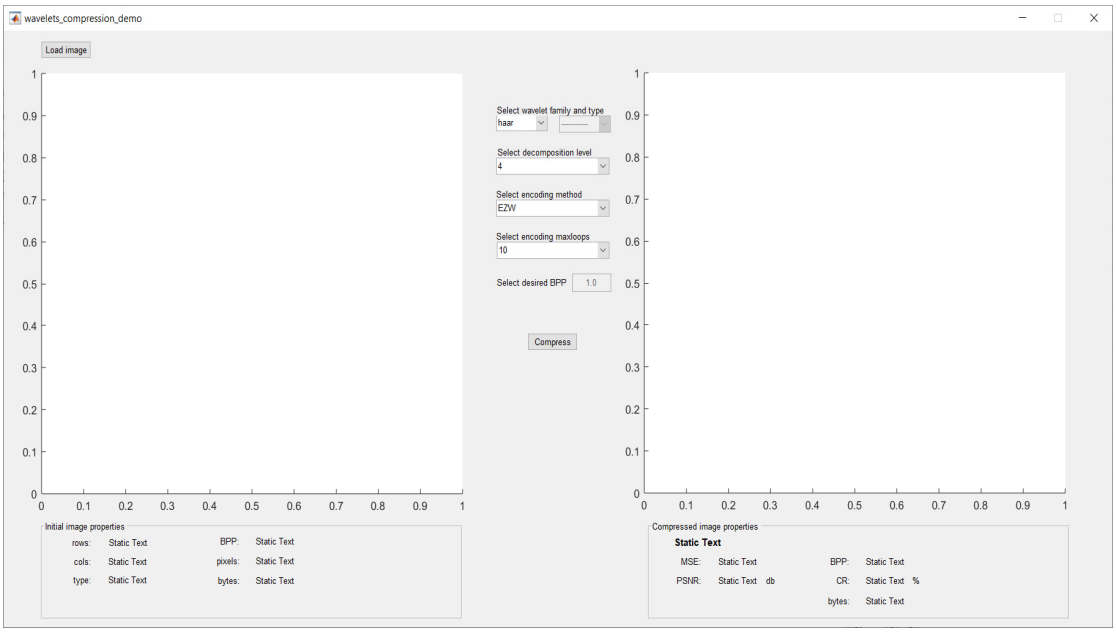
10.1.1 Βασικό Σενάριο Χρήσης Εφαρμογής Συμπίεσης Εικόνας με Χρήση Wavelets

Στην υποενότητα αυτή παρουσιάζεται το βασικό σενάριο χρήσης της εφαρμογής συμπίεσης εικόνας με χρήση wavelets, το οποίο συνοψίζεται στα εξής βήματα:

1. Φόρτωση εικόνας (διαστάσεων δύναμης του 2) και εμφάνιση ιδιοτήτων της
2. Επιλογή παραμέτρων συμπίεσης, όπως τύπος wavelet κλπ.
3. Εκτέλεση συμπίεσης
4. Εμφάνιση συμπιεσμένης εικόνας και των ιδιοτήτων της
5. Εμφάνιση σύγκρισης ιστογραμμάτων αρχικής και συμπιεσμένης εικόνας

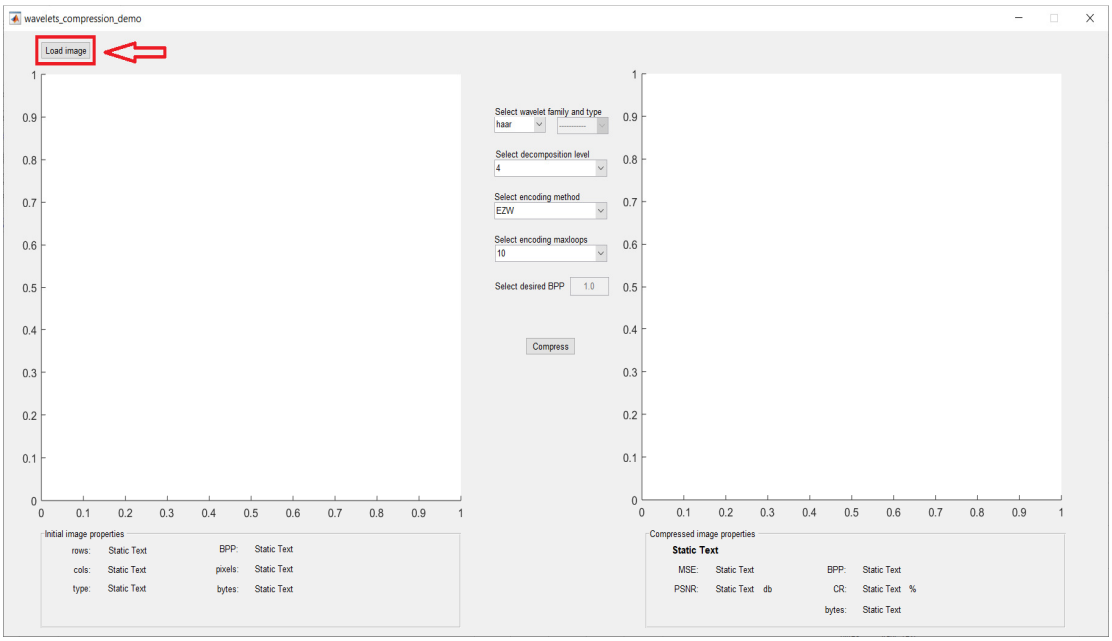
Ακολουθεί η αναλυτική περιγραφή του καθενός βήματος.

Έχοντας εκτελέσει την εφαρμογή μέσα από το περιβάλλον GUIDE του Matlab, ο χρήστης έχει πρόσβαση στην αρχική οθόνη της εφαρμογής. Στην επόμενη εικόνα (**Εικόνα**) φαίνεται η αρχική οθόνη της εφαρμογής.

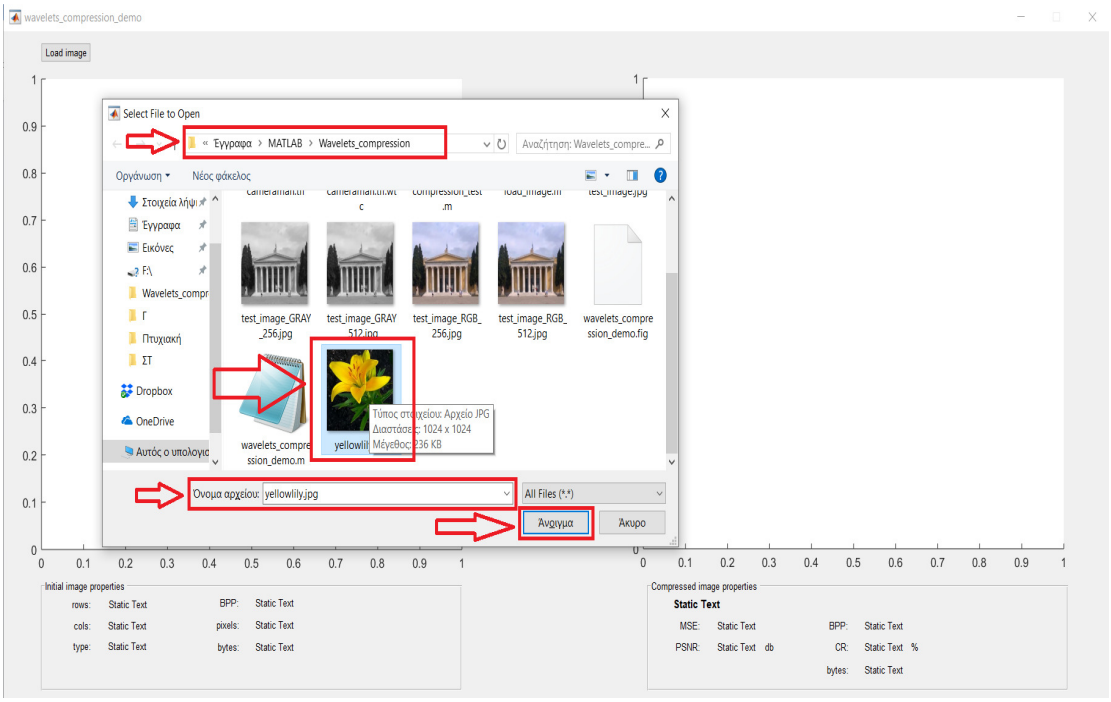


Εικόνα 32: Αρχική οθόνη εφαρμογής

Ακολούθως, πατώντας το πλήκτρο «Load image» εμφανίζεται ένα παράθυρο διαλόγου, όπου ο χρήστης μπορεί να πλοηγηθεί στους φακέλους του, να επιλέξει την εικόνα που επιθυμεί και πατώντας το πλήκτρο «Άνοιγμα» να φορτώσει την επιλεγμένη εικόνα στην εφαρμογή, προκειμένου να την επεξεργαστεί. Στις ακόλουθες εικόνες (**Εικόνα** και **Εικόνα 3**) φαίνεται η διαδικασία αυτή.

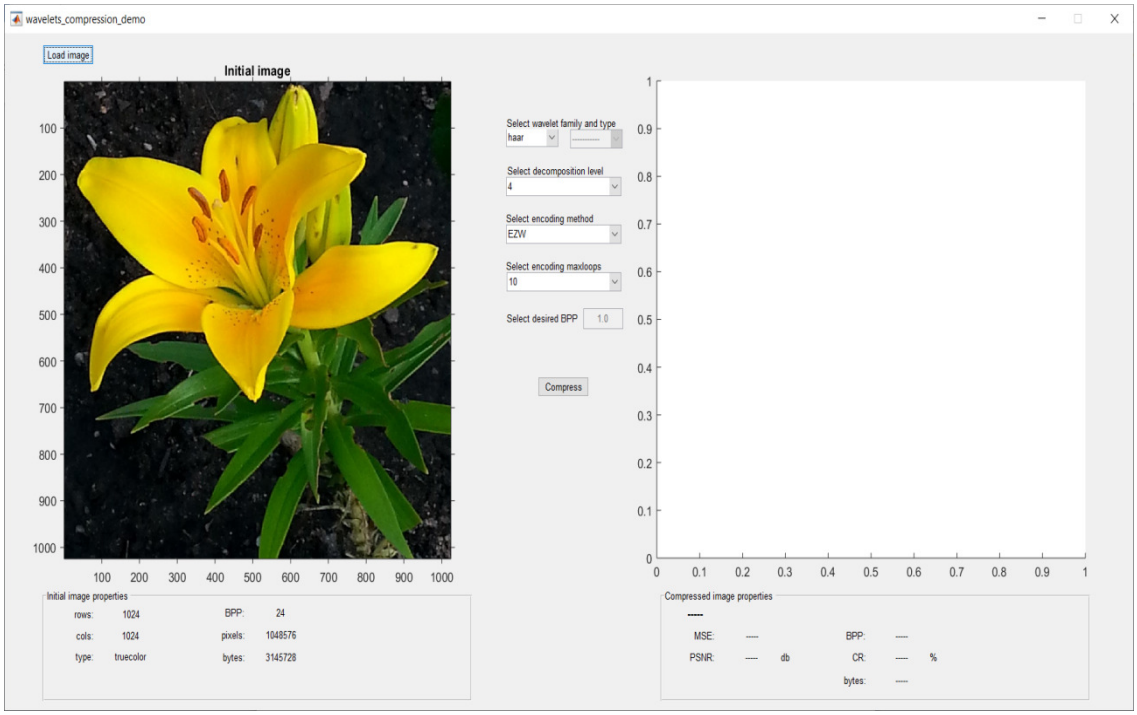


Εικόνα 33: Πλήκτρο «Φόρτωση εικόνας»



Εικόνα 3: Παράθυρο επιλογής και φόρτωσης εικόνας

Μόλις ο χρήστης επιλέξει το πλήκτρο «Άνοιγμα», τότε φορτώνεται η εικόνα και ενημερώνεται η οθόνη της εφαρμογής εμφανίζοντας την επιλεγμένη εικόνα με τις βασικές της ιδιότητες, όπως διαστάσεις, BPP, bytes κλπ, (**Εικόνα**).



Εικόνα 35: Ενημέρωση οθόνης εφαρμογής μετά την φόρτωση της εικόνας

Στη συνέχεια ο χρήστης μπορεί να επιλέξει την οικογένεια και τον τύπο του μετασχηματισμού wavelet που θα χρησιμοποιηθεί για τη συμπίεση. Τα wavelets που υποστηρίζονται είναι τα εξής:

- haar
- db1, .., db10 (Daubechies)
- sym2, .., sym8 (Symlets)
- coif1, .., coif5 (Coiflets)
- bior1.1, bior1.3, bior1.5, …, bior6.8 (BiorSplines)
- rbio1.1, rbio1.3, rbio1.5, …, rbio6.8 (ReverseBior)

Ακολούθως, ο χρήστης πρέπει να εισάγει το πλήθος των επιπέδων αποσύνθεσης του μετασχηματισμού (decomposition levels) επιλέγοντας μία τιμή από το 1 ως το 8 (default τιμή είναι η 4), καθώς και τη μέθοδο της κωδικοποίησης (encoding method) που θα χρησιμοποιηθεί κατά τη συμπίεση. Οι μέθοδοι κωδικοποίησης διακρίνονται στις ακόλουθες 2 κατηγορίες:

- Progressive Coefficients Significance Methods (PCSM) και
- Coefficients Thresholding Methods (CTM)

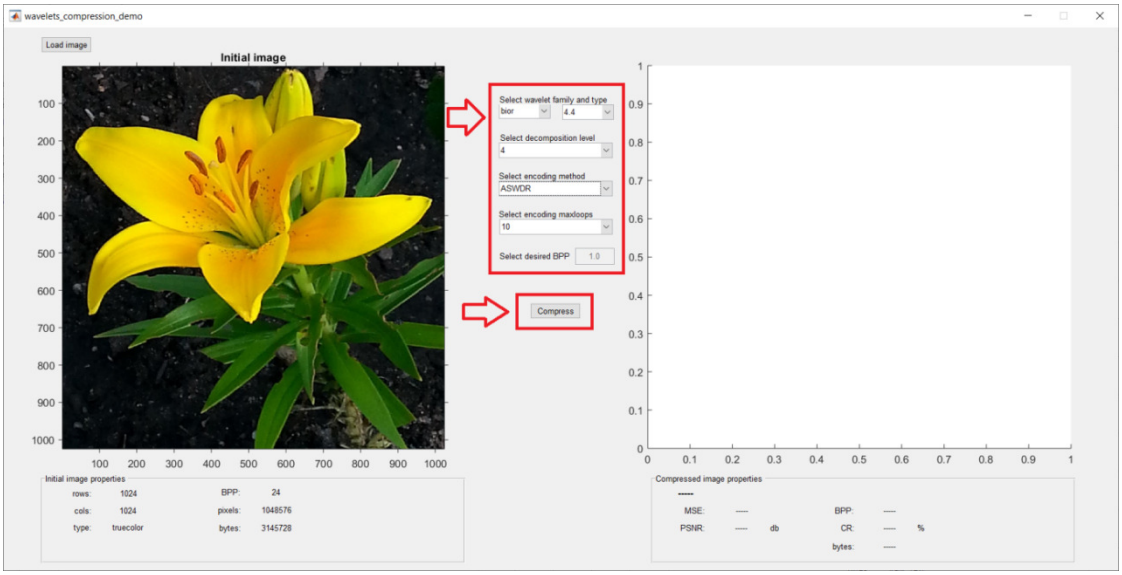
Στον παρακάτω πίνακα (Πίνακας 4) φαίνονται οι μέθοδοι που υποστηρίζονται από την εφαρμογή:

Πίνακας 4: Υποστηριζόμενες μέθοδοι συμπίεσης

Κατηγορία μεθόδου	**Όνομα μεθόδου Matlab**	**Όνομα μεθόδου συμπίεσης**
Progressive Coefficients Significance Methods (PCSM)	'ezw'	Embedded Zerotree Wavelet
	'spiht'	Set Partitioning In Hierarchical Trees
	'stw'	Spatial-orientation Tree Wavelet
	'wdr'	Wavelet Difference Reduction
	'aswdr'	Adaptively Scanned Wavelet Difference Reduction
	'spiht_3d'	Set Partitioning In Hierarchical Trees 3D for truecolor images
Coefficients Thresholding Methods (CTM)	'gbl_mmc_f'	Global thresholding of coefficients and fixed encoding
	'gbl_mmc_h'	Global thresholding of coefficients and Huffman encoding
	'lvl_mmc'	Subband thresholding of coefficients and Huffman encoding

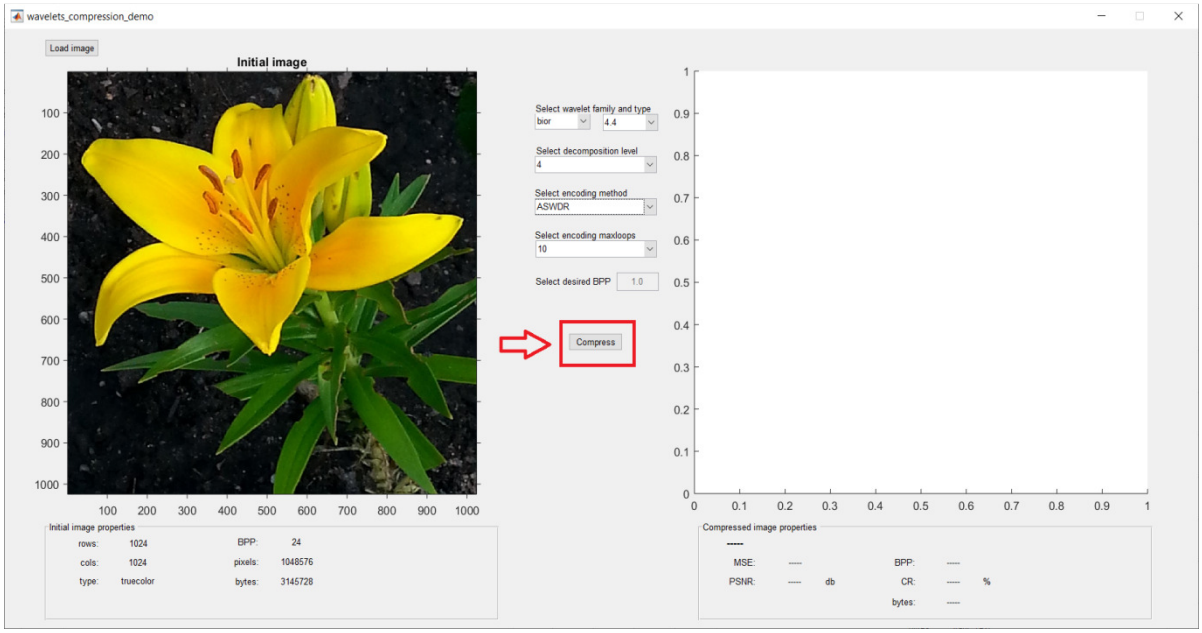
Εφόσον επιλεγεί κάποια από τις μεθόδους PCSM, ο χρήστης καλείται να εισάγει και το μέγιστο αριθμό των βημάτων του αλγορίθμου κωδικοποίησης (encoding maxloops) δίνοντας μία τιμή από 1 ως 19 βήματα (default τιμή είναι η 10). Αν όμως επιλεγεί κάποια από τις μεθόδους CTM, τότε ο χρήστης καλείται να εισάγει τον ρυθμό bits per pixel (desired BPP) που επιθυμεί να πετύχει ο αλγόριθμος συμπίεσης εφόσον είναι δυνατό κάτι τέτοιο (default τιμή είναι η 1.0).

Στην **Εικόνα** φαίνονται οι παράμετροι της συμπίεσης, όπου έχει επιλεγεί το wavelet bior4.4, με 4 επίπεδα αποσύνθεσης, μέθοδο κωδικοποίησης την ASWDR με μέγιστο αριθμό βημάτων κωδικοποίησης 10.



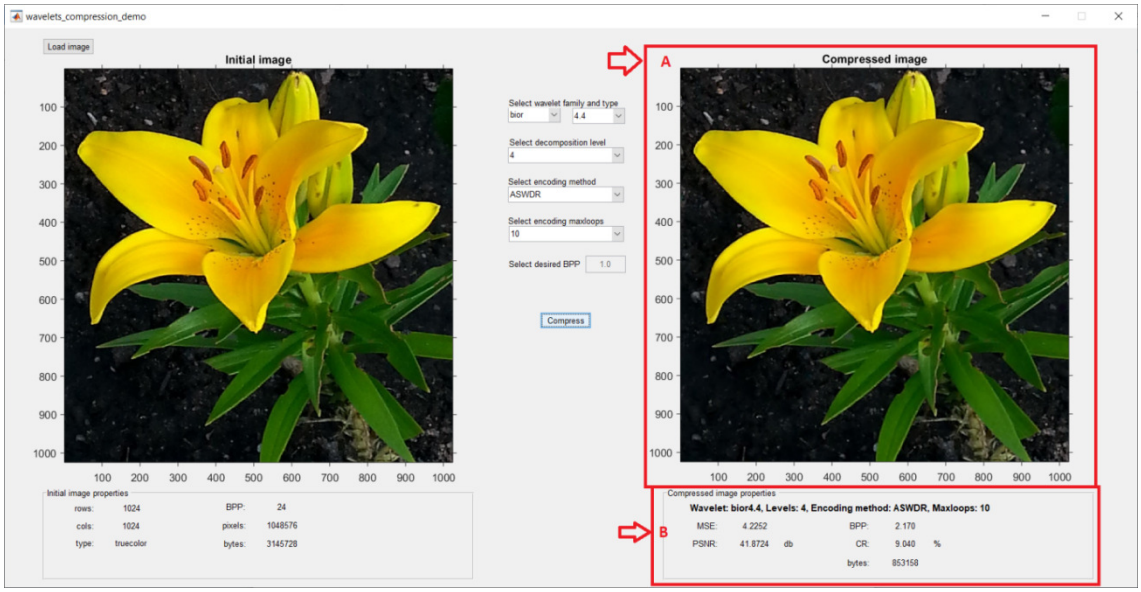
Εικόνα 36: Παράμετροι συμπίεσης

Εφόσον ο χρήστης επιλέξει τις τιμές των παραμέτρων της συμπίεσης με χρήση κάποιου wavelet, μπορεί ακολούθως να πατήσει το πλήκτρο «Compress» (**Εικόνα**), το οποίο θα εκτελέσει τη συμπίεση της εικόνας.



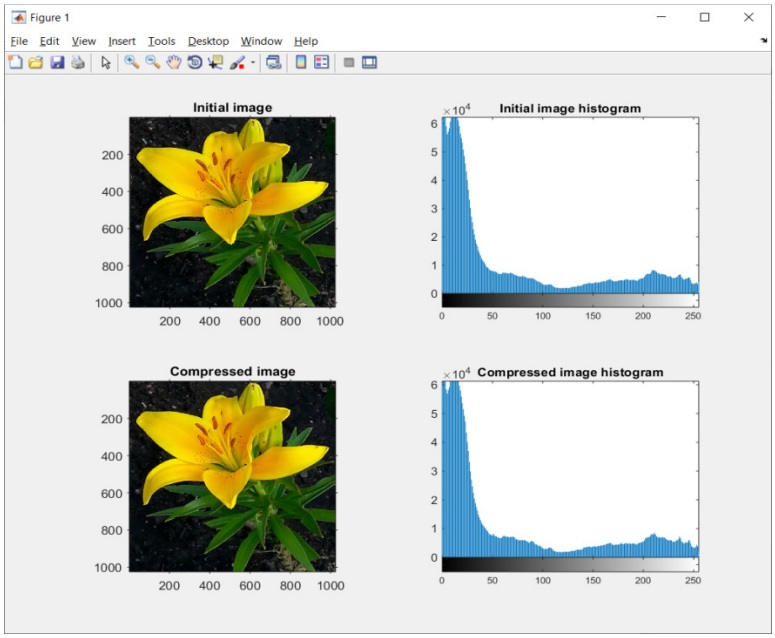
Εικόνα 37: Πλήκτρο «Compress»

Τα αποτελέσματα της συμπίεσης εμφανίζονται στο δεξιό τμήμα του παραθύρου της εφαρμογής (**Εικόνα**), όπου στο τμήμα (A) φαίνεται η συμπιεσμένη εικόνα και στο τμήμα (B) φαίνονται τα χαρακτηριστικά – ιδιότητες της συμπιεσμένης εικόνας, όπως: MSE, PSNR (σε dB), BPP, CR και bytes που απαιτούνται για την αποθήκευση της συμπιεσμένης εικόνας.



Εικόνα 38: Ενημέρωση οθόνης εφαρμογής με τα αποτελέσματα της συμπίεσης

Επίσης, εμφανίζεται ένα παράθυρο με τα ιστογράμματα της αρχικής και της συμπιεσμένης εικόνας, διευκολύνοντας την οπτική σύγκρισή τους (**Εικόνα**).



Εικόνα 39: Σύγκριση εικόνας και ιστογράμματος πριν και μετά τη συμπίεση

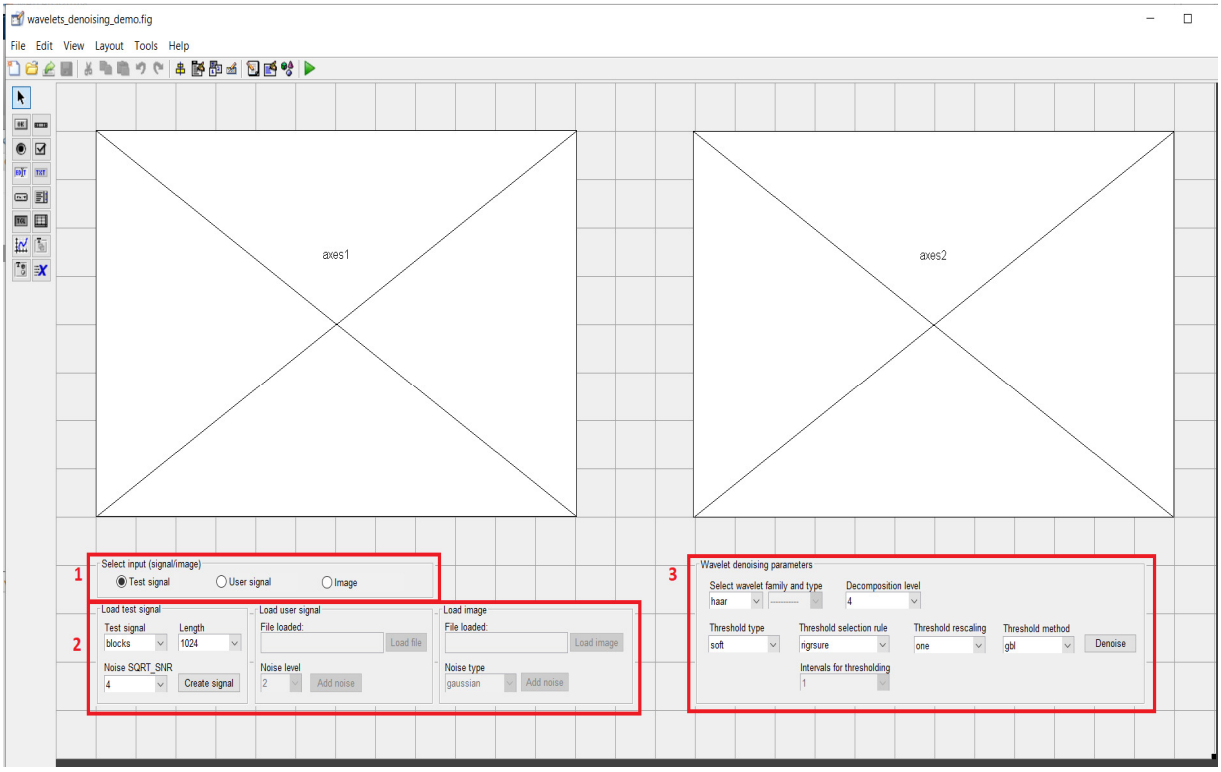
Ο χρήστης μπορεί να φορτώσει μία νέα εικόνα ή να επιλέξει άλλες παραμέτρους για τη συμπίεση και να επαναλάβει την παραπάνω διαδικασία.

10.2 Εφαρμογή Αποθορυβοποίησης με Χρήση Wavelets

Το παράθυρο της εφαρμογής ονομάζεται **wavelets_denoising_demo.fig** και αποτελείται από τρία (3) τμήματα με τα εξής στοιχεία:

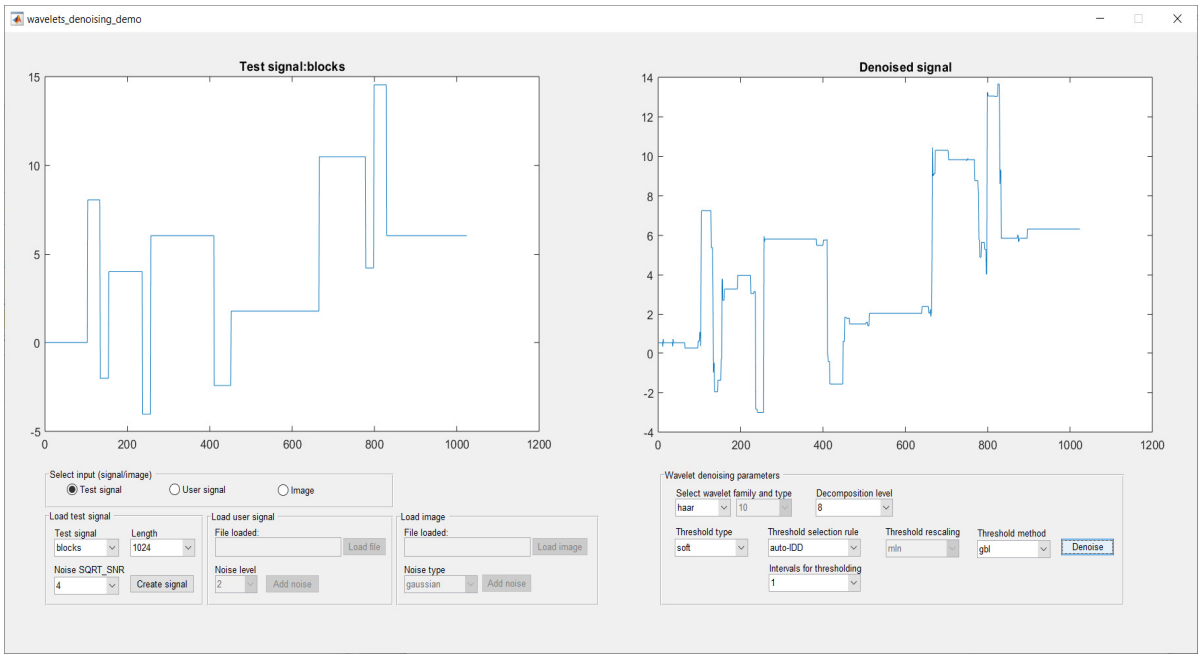
1. Χειριστήρια (radiobuttons) επιλογής τύπου σήματος προς επεξεργασία. Το σήμα μπορεί να είναι: α) ένα από τα έτοιμα δοκιμαστικά 1-D σήματα, β) ένα ηχητικό σήμα που θα εισάγει ο χρήστης ή γ) μία εικόνα (2-D σήμα) που θα εισάγει ο χρήστης.
2. Επιλογή του σήματος εισόδου και των παραμέτρων του θορύβου που θα προστεθεί στο σήμα αυτό. Το σήμα που θα επιλέξει/εισάγει ο χρήστης απεικονίζεται στην περιοχή axes1, ενώ το σήμα μετά την προσθήκη του θορύβου απεικονίζεται στην περιοχή axes2, μαζί με τις μετρικές MSE και PSNR.
3. Περιοχή επιλογής των παραμέτρων αποθορυβοποίησης με χρήση wavelet μετασχηματισμού. Το αποθορυβοποιημένο σήμα απεικονίζεται στην περιοχή axes2, μαζί με τις μετρικές MSE και PSNR.

Στην εικόνα που ακολουθεί (Εικόνα) παρουσιάζεται το παράθυρο της εφαρμογής και τα χειριστήριά του, όπως φαίνονται στο περιβάλλον GUIDE. Οι διακριτές κόκκινες περιοχές αντιπροσωπεύουν τα τμήματα του παραθύρου.



Εικόνα 40: Το παράθυρο εφαρμογής, όπως φαίνεται στο GUIDE του Matlab. Τα τμήματα του παραθύρου εμφανίζονται με κόκκινη σήμανση

Στην εικόνα που ακολουθεί (Εικόνα) παρουσιάζεται το παράθυρο και τα χειριστήρια της εφαρμογής κατά τον χρόνο εκτέλεσης.



Εικόνα 41: Το παράθυρο (wavelets_denoising_demo.fig) κατά τον χρόνο εκτέλεσης

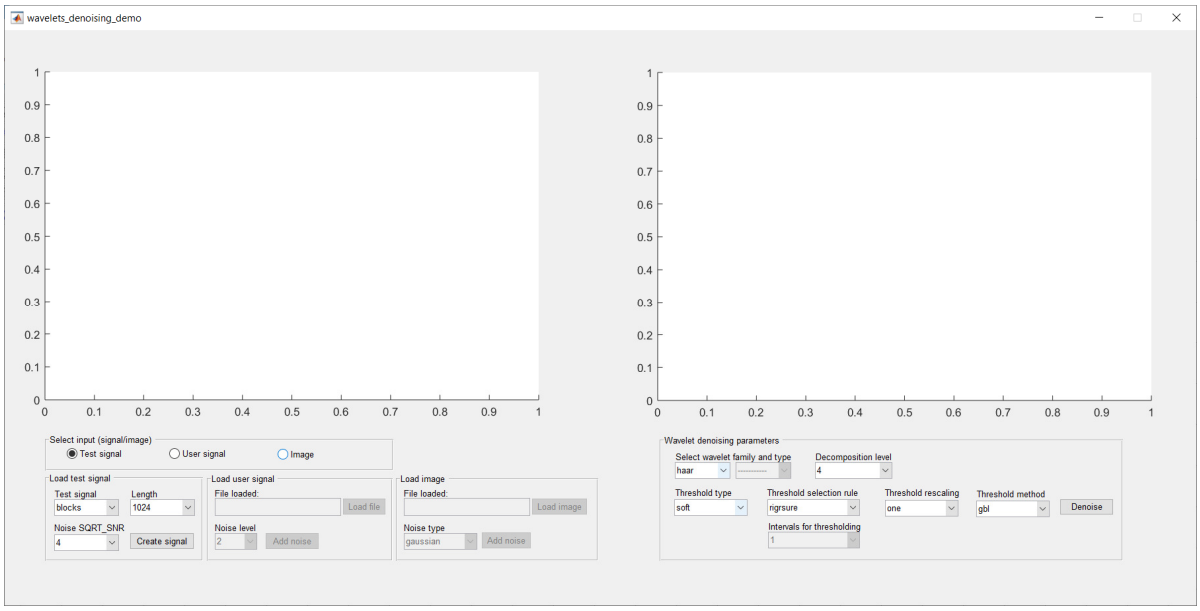
10.2.1 Βασικό Σενάριο Χρήσης Εφαρμογής Αποθορυβοποίησης με Χρήση Wavelets

Στην υποενότητα αυτή παρουσιάζεται το βασικό σενάριο χρήσης της εφαρμογής αποθορυβοποίησης σήματος με χρήση wavelets, το οποίο συνοψίζεται στα εξής βήματα:

1. Επιλογή τύπου σήματος εισόδου (έτοιμο δοκιμαστικό σήμα – test signal, σήμα ορισμένο από τον χρήστη – user signal, εικόνα – image)
2. Επιλογή του επιπέδου του θορύβου που θα προσθέσουμε στο σήμα εισόδου
3. Επιλογή των παραμέτρων φιλτραρίσματος του θορύβου με χρήση wavelet
4. Εκτέλεση της αποθορυβοποίησης

Ακολουθεί η αναλυτική περιγραφή του καθενός βήματος.

Έχοντας εκτελέσει την εφαρμογή μέσα από το περιβάλλον GUIDE του Matlab, ο χρήστης έχει πρόσβαση στην αρχική οθόνη της εφαρμογής. Στην επόμενη εικόνα (**Εικόνα 4**) φαίνεται η αρχική οθόνη της εφαρμογής.



Εικόνα 4: Αρχική οθόνη εφαρμογής

Ακολούθως, ο χρήστης επιλέγει αν θα χρησιμοποιήσει ένα έτοιμο δοκιμαστικό σήμα (test signal) ή αν θα εισάγει ένα δικό του σήμα (user signal, όπως κάποιο ηχητικό σήμα) ή αν θα εισάγει μία εικόνα (image).

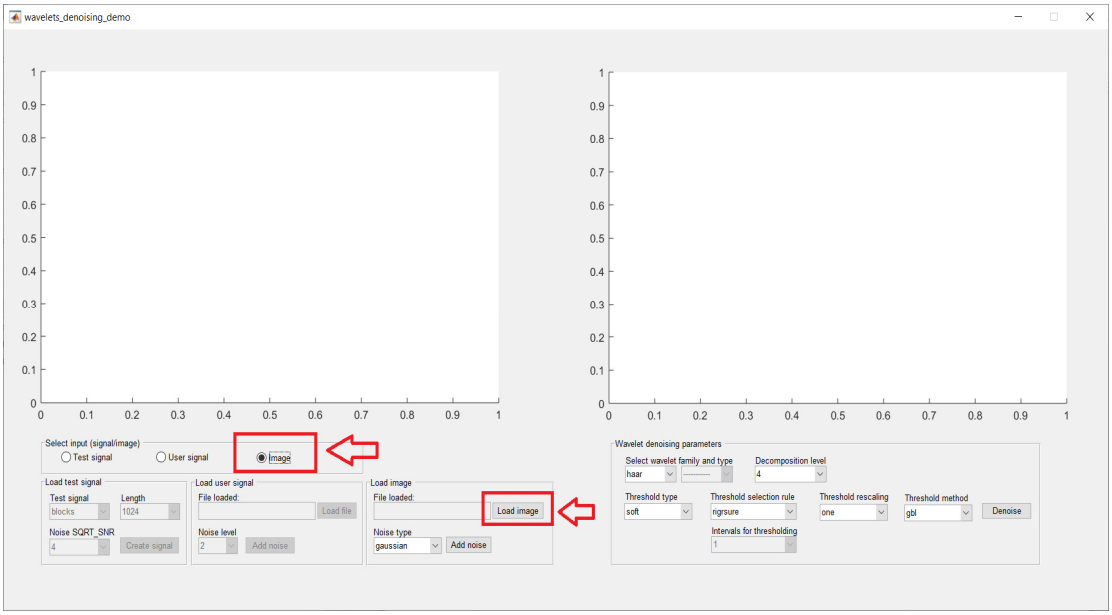
Αν επιλέξει ένα έτοιμο δοκιμαστικό σήμα, θα πρέπει να ορίσει το μήκος του (το πλήθος των δειγμάτων του σήματος είναι δύναμη του 2) και τη στάθμη του θορύβου (noise_SNR_SQRT) που θα προστεθεί σε αυτό. Πατώντας στο πλήκτρο «Create», στην περιοχή axes1 απεικονίζεται το δοκιμαστικό σήμα, ενώ στην περιοχή axes2 απεικονίζεται το σήμα με το θόρυβο. Τα δοκιμαστικά σήματα που υποστηρίζονται είναι τα εξής:

- Blocks
- Bumps,
- Heavy sine,
- Doppler,
- Quadcrisp και
- Mishmash

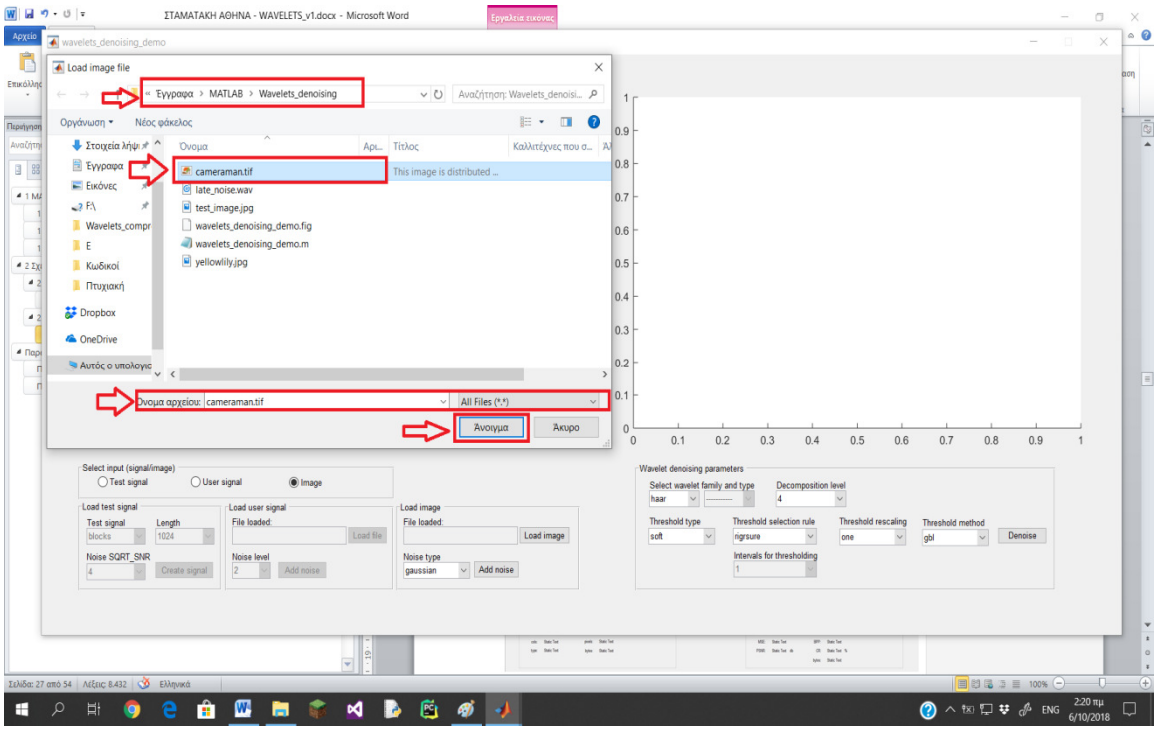
Αν ο χρήστης επιλέξει να εισάγει ένα δικό του σήμα (user signal), τότε ενεργοποιείται το πλαίσιο «Load user signal». Πατώντας στο πλήκτρο «Load file» έχει τη δυνατότητα να φορτώσει ένα υπάρχον αρχείο ήχου, το οποίο απεικονίζεται στην περιοχή axes1 και μετά να επιλέξει το επίπεδο του θορύβου που θα προσθέσει στο σήμα (noise level). Πατώντας το πλήκτρο «Add noise» απεικονίζεται στην περιοχή axes2 το σήμα με το θόρυβο.

Επίσης, αν ο χρήστης επιλέξει να εισάγει μία εικόνα (image), τότε ενεργοποιείται το πλαίσιο «Load image». Πατώντας το πλήκτρο «Load image» εμφανίζεται ένα παράθυρο διαλόγου, όπου ο χρήστης μπορεί να πλοηγηθεί στους φακέλους του, να επιλέξει την εικόνα που επιθυμεί και πατώντας το πλήκτρο «Άνοιγμα» να φορτώσει την επιλεγμένη εικόνα, η οποία θα εμφανιστεί στην περιοχή axes2. Επίσης, μπορεί να επιλέξει τον τύπο θορύβου που θέλει να προσθέσει στην εικόνα (noise type), οπότε πατώντας το πλήκτρο «Add noise»,

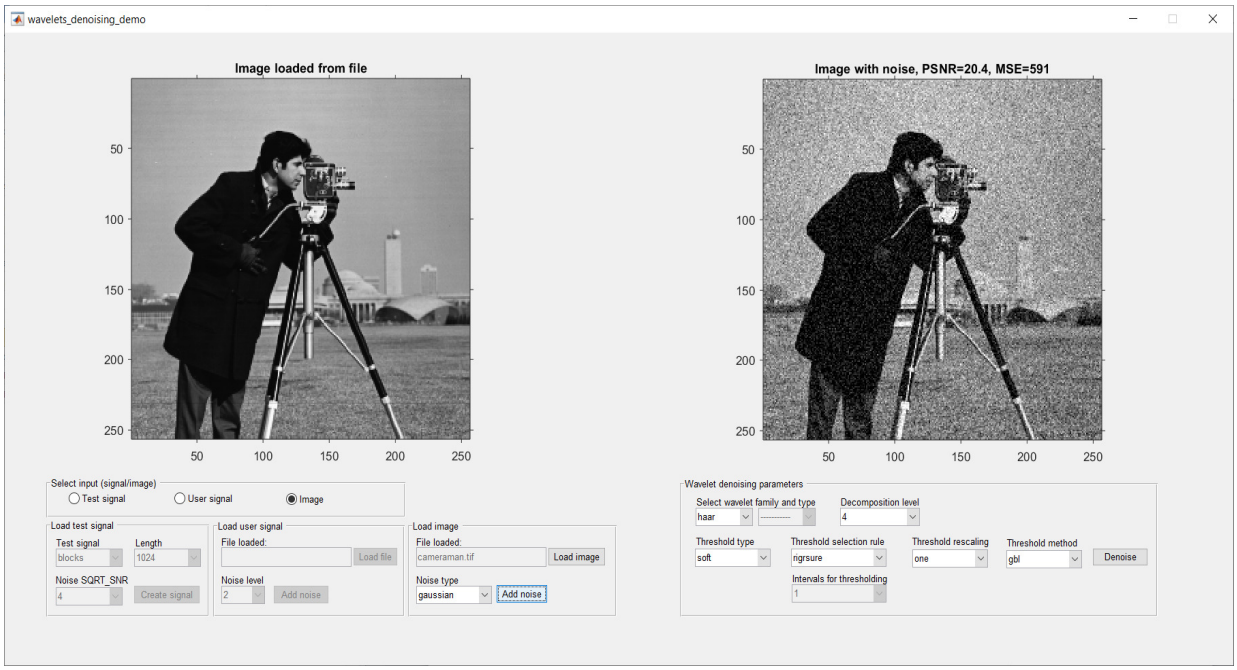
προστίθεται ο θόρυβος στην εικόνα και απεικονίζεται στην περιοχή axes2. Στις ακόλουθες εικόνες (**Εικόνα** ως **Εικόνα**) φαίνεται η διαδικασία αυτή.



Εικόνα 43: Πλήκτρο «Φόρτωση εικόνας»



Εικόνα 5: Παράθυρο επιλογής και φόρτωσης εικόνας



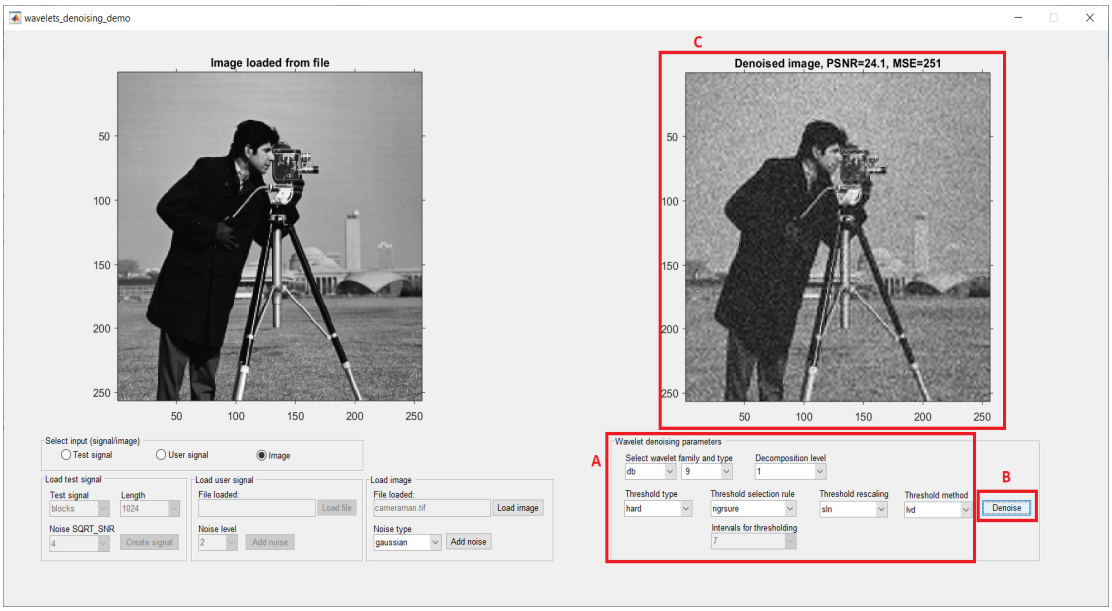
Εικόνα 45: Ενημέρωση οθόνης εφαρμογής μετά την φόρτωση της εικόνας και την προσθήκη του θορύβου

Στη συνέχεια ο χρήστης μπορεί να επιλέξει την οικογένεια και τον τύπο του μετασχηματισμού wavelet που θα χρησιμοποιηθεί για την αποθορυβοποίηση. Τα wavelets που υποστηρίζονται είναι τα εξής:

- haar
- db1, .., db10 (Daubechies)
- sym2, .., sym8 (Symlets)
- coif1, .., coif5 (Coiflets)
- bior1.1, bior1.3, bior1.5, …, bior6.8 (BiorSplines)
- rbio1.1, rbio1.3, rbio1.5, …, rbio6.8 (ReverseBior)

Ακολούθως, ο χρήστης πρέπει να εισάγει το πλήθος των επιπέδων αποσύνθεσης του μετασχηματισμού (decomposition levels) επιλέγοντας μία τιμή από το 1 ως το 8 (default τιμή είναι η 4), καθώς και τον τύπο της κατωφλίωσης των συντελεστών του wavelet μετασχηματισμού (threshold type, με τιμή soft ή hard), τον κανόνα επιλογής του κατωφλίου (threshold selection rule, με τιμή μία από τις εξής: rigrsure, heursure, sqtwolog, minimaxi, auto-IDD), τη μέθοδο κλιμάκωσης του κατωφλίου (threshold rescaling, με τιμή one, sln ή mln) και τη μέθοδο κατωφλίωσης των συντελεστών προσέγγισης του wavelet μετασχηματισμού (threshold method, με τιμή gbl ή lvd). Εφόσον ο χρήστης επιλέξει τις τιμές των παραμέτρων και πατήσει το πλήκτρο «Denoise», θα εκτελεστεί η αποθορυβοποίηση, όπου τα αποτελέσματα εμφανίζονται στην περιοχή του axes2 στο δεξιό τμήμα του παραθύρου της εφαρμογής μαζί με μετρικές ποιότητας, όπως: MSE και PSNR (σε dB).

Στην **Εικόνα 6** φαίνονται οι παράμετροι της αποθορυβοποίησης, όπου έχει επιλεγεί το wavelet db9, με 1 επίπεδο αποσύνθεσης, τύπο κατωφλίωσης hard, επιλογή κανόνα κατωφλίωσης rigrsure και μέθοδο κατωφλίωσης lvd. Επίσης, φαίνονται οι τιμές των μετρικών MSE και PSNR.



Εικόνα 6: Εκτέλεση αποθορυβοποίησης

Ο χρήστης μπορεί να φορτώσει ένα νέο σήμα και να επαναλάβει την όλη διαδικασία της αποθορυβοποίησης.

13.ΣΥΜΠΕΡΑΣΜΑΤΑ

Τα κυματίδια(wavelets) όπως αναλύσαμε και παραπάνω αποτελούν ένα μαθηματικό εργαλείο του μέλλοντος στην ψηφιακή επεξεργασία σήματος όπου με την πάροδο του χρόνου θα εξελίσσονται και θα χρησιμοποιούνται όλο και περισσότερο σε πολλές διεργασίες.Τα κυματίδια αποτελούν ένα εξελιγμένο μοντέλο έναντι του μετασχηματισμού Fourier.Το σημαντικότερο πλεονέκτημα των κυματιδίων(wavelet) είναι ότι προσαρμόζονται ανάλογα με τις ανάγκες που έχουμε και το αντικείμενο που έχουμε επιλέξει. Επιπλέον με τη χρήση κυματιδίων επιτυγχάνεται πολύ καλό αποτέλεσμα της συμπίεσης δεδομένων χωρίς να χάνεται πολύτιμη πληροφορία και καταφέρνοντας όσο το δυνατόν μικρότερη χωρητικότητα στο αρχείο.

Τα κυματίδια wavelets κατάφεραν να ξεπεράσουν το πρόβλημα του περιορισμένου εύρους της ανάλυσης των χρονολογικών σειρών που μέχρι τώρα μπορούσαν να αναλυθούν με τον μετασχηματισμό Fourier.Επομένως έδωσαν δυνατότητα ανάπτυξης και εξέλιξης στην ανάλυση σημάτων. Τέλος δίνουν δυνατότητα ανάλυσης σε συγκεκριμένα χαρακτηριστικά μιας συνάρτησης λόγω της ανάλυσης κλίμακας που διαθέτουν τα οποία δεν ανιχνεύονται με άλλες τεχνικές ανάλυσης.

Το γεγονός ανάπτυξης νέων μαθηματικών εργαλείων και υπολογιστικών μεθόδων ,εγκαθίδρυσε τη λειτουργία των wavelets ως το πιο εξελιγμένο μαθηματικό εργαλείο που χρησιμεύει σε πολλές επιστήμες και διευκολύνει την επεξεργασία και ανάλυση δεδομένων. Τα κύματα αναπτύχθηκαν ανεξάρτητα στους τομείς των μαθηματικών, της κβαντικής φυσικής, της ηλεκτρολογία και της σεισμικής γεωλογίας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Lent, C. S. (2013). *Learning To Program With Matlab: Building GUI Tools.* Wiley & Sons.

Marchand, P., & Holland, T. O. (2003). *Graphics And GUIs With Matlab* (3rd ed.). Washington DC: Chapman & Hall/CRC.

Βαρσάμης, Δ. (2006). MATLAB Graphical User Interfaces: Σημειώσεις GUI για το εργαστήριο του μαθήματος Ασαφή Συστήματα. Σέρρες: ΤΕΙ Σερρών - Τμήμα Πληροφορικής και Επικοινωνιών.

Αθανάσιος Ι. Μάργαρης. (2015).ΣΗΜΑΤΑ & ΣΥΣΤΗΜΑΤΑ: Συνεχούς και Διακριτού Χρόνου,Βελτιωμένη Έκδοση.Θεσσαλονίκη.

Lokenath Debnath, & Firdous Ahmad Shah. (2010).Wavelet Transforms and Their Applications.Department of Mathematics University of Texas (2nd ed.). Pan American Edinburg, TX, USA.

Peter Nickolas. (2017).Wavelets: A Student Guide. Australian Mathematical Society Lecture Series: 24. University of Wollongong, New South Wales.

Paul, S.Addisson. (2017).The Illustrated Wavelet Transform Handbook.Introductory Theory and Applications in Science, Engineering, Medicine and Finance (2nd ed.). Edinburgh, Scotland.

Michael Weeks. (2007).Digital Signal Processing:Using MATLAB and Wavelets.Georgia State

University, Infinity Science Press LLC.

Michel Misiti,Yves Misiti,Georges Oppenheim & Jean-Michel Poggi.(r2016a). Wavelet $Toolbox^{TM}$: Getting Started Guide. MathWorks,Inc.

Stephane Mallat. (1998). A Wavelet Tour of Signal Processing(2nd ed.).

MathWorks.com. (2012). *Image Processing Toolbox*. Ανάκτηση 5 21, 2014, από MathWorks: http://www.mathworks.com/help/images/index.html

MathWorks.com. (2012).Wavelet Families.2018,

από Mathworks: https://www.mathworks.com/help/wavelet/wavelet-families/index.html

Wikipedia.org. Wavelet. Ανάκτηση 8 31, 2018, από Wikipedia:

https://en.wikipedia.org/wiki/wavelet

Wikipedia.org.Haar wavelet. Ανάκτηση 6 14, 2018, από Wikipedia:

https://en.wikipedia.org/wiki/Haar_wavelet

Wikipedia.org.Wavelet transform. Ανάκτηση 6 17, 2018, από Wikipedia:

https://en.wikipedia.org/wiki/ Wavelet_Transform

ΠΑΡΑΡΤΗΜΑ

Π1. Κώδικας Εφαρμογής Συμπίεσης Εικόνας με Χρήση Wavelets

```
function varargout = wavelets_compression_demo(varargin)
% WAVELETS_COMPRESSION_DEMO MATLAB code for wavelets_compression_demo.fig
%      WAVELETS_COMPRESSION_DEMO, by itself, creates a new WAVELETS_COMPRESSION_DEMO
or raises the existing
%      singleton*.
%
%      H = WAVELETS_COMPRESSION_DEMO returns the handle to a new
WAVELETS_COMPRESSION_DEMO or the handle to
%      the existing singleton*.
%
%      WAVELETS_COMPRESSION_DEMO('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in WAVELETS_COMPRESSION_DEMO.M with the given input
arguments.
%
%      WAVELETS_COMPRESSION_DEMO('Property','Value',...) creates a new
WAVELETS_COMPRESSION_DEMO or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before wavelets_compression_demo_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to wavelets_compression_demo_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help wavelets_compression_demo

% Last Modified by GUIDE v2.5 30-Jun-2018 20:57:00

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @wavelets_compression_demo_OpeningFcn, ...
                   'gui_OutputFcn',  @wavelets_compression_demo_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before wavelets_compression_demo is made visible.
function wavelets_compression_demo_OpeningFcn(hObject, eventdata, handles, varargin)
```

```
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to wavelets_compression_demo (see VARARGIN)

% Choose default command line output for wavelets_compression_demo
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes wavelets_compression_demo wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = wavelets_compression_demo_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in load_image_btn.
function load_image_btn_Callback(hObject, eventdata, handles)
% hObject    handle to load_image_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global InitialImageGRAY;
global InitialImageRGB;
% global blankImage;
global InitialImage;
global fname;
global info;
[InitialImageGRAY, InitialImageRGB,info,fname]=load_image();
if ~isempty(InitialImageRGB)
    [r,c,d]=size(InitialImageRGB);
    InitialImage=InitialImageRGB;
else
    [r,c]=size(InitialImageGRAY);
    d=1;
    InitialImage=InitialImageGRAY;
end
if ~isempty(InitialImage)
    bpp = d*8;
    axes(handles.axes1);
    cla(handles.axes1,'reset');
    imshow(InitialImage),title('Initial image');
    cla(handles.axes2,'reset');
    %figure,imshow(InitialImage),title('Initial image');

    set(handles.rows_txt,'String',num2str(r));
    set(handles.cols_txt,'String',num2str(c));
    set(handles.type_txt,'String',info.ColorType);
```

```
    set(handles.BPP_txt,'String',num2str(bpp));
    set(handles.pixels_txt,'String',num2str(r*c));
    set(handles.bytes_txt,'String',num2str(r*c*d));

    set(handles.compression_properties_txt,'String','-----');
    set(handles.compressed_BPP_txt,'String','-----');
    set(handles.compressed_CR_txt,'String','-----');
    set(handles.compressed_MSE_txt,'String','-----');
    set(handles.compressed_PSNR_txt,'String','-----');
    set(handles.compressed_bytes_txt,'String','-----');

end


% --- Executes on selection change in wavelet_family_pmenu.
function wavelet_family_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wavelet_family_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wavelet_family_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from wavelet_family_pmenu
contents = cellstr(get(hObject,'String'));
wfamily=contents{get(hObject,'Value')};
switch wfamily
    case 'haar'
        names=[];
    case 'db'
        names=[1:10]';
    case 'sym'
        names=[2:8]';
    case 'coif'
        names=[1:5]';
    case 'bior'
        names=[1.1,1.3,1.5,2.2,2.4,2.6,2.8,3.1,3.3,3.5,3.7,3.9,4.4,5.5,6.8]';
    case 'rbio'
        names=[1.1,1.3,1.5,2.2,2.4,2.6,2.8,3.1,3.3,3.5,3.7,3.9,4.4,5.5,6.8]';
end
if wfamily=="haar"
    set(handles.wavelet_version_pmenu,'Enable','off');
else
    set(handles.wavelet_version_pmenu,'Enable','on');
    set(handles.wavelet_version_pmenu,'String',num2str(names));
    set(handles.wavelet_version_pmenu,'Value',1);
end

% --- Executes during object creation, after setting all properties.
function wavelet_family_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wavelet_family_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on selection change in wavelet_version_pmenu.
function wavelet_version_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wavelet_version_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wavelet_version_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from wavelet_version_pmenu


% --- Executes during object creation, after setting all properties.
function wavelet_version_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wavelet_version_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in decomposition_level_pmenu.
function decomposition_level_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to decomposition_level_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns decomposition_level_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from decomposition_level_pmenu


% --- Executes during object creation, after setting all properties.
function decomposition_level_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to decomposition_level_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in encoding_method_pmenu.
function encoding_method_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to encoding_method_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns encoding_method_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from encoding_method_pmenu
```

```
contents = cellstr(get(hObject,'String'));
encoding_method=contents{get(hObject,'Value')};
if encoding_method=="GBL_MMC_F" || encoding_method=="GBL_MMC_H" ||
encoding_method=="LVL_MMC"
    set(handles.maxloops_pmenu,'Enable','off');
    set(handles.desired_BPP_edit,'Enable','on');
else
    set(handles.maxloops_pmenu,'Enable','on');
    set(handles.desired_BPP_edit,'Enable','off');
end

% --- Executes during object creation, after setting all properties.
function encoding_method_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to encoding_method_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in maxloops_pmenu.
function maxloops_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to maxloops_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns maxloops_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from maxloops_pmenu


% --- Executes during object creation, after setting all properties.
function maxloops_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maxloops_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in compress_btn.
function compress_btn_Callback(hObject, eventdata, handles)
% hObject    handle to compress_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get all selected data from pmenus
contents = cellstr(get(handles.wavelet_family_pmenu,'String'));
wfamily=contents{get(handles.wavelet_family_pmenu,'Value')};

contents = cellstr(get(handles.wavelet_version_pmenu,'String'));
```

```
wavelet_version=contents{get(handles.wavelet_version_pmenu,'Value')};

contents = cellstr(get(handles.decomposition_level_pmenu,'String'));
decomposition_level=contents{get(handles.decomposition_level_pmenu,'Value')};

contents = cellstr(get(handles.encoding_method_pmenu,'String'));
encoding_method=contents{get(handles.encoding_method_pmenu,'Value')};

contents = cellstr(get(handles.maxloops_pmenu,'String'));
maxloops=contents{get(handles.maxloops_pmenu,'Value')};

desired_BPP = get(handles.desired_BPP_edit,'Value');

if wfamily=="haar"
    wname=wfamily;
else
    wname=strcat(wfamily,strtrim(wavelet_version));
end
L=str2num(decomposition_level);
enc_method=lower(encoding_method);
loops=str2num(maxloops);
aim_BPP=desired_BPP;

global InitialImage;
global fname;
X=InitialImage;
[r,c]=size(X);
savefile=strcat(fname,'.wtc');
if startsWith(enc_method,["GBL_","LVL_"],'IgnoreCase',true)
    properties_str = strcat('Wavelet:',[' ' wname],', Levels:',[' ' num2str(L)],', Encoding method:',[' '
encoding_method]);
    [CR,BPP]=wcompress('c',X,savefile,enc_method,'wname',wname,'level',L,'bpp',aim_BPP);
else
    properties_str = strcat('Wavelet:',[' ' wname],', Levels:',[' ' num2str(L)],', Encoding method:',[' '
encoding_method],', Maxloops:',[' ' num2str(loops)]);
    [CR,BPP]=wcompress('c',X,savefile,enc_method,'wname',wname,'level',L,'maxloop',loops);
end
Xc = wcompress('u',savefile);
delete(savefile);

% D = abs(double(X)-double(Xc)).^2;
% mse = sum(D(:))/numel(X);
% psnr = 10*log10(255*255/mse);
[psnr_value,mse_value,maxerr_value,L2rat_value] = measerr(X,Xc);

% Display the original and the compressed image
axes(handles.axes2);
cla(handles.axes2,'reset');
imshow(Xc,[]),title('Compressed image');
set(handles.compression_properties_txt,'String',properties_str);
set(handles.compressed_BPP_txt,'String',num2str(BPP,'%3.3f'));
set(handles.compressed_CR_txt,'String',num2str(CR,'%3.3f'));
set(handles.compressed_MSE_txt,'String',num2str(mse_value));
set(handles.compressed_PSNR_txt,'String',num2str(psnr_value));
init_bytes=str2double(get(handles.bytes_txt,'String'));
set(handles.compressed_bytes_txt,'String',num2str(ceil(init_bytes*BPP/8)));

figure
```

```
subplot(2,2,1), imshow(X), title('Initial image')
subplot(2,2,2), imhist(X), title('Initial image histogram')
subplot(2,2,3), imshow(Xc,[]), title('Compressed image')
subplot(2,2,4), imhist(uint8(Xc),256), title('Compressed image histogram')
%subplot(2,2,5), imshow(uint8(double(X)-double(Xc)),[]),title('Differences')



function desired_BPP_edit_Callback(hObject, eventdata, handles)
% hObject    handle to desired_BPP_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of desired_BPP_edit as text
%        str2double(get(hObject,'String')) returns contents of desired_BPP_edit as a double
global info;
val=str2double(get(hObject,'String'));
if (val<0.00001 || val > info.BitDepth)
   disp(strcat('Use a value in range [0.00001, ', num2str(info.BitDepth),']'));
   val=1.0;
end
set(hObject,'Value',val);
set(hObject,'String',num2str(val));

% --- Executes during object creation, after setting all properties.
function desired_BPP_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to desired_BPP_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end
```

Π2. Κώδικας Εφαρμογής Αποθορυβοποίησης με Χρήση Wavelets

```
function varargout = wavelets_denoising1d_demo(varargin)
% WAVELETS_DENOISING_DEMO MATLAB code for wavelets_denoising_demo.fig
%      WAVELETS_DENOISING_DEMO, by itself, creates a new WAVELETS_DENOISING_DEMO or
raises the existing
%      singleton*.
%
%      H = WAVELETS_DENOISING_DEMO returns the handle to a new
WAVELETS_DENOISING_DEMO or the handle to
%      the existing singleton*.
%
%      WAVELETS_DENOISING_DEMO('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in WAVELETS_DENOISING_DEMO.M with the given input
arguments.
%
%      WAVELETS_DENOISING_DEMO('Property','Value',...) creates a new
WAVELETS_DENOISING_DEMO or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before wavelets_denoising1d_demo_OpeningFcn gets called.  An
```

```
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to wavelets_denoising1d_demo_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help wavelets_denoising_demo

% Last Modified by GUIDE v2.5 06-Jul-2018 12:29:31

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',  mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @wavelets_denoising_demo_OpeningFcn, ...
                   'gui_OutputFcn',  @wavelets_denoising_demo_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before wavelets_denoising_demo is made visible.
function wavelets_denoising_demo_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to wavelets_denoising_demo (see VARARGIN)

% Choose default command line output for wavelets_denoising_demo
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes wavelets_denoising_demo wait for user response (see UIRESUME)
% uiwait(handles.wavelets_denoising_demo);


% --- Outputs from this function are returned to the command line.
function varargout = wavelets_denoising_demo_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
```

```
varargout{1} = handles.output;



% --- Executes on selection change in test_signal_pmenu.
function test_signal_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to test_signal_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns test_signal_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from test_signal_pmenu


% --- Executes during object creation, after setting all properties.
function test_signal_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to test_signal_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in length_pmenu.
function length_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to length_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns length_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from length_pmenu


% --- Executes during object creation, after setting all properties.
function length_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to length_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in SQRT_SNR_pmenu.
function SQRT_SNR_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to SQRT_SNR_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns SQRT_SNR_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from SQRT_SNR_pmenu
```

```
% --- Executes during object creation, after setting all properties.
function SQRT_SNR_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to SQRT_SNR_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in create_signal_btn.
function create_signal_btn_Callback(hObject, eventdata, handles)
% hObject    handle to create_signal_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
global xn;

contents = cellstr(get(handles.test_signal_pmenu,'String'));
signal_name = contents{get(handles.test_signal_pmenu,'Value')};
signal_id = get(handles.test_signal_pmenu,'Value');

contents = cellstr(get(handles.length_pmenu,'String'));
signal_length = str2double(contents{get(handles.length_pmenu,'Value')});
signal_N = log2(signal_length);

contents = cellstr(get(handles.SQRT_SNR_pmenu,'String'));
signal_SQRT_SNR = str2double(contents{get(handles.SQRT_SNR_pmenu,'Value')});


%seed = now;
[x,xn] = wnoise(signal_id,signal_N,signal_SQRT_SNR);

axes(handles.axes1);
plot(x),title(strcat('Test signal: ',signal_name));
axes(handles.axes2);
plot(xn),title('Test signal with noise');



% --- Executes on button press in load_signal_btn.
function load_signal_btn_Callback(hObject, eventdata, handles)
% hObject    handle to load_signal_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
global fs;

[filename, pathname] = uigetfile( ...
    {'*.wav', 'All wav-Files (*.wav)'; ...
       '*.*','All Files (*.*)'}, ...
    'Select Wave File');
```

```
if isequal(filename,0)
    disp('>load_from_file(): Δεν επιλέξατε κάποιο αρχείο');
else
    % Φορτώνω το αρχείο
    file = fullfile(pathname,filename);
    [x,fs]=audioread(file);
    %x2=x1(1:65536);
    axes(handles.axes1);
    cla(handles.axes1,'reset');
    plot(x), title('Signal loaded from file');
    set(handles.file_loaded_edit,'String',filename);
end



function file_loaded_edit_Callback(hObject, eventdata, handles)
% hObject    handle to file_loaded_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of file_loaded_edit as text
%        str2double(get(hObject,'String')) returns contents of file_loaded_edit as a double


% --- Executes during object creation, after setting all properties.
function file_loaded_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to file_loaded_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in noise_SNR_pmenu.
function noise_SNR_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to noise_SNR_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns noise_SNR_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from noise_SNR_pmenu


% --- Executes during object creation, after setting all properties.
function noise_SNR_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to noise_SNR_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in add_noise_to_signal_btn.
function add_noise_to_signal_btn_Callback(hObject, eventdata, handles)
% hObject    handle to add_noise_to_signal_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
global xn;
contents = cellstr(get(handles.noise_SNR_pmenu,'String'));
noise_SNR = str2double(contents{get(handles.noise_SNR_pmenu,'Value')});
xn=awgn(x,noise_SNR,'measured');
axes(handles.axes2);
cla(handles.axes2,'reset');
plot(xn), title('Signal with AWGN noise');


% --- Executes on button press in load_test_signal_rb.
function load_test_signal_rb_Callback(hObject, eventdata, handles)
% hObject    handle to load_test_signal_rb (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of load_test_signal_rb
global x;
global xn;
x=[];
xn=[];
cla(handles.axes1,'reset');
cla(handles.axes2,'reset');
set(handles.file_loaded_edit,'String','');
set(handles.image_loaded_edit,'String','');

set(handles.test_signal_pmenu,'Enable','on');
set(handles.length_pmenu,'Enable','on');
set(handles.SQRT_SNR_pmenu,'Enable','on');
set(handles.create_signal_btn,'Enable','on');
set(handles.load_signal_btn,'Enable','off');
set(handles.noise_SNR_pmenu,'Enable','off');
set(handles.add_noise_to_signal_btn,'Enable','off');
set(handles.load_image_btn,'Enable','off');
set(handles.noise_type_pmenu,'Enable','off');
set(handles.add_noise_to_image_btn,'Enable','off');


% --- Executes on button press in load_user_signal_rb.
function load_user_signal_rb_Callback(hObject, eventdata, handles)
% hObject    handle to load_user_signal_rb (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of load_user_signal_rb
global x;
global xn;
x=[];
xn=[];
cla(handles.axes1,'reset');
cla(handles.axes2,'reset');
```

```
set(handles.file_loaded_edit,'String','');
set(handles.image_loaded_edit,'String','');

set(handles.test_signal_pmenu,'Enable','off');
set(handles.length_pmenu,'Enable','off');
set(handles.SQRT_SNR_pmenu,'Enable','off');
set(handles.create_signal_btn,'Enable','off');
set(handles.load_signal_btn,'Enable','on');
set(handles.noise_SNR_pmenu,'Enable','on');
set(handles.add_noise_to_signal_btn,'Enable','on');
set(handles.load_image_btn,'Enable','off');
set(handles.noise_type_pmenu,'Enable','off');
set(handles.add_noise_to_image_btn,'Enable','off');




% --- Executes on selection change in wavelet_family_pmenu.
function wavelet_family_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wavelet_family_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wavelet_family_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from wavelet_family_pmenu
contents = cellstr(get(hObject,'String'));
wfamily=contents{get(hObject,'Value')};
switch wfamily
    case 'haar'
        names=[];
    case 'db'
        names=[1:10]';
    case 'sym'
        names=[2:8]';
    case 'coif'
        names=[1:5]';
    case 'bior'
        names=[1.1,1.3,1.5,2.2,2.4,2.6,2.8,3.1,3.3,3.5,3.7,3.9,4.4,5.5,6.8]';
    case 'rbio'
        names=[1.1,1.3,1.5,2.2,2.4,2.6,2.8,3.1,3.3,3.5,3.7,3.9,4.4,5.5,6.8]';
end
if wfamily=="haar"
    set(handles.wavelet_version_pmenu,'Enable','off');
else
    set(handles.wavelet_version_pmenu,'Enable','on');
    set(handles.wavelet_version_pmenu,'String',num2str(names));
    set(handles.wavelet_version_pmenu,'Value',1);
end


% --- Executes during object creation, after setting all properties.
function wavelet_family_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wavelet_family_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on selection change in wavelet_version_pmenu.
function wavelet_version_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wavelet_version_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wavelet_version_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from wavelet_version_pmenu


% --- Executes during object creation, after setting all properties.
function wavelet_version_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wavelet_version_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in decomposition_level_pmenu.
function decomposition_level_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to decomposition_level_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns decomposition_level_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from decomposition_level_pmenu


% --- Executes during object creation, after setting all properties.
function decomposition_level_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to decomposition_level_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in wthresh_type_pmenu.
function wthresh_type_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wthresh_type_pmenu (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wthresh_type_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from wthresh_type_pmenu


% --- Executes during object creation, after setting all properties.
function wthresh_type_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wthresh_type_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in wthreshold_rule_pmenu.
function wthreshold_rule_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wthreshold_rule_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wthreshold_rule_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from wthreshold_rule_pmenu
contents = cellstr(get(hObject,'String'));
val = contents{get(hObject,'Value')};
if val == 'auto-IDD'
    set(handles.wthreshold_rescaling_pmenu,'Enable','off');
    set(handles.number_of_intervals_pmenu,'Enable','on');
else
    set(handles.wthreshold_rescaling_pmenu,'Enable','on');
    set(handles.number_of_intervals_pmenu,'Enable','off');
end

% --- Executes during object creation, after setting all properties.
function wthreshold_rule_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wthreshold_rule_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in wthreshold_rescaling_pmenu.
function wthreshold_rescaling_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wthreshold_rescaling_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: contents = cellstr(get(hObject,'String')) returns wthreshold_rescaling_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from wthreshold_rescaling_pmenu


% --- Executes during object creation, after setting all properties.
function wthreshold_rescaling_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wthreshold_rescaling_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in denoise_btn.
function denoise_btn_Callback(hObject, eventdata, handles)
% hObject    handle to denoise_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
global xn;


% Get all selected data from pmenus
contents = cellstr(get(handles.wavelet_family_pmenu,'String'));
wfamily=contents{get(handles.wavelet_family_pmenu,'Value')};

contents = cellstr(get(handles.wavelet_version_pmenu,'String'));
wavelet_version=contents{get(handles.wavelet_version_pmenu,'Value')};

contents = cellstr(get(handles.decomposition_level_pmenu,'String'));
decomposition_level=contents{get(handles.decomposition_level_pmenu,'Value')};

contents = cellstr(get(handles.wthresh_type_pmenu,'String'));
wthresh_type=contents{get(handles.wthresh_type_pmenu,'Value')};

contents = cellstr(get(handles.wthreshold_rule_pmenu,'String'));
wthreshold_rule=contents{get(handles.wthreshold_rule_pmenu,'Value')};

contents = cellstr(get(handles.wthreshold_rescaling_pmenu,'String'));
wthreshold_rescaling=contents{get(handles.wthreshold_rescaling_pmenu,'Value')};

contents = cellstr(get(handles.number_of_intervals_pmenu,'String'));
number_of_intervals=str2double(contents{get(handles.wthreshold_rescaling_pmenu,'Value')});

contents = cellstr(get(handles.wthreshold_method_pmenu,'String'));
wthreshold_method=contents{get(handles.wthreshold_method_pmenu,'Value')};


if wfamily=="haar"
    wname=wfamily;
else
    wname=strcat(wfamily,strtrim(wavelet_version));
end
```

```
L=str2double(decomposition_level);

if (wthresh_type=='soft')
    SORH = 's';
else
    SORH = 'h';
end
TPTR = wthreshold_rule;
SCAL = wthreshold_rescaling;
nb_Int = number_of_intervals;

if get(handles.load_image_rb,'Value')==0
    if TPTR == 'auto-IDD'
        [xd,coefs,thrParams,int_DepThr_Cell,BestNbOfInt]=cmddenoise(xn,wname,L,SORH,nb_Int);
        disp(strcat('Best number of intervals:',num2str(BestNbOfInt)));
    else
        xd = wden(xn,TPTR,SORH,SCAL,L,wname);
    end
else

    if wthreshold_method=='gbl'
        [thr,sorh,keepapp] = ddencmp('den','wv',xn);
        xd = wdencmp(wthreshold_method,xn,wname,L,thr,sorh,keepapp);
    else
        [C,S] = wavedec2(xn,L,wname);
        thr = wthrmngr('dw2ddenoLVL','penalhi',C,S,L);
        xd = wdencmp(wthreshold_method,xn,wname,L,thr,SORH);
    end



end

%if SCAL=='mlm'
%    xd = wden(xn-xn(1),TPTR,SORH,SCAL,L,wname) + xn(1);
%else
%    xd = wden(xn,TPTR,SORH,SCAL,L,wname);
%end

axes(handles.axes2);
cla(handles.axes2,'reset');
if get(handles.load_image_rb,'Value')==0
    plot(xd), title('Denoised signal');
else
    [psnr_value,mse_value,~,~] = measerr(x,xd);
    imshow(xd,[]), title(strcat(['Denoised image, PSNR=',num2str(psnr_value,3),', MSE=',
num2str(mse_value,3)]));
end


% --- Executes on selection change in number_of_intervals_pmenu.
function number_of_intervals_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to number_of_intervals_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns number_of_intervals_pmenu contents as cell
array
```

```
%        contents{get(hObject,'Value')} returns selected item from number_of_intervals_pmenu


% --- Executes during object creation, after setting all properties.
function number_of_intervals_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to number_of_intervals_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in load_image_btn.
function load_image_btn_Callback(hObject, eventdata, handles)
% hObject    handle to load_image_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
[filename, pathname] = uigetfile({'*.png';'*.jpg';'*.tiff';'*.*'},'Load image file');
if isequal(filename,0)
    disp('>load_from_file(): Δεν επιλέξατε κάποιο αρχείο');
else
    % Φορτώνω το αρχείο
    file = fullfile(pathname,filename);
    info=imfinfo(file);
    if info.ColorType == 'truecolor'
        x = rgb2gray(imread(file));
    else
        x = imread(file);
    end
    axes(handles.axes1);
    cla(handles.axes1,'reset');
    imshow(x), title('Image loaded from file');
    set(handles.image_loaded_edit,'String',filename);
end


function image_loaded_edit_Callback(hObject, eventdata, handles)
% hObject    handle to image_loaded_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of image_loaded_edit as text
%        str2double(get(hObject,'String')) returns contents of image_loaded_edit as a double


% --- Executes during object creation, after setting all properties.
function image_loaded_edit_CreateFcn(hObject, eventdata, handles)
% hObject    handle to image_loaded_edit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on selection change in noise_type_pmenu.
function noise_type_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to noise_type_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns noise_type_pmenu contents as cell array
%        contents{get(hObject,'Value')} returns selected item from noise_type_pmenu


% --- Executes during object creation, after setting all properties.
function noise_type_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to noise_type_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in add_noise_to_image_btn.
function add_noise_to_image_btn_Callback(hObject, eventdata, handles)
% hObject    handle to add_noise_to_image_btn (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global x;
global xn;
contents = cellstr(get(handles.noise_type_pmenu,'String'));
noise_type = contents{get(handles.noise_type_pmenu,'Value')};
xn=imnoise(x,noise_type);

axes(handles.axes2);
cla(handles.axes2,'reset');
[psnr_value,mse_value,~,~] = measerr(x,xn);
imshow(uint8(xn)), title(strcat('Image with noise, PSNR=',num2str(psnr_value,3),', MSE=',
num2str(mse_value,3)));


% --- Executes on button press in load_image_rb.
function load_image_rb_Callback(hObject, eventdata, handles)
% hObject    handle to load_image_rb (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of load_image_rb
global x;
global xn;
x=[];
xn=[];
cla(handles.axes1,'reset');
```

```
cla(handles.axes2,'reset');
set(handles.file_loaded_edit,'String','');
set(handles.image_loaded_edit,'String','');

set(handles.test_signal_pmenu,'Enable','off');
set(handles.length_pmenu,'Enable','off');
set(handles.SQRT_SNR_pmenu,'Enable','off');
set(handles.create_signal_btn,'Enable','off');
set(handles.load_signal_btn,'Enable','off');
set(handles.noise_SNR_pmenu,'Enable','off');
set(handles.add_noise_to_signal_btn,'Enable','off');
set(handles.load_image_btn,'Enable','on');
set(handles.noise_type_pmenu,'Enable','on');
set(handles.add_noise_to_image_btn,'Enable','on');



% --- Executes on selection change in wthreshold_method_pmenu.
function wthreshold_method_pmenu_Callback(hObject, eventdata, handles)
% hObject    handle to wthreshold_method_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns wthreshold_method_pmenu contents as cell
array
%        contents{get(hObject,'Value')} returns selected item from wthreshold_method_pmenu


% --- Executes during object creation, after setting all properties.
function wthreshold_method_pmenu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to wthreshold_method_pmenu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```