ΠΟΛΥΤΕΧΝΕΙΟ
ΚΡΗΤΗΣ

ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ & ΔΙΟΙΚΗΣΗΣ

Υπολογιστική μελέτη απεικονιστικών τεχνικών βάσει της χρονικής αντιστρεψιμότητας ελαστικών κυμάτων

Μάριος Ι. ΜΑΥΡΙΚΗΣ

Επιβλέπων
ΓΕΩΡΓΙΟΣ Ε. ΣΤΑΥΡΟΥΛΑΚΗΣ, Καθηγητής Πολυτεχνείου Κρήτης
Συνεπιβλέπων
ΧΡΗΣΤΟΣ Γ. ΠΑΝΑΓΙΩΤΟΠΟΥΛΟΣ, Δρ. Πολιτικός Μηχανικός, Συνεργάτης ΙΤΕ και ΕΛ.ΜΕ.ΠΑ
Μέλος επιτροπής
ΑΡΙΣΤΟΜΕΝΗΣ ΑΝΤΩΝΙΑΔΗΣ, Καθηγητής Πολυτεχνείου Κρήτης

25 Σεπτεμβρίου 2019

Περίληψη

Τα τελευταία χρόνια ο έλεγχος και η παρακολούθηση της λειτουργίας δυναμικών συστημάτων αποτελεί μια συνηθισμένη διαδικασία. Στην εργασία αυτή γίνεται μελέτη εφαρμογής μεθόδων παρακολούθησης της δομικής ακεραιότητας (Structural Health Monitoring) και γραφικής αποτύπωσης σημείων διέγερσης και βλαβών. Μέσω αυτών των διαδικασιών κατορθώνουμε να ανιχνεύσουμε πηγές διέγερσης ή πιθανής δυσλειτουργίας (λόγω αλλοίωσης ή φθοράς) πάνω στην κατασκευή. Επιπλέον μελετάμε τη δυνατότητα χωρικού εντοπισμού των σημείων διέγερσης ή/και βλάβης. Η διέγερση του συστήματος αλλά και η συλλογή δεδομένων απόκρισης μπορεί να γίνει με χρήση πιεζοηλεκτρικών κεραμικών υλικών (PZT), τα οποία θα συνθέτουν διατάξεις αισθητήρων/διεγερτών. Ο λόγος σήμα-προς-θόρυβο(Signal-to-Noise Ratio) αποτελεί, επίσης, αντικείμενο μελέτης της παρούσας εργασίας και έχει ως στόχο την εύρεση του βέλτιστου συνδυασμού αισθητήρων/διεγερτών που θα πρέπει να χρησιμοποιηθεί σε μια κατασκευή ώστε να μειωθεί η επιρροή των σημάτων θορύβου. Τα συστήματα που μας απασχολούν είναι κυρίως πλαισιακές κατασκευές. Η προσέγγιση που ακολουθούμε είναι αμιγώς υπολογιστική, με χρήση της μεθόδου των πεπερασμένων στοιχείων για την επίλυση προβλημάτων δυναμικής στο πεδίο του χρόνου. Τέλος, η συγκεκριμένη εργασία συνδυάζει διάφορους επιστημονικούς κλάδους, όπως είναι αυτός της Μηχανικής, της Δυναμικής των Κατασκευών, της Υπολογιστικής Μηχανικής, των Εφαρμοσμένων Μαθηματικών και του Προγραμματισμού.

Λέξεις - φράσεις κλειδιά — Μη καταστροφικός έλεγχος, παρακολούθηση δοκιμής ακεραιότητας κατασκευών, διέγερση, δυσλειτουργία, εντοπισμός θέσης, υπολογιστικές μέθοδοι, Μέθοδος Πεπερασμένων Στοιχείων, λόγος σήμα-προς-θόρυβο.

Abstract

In recent years, control and monitoring of dynamic system operation is a common process. In this paper, we deal with the implementation of structural integrity monitoring (Structural Health Monitoring) and graphical stamping of excitation and damage points. Through these processes, we are able to detect sources of excitation or possible malfunction (due to spoilage or wear) on the construction. In addition, we deal with the possibility of spatially locating excitation and/or failure points. The stimulation of the system as well as the collection of response data can be done using piezoelectric ceramic materials (PZT) which will synthesize sensor/stimulator devices. Signal-to-Noise Ratio also studied in this work and the main aim is the foundation of the optimal receivers/stimulators set which should use on the structure to reduce the influence of the «ghosts». The systems we employ are mainly frame constructions. The approach we follow is purely computational, using the finite element method to solve dynamic problems in the field of time. To sum up, this work combines a variety of scientific areas, such as Mechanics, Structural Dynamics, Computational Mechanics, Applied Mathematics and Programming.

Keywords — Non-destructive testing, Structural Health Monitoring, stimulation, malfunction, positioning, computational methods, Finite Element Method, Signal-to-Noise Ratio.

Ευχαριστίες

Θα ήθελα να απευθύνω θερμές ευχαριστίες στον επιβλέποντα καθηγητή κύριο Γεώργιο Ε. Σταυρουλάκη για την ανάθεση της συγκεκριμένης εργασίας και για την εμπιστοσύνη που μου έδειξε καθ΄ όλη την διάρκεια εκπόνησης της.
Επιπλέον, οφείλω ένα μεγάλο ευχαριστώ στον συνεπιβλέποντα Δρ. Χρήστο Γ. Παναγιωτόπουλο που μου έδωσε την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα και για τη συνεχή βοήθεια, καθοδήγηση και στήριξη του σε όλη τη διάρκεια υλοποίησης της παρούσας διπλωματικής εργασίας.
Ευχαριστώ επίσης τα μέλη της οικογένειας μου που με την βοήθεια, εμπιστοσύνη και συμπαράσταση τους κατάφερα να φτάσω μέχρι εδώ.
Τέλος, δεν μπορώ να μην αναφερθώ στους φίλους και συμφοιτητές μου που μαζί τους πορεύτηκα σε αυτό το ταξίδι.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Έννοιες και παραδείγματα των Μη Καταστροφικών Ελέγχων

Ο έλεγχος ενός συνόλου ή τμήματος μιας κατασκευής αποτελεί αναπόσπαστο κομμάτι της δουλειάς ενός μηχανικού. Είναι σημαντικό να υπάρχει ανά πάσα στιγμή η γνώση της καλής ή κακής λειτουργίας μιας κατασκευής, ποιος είναι ο εκτιμώμενος χρόνος ζωής και ποια η δομική της ακεραιότητα. Η ανάγκη, λοιπόν, για αξιόπιστο χαρακτηρισμό υλικών και δομών έχει δημιουργήσει μια πληθώρα τεχνικών και οργανολογίας. Όμως, η πλειοψηφία αυτών απαιτεί την κατεργασία της κοπής για να δοθούν ακριβή αποτελέσματα, πράγμα που σημαίνει οτι η δομή του υλικού θα καταστραφεί και προφανώς το κόστος της διαδικασίας θα αυξηθεί. Αυτό μας οδηγεί στο συμπέρασμα πως ιδανικές μέθοδοι για τον έλεγχο της μικροδομής και της μακροδομής ενός υλικού είναι οι μέθοδοι Μη Καταστροφικού Ελέγχου (ΜΚΕ). Απ΄ την ονομασία μπορεί να γίνει αντιληπτό πως πρόκειται για φυσικές διαδικασίες που δεν καταστρέφουν την σύσταση των υλικών ή την λειτουργικότητα τους, δίνοντας μετρήσεις και αποτελέσματα με σκοπό την ανίχνευση πηγών ενέργειας, ατελειών, διαρροών ή οποιασδήποτε άλλης αστοχίας.

Ο ΜΚΕ έχει ένα αρκετά μεγάλο εύρος εφαρμογών, όπως για παράδειγμα στη ναυπηγική, στη μηχανική, στην αεροδιαστημική, στην αυτοκινητοβιομηχανία, στις θαλάσσιες κατασκευές, σε μνημεία πολιτιστικής κληρονομιάς. Αυτό οφείλεται στο γεγονός ότι επιφέρει αποτελέσματα στη μελέτη των υλικών όσον αφορά τις φυσικές ιδιότητες, την μικροδομή, τη μακροδομή, τη μηχανική τάση, τα ελαττώματα και τις διαρροές. Ο ΜΚΕ αναφέρεται σε διαφορετικές μεθόδους που βασίζονται σε ελαστικά κύματα, διεισδύουσες ακτινοβολίες, φως, ηλεκτρικά και μαγνητικά πεδία, κ.α. Επιπλέον, αναπτυσσόμενος τομέας των ΜΚΕ είναι και η ιατρική. Για παράδειγμα, σε ιατρικές μεθόδους ανίχνευσης πέτρας στα νεφρά, ενώ σημαντικό κίνητρο αποτελεί ο εντοπισμός ή η θεραπευτική αγωγή καρκινικών ιστών.

Ο ρόλος των ΜΚΕ βασίζεται σε δύο κύριους άξονες. Ο πρώτος είναι η αποτροπή της καταστροφικής αστοχίας των κατασκευών και ο δεύτερος άξονας αφορά την ασφαλή και οικονομική διαχείριση των κατασκευών. Αναφερόμενοι στον πρώτο άξονα μπορούμε να παραθέσουμε αρκετά παραδείγματα αστοχιών στην ιστορία του τεχνικού πολιτισμού τα οποία συνέβησαν λόγω αδυναμίας ανίχνευσης της βλάβης σε αρχικό στάδιο. Φυσικά, η πλειοψηφία αυτών έφεραν σημαντικές απώλειες ανθρώπων αλλά και σε δευτερεύον επίπεδο την απώλεια κεφαλαίου. Ένα εύστοχο παράδειγμα από την αεροναυπηγική είναι η βλάβη του αεροσκάφους «Comet» το 1952, όπου οι ρωγμές ξεκίνησαν από τις ακμές των παραθύρων. Συγκεκριμένα, η τοποθεσία του κινητήρα σε συνδυασμό με τα τετραγωνικά παράθυρα προκάλεσαν αυξημένες τάσεις στα μεταλλικά μπουλόνια των ακμών των παραθύρων, με καταστροφικό αποτέλεσμα. Εκείνα τα χρόνια δεν υπήρχε η τεχνογνωσία για τον προσδιορισμό των παραμορφώσεων με συνέπεια να μην είναι εμφανής η συγκέντρωση τάσεων στις ακμές. Έκτοτε, τα αεροσκάφη έχουν παράθυρα σχήματος οβάλ. Επιπλέον, μεγάλα ατυχήματα προκλήθηκαν στα σιδηροδρομικά δίκτυα λόγω ρωγμών στους άξονες ή σε άλλα κομμάτια των τρένων. Οι αφορμές ήταν αρκετές κι έτσι ο επιστημονικός κόσμος ξεκίνησε να ερευνά και να αναπτύσσει τεχνικές με τις οποίες θα μπορούσε κανείς να εντοπίσει την βλάβη/ρωγμή σε αρχικό στάδιο. Οι κλάδοι που αναπτύχθηκαν ήταν της θραυστομηχανικής για καλύτερο σχεδιασμό των κατασκευών, της επιστήμης των υλικών για μεγαλύτερη αντοχή των υλικών και μεταγενέστερα διάφορες μέθοδοι ΜΚΕ για την έγκαιρη ανίχνευση της βλάβης. Ο δεύτερος άξονας των ΜΚΕ που αναφέραμε παραπάνω είναι αρκετά σημαντικός και ειδικότερα στις μέρες μας που η χρήση διάφορων κατασκευών (πχ. οδικά δίκτυα, γέφυρες, κτίρια) γίνεται σε καθημερινό επίπεδο. Όσο μεγαλύτερη εξέλιξη υπάρχει στον τομέα των ΜΚΕ τόσο μεγαλύτερη πρόληψη επιτυγχάνεται με αποτέλεσμα την μείωση των θανατηφόρων ατυχημάτων αλλά και την μείωση του κόστους συντήρησης ενός δομικού έργου.

Σκεπτόμενοι τους τρόπους ΜΚΕ των κατασκευών μπορούμε να ξεχωρίσουμε τις μεθόδους παρακολούθησης δομικής ακεραιότητας (Structural Health Monitoring methods). Δηλαδή, η συνεχής παρακολούθηση σε πραγματικό χρόνο μιας κατασκευής ή ακόμα και μιας γραμμής παραγωγής έχει ως αποτέλεσμα τον έγκαιρο εντοπισμό μιας πιθανής βλάβης ή δυσλειτουργίας. Φυσικά αυτό στοχεύει στην άμεση επιδιόρθωση βοηθώντας έτσι στη μείωση του κόστους επιθεώρησης και του κόστους συντήρησης. Δύο αξιόπιστες τεχνικές του τομέα παρακολούθησης της δομικής ακεραιότητας των κατασκευών είναι α) η τεχνική βασισμένη στην διάδοση κυμάτων και β) η τεχνική που βασίζεται στις δονήσεις. Στις τεχνικές αυτές θα πρέπει να έχουμε αποτελέσματα με μεγάλη ακρίβεια για να μπορέσει ο εκάστοτε μηχανικός να προβεί σε σωστά συμπεράσματα.

Για να δοθούν αξιόπιστα και ακριβή αποτελέσματα όσον αφορά τις μετατοπίσεις ή και τις ταχύτητες ενός συστήματος, χρησιμοποιείται κατάλληλη οργανολογία, για παράδειγμα διατάξεις πιεζοηλεκτρικών αισθητήρων/διεγερτών. Τα χαρακτηριστικά τους θα πρέπει να είναι συγκεκριμένα ώστε να πληρούν κάποιες προδιαγραφές όσον αφορά: α) την ακρίβεια: κάθε αισθητήρας θα πρέπει να διαθέτει υψηλή ακρίβεια ώστε να καλύπτεται ο προσδιορισμός των ιδιοτήτων του υλικού ή της παρουσίας βλαβών/ατελειών, β) την αξιοπιστία: το όργανο θα πρέπει να ανιχνεύει ατέλειες/βλάβες ή ιδιότητες του υλικού με μεγάλο βαθμό αξιοπιστιάς. Σε διαφορετική περίπτωση, ο αισθητήρας δεν θα εντοπίσει κάποια βλάβη/ατέλεια που μπορεί να οδηγήσει σε αστοχία του υλικού, γ) την απλότητα: η συχνή χρήση οργανολογίας η οποία χρησιμοποιείται από τεχνικούς σε βιομηχανίες/εργοστάσια και όχι από εξειδικευμένους χειριστές και δ) το χαμηλό κόστος: το κόστος ενός οργάνου θεωρείται χαμηλό όταν αποτελεί ένα ποσοστό του κόστους του υπό επεξεργασίας εξαρτήματος. Για παράδειγμα, στον τομέα της αεροναυπηγικής, έως και το 12% του τελικού κόστους του εξαρτήματος μπορεί να δαπανηθεί για την επιθεώρηση του και κρίνεται ως κρίσιμο για την ασφαλή πτήση του αεροσκάφους.

1.2 Εισαγωγή στη δυναμική των κατασκευών

Τα δυναμικά φορτία προέρχονται είτε από φυσικά αίτια είτε από ανθρώπινες δραστηριότητες και μπορούν να αποφέρουν μέχρι και καταστροφικά αποτελέσματα, ειδικά όταν πρόκειται για έναν σεισμό. Ο άνθρωπος μπορεί να προκαλέσει κυρίως προβλήματα λειτουργικότητας σε μια κατασκευή, αλλοιώνοντας, για παράδειγμα, με ένα χτύπημα (παλμό) τα μηχανικά χαρακτηριστικά της. Γενικά, οι φορτίσεις μπορεί να είναι αιτιοκρατικές (ντετερμινιστικές) ή στοχαστικές (τυχαίες). Στην πρώτη κατηγορία εντάσσονται φαινόμενα που εμπεριέχεται η ανθρώπινη δραστηριότητα και η χρονική μεταβολή των φορτίων μπορεί να είναι γνωστή ή να περιγράφεται από μια συνάρτηση ή ακόμα και να μπορει να παρασταθεί από μια σειρά αριθμητικών σειρών για κάποια διακριτά χρονικά βήματα. Αντίθετα, η σεισμική δόνηση αποτελεί στοχαστική φόρτιση, διότι δεν μπορεί να μελετηθεί χωρίς τη χρήση μετρήσεων και πολλών καταγραφών ώστε να μπορεί να προσδιοριστεί. Δηλαδή, η χρονική μεταβολή είναι άγνωστη και μπορεί να περιγραφεί στοχαστικά.

Μια απλή μορφή δυναμικής φόρτισης είναι αυτή της αρμονικής (ημιτονοειδής) ταλάντωσης, η οποία παρέχει την κατάλληλη θεωρία για την επίλυση των διαφορικών εξισώσεων που την περιγράφουν, ενώ η χρήση πειραματικών μετρήσεων μπορεί να αποδώσει πληροφορίες για το εύρος και τη συχνότητα της. Ωστόσο, υπάρχουν και πιο σύνθετες μορφές δυναμικής φόρτισης, οι οποίες μπορεί να είναι μη-περιοδικές ή η διάρκεια τους να είναι κάποια δευτερόλεπτα. Για παράδειγμα, οι φορτίσεις τύπου πλήγματος ή ο σεισμική δόνηση, η οποία αποτελεί μη-περιοδική φόρτιση μεγάλης χρονικής διάρκειας. Μια κατηγοριοποιήση των δυναμικών φορτίων με βάση την προέλευση τους και την χρονική τους εξέλιξη είναι :

1. Περιβαλλοντικά / Ανθρώπινης δραστηριότητας

2. Περιοδικά / Μη-περιοδικά

3. Πλήγματα / Μεγάλης χρονικής διάρκειας

4. Στοχαστικά / Αιτιοκρατικά

Η επίλυση δυναμικών συστημάτων έχει περάσει πλέον σε επίπεδο καθημερινότητας, γιατί η χρήση ηλεκτρονικών υπολογιστών (Η/Υ) παρέχει τη δυνατότητα ανάπτυξης λογισμικών, τα οποία μπορούν να επιλύουν σύνθετα δυναμικά προβλήματα με μεγάλο αριθμό βαθμών ελευθερίας (ΒΕ). Φυσικά, τα απλά γραμμικά συστήματα μπορούν να επιλυθούν χρησιμοποιώντας μόνο τις μεθόδους αναλυτικής επίλυσης εξισώσεων κίνησης βασιζόμενα στην κλασσική θεωρία της δυναμικής.

Το δυναμικό σύστημα, η φόρτιση και η απόκριση του θεωρούνται οι τρεις βασικοί παράγοντες ενός προβλήματος. Συνήθως, η φόρτιση και το σύστημα θεωρούνται γνωστά και το ζητούμενο είναι ο προσδιορισμός της απόκρισης, ενώ υπάρχουν και περιπτώσεις που γνωρίζουμε την επιβαλλόμενη δύναμη και τις αποκρίσεις που προκύπτουν και το ζητούμενο είναι ο προσδιορισμός των παραμέτρων του συστήματος. Τέτοια παραδείγματα είναι όταν σε κάποιο δομημένο περιβάλλον (π.χ. γέφυρες, κτίρια) τοποθετούμε όργανα μέτρησης της απόκρισης της κατασκευής για κάποια συγκεκριμένη φόρτιση με κύριο στόχο την επαλήθευση των παραμέτρων που χρησιμοποιήθηκαν για την κατασκευή του συστήματος.

Κεφάλαιο 2

Η δυναμική απόκριση των πλαισιακών κατασκευών και η εφαρμογή της μεθόδου χρονικής αντιστρεψιμότητας

Το παρόν κεφάλαιο αποτελεί την εισαγωγή στις έννοιες της δυναμικής των κατασκευών που χρησιμοποιήθηκαν στην παρούσα εργασία. Επιπλέον, αναλύονται οι δύο περιπτώσεις που μελετάμε στην παρούσα διπλωματική εργασία. Η πρώτη περίπτωση (*Ενότητα 2.4*) αφορά μια δοκό μιας διάστασης και συγκεκριμένα με διεύθυνση κατά μήκος του άξονα x, ενώ στη δεύτερη περίπτωση (*Ενότητα 2.5*) ασχολούμαστε με μια κατασκευή δυο διαστάσεων, απαρτιζόμενη από πεπερασμένες δοκούς. Και στις δυο περιπτώσεις επιλύονται τα προβλήματα εντοπισμού της θέσης της δύναμης διέγερσης και της τοποθεσίας κάποιας ατέλειας, διαρροής ή αλλοίωσης πάνω στην κατασκευή. Επιπλέον, μελετάτε στη μονοδιάστατη περίπτωση, η συμπεριφορά του σημείου διέγερσης μετά την προσθήκη τυχαίων τιμών Γκαουσιανού θορύβου.

2.1 Δυναμική συμπεριφορά κατασκευών

Η θεωρία της δυναμικής των κατασκευών παρέχει μεθόδους επίλυσης εξισώσεων κίνησης μιας κατασκευής, οι οποίες περιορίζονται σε περιπτώσεις επίλυσης απλών γραμμικών συστημάτων. Ωστόσο, η καθημερινή χρήση των ηλεκτρονικών υπολογιστών (Η/Υ) σε συνδυασμό με την ανάπτυξη λογισμικού για αριθμητική ανάλυση, καθιστούν πλέον πολύ εύκολη την επίλυση ρεαλιστικών προσομοιωμάτων με πολύ μεγάλο αριθμό βαθμών ελευθερίας[1] (ΒΕ).

Θεωρία Euler - Bernulli

Η μέθοδος Euler - Bernulli[2] είναι συνυφασμένη με την υπόθεση ότι τα επίπεδα τμήματα παραμένουν επίπεδα μετά από κάποια επιβαλλόμενη φόρτιση. Ουσιαστικά, το παραπάνω σημαίνει πως δεν υπάρχουν διατμητικές παραμορφώσεις σε κάποιο τμήμα ενός επιπέδου (π.χ.

[1] Ο Βαθμός Ελευθερίας για κάθε κόμβο είναι οι τρεις μεταφορικοί και οι τρεις στροφές κατά και γύρω από τους άξονες x, y, z αντίστοιχα, ενώ αναπαρίστανται (η «διεύθυνση») με τους ακέραιους αριθμούς από 1 έως 6 [1].

βαθμός ελευθερίας	u_x	u_y	u_z	r_x	r_y	r_z
«διεύθυνση»	1	2	3	4	5	6

[2] https://en.wikipedia.org/wiki/Bending

μια δοκό). Σημαντική προϋπόθεση αποτελεί το γεγονός ότι η μέγιστη καταπόνηση στη συγκεκριμένη μέθοδο δεν πρέπει να ξεπερνά το όριο διαρροής του υλικού.

Χρησιμοποιώντας την μέθοδο Euler - Bernulli για να μελετήσουμε μια δοκό, θα πρέπει η δοκός να τηρεί κάποιες προϋποθέσεις. Αυτές είναι οι εξής:

- λεπτό και ευθύ σχήμα δοκού
- υλικό που αντέχει φορτίσεις κάτω του ορίου διαρροής του, ισοτροπικό και με ομογενή διατομή
- μικρή εκτροπή

Η εξισώση της θεωρίας Euler - Bernulli που μπορεί να περιγράψει μια τέτοια δοκό είναι η ακόλουθη:

$$\frac{d^2w(x)}{dx^2}=\frac{M(x)}{E(x)I(x)} \tag{2.1}$$

όπου, E είναι το μέτρο ελαστικότητας, I η ροπή αδράνειας, M η εσωτερική ροπή κάμψης της δοκού.
Επιπλέον, αν εφαρμοστεί ένα εγκάρσιο φορτίο σε μια ομογενή δοκό με σταθερή διατομή σε όλο το μήκος της, η εξίσωση που περιγράφει τη δύναμη είναι:

$$EI\frac{d^4w(x)}{dx^4}=q(x) \tag{2.2}$$

Συνεπώς, καταλήγουμε στις ακόλουθες εξισώσεις για την ροπή κάμψης M και για την διατμητική δύναμη Q,

$$-EI\frac{d^2w}{dx^2}=M(x), \tag{2.3α'}$$

$$\frac{dM}{dx}=Q(x) \tag{2.3β'}$$

Θεωρία Timoshenko

Η θεωρία Timoshenko λαμβάνει υπ΄ όψιν της τρεις βαθμούς ελευθερίας. Οι δύο αφορούν τις παραμορφώσεις διάτμησης (αξονικές και εγκάρσιες) και ο τρίτος την περιστροφική κάμψη. Επίσης, αποτελεί ένα ανάπτυγμα της θεωρίας Euler-Bernulli εισάγοντας τις διατμητικές τάσεις στα δομικά στοιχεία της δοκού. Οι δυο βασικές υποθέσεις της θεωρίας είναι:

- η παραμορφωμένη δοκός παραμένει ευθεία κατά την διεύθυνση του άξονα της και
- το πάχος της δοκού μένει αναλλοίωτο μετά την παραμόρφωση.

Σύμφωνα με τις υποθέσεις που κάναμε και στην μέθοδος Euler-Bernulli, η εξίσωση που περιγράφει την κυρτότητα της δοκού είναι:

$$EI\frac{d^4w}{dx^4}=q(x)-\frac{EI}{\kappa AG}\frac{d^2q}{dx^2} \tag{2.4}$$

όπου, I είναι η ροπή αδράνειας, A το εμβαδόν διατομής, G το μέτρο διάτμησης, κ ο συντελεστής διάτμησης και $q(x)$ η εγκάρσια δύναμη. Αν ο λόγος Poisson είναι κοντά στο $0,3$, τότε ο συντελεστής διάτμησης κ για ορθογώνιες διατομές ισούται:

$$\kappa = \frac{5+5\nu}{6+5\nu} \tag{2.5}$$

ενώ, για κυκλικές διατομές είναι:

$$\kappa = \frac{6+12\nu+6\nu^2}{7+12\nu+4\nu^2} \tag{2.6}$$

Επιπλέον, η περιστροφή $\phi(x)$ περιγράφεται από την εξίσωση:

$$\frac{d\phi}{dx} = -\frac{d^2w}{dx^2} - \frac{q(x)}{\kappa AG} \tag{2.7}$$

Άρα, η ροπή κάμψης M και η διατμητική δύναμη Q θα είναι:

$$-EI\frac{d\phi}{dx} = M(x), \tag{2.8α'}$$

$$\kappa AG(\frac{dw}{dx} - \phi) = -EI\frac{d^2\phi}{dx^2} = \frac{dM}{dx} = Q(x) \tag{2.8β'}$$

2.2 Χρονική αντιστρεψιμότητα σε κατασκευές

Η χρονική αντιστροφή των αποκρίσεων σε κάποιο μέσο έχει ως στόχο την εστίαση της ενέργειας των κυμάτων, για παράδειγμα ελαστικών, σε κάποιο στόχο (πηγή)[2]. Για να επιτευχθεί αυτό θα πρέπει αρχικά το υπό εξέταση μέσο να διεγερθεί από κάποια δύναμη είτε στην επιφάνεια του είτε σε κάποιο βάθος. Έπειτα, τα κύματα ταξιδεύοντας εντός του μέσου προς όλες τις κατευθύνσεις, θα χάσουν κάποιο μέρος της ενέργειας τους και αυτό οφείλεται σε τρεις θεμελιώδεις παράγοντες εξασθένησης: λόγω της αποσβεννύμενης ακτινοβολίας εξαιτίας της διαστολής της γεωμετρίας του δοκιμίου, λόγω της φυσικής εξασθένησης της κατασκευής ή της απόσβεσης του υλικού, τα οποία προκύπτουν από την εσωτερική τριβή και τέλος, λόγω σημείων με διαφορετικά μηχανικά ή γεωμετρικά χαρακτηριστικά. Για να επιτευχθεί η αποφυγή των προαναφερόμενων παραγόντων εξασθένησης της ενέργειας του κύματος θα πρέπει να χρησιμοποιηθούν όσο το δυνατό περισσότερες πηγές ώστε να αυξηθούν τα ποσά ενέργειας που φτάνουν στο στόχο.

Σ' ένα αντίστροφο πρόβλημα, η ανάλυση είναι συνυφασμένη με διάφορες δυσκολίες, οι οποίες συνήθως συμβαίνουν σε όλα τα προβλήματα χρονικής αντιστροφής. Ωστόσο, σημαντικό κομμάτι απότελεί η γνώση των ιδιοτήτων του χωρίου και του στόχου (πηγή διέγερσης). Στην πραγματικότητα, η ποιότητα τον αποτελεσμάτων μιας τέτοιας επίλυσης εξαρτάται από τις πληροφορίες που έχουμε για τις ιδιότητες του χωρίου και του στόχου. Με άλλα λόγια, όσο καλύτερα γνωρίζουμε τα δεδομένα, τόσο ακρίβεια θα υπάρχει στον εντοπισμό του στόχου ή των στόχων.

Η ιδέα της χρονικής αντιστροφής για τις περιπτώσεις της παρούσας εργασίας ακολουθεί τις ακόλουθες εξισώσεις:
στο ευθύ πρόβλημα,

$$q(x,t) = \sum_{s=1}^{N_s} P(t)\delta(x-x_s) \tag{2.9α'}$$

$$w(x,t) = \sum_{s=1}^{N_s} P(t)\delta(x-x_s) \tag{2.9β'}$$

όπου, $P(t)$ η δύναμη διέγερσης στο χρόνο, $\delta(x-x_s)$ η συνάρτηση Dirac.
Στο αντίστροφο πρόβλημα οι συναρτήσεις που εκφράζουν τις μετατοπίσεις των αξονικών και των καμπτικών κυμάτων είναι:

$$q(x,\tau){=}\sum_{r=1}^{N_r} u_{inc}(T-t)\delta(x-x_r) \tag{2.10α΄}$$

$$w(x,\tau){=}\sum_{r=1}^{N_r} v_{inc}(T-t)\delta(x-x_r) \tag{2.10β΄}$$

όπου, $\tau{=}T-t_0$.
Επίσης, παρόμοια διαδικασία ακολουθείται και στην περίπτωση του εντοπισμού κάποιας αλλοίωσης ή φθοράς πάνω στην κατασκευή. Συγκεκριμένα, το ευθύ πρόβλημα επιλύεται για την αλλοιωμένη δοκό (Total Field) κι έπειτα επιλύεται το χρονικά ανεστραμμένο πρόβλημα. Όμως, οι τιμές που στέλνουμε πίσω στην «υγιή» κατασκευή από κάθε αισθητήρα υπολογίζονται από τη σχέση:

$$u_{sc}{=}u_{tot}-u_{inc} \tag{2.11}$$

Συνεπώς, οι εξισώσεις που επιλύονται στο αντίστροφο πρόβλημα είναι οι ακόλουθες:

$$q(x,\tau){=}\sum_{r=1}^{N_r} u_{sc}(\tau)\delta(x-x_r) \tag{2.12α΄}$$

$$w(x,\tau){=}\sum_{r=1}^{N_r} v_{sc}(\tau)\delta(x-x_r) \tag{2.12β΄}$$

Επιπρόσθετα, στα προβλήματα χρονικής αντιστροφής οι αισθητήρες έχουν διπλή ιδιότητα. Αρχικά, ως αισθητήρες καταγραφής αποτελεσμάτων και εν συνεχεία ως διεγέρτες γιατί για να εντοπιστεί μια πηγή διέγερσης σε μια κατασκευή θα πρέπει να επιλυθούν δύο προβλήματα. Στο πρώτο πρόβλημα (ευθεία επίλυση) οι αισθητήρες χρησιμοποιούνται για να καταγράψουν τα αποτελέσματα κάθε ΒΕ από μια συγκεκριμένη θέση από τη στιγμή που δρα κάποια δύναμη μέχρι να ολοκληρωθεί το φαινόμενο, ενώ στο δεύτερο πρόβλημα η τοποθεσία κάθε αισθητήρα αλλά και κάθε ΒΕ χρησιμοποιούνται για την διέγερση της κατασκευής, στέλνοντας τις τιμές χρονικά ανεστραμμένες.

2.3 Μέθοδος απεικόνισης σε κατασκευές βασιζόμενη στη χρονική αντιστρεψιμότητα

Η απεικόνιση είναι ένας κλάδος στη μηχανική και γενικότερα στην επιστήμη, κατά την οποία γίνεται η παρουσίαση των καταγραφών των κυμάτων που διαδόθηκαν σε κάποιο μέσο ή μια κατασκευή. Το εύρος των περιπτώσεων που χρησιμοποιούνται οι μέθοδοι απεικόνισης είναι αρκετά μεγάλο, όπως για παράδειγμα στην ιατρική, στην οπτική, ή τη γεωφυσική. Ωστόσο, στην μηχανική των κατασκευών είναι σε πρώιμο στάδιο αλλά παρατηρείται συνεχή εξέλιξη των μεθόδων απεικόνισης. Στη συγκεκριμένη διπλωματική εργασία, κατασκευάζονται δύο κώδικες μετα-επεξεργασίας των αποτελεσμάτων (*Κώδικες A.3, A.4*) όχι μόνο για τον εντοπισμό της δύναμης διέγερσης αλλά και για την ανίχνευση φθορών πάνω στην κατασκευή. Επιπλέον, να σημειωθεί εδώ πως όλα τα σχήματα που θα αναλυθούν σε επόμενα κεφάλαια είναι αποτέλεσμα του συνδυασμού αυτών των δύο κομματιών κώδικα.

2.3.1 Μεταβλητές απεικόνισης

Στα προβλήματα χρονικής αντιστρεψιμότητας δεν γνωρίζουμε εκ των προτέρων την χρονική στιγμή κατά την οποία το σήμα συσσωρεύεται στον κόμβο της πηγής διέγερσης ή της αλλοιωμένης περιοχής, με αποτέλεσμα να χρησιμοποιούμε κάποια μεταβλητή απεικόνισης των αποκρίσεων. Συνήθως, η χρονική στιγμή απεικόνισης είναι ευδιάκριτη και επιτυγχάνεται ο εντοπισμός της πηγής ή των πηγών διέγερσης. Ωστόσο, υπολογίζοντας το χωρικό ολοκλήρωμα της εκάστοτε μεταβλητής απεικόνισης επιτυγχάνουμε την ανίχνευση κάποιας πηγής διέγερσης στην κατασκευή χωρίς να πρέπει να αναζητήσουμε τη χρονική στιγμή. Η σχέση που το περιγράφει είναι η ακόλουθη:

$$Im\bar{Var}(x)=\int_0^T ImVar(x,t)dt, \tag{2.13}$$

όπου $ImVar(x,t)$ είναι οι τιμές κάθε κόμβου στο χρόνο εκφρασμένες από την επιλεγμένη μεταβλητή απεικόνισης.

Επιπλέον, υπάρχουν πολλές μεταβλητές που μπορούν να χρησιμοποιηθούν στην απεικόνιση των συγκεκριμένων προβλημάτων. Στην πιο απλή μορφή η απεικόνιση των μετατοπίσεων (u_x, u_y, ψ) ή των ταχυτήτων $(\dot{u_x}, \dot{u_y}, \dot{\psi})$ σε κάθε ΒΕ μπορεί να αποδώσει κάποιο αξιόλογο στιγμιότυπο του σήματος. Ακόμη μια μεταβλητή είναι αυτή της Ευκλείδιας νόρμας,

$$I_{eucl}(t)=w_e(u_x^2+u_y^2+\psi^2)+w_k(\dot{u}_x^2+\dot{u}_y^2+\dot{\psi}^{)} \tag{2.14}$$

όπου, w_e και w_k οι συντελεστές ελαστικής και κινητικής ενέργειας, αντίστοιχα.
Η *Εξ. 2.15* δηλώνει την ενεργειακή ποσότητα που μπορεί να χρησιμοποιηθεί ως μεταβλητή απεικόνισης, ενώ οι *Εξ. 2.16* και *2.17* αποτελούν προσεγγίσεις της, με την δεύτερη να είναι η πιο πρόχειρη εκδοχή της.

$$I_{ed}(t)=\frac{1}{2}\left(EA\left(\frac{\partial u}{\partial x}\right)^2+\kappa GA\left(\frac{\partial v}{\partial x}-\psi\right)^2+EI\left(\frac{\partial \psi}{\partial x}\right)^2\right)+\frac{1}{2}\left(\rho A\left(\frac{\partial u}{\partial t}\right)^2+\rho A\left(\frac{\partial v}{\partial t}\right)^2+\rho I\left(\frac{\partial \psi}{\partial t}\right)^2\right) \tag{2.15}$$

$$\tilde{I}_{ed}(t)=w_{et}(\tfrac{EA+\kappa GA}{2})\left((\tfrac{\Delta u_x}{L})^2+(\tfrac{\Delta u_y}{L})^2\right)+w_{er}EI(\tfrac{\Delta\psi}{L})^2+w_{kt}\rho A(\dot{u}_x^2+\dot{u}_y^2)+w_{kr}\rho I\dot{\psi}^2 \tag{2.16}$$

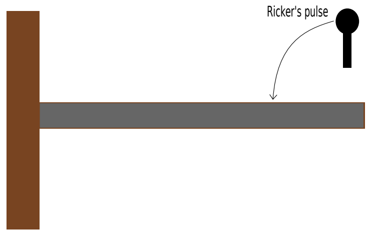
$$\tilde{I}_{ed}(t)=w_{et}(\frac{EA+\kappa GA}{2})\left(u_x^2+u_y^2\right)+w_{er}EI\psi^2+w_{kt}\rho A(\dot{u}_x^2+\dot{u}_y^2)+w_{kr}\rho I\dot{\psi}^2 \tag{2.17}$$

όπου, E το μέτρο ελαστικότητας, A το εμβαδόν διατομής, G το μέτρο διάτμησης και κ ο συντελεστής διάτμησης.

2.4 Ανάλυση περίπτωσης μονοδιάστατης δοκού

Η πρώτη περίπτωση που μελετήθηκε ήταν αυτή της καμπτικής δοκού βασιζόμενη στη θεωρία Timoshenko[3]. Όσον αφορά τις συνοριακές συνθήκες χρησιμοποιήθηκαν με βάση των συνθηκών της μονόπακτης δοκού, οπότε η δοκός πακτώθηκε στο αριστερό της άκρο, ενώ το δεξί άκρο αφέθηκε ελεύθερο. Αυτό σημαίνει ότι στο αριστερό άκρο οι τιμές και των τριών βαθμών ελευθερίας είναι μηδενικές.
Στο *Σχήμα 2.1* απεικονίζεται το σκαρίφημα της δοκού του προβλήματος:

[3]Πηγή: https://en.wikipedia.org/wiki/Timoshenko_beam_theory



Σχήμα 2.1: Θεωρητική δοκός.

Όπως φαίνεται και στο *Σχήμα 2.1* μια σύντομη περιγραφή του ευθέος προβλήματος είναι, ότι η δοκός διεγείρεται σε κάποιο επιλεγμένο σημείο. Το φαινόμενο εξελίσσεται για συγκεκριμένο χρονικό διάστημα και ταυτόχρονα καταγράφεται η απόκριση της δοκού από αισθητήρες x_a (receivers).
Τα γεωμετρικά και μηχανικά χαρακτηριστικά της δοκού που χρησιμοποιήθηκαν στο συγκεκριμένο πρόβλημα ήταν το μήκος L=$30m$, το μέτρο ελαστικότητας E=$1Pa$, ο λόγος Poisson ν=0.25, η πυκνότητα ρ=$1\frac{kg}{m^3}$. Η διατομή της δοκού ήταν ίση με A=$1m^2$, ενώ η θεωρητική διατομή ισούταν με A_s=$\kappa * A$, όπου κ=$\frac{5}{6}$. Η δοκός απαρτίζεται από 1949 πεπερασμένα στοιχεία (βλ. *Κεφάλαιο 3*), δηλαδή στη προσομοίωση της δοκού χρημοποιήθηκαν N_x=1950 κόμβοι. Αυτό σημαίνει ότι το μήκος κάθε πεπερασμένου στοιχείου είναι dx=$0.0154m$.
Η δύναμη διέγερσης στο χρόνο (Ricker's pulse) η οποία έχει συνιστώσες κατά την αξονική και εγκάρσια κατεύθυνση και ασκείται στα $\frac{3}{4}$ του L είναι της μορφής:

$$P(x,t)=\alpha\delta\left(x-x_s\right)\left(1-2\pi^2 s^2\left(t-t_0\right)^2\right)e^{-\pi^2 s^2 (t-t_0)^2} \tag{2.18}$$

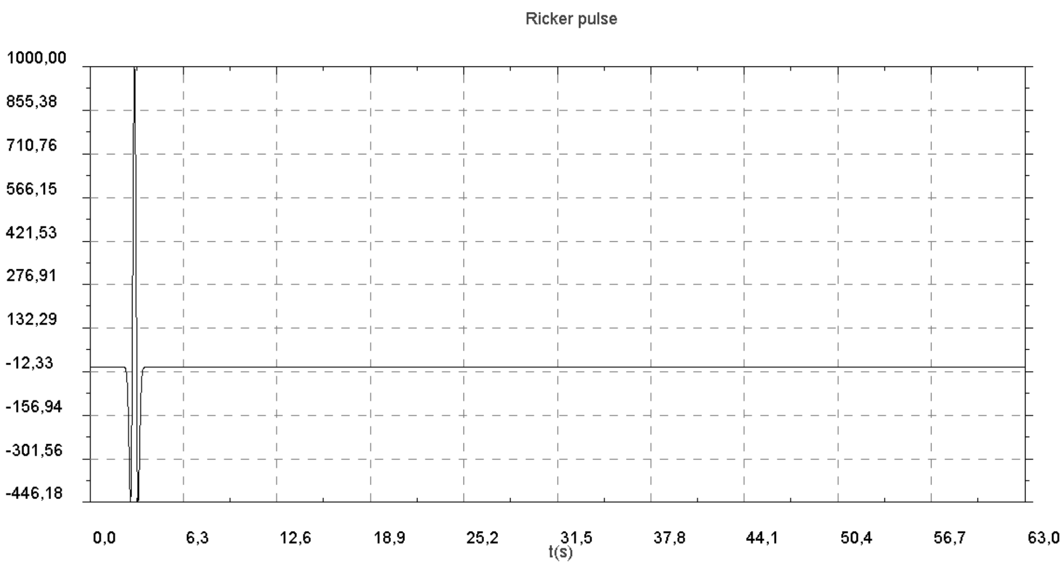
όπου t_0=3, s=1.5 και εύρος α=1000.
Επιπλέον, ο συνολικός χρόνος του φαινομένου ορίσθηκε ίσος με T=$t_0+2\frac{L}{c_1}$=$63sec$. Ο χρόνος αυτός διακριτοποιήθηκε σε N_t=4095 (2^{12}) διακριτά χρονικά βήματα, το οποίο σημαίνει ότι το χρονικό βήμα είναι ίσο με dt=$0.0154s$.
Στην δοκό υπάρχουν τρεις βαθμοί ελευθερίας, ο αξονικός, ο εγκάρσιος και ο περιστροφικός. Άρα, υπάρχουν οι αντίστοιχες αξονικές και καμπτικές ταχύτητες διάδοσης κύματος. Η ταχύτητα διάδοσης κύματος κατά την αξονική διεύθυνση δίνεται από τον τύπο [3]:

$$c_1=\sqrt{\frac{E}{\rho}} \tag{2.19}$$

Ενώ, η ταχύτητα διάδοσης των καμπτικών κυμάτων δίνεται από τον τύπο:

$$c_2 = \sqrt{\frac{\kappa G}{\rho}} \tag{2.20}$$

Στο σημείο αυτό, θα πρέπει να εξηγήσουμε τον τρόπο που επιλέχθηκε η χρονική διακριτοποίηση. Η μεγαλύτερη από τις ταχύτητες διάδοσης είναι αυτή του αξονικού κύματος c_1, άρα θα πρέπει να εξετάσουμε αν ο χρόνος που χρειάζεται το αξονικό κύμα για να διασχύσει ένα πεπερασμένο στοιχείο μήκους dx είναι μικρότερος ή ίσος από το dt. Συνεπώς, μετά από έναν αριθμό δοκιμών καταλήξαμε στις προαναφερόμενα δεδομένα διακριτοποίησης.
Συνεπώς, έχοντας όλα τα παραπάνω μπορούμε να φτάσουμε στον υπολογισμό της διακριτής χρονικής στιγμής που δρα η δύναμη. Αυτό επιτυγχάνεται διαιρώντας την αρχική χρονική τιμή t_0 με το χρονικό βήμα dt. Άρα, ο διακριτός χρόνος διέγερσης στο ευθύ πρόβλημα (Forward Step) είναι ο 195. Σύμφωνα με τη μέθοδο της χρονικής αντιστρεψιμότητας, η εύρεση της χρονικής τιμής διέγερσης προσεγγίζεται από την αφαίρεση του συνολικού αριθμού χρονικών διακριτών βημάτων N_t με τον διακριτό χρόνο διέγερσης στο ευθύ πρόβλημα. Έτσι, ο διακριτός χρόνος διέγερσης προκύπτει ίσος με 3900, δηλαδή για t=60s. Στο σημείο αυτό να σημειωθεί πως τα διαγράμματα των μετατοπίσεων έγιναν στο παραπάνω χρονικό διακριτό βήμα, ενώ στις ταχύτητες ο χρόνος εντοπισμού της δύναμης είναι κατά 10 χρονικά βήματα μικρότερος, δηλαδή ισούται με 3890.
Στο *Σχήμα 2.2* απεικονίζεται η χρονική εξέλιξη δράσης του παλμού. Παρατηρούμε ότι τη χρονική στιγμή t_0=3s, η δύναμη δρα πάνω στη δοκό.



Σχήμα 2.2: Παλμός Ricker.

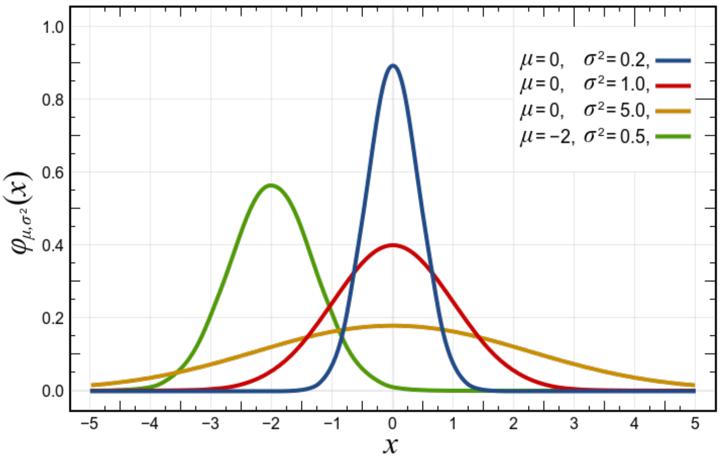
2.4.1 Προσθήκη τυχαίων τιμών Γκαουσιανού θορύβου

Ο θόρυβος αποτελεί μια έννοια πολύ συνηθισμένη στην μελέτη και τον έλεγχο κατασκευών. Εξ' ορισμού θόρυβος είναι μια ανεπιθύμητη ενέργεια συνήθως τυχαίου χαρακτήρα η οποία έχει τη δυνατότητα της υπερκάλυψης αλλά και αποδυνάμωσης του σήματος της διέγερσης. Τα είδη θορύβου χωρίζονται σε δύο μεγάλες κατηγορίες. Η πρώτη κατηγορία είναι αυτή των εξωτερικών θορύβων (π.χ. βιομηχανικά και ατμοσφαιρικά παράσιτα) και η δεύτερη των εσωτερικών (π.χ. θερμικός θόρυβος). Ο λευκός θόρυβος ανήκει στους εξωτερικούς παράγοντες που σχετίζονται με την διαδικασία σημάτων και αποτελεί ένα τυχαίο σήμα ή μια τυχαια διαδικασία η οποία έχει ίδια ένταση σε διαφορετικές συχνότητες. Αυτό έχει ως αποτέλεσμα να δίνει μια σταθερή φασματική πυκνότητα ισχύος. Το όνομά του προέρχεται από το λευκό φως παρόλο που η φασματική πυκνότητα του λευκού φωτός δεν είναι πάνω απ΄ του ορατού. Δηλαδή, αν σκεφτούμε ότι μια ακτίνα λευκού φωτός που προσπίπτει σ΄ ένα πρίσμα θα αναλυθεί στα έξι χρώματα του ορατού φωτός με μήκη κύματος που κυμαίνονται μεταξύ $400nm$ (ιώδες) και $700nm$ (ερυθρό) και με συχνότητες από $430x10^{12}Hz$ (ερυθρό) έως $750x10^{12}Hz$ (ιώδες), μπορούμε να κατανοήσουμε καλύτερα το εύρος του φάσματος της πυκνότητας ισχύος. Η ονομασία του Γκαουσιανού θορύβου (Gaussian Noise)[4] δόθηκε από τον Γερμανό μαθηματικό και επιστήμονα Carl Friedrich Gauss. Αποτελεί έναν στατιστικό θόρυβο ο οποίος είναι ίσος με τη συνάρτηση πυκνότητας πιθανότητας της κανονικής κατανομής[5]. Μ΄ αυτόν τον τρόπο μπορούν να βρεθούν οι κατανομές τυχαίων μεταβλητών που δεν είναι γνωστές. Η συνάρτηση πυκνότητας πιθανότητας του Γκαουσιανού θορύβου δίνεται από την ακόλουθη σχέση:

$$p_G(z)=\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(z-\mu)^2}{2\sigma^2}} \tag{2.21}$$

όπου,
σ^2 η διασπορά και,
μ η μέση τιμή.

Στο *Σχήμα 2.3* φαίνονται οι καμπανοειδείς μορφές της κανονικής κατανομής για διάφορες τιμές της μέσης τιμής μ και της διασποράς σ^2.



Σχήμα 2.3: Κανονική κατανομή

Για την επίλυση του προβλήματος του θορύβου γίνεται χρήση του *Κώδικα 2.1* ο οποίος παρεμβάλλεται στον κυρίως κώδικα επίλυσης A.2 του αντίστροφου προβλήματος (Backward Step) (βλ. Παράρτημα Α΄). Πιο συγκεκριμένα, επιλέχθηκε η διέγερση της δοκού να γίνει στην ακριβή τοποθεσία του κάθε διεγέρτη/ελεγκτή (receiver) για δύο βαθμούς ελευθερίας, τον αξονικό και τον εγκάρσιο. Δηλαδή, πραγματοποιηθήκαν έξι (6) επιλύσεις για το συγκεκριμένο πρόβλημα χρησιμοποιώντας κάθε φορά την ίδια λίστα τυχαίων τιμών Γκαουσιανού

[4]Πηγή: https://en.wikipedia.org/wiki/Gaussian_noise
[5]Πηγή: https://en.wikipedia.org/wiki/Normal_distribution

θορύβου. Τα αποτελέσματα των μετατοπίσεων και των ταχυτήτων αποθηκεύθηκαν σε κατάλληλα επιλεγμένα σημεία πάνω στην κατασκευή, τα οποία θα ονομάζουμε imagers. Ο καθορισμός της θέσης των imagers φαίνεται στις γραμμές 58 έως 63 του *Κώδικα A.2*. Στη συνέχεια, παρουσιάζεται το κομμάτι του *Κώδικα 2.1*.

Listing 2.1: Random Gaussian Noise code

```
1  import java.util.Random
2
3  Random rand = new Random()
4  ListOfNoise = []
5  Nt = 2**12-1
6
7  while (ListOfNoise.size()<=Nt){
8    double val = rand.nextGaussian()*1.0+0.0
9
10   //accept only absolute values which are equal or smaller than 1
11   if (Math.abs(val)<=1){
12     ListOfNoise.add(val)
13   }
14 }
15 println ListOfNoise
16
17
18 //Plot your random Gaussian list
19 RandGaussianNoisePlot = new PlotFrame()
20 y = new plotfunction(ListOfNoise as double[])
21 RandGaussianNoisePlot.addFunction(y)
22 RandGaussianNoisePlot.show()
```

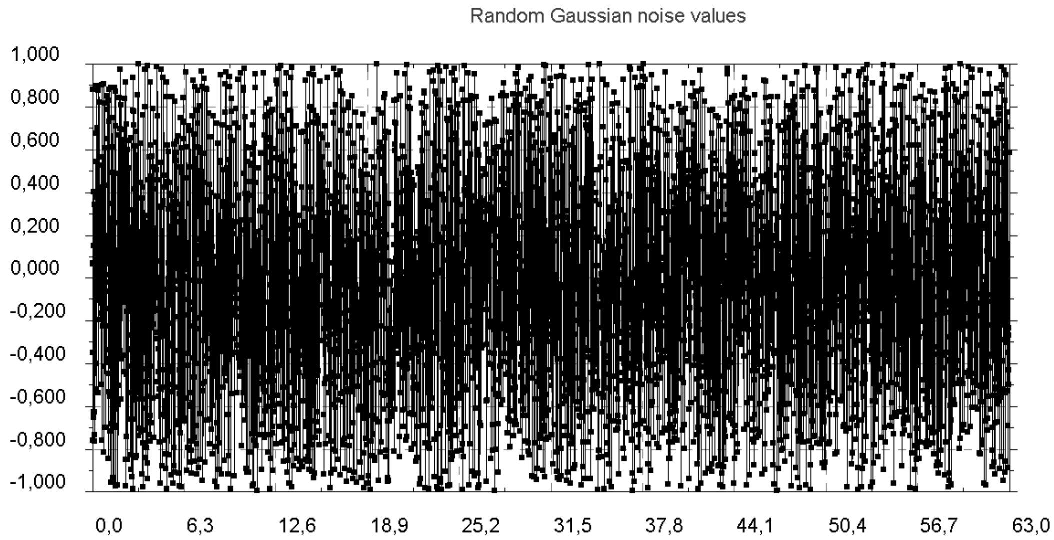
Στις γραμμές 1 και 3 (*Κώδικας 2.1*) καθορίζεται η συνάρτηση δημιουργίας τυχαίων τιμών. Με την εντολή *double val = rand.nextGaussian()*1.0+0.0* εκχωρείται κάθε τυχαία τιμή Γκαουσιανού θορύβου με μέση τιμή 0 και τυπική απόκλιση 1 στην μεταβλητή *val*. Κατά ποσοστό 70% οι τυχαίες τιμές θα είναι μεταξύ του -1 και του 1. Επειδή, αυτό αποτελεί έναν απ' τους περιορισμούς του προβλήματος χρησιμοποείται η συνθήκη if στις γραμμές 11 έως και 13.

Στο *Σχήμα 2.4* απεικονίζονται οι τιμές του Γκαουσιανού θορύβου που χρησιμοποιήθηκαν στην επίλυση του προβλήματος. Κάθε τιμή αντιστοιχεί σε ένα χρονικό διακριτό βήμα, πράγμα που σημαίνει ότι θα έχουμε 4095 τυχαίες τιμές. Στις γραμμές 18 έως 22 του *Κώδικα 2.1* βρίσκονται οι εντολές για την απεικόνιση του *Σχήματος 2.4*.

Ο συντελεστής που χρησιμοποιήθηκε για να επιτευχθεί η αναλογία μεταξύ των αποκρίσεων χωρίς θόρυβο με αυτών του θορύβου ήταν ίσος με το 10% της μέγιστης απόλυτης τιμής όλων των αποκρίσεων για έναν συγκεκριμένο ΒΕ ενός αισθητήρα/διεγέρτη (receiver). Για παράδειγμα, αν επιλεγεί η απεικόνιση των αποτελεσμάτων του αισθητήρα/διεγέρτη με id=1950 κατά τον αξονικό βαθμό ελευθερίας τότε για κάθε imager αποθηκεύεται η μέγιστη τιμή (κατ' απόλυτο) και στη συνέχεια επιλέγεται η μέγιστη τιμή αυτών. Στον *Κώδικα A.4* μεταξύ των γραμμών 93 και 127 φαίνεται η διαδικασία που ακολουθείται για τον εντοπισμό της μέγιστης απόκρισης.

2.4.2 Εύρεση χρονικής διακριτής τιμής εντοπισμού της περιοχής αλλοίωσης/φθοράς (Scattered field)

Στην ενότητα αυτή τροποποιούνται οι μηχανικές ιδιότητες σε κάποιο τυχαίο σημείο της δοκού και η διαδικασία επίλυσης επαναλαμβάνεται. Η περιοχή που επιλέχθηκε είναι αρκετά μικρή και αποτελείται από τους πεπερασμένους κόμβους 350 (0.1795 του L) και 351 (0.18



Σχήμα 2.4: Τυχαίος Γκαουσιανός Θόρυβος (Random Gaussian Noise) συναρτήσει του χρόνου.

του L). Το πεπερασμένο στοιχείο έχει μέτρο ελαστικότητας ίσο με E_{scat}=$0.2Pa$, ο λόγος Poisson είναι ίσος με ν=0.25, η πυκνότητα ισούται με ρ_{scat}=$1\frac{kg}{m^3}$ και η διατομή A_{scat}=$0.5m^2$. Δηλαδή, οι διαφοροποίησεις με το πρόβλημα σε «κανονικές» συνθήκες είναι η μείωση του μέτρου ελαστικότητας E στο $\frac{1}{5}$ του αρχικού καθώς και κάποια ελάττωση της διατομής A, για παράδειγμα το εμβαδόν στο μισό του αρχικού.
Στην πηγή διέγερσης δημιουργούνται τρία κύματα, όσα δηλαδή και οι βαθμοί ελευθερίας. Το καθένα διαδίδεται με διαφορετική ταχύτητα στη δοκό. Αυτό σημαίνει πως θα έχουμε διαφορετικούς χρόνους άφιξης για το καθένα στην περιοχή αλλοίωσης. Ο συλλογισμός που ακολουθήθηκε για τον υπολογισμό κάθε διακριτού χρονικού βήματος είναι ο ακόλουθος: α)υπολογισμός απόστασης μεταξύ τοποθεσίας διέγερσης και σκεδαστή, β) εύρεση χρόνου διάδοσης κάθε κύματος ξεχωριστά και γ) μετατροπή χρόνου διάδοσης σε διακριτό χρονικό βήμα. Συνεπώς, για το αξονικό κύμα έχουμε:

$$l_{sd}{=}x_s - x_d{=}0.75m - 0.18m{=}0.57 \tag{2.22}$$

$$l_{sd} \times 30m{=}17.1m \tag{2.23}$$

$$c_1{=}\frac{l_{sd} \times 30}{t}{=}{>}t{=}\frac{l_{sd} \times 30}{c_1}{=}17.1s \tag{2.24}$$

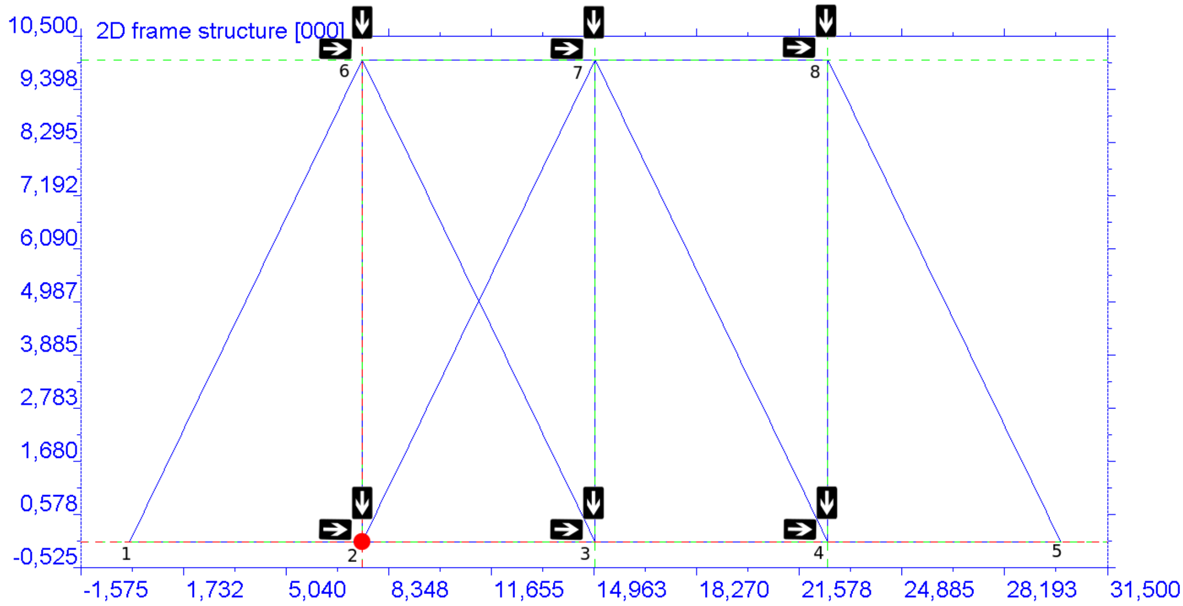
$$N_{t_{scat}}{=}\frac{17.1s + t_0}{d_t}{=}\frac{(17.1 + 3)s}{0.015385s}{=}1306.5 \tag{2.25}$$

Άρα, για το αντίστροφο πρόβλημα (Backward Step) έχουμε:

$$4095 - 1306.5{\simeq}2788 \tag{2.26}$$

2.5 Ανάλυση περίπτωσης δισδιάστατης πλαισιακής κατασκευής

Για την καλύτερη κατανόηση των μεθόδων επίλυσης και εντοπισμού είτε της δύναμης διέγερσης είτε κάποιας αλλοιωμένης περιοχής, μελετήσαμε μια πιο σύνθετη περίπτωση. Γεωμετρικά αποτελούσε μια πλαισιακή κατασκευή δύο διαστάσεων σχηματισμένη με πεπερασμένες δοκούς, η οποία μπορεί να παρατηρηθεί στο *Σχήμα 2.5*. Επίσης, απεικονίζονται οι θέσεις των αισθητήρων/διεγερτών με τα βέλη, ενώ η κόκκινη κουκίδα δείχνει την τοποθεσία της πηγής διέγερσης. Τα βελάκια κατά μήκος του άξονα x δηλώνουν την οριζόντια διεύθυνση καταγραφής ενός αισθητήρα/διεγέρτη και η κατακόρυφη διεύθυνση καταγραφής περιγράφεται από τα βέλη που πέφτουν κάθετα στα δύο οριζόντια επίπεδα της διαδιάστατης κατασκευής.



Σχήμα 2.5: Δισδιάστατη πλαισιακή κατασκευή. Τα βέλη δηλώνουν τους αισθητήρες, ενώ η κόκκινη κουκίδα την τοποθεσία της δύναμης διέγερσης.

Η δύναμη διέγερσης περιγράφεται από την ίδια συνάρτηση [2.4] με αυτή της μονοδιάστατης περίπτωσης και ξεκινά να δρά την χρονική στιγμή t_0=$3s$ στον βασικό κόμβο με id ίσο με 2 (βλ. *Σχήμα 2.5*). Οι δύο συνιστώσες της εφαρμοζόμενης δύναμης ακολουθούν τη κατεύθυνση των αξόνων x και y.

Το μήκος της κάτω οριζόντιας πλευράς αποτελείται από 4 ισομήκεις δοκούς $7.5m$, άρα το συνολικό μήκος είναι L=$30m$. Το ύψος H ισούται με $10m$, ενώ το μήκος των επικλινών δοκών μπορεί να υπολογιστεί από το Πυθαγόρειο θεώρημα ($12.5m$ κάθε επικλινής δοκός). Τα μηχανικά χαρακτιριστικά της κατασκευής είναι το μέτρο ελαστικότητας E=$1Pa$, ο λόγος Poisson ν=0.25, η πυκνότητα ρ=$1\frac{kg}{m^3}$ και το εμβαδόν διατομής A=$1m^2$. Και στην περίπτωση της δισδιάστατης κατασκευής ο συντελεστής διάτμησης είναι k=$\frac{5}{6}$, άρα το θεωρητικό εμβαδόν διατομής ισούται με A_s=$0.8333m^2$. Όσον αφορά την χωρική διακριτοποίηση, κάθε δοκός αποτελείται από ένα πλέγμα (mesh) 200 κόμβων, το οποίο σημαίνει 199 πεπερασμένα στοιχεία. Λόγω των διαφορών που υπάρχουν στα μήκη των τριών διαφορετικών δοκών που διαθέτει η κατασκευή, είναι προφανές πως και το μήκος κάθε πεπερασμένου στοιχείου θα διαφέρει από δοκό σε δοκό. Επομένως, για τις οριζόντιες δοκούς έχουμε ένα μήκος στοιχείων ίσο με $0.0375m$, στις κάθετες τα στοιχεία έχουν μήκος $0.05m$ και στις διαγώνιες το μήκος ενός στοιχείο είναι $0.0625m$.

Οι ταχύτητες διάδοσης των κυμάτων είναι οι c_1=$1\frac{m}{s}$ και c_2=$0.277\frac{m}{s}$. Επίπλέον, η συνολική διάρκεια εξέλιξης του φαινομένου ορίσθηκε ίδια με της μονοδιάστατης περίπτωσης και είναι ίση με T=63sec, ενώ το χρονικό βήμα που επιλέχθηκε για να υπάρχει ευστάθεια στο πρόβλημα ήταν dt=0.0308s, το οποίο δίνει ως αποτέλεσματα 2048 (2^{11}) χρονικά διακριτά βήματα.

2.5.1 Εύρεση χρονικής διακριτής τιμής εντοπισμού της περιοχής αλλοίωσης/φθοράς (Scattered field)

Το πρόβλημα ανίχνευσης κάποιας δυσλειτουργίας στο υλικό της κατασκευής μελετήθηκε και στην δισδιάστατη περίπτωση. Στο *Σχήμα 2.6* φαίνεται η περιοχή με τη φθορά μεταξύ των κόμβων με id ίσο με 1491 και 1501. Το μέτρο ελαστικότητας E_{scat} στην συγκεκριμένη τοποθεσία ισούται με $0.2Pa$, δηλαδή $E_{scat}=\frac{E}{5}$. Τα υπόλοιπα μηχανικά και γεωμετρικά χαρακτηριστικά της δοκού παρέμειναν αναλλοίωτα.
Για την εύρεση και απεικόνιση της συγκεκριμένης τοποθεσίας απαιτείται ο υπολογισμός του διακριτού χρονικού βήματος κατά το οποίο γίνεται η άφιξη του ταχύτερου κύματος. Επομένως, ακολουθήσαμε την υπολογιστική διαδικασία της μονοδιάστατης περίπτωσης. Η απόσταση του κόμβου 2 με τον κόμβο 1491 (βλ. *Σχήμα 2.6*) είναι:

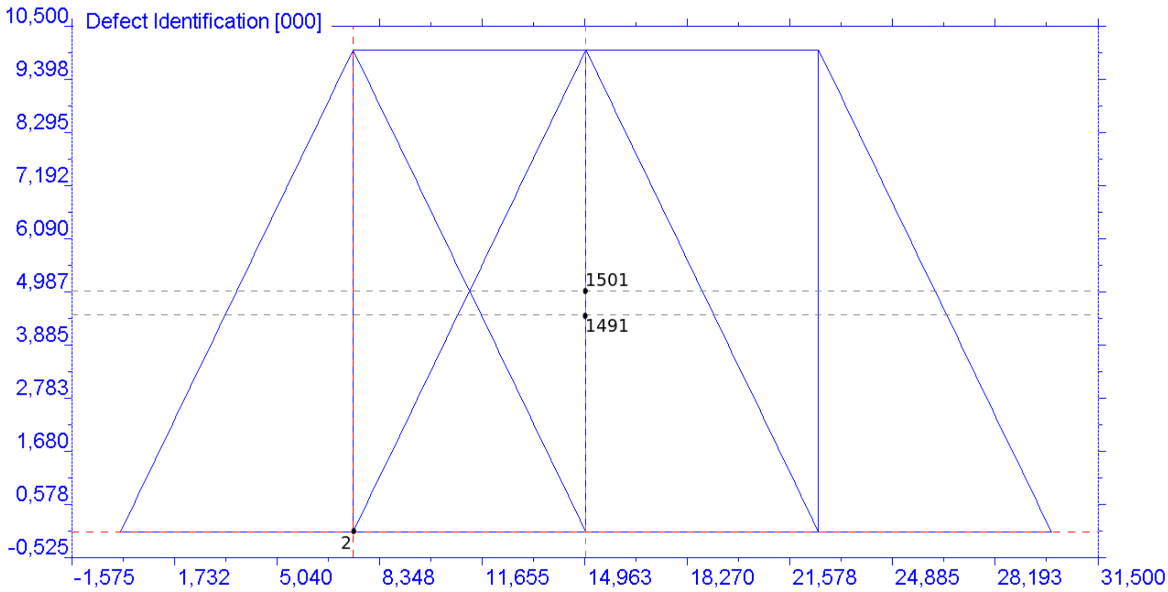
$$l_{sd}=x_{2,3}+x_{3,1491}=7.5m+4.5m=12.0m \tag{2.27}$$

$$c_1=\frac{l_{sd}}{t}=>t=\frac{l_{sd}}{c_1}=12.0s \tag{2.28}$$

$$N_{t_{scat}}=\frac{12.0s+t_0}{dt}=\frac{(12+3)s}{0.0308s}=487 \tag{2.29}$$

Υπολογίσαμε ότι το κύμα με την μεγαλύτερη ταχύτητα χρειάζεται $12s+t_0$ για να φτάσει στην αλλοιωμένη περιοχή στο ευθύ πρόβλημα και μέσω της *Εξίσωσης 2.29* μετατράπηκε σε χρονικό διακριτό βήμα. Άρα, σύμφωνα με την θεωρία της μεθόδου χρονικής αντιστροφής το διακριτό χρονικό βήμα που θα χρησιμοποιήσουμε στην απεικόνιση των αποτελεσμάτων είναι:

$$2047-487\simeq1560 \tag{2.30}$$



Σχήμα 2.6: Απεικόνιση της αλλοιωμένης περιοχής (εντός των κόμβων 1491 και 1501) και της πηγής διέγερσης (κόμβος 2).

Κεφάλαιο 3

Μέθοδος Πεπερασμένων Στοιχείων, Περιβάλλον εργασίας και προγραμματισμού

3.1 Ιστορική αναδρομή της μεθόδου των πεπερασμένων στοιχείων

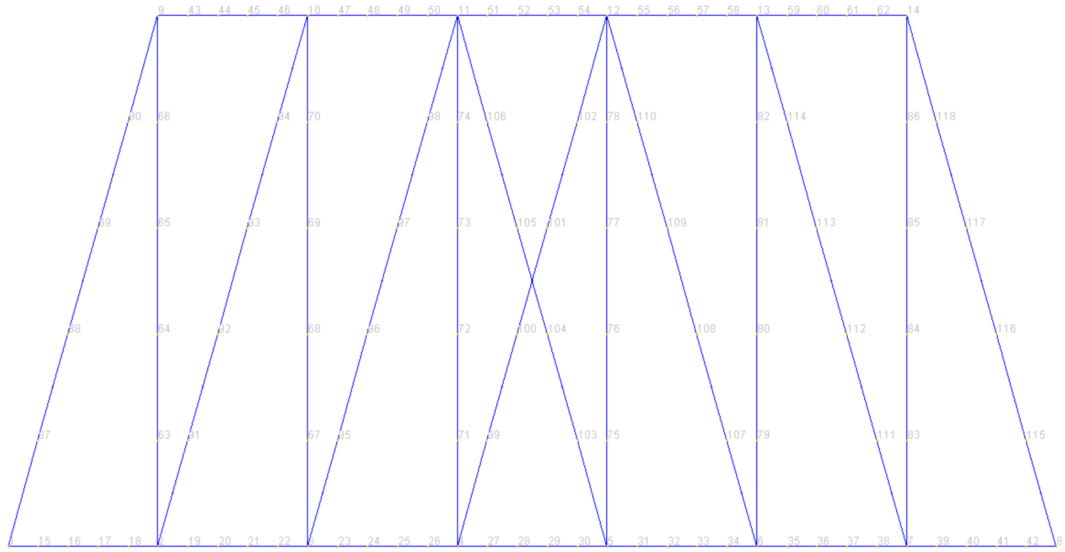
Η μέθοδος που χρησιμοποείται στην επίλυση των προβλημάτων της παρούσας εργασίας είναι η Μέθοδος των Πεπερασμένων Στοιχείων (ΜΠΣ).

Η Μέθοδος Πεπερασμένων Στοιχείων (ΜΠΣ) πρωτοεμφανίστηκε στην αρχαιότητα όταν οι πρώτοι γεωμέτρες προσέγγισαν με πεπερασμένα στοιχεία την τιμή του π ($\cong 3.14$) και αποτελεί αντικείμενο έρευνας και ανάπτυξης από το 1950 περίπου και έπειτα. Μεγάλοι επιστήμονες της εποχής πρωτοασχολήθηκαν με την εξέλιξη της μεθόδου όπως ο Ιωάννης Αργύρης, ο Boris Galerkin, ο Walther Ritz και άλλοι. Φτάνοντας στο σήμερα μπορούμε να πούμε ότι εξακολουθεί να είναι δύσκολο να δοθεί μια αντικειμενική εικόνα της εξέλιξης της μεθόδου των πεπερασμένων στοιχείων. Αυτό συμβαίνει διότι i) υπήρξε ταχεία πρόοδος της μεθόδου και ii) η επιστημονική προέλευση των ερευνητών δεν ήταν κοινή.

Η αδιάκοπη έρευνα πάνω στην μέθοδο των πεπερασμένων στοιχείων οδήγησε στο συμπέρασμα πως υπάρχουν πέντε ισοδύναμοι μέθοδοι διατύπωσης της μεθόδου. Μια προς μια δίνουν ως αποτέλεσμα τις ίδιες ακριβώς μητρωϊκές εξισώσεις κίνησης. Αυτές είναι: 1) η Αρχή των Δυνατών Έργων, 2) η Ελαχιστοποίηση της Ολικής Ενέργειας, 3) η μέθοδος Galerkin, 4) η Αρχή του Hamilton και 5) η μέθοδος των εξισώσεων Lagrange. Όπως αναφέρει στο βιβλίο του ο Clough (Clough 1975), «*η μέθοδος που πρέπει να χρησιμοποείται σε κάθε περίπτωση είναι κυρίως θέμα ευκολίας και προσωπικής προτίμησης· η εκλογή γενικά εξαρτάται από τη φύση του θεωρούμενου δυναμικού συστήματος*». Επίσης, σημαντικό ρόλο παίζει και το υπόβαθρο του εκπαιδευόμενου μιας και η μέθοδος των πεπερασμένων στοιχείων μπορεί να χρησιμοποοιηθεί από διάφορα επιστημονικά πεδία. Για παράδειγμα, η ενασχόληση των μηχανολόγων ή ναυπηγών με θερμικά και ακουστικά προβλήματα καθώς επίσης και η μηχανική των ρευστών, επιβάλλουν την εισαγωγή της μεθόδου Galerkin, ενώ η εκπαίδευση των πολιτικών μηχανικών θα μπορούσε ενδεχομένως να περιοριστεί στη διδασκαλία μόνο της Αρχής των Δυνατών Έργων ή της Αρχής της Ελαστικής Ολικής Δυνητικής Ενέργειας [4].

Η υποδιαίρεση των κατασκευών σε μεγάλο πλήθος μικρών στοιχείων (πεπερασμένα στοιχεία) ξεκίνησε από τον τομέα της δομοστατικής (structural engineering). Οι πρώτες προσπάθειες έγιναν κατά της διάρκεια της δεκαετίας του 1940, με την ανάπτυξη αναλογιών μεταξύ πραγματικών διακριτών στοιχείων, για παράδειγμα ράβδους και δοκούς, με στόχο τον δομικό σχεδιασμό αεροσκαφών. Για το σκοπό αυτό, αναπτύχθηκαν μητρωϊκές μέθοδοι. Ωστόσο, οι αυξημένες απαιτήσεις ανάλυσης των συγκεκριμένων προβλημάτων κατέστησαν τις ημι-αναλυτικές μεθόδους ανεπαρκείς και τότε ξεκινά η έρευνα νέων και πιο αξιόπιστων προσεγγίσεων.

Με απλά λόγια τα πεπερασμένα στοιχεία είναι η αναπαράσταση ενός συνεχούς μέσου με μονοδιάστατα, δισδιάστατα ή τρισδιάστατα γεωμετρικά σχήματα ή «στοιχεία» (βλ. *Σχήμα 3.1*). Στη γλώσσα των μαθηματικών μεταφράζεται ως η αντικατάσταση συνεχών συναρτήσεων με τμηματικές προσεγγίσεις. Το σύνολο των τμηματικών προσεγγίσεων οδηγούν στην συνολική λύση του προβλήματος.



Σχήμα 3.1: Αναπαράσταση μονοδιάστατων Πεπερασμένων Στοιχείων σε πλαισιακή κατασκευή με δοκούς με χρήση του λογισμικού ανοιχτού κώδικα SDE.

3.2 Μέθοδοι χρονικής ολοκλήρωσης

Στην παρούσα ενότητα θα αναλύσουμε συνοπτικά τις μεθόδους χρονικής ολοκλήρωσης. Τα κρίσιμα χαρακτηριστικά ενός αλγορίθμου χρονικής ολοκλήρωσης είναι α) η αριθμητική ευστάθεια και β) η ακρίβεια του. Η αριθμητική ευστάθεια είναι επιβεβλημένη προκειμένου να εξασφαλίζεται η σύγκλιση των αποτελεσμάτων του αλγορίθμου. Αν η απαίτηση της αριθμητικής ευστάθειας εισάγει περιορισμούς στο μέγεθος της χρονικής παραμέτρου διακριτοποίησης Δt τότε ο αλγόριθμος χαρακτηρίζεται ως υπό συνθήκη ευσταθής. Σε αντίθετη περίπτωση, χαρακτηρίζεται ως απόλυτα ευσταθής. Η δέσμευση του μεγέθους του χρονικού βήματος Δt ($[sec]$) αποτελεί αναγκαία συνθήκη ώστε να εξασφαλισθεί η σύγκλιση του αλγορίθμου αλλά όχι ικανή. Συνεπώς, η επιθυμητή ακρίβεια μπορεί να απαιτεί ακόμα μικρότερο μέγεθος του βήματος χρονικής διακριτοποίησης Δt.

Σε γενικές γραμμές, οι αλγόριθμοι χρονικής ολοκλήρωσης μπορούν να καταταχθούν σε δύο θεμελιώδεις κατηγορίες:

1. Πεπλεγμένες (implicit), όπου η εξίσωση διαφορών έχει στο ένα της σκέλος περισσότερα του ενός άγνωστα μεγέθη του διακριτού χρόνου, με αποτέλεσμα να μην γίνεται άμεσα ο διαχωρισμός γνωστών από αγνώστους, άρα και η επίλυση της.

2. Ρητές (explicit), όπου η αντίστοιχη εξίσωση διαφορών έχει στο ένα της σκέλος μόνο έναν άγνωστο του διακριτού χρόνου, οπότε και επιτυγχάνεται ο διαχωρισμός και η επίλυση ως προς το μέγεθος αυτό.

Μέθοδος β-Newmark

Η μέθοδος β-Newmark εντάσσεται στην οικογένεια των μεθόδων αριθμητικής ολοκλήρωσης, οι οποίες επιλύουν προβλήματα στη δυναμική των κατασκευών. Επίσης, αποτελεί μια ρητή (implicit) μέθοδο, όπως είναι και η μέθοδος των Κεντρικών Διαφορών, ενώ άλλες μέθοδοι επίλυσης άρρητων (explicit) συναρτήσεων, είναι πιο αποτελεσματικές για την ανάλυση των περιπτώσεων της παρούσας διπλωματικής εργασίας μιας και ασχολούμαστε με προβλήματα διακυματικής διάδοσης. Ωστόσο, η αριθμητική μέθοδος β-Newmark είναι αρκετά εύκολη στον προγραμματισμό της και αποδίδει αλγορίθμους χωρίς συνθήκη ευστάθειας στην περίπτωση που δοθούν κατάλληλες παράμετροι. Οι δυο αυτές παράμετροι είναι οι συντελεστές β και γ, οι οποίοι διαμορφώνουν την μεταβολή της επιτάχυνσης στα άκρα κάθε χρονικού βήματος και έτσι μπορούν να προσεγγιστούν οι μετακινήσεις και οι ταχύτητες στο συγκεκριμένο βήμα. Η σχέση των μετατοπίσεων δίνεται από τον τύπο,

$$u_{n+1}{=}u_n + \Delta t\dot{u}_n + (\frac{1}{2} - \beta)\Delta t^2\ddot{u}_n + \beta\Delta t^2\ddot{u}_{n+1}, \tag{3.1}$$

και για την ταχύτητα,

$$\dot{u}_{n+1}{=}\dot{u}_n + (1-\gamma)\Delta t\ddot{u}_n + \gamma\Delta t\ddot{u}_{n+1}. \tag{3.2}$$

Επιλύοντας την *Εξ. 3.1* ως προς $\ddot{u}_{n+1}$ και αντικαθιστώντας στην επόμενη σχέση των ταχυτήτων (*Εξ. 3.2*) θα δημιουργηθεί μια έκφραση της επιτάχυνσης και της ταχύτητας στο χρονικό βήμα $n+1$, συναρτήσει των συντελεστών β και γ, αλλά και της μετακίνησης, της ταχύτητας και της επιτάχυνσης στο προηγούμενο χρονικό βήμα n καθώς και της μετακίνησης στο χρονικό βήμα $n+1$.
Η ανισότητα $0.5 \leqslant \gamma \leqslant 2\beta$ εκφράζει την σχέση που υπάρχει ανάμεσα στις δυο παραμέτρους της μεθόδου β-Newmark, έτσι ώστε να δίνει ευσταθή σχήματα χρονικής ολοκλήρωσης άνευ συνθηκών. Διαφορετικά, η ευστάθεια στο βήμα χρονικής διακριτοποίησης δίνεται από τη συνθήκη:

$$\Delta t \leqslant \Delta t_{cr}{=}\frac{1}{\omega_0}\frac{\xi(\gamma-0.5)+\sqrt{0.5\gamma-\beta+\xi^2(\gamma-0.5)^2}}{0.5\gamma-\beta} \tag{3.3}$$

Δυο συνδυασμοί των παραμέτρων β και γ είναι:
(α) η μέθοδος μέσης σταθερής επιτάχυνσης με $\beta{=}\frac{1}{4}, \gamma{=}\frac{1}{2}$ και
(β) η μέθοδος γραμμικής επιτάχυνσης με $\beta{=}\frac{1}{6}, \gamma{=}\frac{1}{2}$.
Σε όλες τις επιλύσεις της παρούσας εργασίας χρησιμοποιήθηκε η περίπτωση (α), η οποία είναι ευσταθής χωρίς συνθήκες.

Συνοριακές συνθήκες

Ο καθορισμός των συνοριακών συνθηκών έγινε σύμφωνα την μεθόδο της ποινής[1] (Penalty method). Ουσιαστικά, πρόκειται για μια μέθοδο επίλυσης προβλημάτων βελτιστοποίησης

[1]Πηγή: https://en.wikipedia.org/wiki/Penalty_method

υπό περιορισμούς. Η μέθοδος που ακολουθείται αντικαθιστά το δοθέν πρόβλημα σε ένα χωρίς περιορισμούς, του οποίου οι λύσεις προσεγγίζουν αυτές του αρχικού προβλήματος. Η παραπάνω μετατροπή απαιτεί τον καθορισμό της συνάρτησης της ποινής (Penalty function), στην οποία εντάσσεται η παράμετρος της ποινής πολλαπλασιασμένη με την τιμή παραβίασης του περιορισμού. Συνήθως, είναι μια μεγάλη τιμή.

3.3 Γλώσσα προγραμματισμού Java

Στις μέρες μας ο προγραμματισμός αποτελεί αναπόσπαστο κομμάτι τις καθημερινότητας των ανθρώπων. Τα πάντα κρύβουν πίσω τους κάποιο εκτελέσιμο αρχείο με εκατοντάδες ή και χιλιάδες γραμμές κώδικα. Ένα απλό παράδειγμα είναι η χρήση του ηλεκτρονικού υπολογιστή, που πλέον καθένας μας έχει τουλάχιστον έναν και τον χρησιμοποιεί καθημερινά. Όμως η πλειοψηφία απ΄ τους κοινούς χρήστες δεν γνωρίζει ότι για κάθε λειτουργία του απαιτεί την δημιουργία κώδικα σε κάποια συγκεκριμένη γλώσσα προγραμματισμού. Οι γλώσσες προγραμματισμού ποικίλλουν. Μερικές από τις πιο δημοφιλείς είναι η Java[2], η Python, η Ruby, η SQL κ.α. Η Java χαρακτηρίζεται από πολλούς μια από τις πιο σημαντικές γλώσσες αντικειμενοστραφούς προγραμματισμού μιας και μπορεί να χρησιμοποιηθεί με ευκολία σε κάθε λειτουργικό σύστημα (Operating System) χωρίς να γίνει μεταγλώττιση ή κάποια τροποποίηση στον πηγαίο κώδικα. Φυσικά όλα τα παραπάνω προϋποθέτουν την εγκατάσταση ενός λογισμικού ανάπτυξης της Java (Software Development Kit (SDK)) στον εκάστοτε ηλεκτρονικό υπολογιστή, το οποίο περιέχει έναν μεταγλωττιστή (compiler), έναν διερμηνέα (interpeter) και διάφορα άλλα εργαλεία (tools) που χρησιμοποιούνται για την ανάπτυξη μιας εφαρμογής. Ένας καλός τρόπος διευκόλυνσης αλλά και ελαχιστοποίησης του χρόνου ανάπτυξης εφαρμογών είναι η χρήση διάφορων ολοκληρωμένων περιβάλλοντων ανάπτυξης (Integrated Development Environments (IDE)) τα οποία προσφέρουν γραφικό περιβάλλον με επιλογές έτοιμων εντολών, πλαίσια κειμένου και άλλα αντικείμενα (objects) μέσω των οποίων υπάρχει μεγάλη διαδραστικότητα της πλατφόρμας με το χρήστη. Ευρέως γνωστά IDE είναι τα Netbeans, Eclipse, JBuilder.

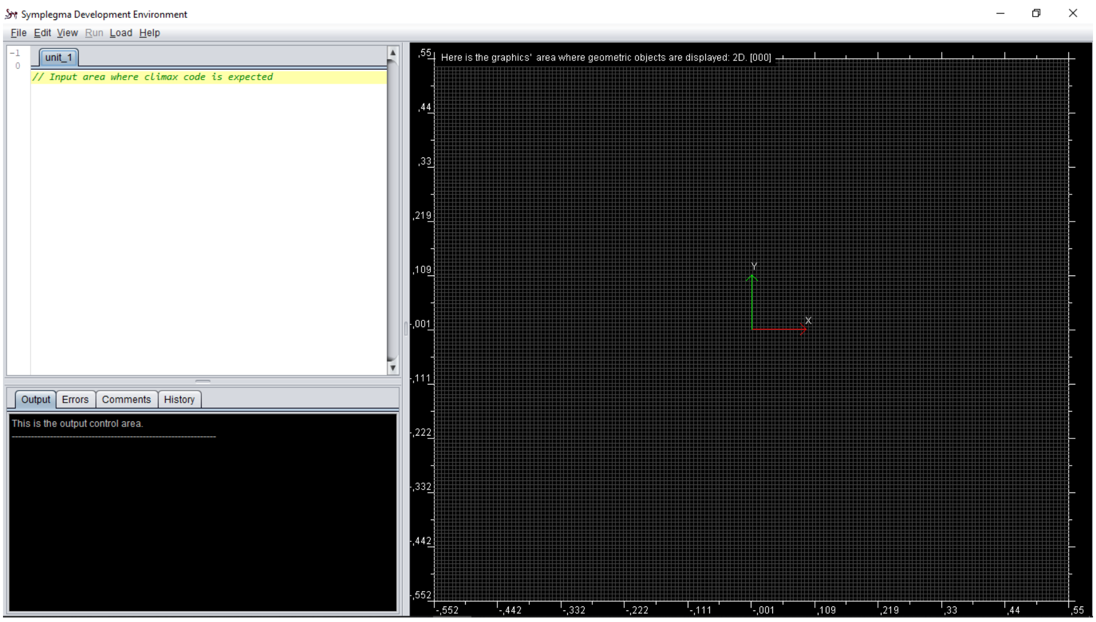
3.4 Γλώσσα προγραμματισμού Groovy

Η γλώσσα προγραμματισμού ανοιχτού κώδικα Groovy[3] αποτελεί μια παραλλαγή της Java και αναμφίβολα είναι αναγνωρισμένη από την πλειοψηφία των προγραμματιστών και ειδικότερα αυτοί που έχουν ένα υπόβαθρο στη Java. Είναι μια δυναμική, αντικειμενοστραφής γλώσσα προγραμματισμού και μοιάζει ως προς το συντακτικό της με την Python. Η αλληλεπίδραση της με τα πακέτα της Java είναι άμεση και «τρέχει» μέσω της εικονικής μηχανής της (Java Virtual Machine). Επίσης, η Groovy μπορεί να χρησιμοποιηθεί σαν γλώσσα σεναρίων scripting. Μερικά χαρακτηριστικά που κάνουν τις δύο γλώσσες να διαφέρουν είναι η χρήση της δήλωσης import στην Groovy, όπως επίσης η χρήση μιας μεθόδου (π.χ. η δήλωση μιας μεταβλητής) γίνεται κατά το χρόνο εκτέλεσης, ενώ στην Java γίνεται την ώρα της μεταγλώττισης. Επιπλέον, στην Groovy η αρχικοποίηση μιας διάταξης/πίνακα (array) γίνεται με τη χρήση αγκυλών ([...]) και όχι με τη χρήση αγκίστρων ({...}) και η συμπεριφορά του «==» στην Java σημαίνει ισότητα ή ότι δύο αντικείμενα ταυτίζονται, ενώ στην Groovy χρησιμοποιείται για τη σύγκριση δύο αντικειμένων αν είναι ίσα. Τέλος, οι κώδικες των περιπτώσεων της παρούσας εργασίας έχουν δημιουργηθεί στη γλώσσα προγραμματισμού Groovy και μπορεί κανείς να τους μελετήσει είτε στο *Παράρτημα Α΄* είτε στην ενότητα που ακολουθεί.

[2]Πηγή: https://www.java.com/en/
[3]Πηγή: http://groovy-lang.org/

3.5 Symplegma Development Environment (SDE)

Το Symplegma Development Environment (SDE)[5] και τα πακέτα Climax[4] είναι τα βασικά εργαλεία ανάπτυξης όλων των προγραμμάτων που βοήθησαν στην παρούσα πτυχιακή εργασία. Το περιβάλλον εργασίας SDE είναι αποκλειστικά υλοποιημένο σε Java αξιοποιώντας ταυτόχρονα την γλώσσα προγραμματισμού Groovy. Επιπλέον, αξίζει να σημειωθεί πως κάθε σχήμα που σχολιάζεται σε επόμενα κεφάλαια του τεύχους, όπως επίσης και όλες οι αναλύσεις των προβλημάτων είναι φτιαγμένα και επιλυμένα από τα προγράμματα και τις βιβλιοθήκες του SDE.
Στη συνέχεια, απεικονίζεται το Symplegma Development Environment (βλ. *Σχήμα 3.2*). Τα τρία παράθυρα που μπορούμε να δούμε είναι: το παράθυρο εντολών (πάνω αριστερά), το παράθυρο εκτύπωσης αποτελεσμάτων και σφαλμάτων (κάτω αριστερά) και το παράθυρο απεικόνισης (GUI) (δεξιά).



Σχήμα 3.2: Symplegma Development Environment (SDE): Περιβάλλον ανάπτυξης κώδικα και απεικόνισης αποτελεσμάτων.

Το ευθύ πρόβλημα για το προσπίπτον κύμα - Ανάλυση *Κώδικα A.1*

Όπως ανεφέρθηκε και σε προηγούμενο κεφάλαιο, η πρώτη περίπτωση μελέτης της συγκεκριμένης πτυχιακής είναι η μονόπακτη δοκός. Δηλαδή, το αριστερό άκρο της είναι δεσμευμένο κατά τους βαθμούς ελευθερίας αξονικό, εγκάρσιο και περιστροφικό σύμφωνα με τις συνοριακές συνθήκες Dirichlet, ενώ το δεξί άκρο παραμένει ελεύθερο καθόλη την διάρκεια του φαινομένου. Υπολογιστικά, λοιπόν, στο πρώτο στάδιο του κώδικα ορίζεται μια απαραίτητη βιβλιοθήκη για την επίλυση του προβλήματος η οποία αφορά τα πεπερασμένα στοιχεία. Η εισαγωγή της στο περιβάλλον SDE γίνεται με την εντολή `import` όπως φαίνεται και παρακάτω:

import jfem.* (*Κώδικας A.1, γραμμή 9*)

[4]Αμφότερα τα πακέτα Climax και SDE είναι ελεύθερα και ανοιχτού κώδικα λογισμικά τα οποία έχουν κυρίως υλοποιηθεί από τον Δρ. Χρήστο Γ. Παναγιωτόπουλο.

Επιπλέον, ένα ακόμη πρωταρχικό στοιχείο του κώδικα επίλυσης των προβλήματος είναι η δημιουργία ενός υπολογιστικού χωρίου (εδω αναφέρεται ως aDomain[5]) πεπερασμένων στοιχείων. Σε μορφή κώδικα εισάγεται ως εξής:

aDomain=theUniverse.FEMDomain() (*Κώδικας A.1, γραμμή 11*)

Το επόμενο κομμάτι του *Κώδικα A.1* αφορά τον καθορισμό της γεωμετρίας και των μηχανικών ιδιοτήτων της δοκού όπως επίσης και τον ορισμό άλλων παραμέτρων. Δηλαδή, η μεταβλητή L (*γραμμή 14*) δείχνει το μήκος της δοκού και η μεταβλητή Nx (*γραμμή 15*) δηλώνει το συνολικό πλήθος των κόμβων των πεπερασμένων στοιχείων. Από την *γραμμή 18* μέχρι την *γραμμή 20* ορίζεται το μέτρο ελαστικότητας E, η πυκνότητα ρ και ο λόγος Poisson ν της δοκού. Οπότε, στο στάδιο αυτό μπορούν εύκολα να υπολογιστούν η ταχύτητα διάδοσης των ελαστικών κυμάτων κατά την αξονική διεύθυνση και η απόσταση που πρέπει να διανύσει το κύμα από τον αρχικό κόμβο ενός πεπερασμένου στοιχείου μέχρι τον τελικό. Ο μαθηματικός τύπος διάδοσης του αξονικού κύματος είναι:

$$c_1=\sqrt{\frac{E}{\rho}} \tag{3.4}$$

και μπορεί κανείς να τον εντοπίσει στην *γραμμή 21*. Η απόσταση των πεπερασμένων στοιχείων dx (*γραμμή 22*) υπολογίζεται από το πηλίκο της διαίρεσης του συνολικού μήκους L της δοκού με τον αριθμό των κόμβων των πεπερασμένων στοιχείων Nx μειωμένο κατά 1, δηλαδή πρακτικά ο παρονομαστής ισούται με τα πεπερασμένα στοιχεία.

Επόμενο βήμα στον προγραμματισμό του προβλήματος είναι ο καθορισμός της συνολικής διάρκειας εξέλιξης του φαινομένου. Αρχικά, ορίζεται η μεταβλητή *init* δίνοντας έτσι την αρχική χρονική συνθήκη του προβλήματος. Ο συνολικός χρόνος υπολογίζεται από το άθροισμα που προκύπτει από τον αρχικό χρόνο και από τον διπλασιασμό του μήκους L δια την ταχύτητα διάδοσης του αξονικού ελαστικού κύματος. Επιπλέον, θα πρέπει να δοθεί ο αριθμός των χρονικών διακριτών βημάτων του φαινομένου για να μπορέσουμε να υπολογίσουμε στη συνέχεια το χρονικό βήμα dt (βλ. *γραμμές 25 έως 28*). Εδώ πρέπει να σημειωθεί ότι το χρονικό βήμα dt που υπολογίσαμε παραπάνω δεν μπορεί να είναι μεγαλύτερο από το χρονικό βήμα που προκύπτει διαιρώντας τη μεταβλητή dx με την ταχύτητα διάδοσης του αξονικού κύματος. Δηλαδή, το κύμα με την μεγαλύτερη ταχύτητα, που στην προκειμένη περίπτωση είναι κατά την αξονική διεύθυνση, θα πρέπει να μεταβαίνει από ένα πεπερασμένο στοιχείο στο άλλο σε χρόνο μεγαλύτερο του χρονικού βήματος dt που έχει οριστεί από τον συνολικό χρόνο του φαινομένου και τα διακριτά χρονικά βήματα. Για τον λόγο αυτό υπολογίζεται η μεταβλητή με όνομα dtd (*γραμμή 31*) και ο προαπαιτούμενος έλεγχος ευστάθειας γίνεται στην ακριβώς επόμενη γραμμή (*γραμμή 32*).

Συνεχίζοντας την ανάλυση του κώδικα για το ευθύ πρόβλημα, καθορίζεται η ακριβής τοποθεσία της δράσης του παλμού Ricker pulse[6] και κατά ποιόν ή ποιούς βαθμούς ελευθερίας θα εφαρμοστεί. Στις *γραμμές 39 έως 44* βλέπουμε ότι η δύναμη εφαρμόζεται στο 0.75 του συνολικού μήκους L της δοκού (=22.5 μονάδες απόστασης) κατά τον αξονικό (βλ. *γραμμή 43*) και τον εγκάρσιο (βλ. *γραμμή 44*) ΒΕ.

[5]Αποτελεί ένα αντικείμενο στο οποίο ενσωματώνονται όλα τα δεδομένα που εισάγονται με τη μορφή κώδικα. Επιπλέον, κάθε ανάλυση ή επίλυση γίνεται στο συγκεκριμένο τομέα/χωρίο (domain).

[6]Πηγή: http://subsurfwiki.org/wiki/Ricker_wavelet

Σημαντικό κομμάτι της ανάλυσης είναι ο αριθμός των αισθητήρων/διεγερτών (receivers) και η θέση του καθενός στο χώρο. Οι αισθητήρες αυτοί έχουν την δυνατότητα να καταγράφουν τιμές για τις μετατοπίσεις και τις ταχύτητες των κόμβων των πεπερασμένων στοιχείων για κάθε χρονικό διακριτό βήμα και για όλους τους βαθμούς ελευθερίας του συστήματος. Στις *γραμμές 47 έως 51* είναι ευδιάκριτο ότι χρησιμοποιήθηκαν 3 αισθητήρες/διεγέρτες και με τη χρήση της επαναληπτικής διαδικασίας `each` εκχωρούνται στον πίνακα με όνομα `receivers` οι συντεταγμένες κάθε αισθητήρα/διεγέρτη κατά την οριζόντια διεύθυνση.

Όσον αφορά τα μηχανικά χαρακτηριστικά της δοκού θα πρέπει να δώσουμε το εμβαδόν διατομής A και το θεωρητικό εμβαδόν As με σκοπό τον υπολογισμό της ροπής αδράνειας, η οποία ισούται με $I{=}\frac{bh^3}{12}$. Στο περιβάλλον SDE μπορεί κανείς να ορίσει κάποιο στοιχείο διατομής με την ακόλουθη εντολή:

aSect=new CrossSection(1, A, As, Imi) (*Κώδικας A.1, γραμμή 61*)

Ανατρέχοντας στα αρχικά δεδομένα του προβλήματος, η δοκός επιλέχθηκε να χωριστεί σε $Nx = 1950$ κόμβους πεπερασμένων στοιχείων. Όλα τα στοιχεία έχουν το ίδιο υλικό, την ίδια διατομή και το μήκος τους είναι ίσο με dx. Η αναπαράσταση τους στο χώρο κατά τον άξονα x γίνεται με τις εντολές μεταξύ των *γραμμών 91 έως 95.* Για κάθε πεπερασμένο στοιχείο δημιουργούμε έναν κόμβο με την εντολή:

aNode=new Node(++id, it*L/Nx) (*Κώδικας A.1, γραμμή 93*)

Το πρώτο όρισμα της κλάσης `new Node (++id)` δηλώνει των αύξων αριθμό του κόμβου και αρχικά του εκχωρείται η τιμή 0, ενώ το δεύτερο όρισμα (`it*L/Nx`) αποτελεί μια αριθμητική πράξη για τον υπολογισμό της συντεταγμένης του κάθε κόμβου κατά μήκος της δοκού. Κάθε κόμβος εντάσσεται στο υπολογιστικό χωρίο (εδώ `aDomain`) των πεπερασμένων στοιχείων (βλ. *γραμμή 94*). Στη συνέχεια, δημιουργούνται επαναληπτικά τα στοιχεία της δοκού χρησιμοποιώντας την κλάση `new EBeam2d`. Τα πέντε ορίσματα της κλάσης είναι το στοιχείο `it` της επαναληπτικής μεθόδου που βοηθάει στην ανίχνευση των κόμβων εντός του χωρίου, ο κόμβος με `id` ίσο με `it`, ο κατά αυξουσα σειρά επόμενος του (`it+1`), το υλικό και η διατομή. Τα παραπάνω γίνονται αντιληπτά στις *γραμμές 101 έως 104.*

Στο σημείο αυτό θα πρέπει να εντοπίσουμε τις ακριβείς συντεταγμένες της εφαρμοζόμενης δύναμης και να την ορίσουμε συναρτήσει των χρονικών διακριτών βημάτων. Ανατρέχοντας στο πρόγραμμα και συγκεκριμένα στις *γραμμές 112 έως 142* παρατηρούμε αρχικά την εκχώριση της κλάσης `new LoadCase()` με πρώτο ορίσμα τη μεταβλητή που μας δείχνει τον αριθμό των δυνάμεων στο σύστημα (`id`) και ως δεύτερο τα χρονικά διακριτά βήματα (`Nt`). Πριν συνεχίσουμε σε παραιτέρω ανάλυση του κώδικα θα πρέπει να γίνει μια συνοπτική αναφορά στο μοντέλο του σεισμικού κύματος Ricker wavelet. Η συνάρτηση $P(x,t)$ δίνεται από τον ακόλουθο τύπο:

$$P(x,t){=}\alpha\delta\left(x-x_s\right)\left(1-2\pi^2 s^2\left(t-t_0\right)^2\right)e^{-\pi^2 s^2 (t-t_0)^2} \tag{3.5}$$

Στις *γραμμές 115 και 116* δίνονται οι τιμές του συντελεστή s (στον *Κώδικα A.1* δηλώνεται ως $sigma$) και του εύρους a (στον *Κώδικα A.1* δηλώνεται ως $amplit$) αντίστοιχα. Η εύρεση των συντεταγμένων της τοποθεσίας της δύναμης έχει νόημα γιατί παρακάτω θα χρησιμοποιηθούν ως όρισμα της κλάσης `new Load()`. Άρα, πληκτρολογώντας την εντολή εδώ `aDomain.find(coords)`, ζητάμε από το υπολογιστικό χωρίο να εντοπίσει τον κόμβο που αντιστοιχεί στις συντεταγμένες `coords` και εν συνεχεία εκχωρείται στην μεταβλητή `nid` (βλ. *γραμμές 117 έως 119*). Το τελευταίο στάδιο της διαδικασίας αυτής είναι ο καθορισμός της τιμής της συνάρτησης P σε κάθε χρονικό διακριτό βήμα και σε κάθε ΒΕ. Δηλαδή, δημιουργούμε την κλαση `new Load` με ορίσματα την τιμή της συνάρτησης σε κάποιο χρονικό διακριτό βήμα, τον κόμβο εφαρμογής, τον ΒΕ και το ίδιο το χρονικό διακριτό βήμα και την εισάγουμε στην κλαση `LoadCase()`. Με το πέρας της επαναληπτικής διαδικασίας, απαραίτητη ενέργεια αποτελεί η ένταξη της `LoadCase()` στο υπολογιστικό χωρίο (εδώ `aDomain`). Η εντολή που χρησιμοποιήθηκε είναι:

aDomain.putLoadCase(LC) (*Κώδικας A.1, γραμμή 137*)

Το επόμενο στάδιο του προγράμματος αποτελείται από τον προσδιορισμό των συνοριακών συνθηκών. Η κλάση `new ConstraintElement` έχει προγραμματιστεί στο περιβάλλον SDE βάσει της μεθόδου ποινής (Penalty method), για την οποία μιλήσαμε στην (*Ενότητα 2.4*). Συνοπτικά, μπορούμε να πούμε ότι με τη μέθοδο αυτή αλλοιώνουμε κάποιο στοιχείο της δοκού, θέτοντας κάποιο συντελεστή ποινής. Πιο συγκεκριμένα, στην *γραμμή 146* ζητάμε από το υπολογιστικό χωρίο (εδώ `aDomain`) να πάρει το μέγιστο στοιχείο από τα μητρώα δυσκαμψίας και μάζας και να το εκχωρήσει στον πίνακα (array) `mc`. Εν συνεχεία, ορίζουμε κατά βούληση μια αρκετά μεγάλη τιμή, όπως για παράδειγμα $10.0e^{20}$, με σκοπό να υπολογίσουμε την μεταβλητή `Am`. Στις *γραμμές 150, 153 και 156* δηλώνονται τα στοιχεία που θα τεθεί ο περιορισμός με την χρήση της κλάσης `new ConstraintElement`. Στο συγκεκριμένο πρόβλημα βλέπουμε πως ο κόμβος 1 δεσμεύεται κατά τον αξονικό, εγκάρσιο και περιστροφικό ΒΕ, ενώ ταυτόχρονα εισάγονται (εντολή `set()`) και οι παραπάνω τιμές, `Am` και `Ak` (βλ. *γραμμές 151, 154 και 157*). Τέλος, οι παραπάνω περιορισμοί (βλ. *γραμμές 152, 155 και 158*) εισάγονται στο υπολογιστικό χωρίο με την εντολή `aDomain.putConstraintElement(aCE)`.

Το συγκεκριμένο πρόβλημα της δοκού υπόκειται σε δυναμική φόρτιση και για την επίλυση χρησιμοποιήσουμε τον αλγόριθμο β-Newmark[7] με μεταβλητές ορίσματος το υπολογιστικό χωρίο, την αριθμητική τιμή του χρονικού βήματος και τις αριθμητικές τιμές της μεθόδου β-Newmark (βλ. *Υποενότητα 3.2*). Σε μορφή κώδικα θα είναι:

theAnalysis = new BetaNewmarkAnalysis() (*Κώδικας A.1, γραμμή 160*)

Τέλος, στην *γραμμή 162* ζητάμε να πραγματοποιηθεί η ανάλυση και στην ακριβώς επόμενη την ολοκληρώνουμε.

Στο τελευταίο στάδιο του προγράμματος εμφανίζεται μια επαναληπτική διαδικασία, η οποία βοηθάει στην άμεση αποθήκευση των αποτελεσμάτων. Συγκεκριμένα για κάθε διεγέρτη/αισθητήρα και για κάθε βαθμό ελευθερίας αποθηκεύονται οι αριθμητικές τιμές των μετατοπίσεων και των ταχυτήτων σε κάθε χρονικό διακριτό βήμα. Στον Κώδικα A.1 και στις *γραμμές 171 έως 208* εμφανίζεται η παραπάνω διαδικασία.

Το αντίστροφο πρόβλημα για το προσπίπτον κύμα - Ανάλυση *Κώδικα A.2*

Το δεύτερο στάδιο της επίλυσης των δύο περιπτώσεων της συγκεκριμένης εργασίας είναι η χρονικά αντίστροφη διαδικασία επίλυσης με στόχο τον εστιασμό του σήματος είτε στην

[7]Πηγή: https://en.wikipedia.org/wiki/Newmark-beta_method

πηγή διέγερσης είτε σε κάποια αλλοιωμένη περιοχή. Για το λόγο αυτό δημιουργήθηκε ο *Κώδικας A.2*, ο οποίος διαφέρει σε δύο σημεία από τον κώδικα επίλυσης του ευθέος προβλήματος (*Κώδικας A.1*). Το πρώτο είναι ο τρόπος που ορίζεται η δύναμη διέγερσης. Να υπενθυμίσουμε εδώ ότι στην αντίστροφη διαδικασία ως πηγή ορίζεται η τοποθεσία και η διεύθυνση καταγραφής κάποιου αισθητήρα με αποτέλεσμα ο αισθητήρας έκτοτε να δρα σαν διεγέρτης, ενώ το δεύτερο στάδιο είναι ο καθορισμός των κόμβων που θα αποθηκευτούν οι αποκρίσεις του αντίστροφου βήματος.

Πιο συγκεκριμένα, στις *γραμμές 19* και *21* του *Κώδικα A.2* καθορίζονται κατ΄ αντιστοιχία το id του αισθητήρα και ο βαθμός ελευθερίας που θα χρησιμοποιηθεί στη συγκεκριμένη επίλυση. Στον κώδικα, για παράδειγμα, φαίνεται ότι θα χρησιμοποιηθούν οι αποκρίσεις του ευθέος προβλήματος όπως τις κατέγραψε ο αισθητήρας αξονικής διεύθυνσης καταγραφής στον κόμβο με id ίσο με 1950. Επιπλέον, η χρονική αντιστροφή των τιμών και η υπολογιστική εφαρμογή της δύναμης γίνεται μεταξύ των *γραμμών 126* και *150* όπου ο κώδικας διαβάζει σε κάποιο τοπικό φάκελο του Η/Υ τα αποτελέσματα της ευθείας διαδικασίας, σύμφωνα με τις παραπάνω παραμέτρους, τις αντιστρέφει (εντολή: `fback[Nt-i]`, *γραμμή 147*) και έπειτα τις τοποθετεί στην κλάση `new LoadCase()` (*γραμμή 148*) για καθένα χρονικό διακριτό βήμα.

Η δεύτερη διαφοροποίηση αφορά τα σημεία αποθήκευσης των αποτελεσμάτων. Στις *γραμμές 58* έως *63* φαίνεται το κομμάτι του εν λόγω κώδικα. Η *γραμμή 59* δηλώνει ότι το πλήθος των imagers ισούται με τον συνολικό αριθμό των κόμβων διαιρεμένο δια 5, άρα για κάθε 5 κόμβους θα γίνεται καταγραφή. Τέλος, τα σημεία αποθήκευσης εισέρχονται στον πίνακα imagers για να χρησιμοποιηθούν στο τέλος του κώδικα που γίνεται η διαδικασία αποθήκευσης όλων των αποκρίσεων της ανάλυσης.

Απεικόνιση - Ανάλυση *Κώδικα A.3*

Ο Κώδικας A.3 αποτελεί το προπαρασκευαστικό-βοηθητικό κομμάτι για την απεικόνιση των αποτελεσμάτων που προέκυψαν από την επίλυση του αντίστροφου προβλήματος και συνδυάζεται άμεσα με τον Κώδικα A.4, που θα αναλυθεί στην επόμενη ενότητα. Τελικά, η αλληλεπίδραση των δύο προγραμμάτων δίνει τα επιθυμητά διαγράμματα και βίντεο στο Graphical Panel του SDE.
Αρχικά, στην *γραμμή 1* γίνεται εισαγωγή ενός μηχανισμού contraption, ο οποίος είναι σχεδιασμένος με τη χρήση των βιβλιοθηκών της Climax. Ο μηχανισμός αυτός έχει ενσωματωμένες όλες τις σχετικές μεθόδους που θα χρησιμοποιηθούν στον Κώδικα A.3 χωρίς όμως να έχουν αρχικά κάποια εντολή στο εσωτερικό τους. Ο εν λόγω κώδικας είναι ο ακόλουθος:

Listing 3.1: Contraption

```java
package climax;

import java.awt.Graphics2D;

/**
 *
 * @author pchr
 */

public interface contraption {
    public int getID();
    public void SelfPortrait(Graphics2D g2, double ymin, double ymax, int ye, int ys, double xmin, double xmax, int xe, int xs);
```

```
    public void Motion(Graphics2D g2, double ymin, double ymax, int ye, int
    ys, double xmin, double xmax, int xe, int xs, int step, double scale);
    public double maximum_X_coordinate();
    public double minimum_X_coordinate();
    public double maximum_Y_coordinate();
    public double minimum_Y_coordinate();
}
```

Συνεχίζοντας την ανάλυση, βλέπουμε ότι δημιουργείται η κλάση `ImagingFrame`, στην οποία εντάσσονται όλες οι χρήσιμες συναρτήσεις. Η πρώτη συνάρτηση `SelfPortrait` δεν επιστρέφει αρχικά κάποια τιμή, παραμόνο όταν εκτελεστεί κάποια από τις ενσωματωμένες συναρτήσεις `if`. Στις *γραμμές 39 έως 44* υπάρχει η πρώτη συνάρτηση, η οποία σχεδιάζει την γεωμετρία της κατασκευής κάνοντας σύνδεση των κόμβων. Δηλαδή, αν η δήλωση (def) `connectivity` δεν είναι άδεια, τότε ο κώδικας εντοπίζει ανά δύο τους κόμβους και εισάγει στο αντικείμενο (object) τις συντεταγμένες τους. Η ίδια ακριβώς διαδικασία ακολουθείτε για να εμφανιστούν στο γραφικό περιβάλλον οι θέσεις των αισθητήρων/διεγερτών (receivers) με γραμμές πράσινου χρώματος, οι θέσεις των πηγών διέγερσης (sources) με κόκκινο χρώμα και με γραμμές χρώματος γκρι εμφανίζεται η περιοχή της αλλοίωσης/φθοράς (scatterers). Οι χρωματικές ενδείξεις δίνονται με την εντολή `g2.setColor(Color.red)` (π.χ. για το κόκκινο χρώμα).

Στις *γραμμές 84* έως *147* αναπτύσσεται η συνάρτηση (Motion), η οποία βοηθά στη σχεδίαση των κόμβων σε κατάσταση διέγερσης. Εδώ, υπάρχουν δύο περιπτώσεις απεικόνισης των κόμβων και γι' αυτό το λόγο χρησιμοποιείται η δήλωση `switch`. Η περίπτωση 0 (`case: 0`) δείχνει τους κόμβους σε μορφή κύματος (βλ. σχήματα στο *Κεφάλαιο 4*), ενώ η περίπτωση 1 (`case: 1`) απεικονίζει τους κόμβους σε μια χρωματική διαβάθμιση (βλ. σχήματα στο *Κεφάλαιο 5*). Ειδικότερα, στην `case: 0`, η δήλωση (def) `connectivity` έχει τον αντίστοιχο ρόλο με παραπάνω, μόνο που για να σχεδιαστεί η γραμμή που συνδέει τους προς εξέταση κόμβους, θα πρέπει στις συντεταγμένες τους (σε αδράνεια) να προσθέσουμε τις αποκρίσεις του αντίστροφου προβλήματος υπό μια κλίμακα (scale). Η διαδικασία αυτή γίνεται, επίσης, σε κάθε κόμβο για κάθε χρονικό διακριτό βήμα του φαινομένου. Στην `case: 1`, λόγω της χρωματικής διαβάθμισης, η διαδικασία γίνεται με λίγο διαφορετικό τρόπο. Δηλαδή, υπολογίζεται ο μέσος όρος των αποκρίσεων των δύο κόμβων σύνδεσης σε κάποιο χρονικό βήμα (*γραμμή 132*) με στόχο την χρωματική διαβάθμιση της τιμής. Όσο πιο μεγάλη τιμή έχει η συγκεκριμένη σύνδεση τόσο εντονότερο θα είναι το χρώμα. Κάθε χρώμα εισάγεται στο γραφικό περιβάλλον με την εντολή `g2.setColor(theGP.colorArray[colorIndex])`.

Απεικόνιση - Ανάλυση *Κώδικα A.4*

Ο *Κώδικας A.4* αποτελεί τον δεύτερο κώδικα απεικόνισης των αποτελεσμάτων στον οποίο ο χρήστης μπορεί να καθορίσει ποια από τις δύο περιπτώσεις θα μελετήσει, ποιον ή ποιους αισθητήρες/διεγέρτες θα χρησιμοποιήσει, την μεταβλητή απεικόνισης που επιθυμεί, όπως επίσης και την προσθήκη θορύβου ή όχι. Ουσιαστικά ο χρήστης θα πρέπει να παρέμβει στον κώδικα ώστε να τροποποιήσει κατά βούληση τις εντολές μεταξύ των *γραμμών 1* και *93*. Αρχικά, δημιουργείται ένα νέο `ImagingFrame` (*γραμμή 2*), το οποίο αλληλεπιδρά άμεσα με τον *Κώδικα A.3*. Λίγο παρακάτω, ο χρήστης θα πρέπει να ορίσει το ακριβές path στο οποίο έχει αποθηκεύσει τις αποκρίσεις της μεθόδου χρονικής αντιστροφής. Κατά κάποιο τρόπο με τον παραπάνω καθορισμό, ο χρήστης επιλέγει και ποιο πρόβλημα θα μελετήσει, είτε αυτό του εντοπισμού της θέσης της δύναμης είτε αυτό της ανίχνευσης της αλλοιωμένης περιοχής.

Επιπρόσθετα, η *γραμμή 19* αποτελεί βασικό κομμάτι του κώδικα, γιατί ο χρήστης θα πρέπει να ορίσει από ποιον ή ποιους αισθητήρες/διεγέρτες θα αντλήσει τις αποκρίσεις για την επιθυμούμενη απεικόνιση. Για αποκρίσεις που προήλθαν από αισθητήρες/διεγέρτες αξονικής διεύθυνσης καταγραφής (στην 1Δ περίπτωση) ή οριζόντιας διεύθυνσης καταγραφής (στην 2Δ περίπτωση) χρησιμοποιείται το γράμμα «x», ενώ για εγκάρσια διεύθυνση καταγραφής (στην 1Δ περίπτωση) ή κάθετη διεύθυνσης καταγραφής (στην 2Δ περίπτωση) γίνεται χρήση του γράμματος «y». Επιπλέον, επειδή ένας αισθητήρας/διεγέρτης δεν καθορίζεται μόνο από τη διεύθυνση καταγραφής αλλά και από την θέση που είναι τοποθετημένος, θα πρέπει μετά από κάθε γράμμα να συμπληρωθεί και το id του κόμβου που βρίσκεται ο αισθητήρας/διεγέρτης. Ο διαχωρισμός των διευθύνσεων με την θέση γίνεται με τη χρήση της κάτω παύλας (_). Για παράδειγμα, στη *γραμμή 19*, ο χρήστης έχει επιλέξει πέντε αισθητήρες/διεγέρτες, δύο οριζόντιας διεύθυνσης (x_2, x_3) καταγραφής και τρεις κάθετης διεύθυνσης καταγραφής (y_6, y_7, y_8).

Στη συνέχεια, ο χρήστης καλείται να επιλέξει την μεταβλητή που θα χρησιμοποιήσει κατά την απεικόνιση. Στις *γραμμές 35* έως *43*, φαίνονται οι δυνατές επιλογές, οι οποίες είναι όμοιες είτε για τις μετατοπίσεις είτε για τις ταχύτητες. Στην παρούσα εργασία, η μεταβλητή που χρησιποιείται στα σχήματα είναι η Ευκλείδια νόρμα. Επίσης, η συνθήκη ακύρωσης του ενός εκ των δύο σεναρίων είναι το -1, το οποίο εκχωρείται σε όποια μεταβλητή δεν θέλουμε να χρησιμοποιήσουμε. Τέλος, στη *γραμμή 93* υπάρχει η λογική μεταβλητή «noise» με σκοπό την προσθήκη θορύβου ή όχι στις αποκρίσεις. Αν η μεταβλητή είναι αληθής τότε ο κώδικας θα εντοπίσει την μέγιστη τιμή (`MaxVal`) απ΄ όλες τις αποκρίσεις που έχουν καταγραφεί στο αντίστροφο πρόβλημα -είτε για τις μετατοπίσεις είτε για τις ταχύτητες-, με αποτέλεσμα να υπολογιστεί η τιμή του συντελεστή `coefn` (*γραμμή 223*). Η τιμή αυτή ορίζεται ως το 10% της μέγιστης τιμής `MaxVal`.

Εφόσον πραγματοποιηθούν όλα τα παραπάνω από τον χρήστη, ο κώδικας μέσω της δήλωσης `switch` εκτελεί την αντίστοιχη διαδικασία. Δηλαδή, ο διαχωρισμός γίνεται αναλόγως ποια θα είναι η επιθυμητή μεταβλητή απεικόνισης και τελικά κάνει προσθήκη του πίνακα (array) με όνομα `val2video` στην κλάση `ImagingFrame`. Το εν λόγω κομμάτι κώδικα είναι το μεγαλύτερο σε έκταση και μπορεί κανείς να ανατρέξει για αναλυτικές πληροφορίες μεταξύ των *γραμμών 230* και *741*.

Το τελευταίο κομμάτι κώδικα αφορά τις ρυθμίσεις το γραφικού περιβάλλοντος (Graphical Panel) του SDE. Εκεί ο χρήστης μπορεί να καθορίσει τις διαστάσεις των αξόνων x (`theGP.setMargin_x()`) και y (`theGP.setMargin_y()`), τον τίτλο του γραφικού περιβάλλοντος (`theGP.title`), το μέγεθος (`theGP.FontSize`) και το πλήθος των δεκαδικών ψηφίων στους άξονες. Για τον x είναι `theGP.gridnumfx=new DecimalFormat("")`, ενώ για τον y `theGP.gridnumfy=new DecimalFormat("")`. Ακόμα, ο χρήστης θα πρέπει να επιλέξει ποια διακριτή χρονική στιγμή θέλει να προσομοιώσει την κατασκευή. Αυτό επιτυγχάνεται εκχωρώντας την επιθυμητή τιμή στην μεταβλητή `ex_time` και μαζί με κάποια τιμή κλίμακας (`scale`) γίνονται τα ορίσματα της εντολής απεικόνισης `theGP.plotDeform()`.

Κεφάλαιο 4

Πρόβλημα διάδοσης αξονικών και εγκάρσιων κυμάτων σε μονοδιάστατη δοκό

Το παρόν κεφάλαιο αποτελεί ένα από τα πιο σημαντικά κεφάλαια της παρούσας εργασίας διότι αναπαριστούνται και αναλύονται εκτενώς τα αποτελέσματα όλων των περιπτώσεων που μελετήθηκαν στη μονόπακτη δοκό. Αρχικά, η *Ενότητα 4.1* αναφέρεται στον εντοπισμό της πηγής διέγερσης χωρίς καμία επιρροή στην ανάλυση, ενώ στην *Υποενότητα 4.1.2* παρουσιάζονται τα αποτελέσματα του παραπάνω προβλήματος με την προσθήκη τυχαίων τιμών Γκαουσιανού θορύβου. Σημαντικό κομμάτι της παρούσας εργασίας αποτελεί, επίσης, και ο εντοπισμός κάποιας αστοχίας, διαρροής υλικού ή γενικότερα μιας αλλοίωσης της κατασκευής. Στην *Ενότητα 4.2* αναλύονται τα σχετικά αποτελέσματα που προέκυψαν από την επίλυση του χρονικά αντίστροφου προβλήματος. Τέλος, πραγματοποιήθηκε και μια μελέτη για τη συμπεριφορά του λόγου σήμα-προς-θόρυβο (Signal to Noise Ratio), με στόχο την εύρεση της βέλτιστης διάταξης αισθητήρων που μπορούν να τοποθετηθούν στην κατασκευή.

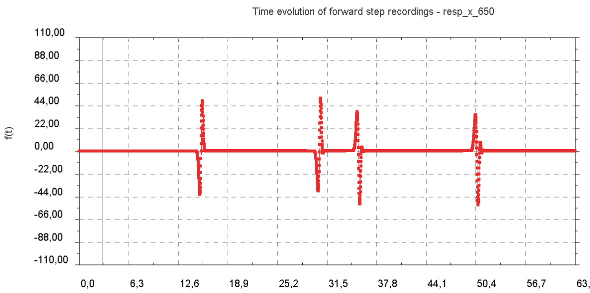
4.1 Εντοπισμός πηγής διέγερσης (Source Localization)

Στην *Ενότητα 4.1* απεικονίζονται τα διαγράμματα εντοπισμού της ακριβής θέσης της διέγερσης του παλμού (Ricker's pulse) για καθένα από τους αισθητήρες/διεγέρτες. Η χωροθέτηση των αισθητήρων/διεγερτών έγινε σε τρία ισαπέχοντα σημεία. Οπότε ο πρώτος βρίσκεται στο 1/3 του L, ο δεύτερος και πιο κοντινός στην πηγή διέγερσης στα 2/3 του L και ο τρίτος τοποθετήθηκε στο ελεύθερο άκρο της δοκού (3/3 του L).

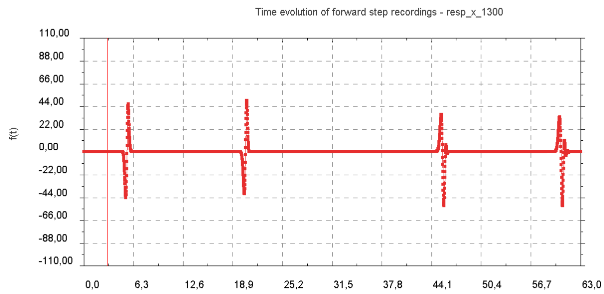
Η διαδικασία εντοπισμού της πηγής διέγερσης είναι αποτέλεσμα δύο αναλύσεων. Στην πρώτη ανάλυση επιλύθηκε το ευθύ βήμα (Forward Step), η οποία αποτελεί μια αμιγώς υπολογιστική διαδικασία και αναλύθηκε στην *Ενότητα 3.5*. Συνοπτικά, ένα συγκεκριμένο σημείο της δοκού διεγείρεται κατά την αξονική και την εγκάρσια διεύθυνση, ενώ με χρήση τριών αισθητήρων x_a(receivers) καταγράφονται τα αποτελέσματα για κάθε ΒΕ της δοκού. Ειδικότερα, αποθηκεύτηκαν οι αποκρίσεις για τις μετατοπίσεις και τις ταχύτητες της αξονικής, της εγκάρσιας και της περιστροφικής κίνησης σε κάθε χρονικό διακριτό βήμα. Στο αντίστροφο πρόβλημα (Backward Step) στέλνονται οι χρονικά αντεστραμμένες αποκρίσεις κάθε ΒΕ ξεχωριστά από κάθε τοποθεσία των αισθητήρων, σημειώνοντας εδώ ότι στο αντίστροφο πρόβλημα οι αισθητήρες λειτουργούν ως διεγέρτες. Τα γεωμετρικά, μηχανικά και δομικά χαρακτηριστικά της δοκού παρουσιάστηκαν με ακρίβεια στο *Κεφάλαιο 2*.

4.1.1 Αποτελέσματα επίλυσης του ευθέος προβλήματος (Forward step)

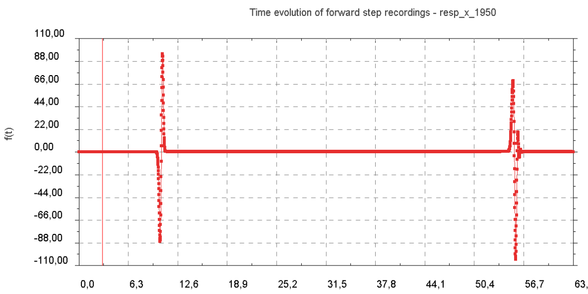
Στην παρούσα υποενότητα απεικονίζονται μερικά αποτελέσματα του ευθέος προβλήματος όπως προέκυψαν με την δυναμικη στο χρονο αναλυση με χρήση του αλγορίθμου β-Newmark. Στα *Σχήματα 4.1* βλέπουμε τις καταγραφές (μετατοπίσεις και ταχύτητες) των αισθητήρων x_a στην πάροδο του χρόνου. Αρχικά, μπορούμε να εντοπίσουμε τη χρονική στιγμή που δρα ο παλμός (t_0=3s) βλέποντας σε οποιοδήποτε διάγραμμα την συνεχή κόκκινη γραμμή. Επόμενη παρατήρηση είναι ότι ο αισθητήρας με την κοντινότερη απόσταση από το σημείο που δρα η δύναμη είναι ο x_a=1300. Αυτό γίνεται αντιληπτό βλέποντας είτε το *Σχήμα 4.1β′* με τις μετακινήσεις είτε το *Σχήμα 4.1ϵ′* με τις ταχύτητες, στα οποία ο παλμός φτάνει περίπου σε χρόνο t=6s, ενώ στους άλλους δύο αισθητήρες ξεπερνά τα 10s. Επίσης, είναι χρήσιμο να σημειωθεί ότι το κύμα δεν περνά μόνο μια φορά από τους αισθητήρες και πιο συγκεκριμένα από τους x_a=650 και x_a=1300 περνά 4 φορές, ενώ από τον αισθητήρα στο ελεύθερο άκρο της δοκού (x_a=1950) περνά 2 φορές.



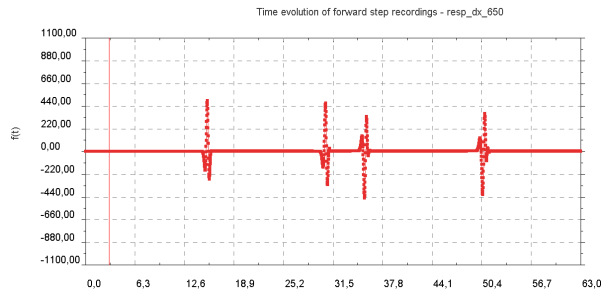
(α′) Αισθητήρας x_a=650



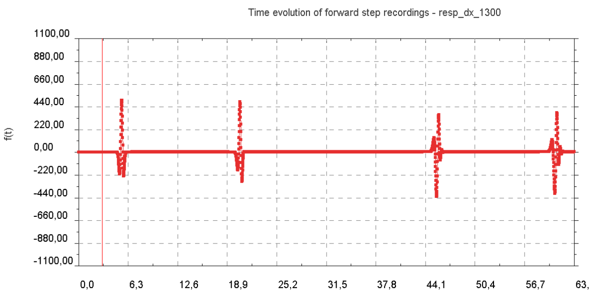
(β′) Αισθητήρας x_a=1300



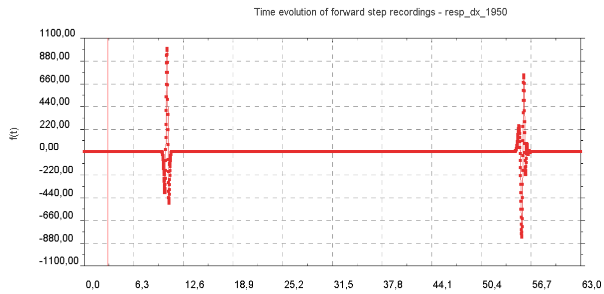
(γ′) Αισθητήρας x_a=1950



(δ′) Αισθητήρας x_a=650



(ε′) Αισθητήρας x_a=1300



(ϛ′) Αισθητήρας x_a=1950

Σχήμα 4.1: Απεικόνιση των αξονικών μετατοπίσεων (α′, β′, γ′) και ταχυτήτων (δ′, ε′, στ′) συναρτήσει του χρόνου όπως καταγράφηκαν από τους αισθητήρες με id ίσο με 650, 1300, 1950. Η συνεχής κόκκινη γραμμή συμβολίζει τη χρονική στιγμή που δρα ο παλμός Ricker.

Στα *Σχήματα 4.2* γίνεται αναπαράσταση της διάδοσης του κύματος κατά την επίλυση του ευθέος βήματος. Όπως μπορούμε να δούμε, στο *Σχήμα 4.2α΄* έχει ξεκινήσει η δράση της δύναμης, ενώ στο *Σχήμα 4.2β΄* και τη χρονική στιγμή t_0=3 ο παλμός φτάνει στη μέγιστη τιμή και στη συνέχεια ξεκινά η διάδοση του προς όλες τις κατευθύνσεις. Επιπλέον, παρατηρούμε ότι τη στιγμή t_0=15.38 (*Σχήμα 4.2ϵ΄*) που το κύμα φτάνει στο ελεύθερο άκρο της δοκού, η πορεία του κύματος αντιστρέφεται και επιστρέφει πίσω χωρίς καμία αλλαγή. Αντίθετα, μόλις το κύμα φτάσει στο δεσμευμένο άκρο της δοκού (*Σχήμα 4.2ζ΄*) ανακλάται ανεστραμμένο προς την αντίθετη κατεύθυνση (*Σχήμα 4.2η΄*) και τελικά στο *Σχήμα 4.2ι΄* βλέπουμε την κατάσταση των κυμάτων μετά το πέρας το φαινομένου.

4.1.2 Αποτελέσματα επίλυσης του αντίστροφου προβλήματος (Backward step)

Στην *Υποενότητα 4.1.2* αναπαριστούνται τα αποτελέσματα της επίλυσης του χρονικά αντίστροφου προβλήματος, ενώ στην *Υποενότητα 4.1.2* σχολιάζεται και απεικονίζεται η επιρροή των τιμών Γκαουσιανού θορύβου στα αποτελέσματα κατά τον εντοπισμό της τοποθεσίας της δύναμης διέγερσης.

Τα *Σχήματα 4.3* δείχνουν σε διάφορες χρονικές στιγμές τη διάδοση των κυμάτων που δημιουργήθηκαν εξαιτίας της επίδρασης των χρονικά αντεστραμμένων τιμών από την τοποθεσία του διεγέρτη με id ίσο με 1950. Αρχικά, στο *Σχήμα 4.3α΄* βλέπουμε το ξεκίνημα της διέγερσης και τη χρονική στιγμή t=7.57s (βλ. *Σχήμα 4.3β΄*), ο παλμός παίρνει την μέγιστη τιμή του και ξεκινά να διαδίδεται κατά μήκος της δοκού. Στη συνέχεια, παρατηρούμε ότι λίγο πριν τα 38s έχει διανύσει το συνολικό μήκος της κατασκευής, φτάνοντας στο δεσμευμένο άκρο, όπου αντιστρέφεται (βλ. *Σχήμα 4.3ϵ΄*). Επιπλέον, αξιοσημείωτη παρατήρηση αποτελεί η δημιουργία ενός ακόμα κύματος ίσου με του αρχικού την χρονική στιγμή t=53.08s, το οποίο θα καταλήξει τελικά στο σημείο της πηγής διέγερσης ταυτόχρονα με το αρχικό και μαζί θα δώσουν την μέγιστη τιμή της μετατόπισης του συγκεκριμένου κόμβου. Η παραπάνω παρατήρηση θα σχολιαστεί περισσότερο στην *Ενότητα 4.3* μιας και επηρεάζεται η γραμμικότητα του λόγου σήμα-προς-θόρυβο. Στο *Σχήμα 4.3θ΄* απεικονίζεται με ακρίβεια ο εντοπισμός της πηγής χρησιμοποιώντας τις «καθαρές» τιμές των μετατοπίσεων.

Τα *Σχήματα 4.4α΄, 4.4β΄ και 4.4γ΄* απεικονίζουν τον εντοπισμό της πηγής διέγερσης κατά το χρονικό διακριτό βήμα 3900 χρησιμοποιώντας ως μεταβλητή απεικόνισης τις αξονικές αποκρίσεις των μετακινήσεων. Η τοποθέτηση αισθητήρων πριν (αριστερά) από το σημείο της δύναμης και η ανάκλαση των κυμάτων από το δεσμευμένο άκρο της δοκού έχουν ως αποτέλεσμα την δημιουργία κάποιων μικρότερων σημάτων με πλάτος ίσο με το μισό της πηγής [6]. Επιπλέον, ο αισθητήρας στο ελεύθερο άκρο της δοκού έχει ως αποτέλεσμα την δημιουργία σήματος ίσου πλάτους με αυτό της πηγής και στο Σχήμα 4.4γ΄ βλέπουμε την συνάντηση των δύο κυμάτων ακριβώς στο σημείο της πηγής διέγερσης και την τοπική ενίσχυση του σήματος.
Οι αισθητήρες x_a που χρησιμοποιήθηκαν είχαν την δυνατότητα καταγραφής και των εγκάρσιων κυμάτων οπότε για την επίλυση του αντίστροφου προβλήματος χρησιμοποιήθηκαν οι χρονικά αντεστραμμένες αποκρίσεις τους. Στο *Σχήμα 4.5* φαίνεται για καθέναν από τους τρεις αισθητήρες η Ευκλείδια νόρμα (βλ. *Εξ. 2.14*) των εγκάρσιων και των περιστροφικών κυμάτων. Επίσης, παρατηρούμε μια επιπλέον διέγερση στο ελεύθερο άκρο της δοκού με μικρότερο πλάτος από την τοποθεσία της δύναμης στην περίπτωση του εγκάρσιου αισθητήρα x_a=1950 (βλ. *Σχήμα 4.5γ΄*).

Επιπρόσθετα, χρησιμοποιώντας μεγαλύτερο πλήθος αισθητήρων επιτυγχάνουμε καλύτερα και πιο ευκρινή αποτελέσματα. Στο *Σχήμα 4.6α΄* έχουν χρησιμοποιηθεί οι δύο αισθητήρες (αριστερά της πηγής) και παρατηρούμε ότι όχι μόνο υπάρχει αύξηση της τιμής του πλάτους του σήματος στο σημείο της δύναμης αλλά και τα δευτερεύοντα υπόκεινται σε μείωση γιατί τυχαίνει μερικοί από τους κόμβους τους να έχουν ίδιες συντεταγμένες, με αποτέλεσμα να αλληλοεξουδετερώνονται. Στην περίπτωση που χρησιμοποιούνται και οι τρεις αισθητήρες (βλ. *Σχήμα 4.6β΄*), το μόνο που συμβαίνει είναι η ενίσχυση του σήματος στο σημείο διέγερσης. Άλλωστε αναφέρθηκε παραπάνω πως όταν χρησιμοποιείται ο αισθητήρας x_a=1950, το δευτερεύον σήμα «πέφτει» ακριβώς στο σημείο της πηγής. Για παράδειγμα, αν υποθέσουμε ότι με χρήση ενός αισθητήρα το πλάτος του κύματος στο σημείο διέγερσης είναι ίσο με 1, τότε η χρήση δυο αισθητήρων θα αποφέρει την τιμή 2 (διπλασιασμός) λόγω της επαλληλίας. Επίσης, χρησιμοποιώντας και τον τρίτο αισθητήρα (στο ελεύθερο άκρο), περιμένουμε ότι το σήμα θα πάρει την τιμή 3 αλλά λόγω του δευτερεύοντος και ίσου πλάτους κύματος έχουμε μεγαλύτερη ενίσχυση, δηλαδή ίση με 4.

Μια ακόμη σημαντική παρατήρηση της συγκεκριμένης μελέτης είναι ότι είναι εφικτό να εντοπίσουμε την δύναμη διέγερσης χρησιμοποιώντας ποικίλες μεταβλητές απεικόνισης (βλ. *Υποενότητα 2.3.1*). Στο *Σχήμα 4.7* βλέπουμε τρεις -μια για κάθε αισθητήρα/διεγέρτη- απεικονίσεις των απόλυτων αποκρίσεων των ταχυτήτων για τα αξονικά κύματα.

Προσθήκη τυχαίων τιμών Γκαουσιανού θορύβου

Εδω μελετάμε την επιρροή που έχει ο θόρυβος-συγκεκριμένα ο Γκαουσιανός-στη μέθοδο της χρονικής αντιστροφής των αποκρίσεων που χρησιμοποιούμε. Η αρχή της επαλληλίας είναι αυτή που συνδέει τις «καθαρές» αποκρίσεις με αυτές του θορύβου ώστε να δημιουργηθούν τα παρακάτω διαγράμματα. Όπως έχει αναφερθεί στο *Κεφάλαιο 2*, ο θόρυβος προστίθεται στις «καθαρές» αποκρίσεις πολλαπλασιασμένος με έναν συντελεστή. Ο συντελεστής αυτός προκύπτει από το 10% της μέγιστης τιμής που εντοπίστηκε στις «καθαρές», είτε αξονικές είτε εγκάρσιες, αποκρίσεις. Στη συνέχεια, παρουσιάζονται τα διαγράμματα για τις μετακινήσεις και τις ταχύτητες των κόμβων.

Στα *Σχήματα 4.8* φαίνεται η επιρροή του θορύβου στον εντοπισμό της πηγής διέγερσης χρησιμοποιώντας τις απόλυτες αξονικές τιμές (*Σχήματα 4.8α΄, 4.8γ΄, 4.8ε΄*) και την Ευκλίδεια νόρμα των εγκάρσιων και περιστροφικών μετακινήσεων (*Σχήματα 4.8β΄, 4.8δ΄, 4.8ϛ΄*). Παρατηρούμε ότι ο θόρυβος δεν επηρεάζει σχεδόν καθόλου τις αξονικές μετατοπίσεις των κόμβων και η τοποθεσία της δύναμης εντοπίζεται με ακρίβεια. Αντίθετα, με την απεικόνιση της Ευκλίδειας νόρμας των εγκάρσιων και των περιστροφικών αποκρίσεων βλέπουμε πως οι κόμβοι επηρεάζονται αρκετά στις τοποθεσίες των αισθητήρων, αλλά και στην προκειμένη περίπτωση η πηγή διέγερσης εντοπίζεται στην αναμενόμενη θέση και στο κατάλληλο χρονικό βήμα. Αξιοσημείωτο είναι το *Σχήμα 4.8ϛ΄*, στο οποίο η μετακίνηση των κόμβων στο ελεύθερο άκρο της δοκού (τοποθεσία αισθητήρα x_a=1950) είναι μεγαλύτερη από της πηγής. Οπότε, θα ήταν δύσκολο κανείς να βγάλει συμπέρασμα για την ακριβή τοποθεσία της δύναμης.

Η απεικόνιση των ταχυτήτων στους κόμβων γίνεται στα *Σχήματα 4.9*. Μια αρχική παρατήρηση είναι ότι ο θόρυβος επηρεάζει περισσότερο τον εντοπισμό της πηγής διέγερσης στις ταχύτητες απ΄ ότι στις μετακινήσεις χωρίς όμως να δημιουργεί αμφιβολίες για την τοποθεσία της. Επιπλέον, όπως φαίνεται στα *Σχήματα 4.9β΄, 4.9δ΄, 4.9ϛ΄*, ο θόρυβος δημιουργεί συνεχόμενες αναταράξεις στο σήμα ενισχύοντας τα δευτερεύοντα σήματα (ghosts).

4.2 Εντοπισμός αλλοίωσης/φθοράς στη μονοδιάστατη περίπτωση (Defect identification)

Σε μια κατασκευή ή και σε τμήμα αυτής υπάρχει πιθανότητα κάποιας αλλοίωσης/φθοράς στο σύνολο ή σε τμήμα αυτής. Αυτό μπορεί να οφείλεται είτε στις συνεχείς καταπονήσεις είτε εξαιτίας κάποιας εξωτερικής δύναμης, όπως είναι ο σεισμός. Η ανίχνευση τους αποτελεί σημαντικό κομμάτι της παρούσας εργασίας και σε αυτή την ενότητα αναλύονται τα αποτελέσματα της συγκεκριμένης μελέτης για την μονοδιάστατη περίπτωση της δοκού.

4.2.1 Αποτελέσματα επίλυσης του ευθέος προβλήματος (Forward step)

Η παρούσα ενότητα αφορά την διάδοση του κύματος στο ευθύ πρόβλημα επίλυσης για την περίπτωση που η δοκός επιδέχεται κάποια αλλοίωση στα μηχανικά χαρακτηριστικά της. Επιπλέον, εντοπίζονται γραφικά οι χρόνοι άφιξης των τριών κυμάτων διάδοσης στο κοντινότερο σύνορο της αλλοιωμένης περιοχής από την πηγή. Στο *Κεφάλαιο 2* είδαμε πως πραγματοποιήθηκε αλλοίωσης και στα μηχανικά χαρακτηριστικά της δοκού και στα γεωμετρικά. Δηλαδή, το μέτρο ελαστικότητας E στην περιοχή της φθοράς θεωρήθηκε ίσο με το $\frac{1}{5}$ του αρχικού και το εμβαδόν διατομής A μειώθηκε στο μισό του αρχικού.

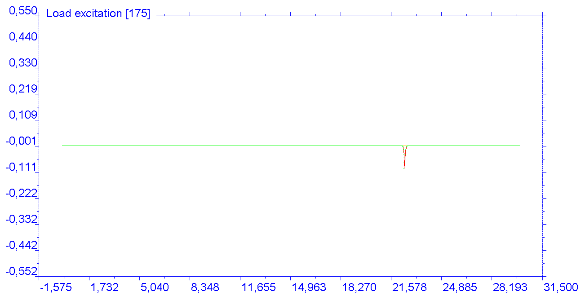
Το *Σχήμα 4.10α΄* παρουσιάζει την διάδοση του αξονικού κύματος στον κόμβο με id=351. Ο εν λόγω κόμβος αποτελεί το ένα από τα δύο σύνορα της περιοχής αλλοίωσης και βρίσκεται πιο κοντά στην πηγή διέγερσης. Η ταχύτητα του αξονικού κύματος έχει υπολογιστεί μεγαλύτερη από αυτές των καμπτικών κυμάτων, άρα το αξονικό κύμα ανιχνεύται πρώτο. Πράγματι, στο προαναφερθέν σχήμα παρατηρούμε ότι το αξονικό κύμα χρειάζεται περίπου $19s$ για να εντοπίσει την περιοχή της φθοράς, ενώ στα *Σχήματα 4.10β΄* και *4.10γ΄*, το εγκάρσιο και το περιστροφικό κύμα φτάνουν περίπου στα $32s$. Επίσης, μπορεί να σημειωθεί ότι και στα τρια σχήματα παρατηρούνται κι άλλες ενισχύσεις του σήματος, το οποίο σημαίνει οτι τα κύματα περνούν από την περιοχή αλλοίωσης περισσότερες από μια φορες.

Η διάδοση κύματος σε μέσο το οποίο έχει φθαρεί σε κάποιο σημείο του διαφέρει από τη διάδοση σε μια αναλλοίωτη δοκό. Η διαφορά είναι στην περιοχή που υπάρχει η φθορά. Η περιοχή αυτή λειτουργεί σαν πηγή κυμάτων την στιγμή όμως που κάποιο από τα κύματα της πηγής διέγερσης περάσει από εκεί. Στο *Σχήμα 4.11α΄* βλέπουμε τη διάδοση των κυμάτων της διεγείρουσας δύναμης και τη στιγμή t=23.08s (βλ. *Σχήμα 4.11β΄*) το αξονικό κύμα έχει ήδη εντοπίσει την περιοχή φθοράς αλλά είναι σχεδόν αδύνατο να το διακρίνουμε διότι η δοκός βρίσκεται στο οριζόντιο επίπεδο. Στο *Σχήμα 4.11δ΄* παρατηρούμε πως όχι μόνο το εγκάρσιο αλλά και το περιστροφικό κύμα έφτασε στο σύνορο της δυσλειτουργικής περιοχής και στα t=32.28s έχουμε μια τοπική ενίσχυση των σημάτων (βλ. *Σχήμα 4.11ε΄*). Στη συνέχεια, δημιουργούνται κύματα προς όλες τις κατευθύνσεις και ταξιδεύουν κατά μήκος της δοκού. Κάθε φορά που κάποιο κύμα φτάνει στην περιοχή με τη φθορά, η παραπάνω διαδικασία επαναλαμβάνεται. Τελικά, το φαινόμενο σταματά στα $63s$ και στο *Σχήμα 4.11ι΄* φαίνεται η θέση και η κατάσταση των κυμάτων.

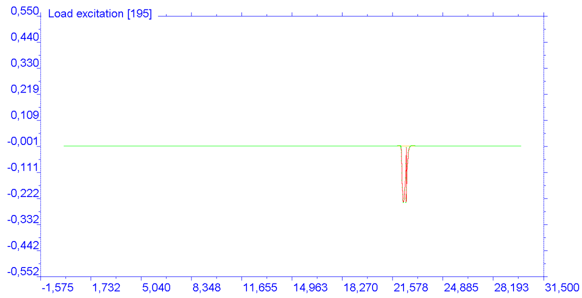
4.2.2 Αποτελέσματα επίλυσης του αντίστροφου προβλήματος (Backward step)

Με την μέθοδο της χρονικής αντιστρεψιμότητας των τιμών είναι εφικτό να εντοπίσουμε όχι μόνο την τοποθεσία της δύναμης διέγερσης αλλά και την περιοχή μιας βλάβης σε κάποια κατασκευή (βλ. *Ενότητα* 2.2). Στην *Ενότητα* 4.2.2 παρουσιάζονται τα αποτελέσματα χρησιμοποιώντας τις Ευκλείδιες νόρμες των μετατοπίσεων και των ταχυτήτων.

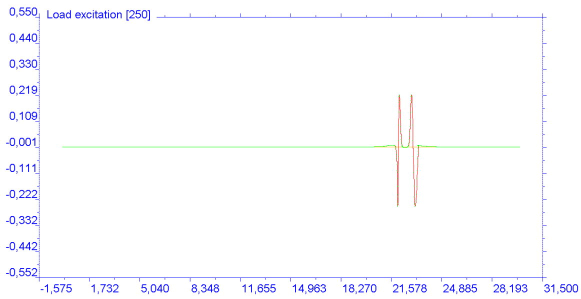
Τα *Σχήματα* 4.12 απεικονίζουν τον εντοπισμό της βλάβης κάνοντας χρήση την Ευκλίδεια νόρμα των μετακινήσεων έχοντας στείλει τις ανεστραμμένες αποκρίσεις από έξι αισθητήρες. Αρχικά, παρατηρούμε ότι από τους αξονικούς αισθητήρες (βλ. *Σχήματα* *4.12α΄*, *4.12γ΄* και *4.12ε΄*) εντοπίζεται με ακρίβεια η περιοχή της φθοράς, ενώ χρησιμοποιώντας τους εγκάρσιους αισθητήρες παρατηρείται ότι μόνο από τους x_a=650 (*Σχήμα* *4.12β΄*) και x_a=1300 (*Σχήμα* *4.12δ΄*) έχουμε εντοπισμό της αλλοίωσης. Αντίθετα, στο *Σχήμα* *4.12ϛ΄* φαίνεται ο εγκάρσιος αισθητήρας x_a=1950, ο οποίος αδυνατεί να δώσει αποτελέσματα γιατί ο χρόνος που εξελίσσεται το φαινόμενο δεν επαρκεί για να φτάσουν τα καμπτικά κύματα στην τοποθεσία της δυσλειτουργίας.
Επιπλέον, αντίστοιχα συμπεράσματα σημειώνονται και για την Ευκλίδεια νόρμα των ταχυτήτων. Στα *Σχήματα* 4.13 εντοπίζεται ακριβώς η τοποθεσία της αλοιωμένης περιοχής χρησιμοποιώντας τρεις αξονικούς αισθητήρες και δύο εγκάρσιους, ενώ για τον τρίτο εγκάρσιο παρατηρούμε το ίδιο ακριβώς πρόβλημα με τις μετατοπίσεις.



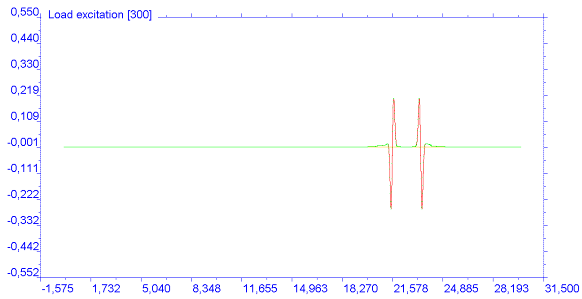
(α′) t=2.69s



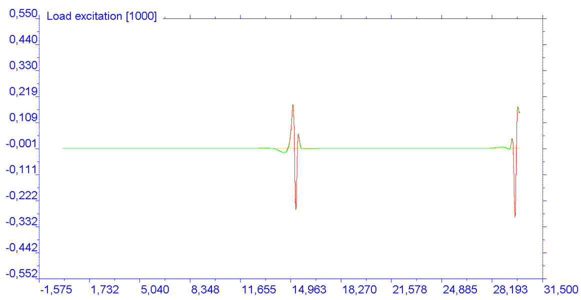
(β′) t=3s



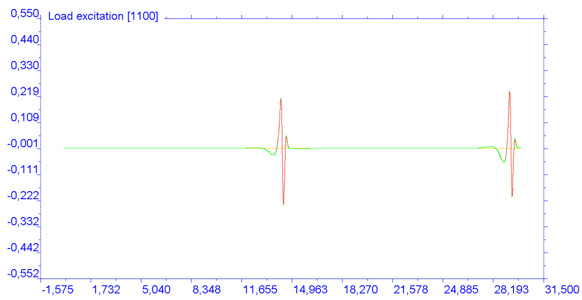
(γ′) t=3.85s



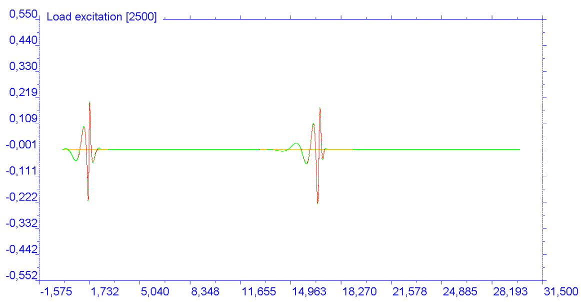
(δ′) t=4.62s



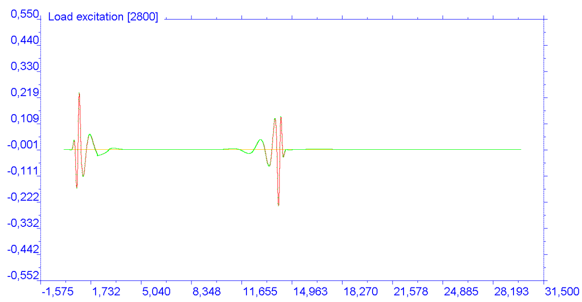
(ε′) t=15.38s



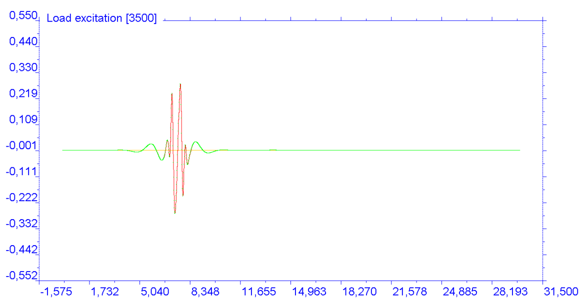
(ϛ′) t=16.92s



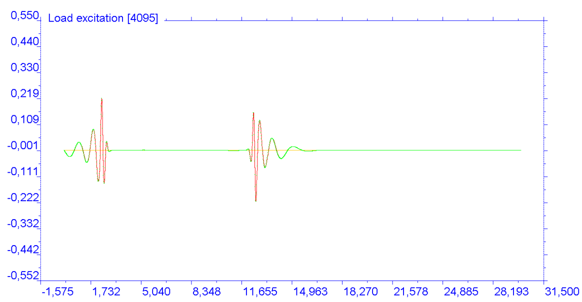
(ζ′) t=38.46s



(η′) t=43.08s

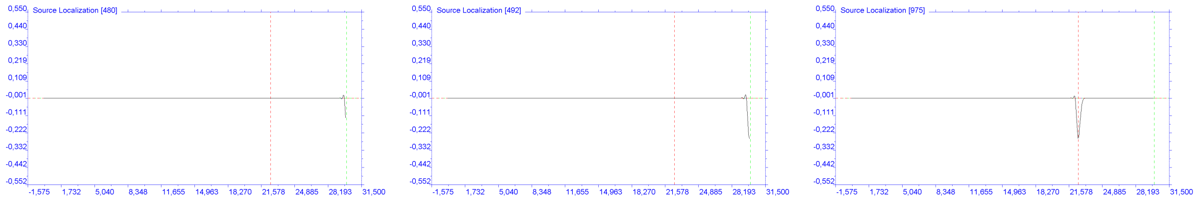


(ϑ′) t=53.85s

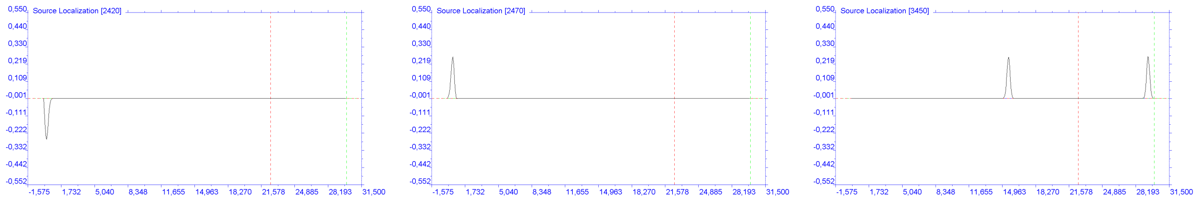


(ι′) t=63.00s

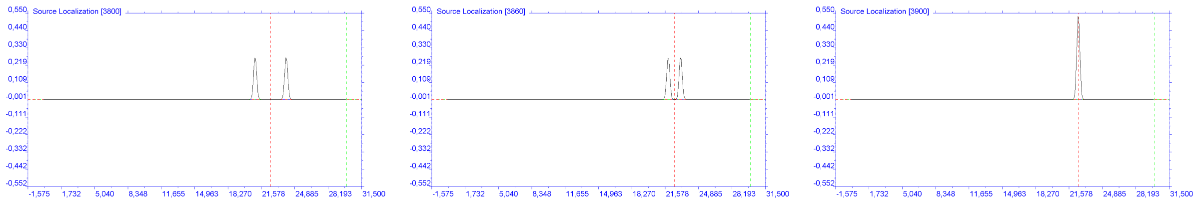
Σχήμα 4.2: Διάδοση κύματος κατά την ευθεία επίλυση (Forward wave propagation). Η παρακολούθηση σχετικού βίντεο μπορεί να γίνει στο ακόλουθο link.



(α′) t=7.38s (β′) t=7.57s (γ′) t=15s

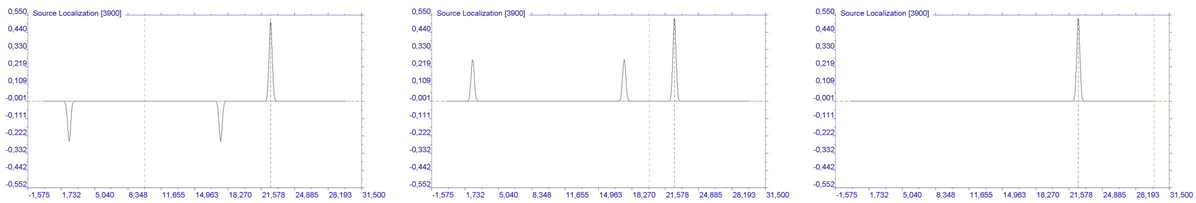


(δ′) t=37.23s (ε′) t=38s (ϛ′) t=53.08s



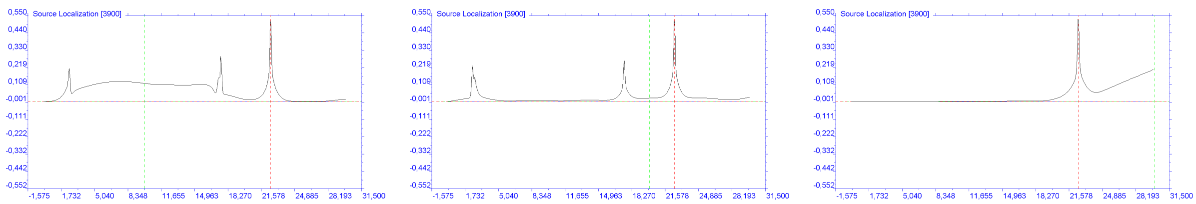
(ζ′) t=58.46s (η′) t=59.39s (ϑ′) t=60s

Σχήμα 4.3: Εντοπισμός της πηγής διέγερσης από τον αισθητήρα/διεγέρτη x_a=1950 έχοντας καταγράψει τις αξονικές μετατοπίσεις των σημείων. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης. Η παρακολούθηση σχετικού βίντεο μπορεί να γίνει στο ακόλουθο link.



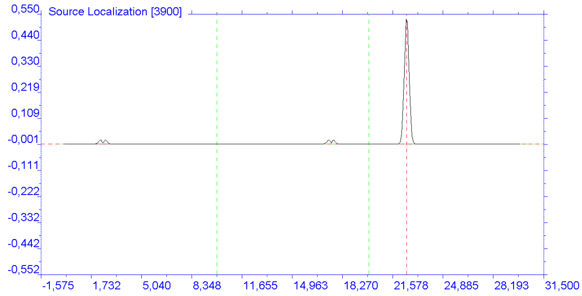
(α′) Αισθητήρας x_a=650 (β′) Αισθητήρας x_a=1300 (γ′) Αισθητήρας x_a=1950

Σχήμα 4.4: Εντοπισμός θέσης διέγερσης με χρήση των αξονικών αποκρίσεων των μετατοπίσεων. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.

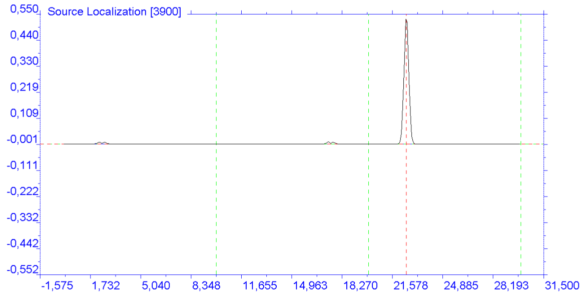


(α′) Αισθητήρας x_a=650 (β′) Αισθητήρας x_a=1300 (γ′) Αισθητήρας x_a=1950

Σχήμα 4.5: Εντοπισμός θέσης διέγερσης με χρήση της Ευκλίδειας νόρμας την εγκάρσιων και περιστροφικών αποκρίσεων των μετατοπίσεων. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.

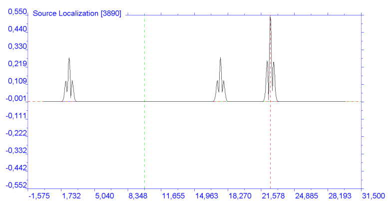


(α′) Αισθητήρες x_a=650, 1300

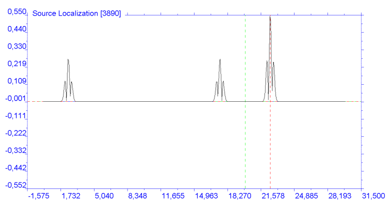


(β′) Αισθητήρες x_a=650, 1300, 1950

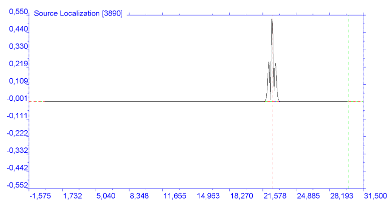
Σχήμα 4.6: Εντοπισμός θέσης διέγερσης με χρήση των απόλυτων αξονικών αποκρίσεων των μετατοπίσεων. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.



(α′) Αισθητήρας x_a=650

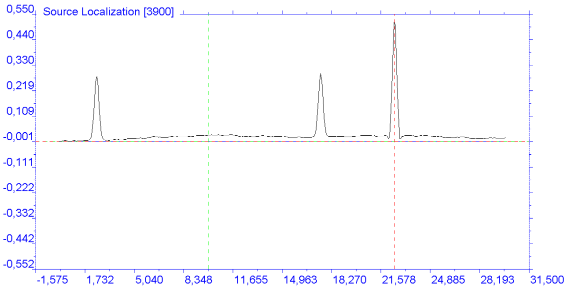


(β′) Αισθητήρας x_a=1300

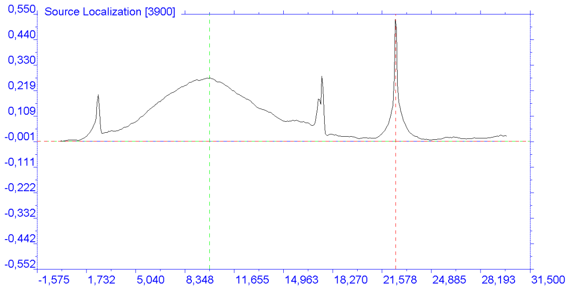


(γ′) Αισθητήρας x_a=1950

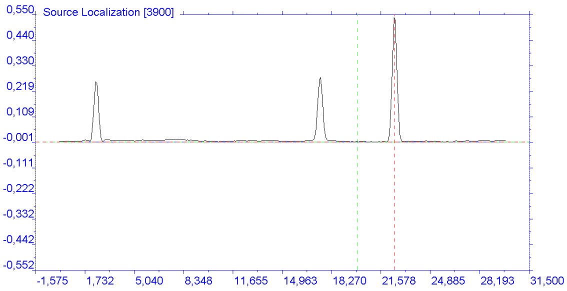
Σχήμα 4.7: Εντοπισμός θέσης διέγερσης με χρήση των απόλυτων αξονικών αποκρίσεων των ταχυτήτων. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.



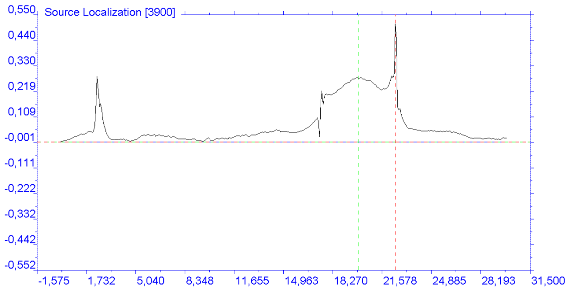
(α′) Αισθητήρας x_a=650



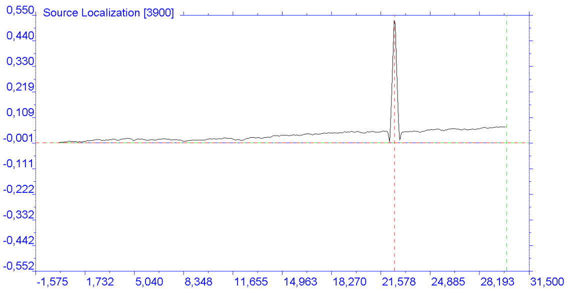
(β′) Αισθητήρας x_a=650



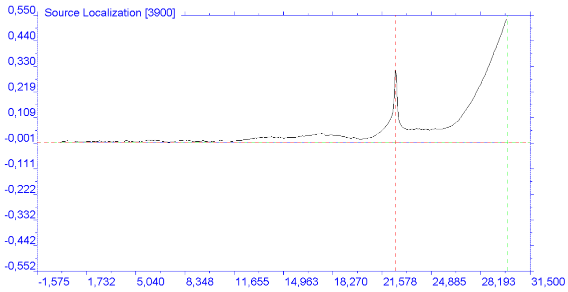
(γ′) Αισθητήρας x_a=1300



(δ′) Αισθητήρας x_a=1300

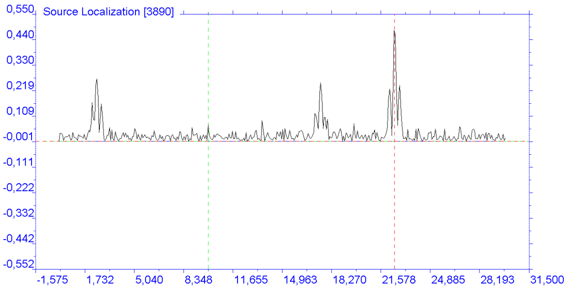


(ε′) Αισθητήρας x_a=1950

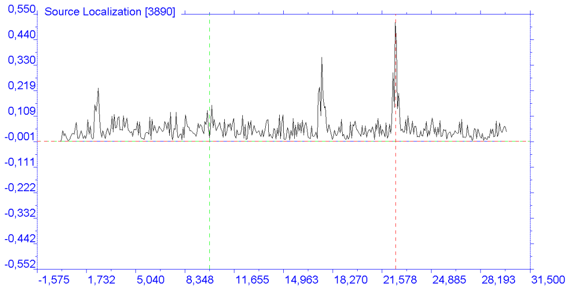


(ϛ′) Αισθητήρας x_a=1950

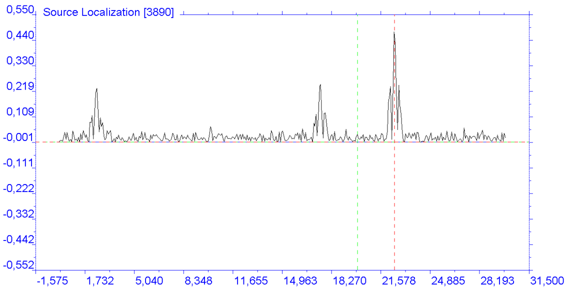
Σχήμα 4.8: Απεικόνιση των απόλυτων αξονικών (αριστερά στήλη) και της Ευκλίδειας νόρμας των εγκάρσιων και περιστροφικών (δεξιά στήλη) μετατοπίσεων με την προσθήκη Γκαουσιανού θορύβου. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.



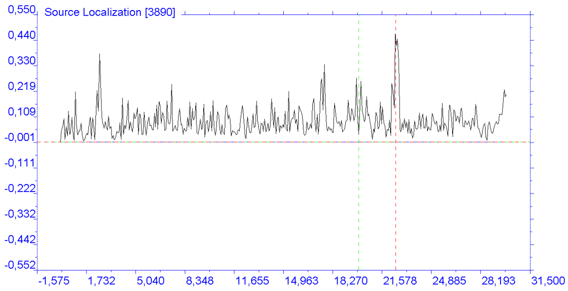
(α′) Αισθητήρας x_a=650



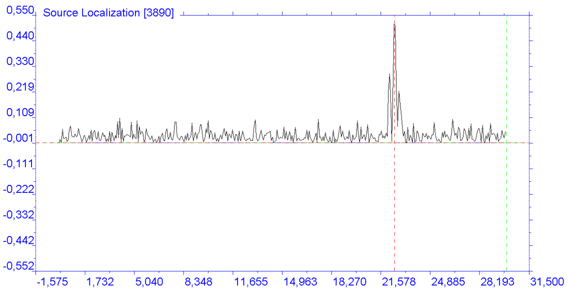
(β′) Αισθητήρας x_a=650



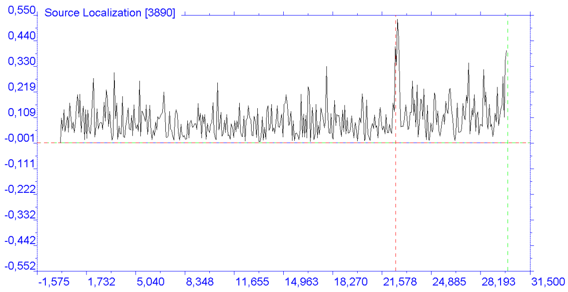
(γ′) Αισθητήρας x_a=1300



(δ′) Αισθητήρας x_a=1300

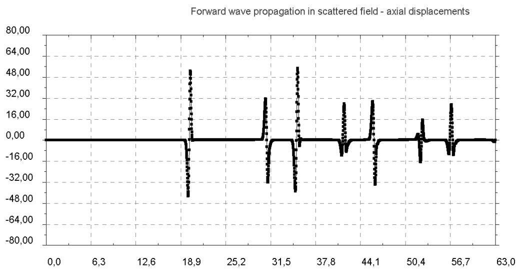


(ε′) Αισθητήρας x_a=1950

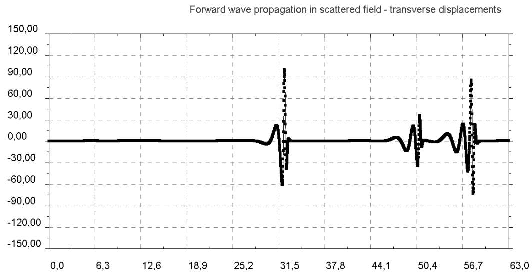


(ϛ′) Αισθητήρας x_a=1950

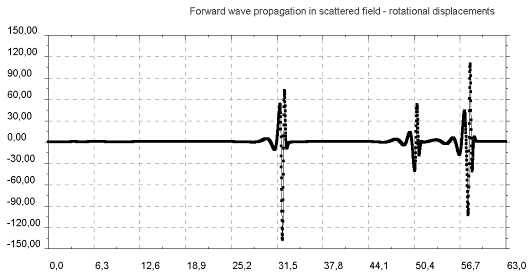
Σχήμα 4.9: Απεικόνιση των απόλυτων αξονικών (αριστερά στήλη) και της Ευκλίδειας νόρμας των εγκάρσιων και περιστροφικών (δεξιά στήλη) ταχυτήτων με την προσθήκη Γκαουσιανού θορύβου. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, ενώ η κόκκινη την τοποθεσία της δύναμης.



(α′)

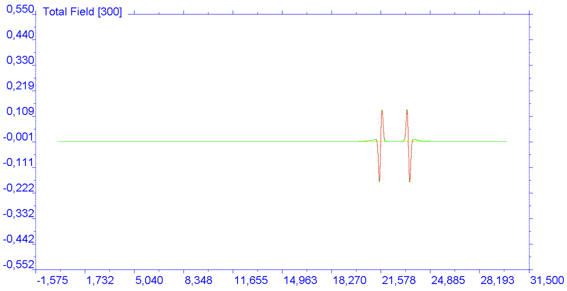


(β′)

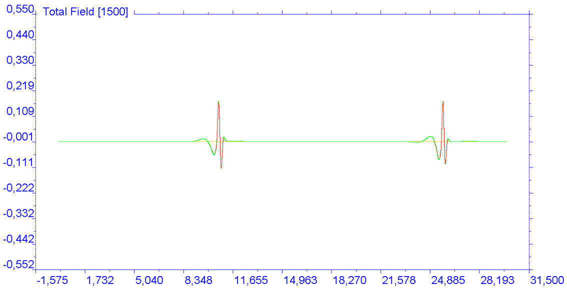


(γ′)

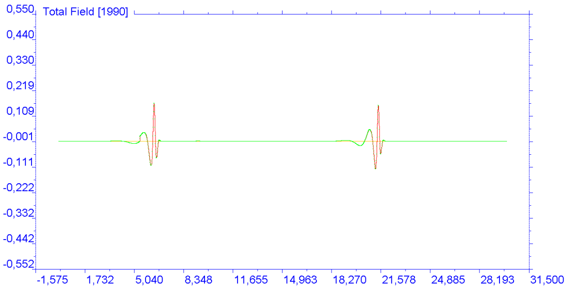
Σχήμα 4.10: Διάδοση κύματος στον κοντινότερο κόμβο της περιοχής αλλοίωσης από την πηγή (κόμβος με id=351) κατά την ευθεία επίλυση του συνολικού πεδίου (Total field). Μονάδες μέτρησης: άξονας y $[10^{-3}m]$, άξονας x $[sec]$.



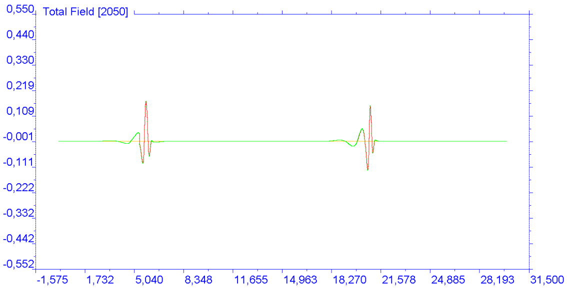
(α′) t=4.62s



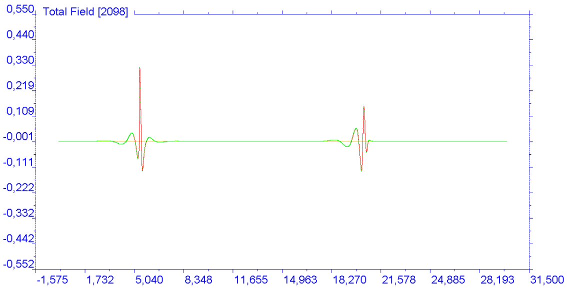
(β′) t=23.08s



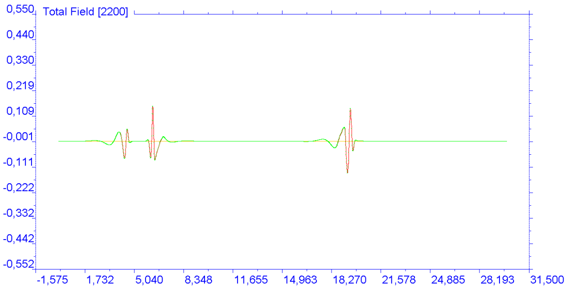
(γ′) t=30.62s



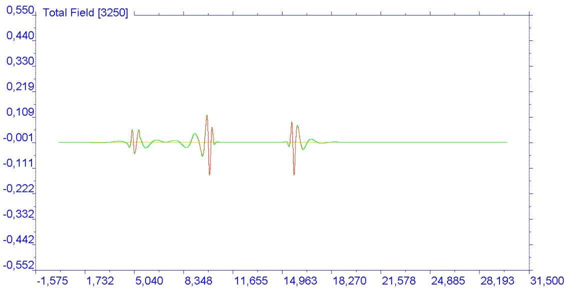
(δ′) t=31.54s



(ε′) t=32.28s

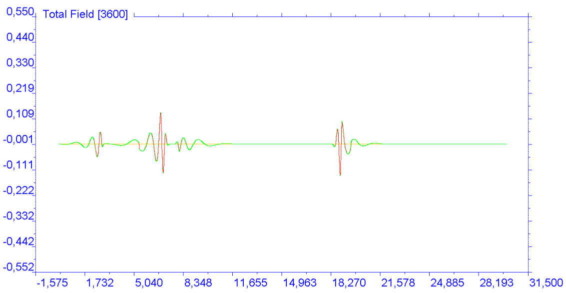


(ϛ′) t=33.85s

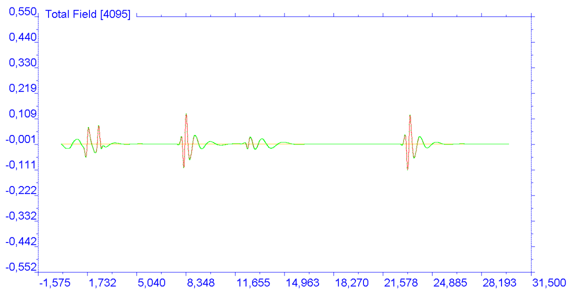


(ζ′) t=50.00s

(η′) t=52.31s

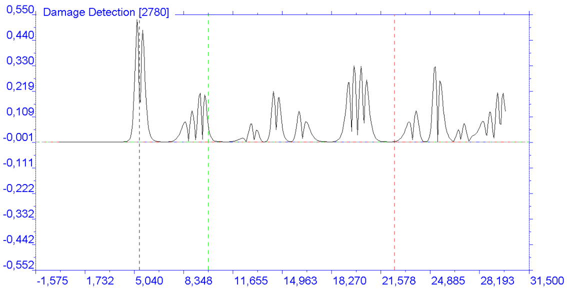


(ϑ′) t=55.38s

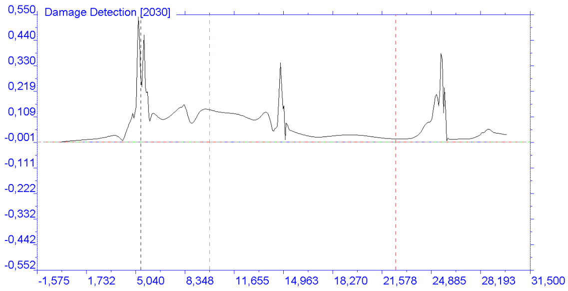


(ι′) t=63s

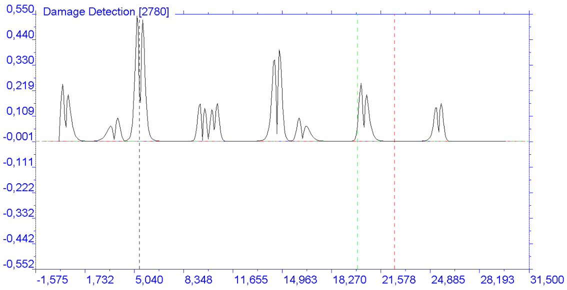
Σχήμα 4.11: Διάδοση κύματος στην αλλοιωμένη δοκό κατά την ευθεία επίλυση του συνολικού πεδίου (Total field). Η παρακολούθηση σχετικού βίντεο μπορεί να γίνει στο ακόλουθο link.



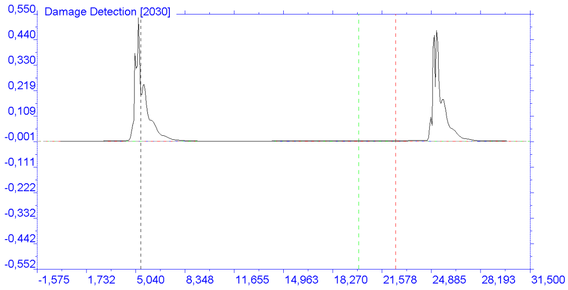
(α′) Αισθητήρας x_a=650



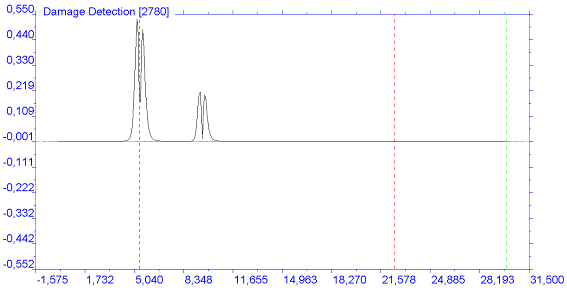
(β′) Αισθητήρας x_a=650



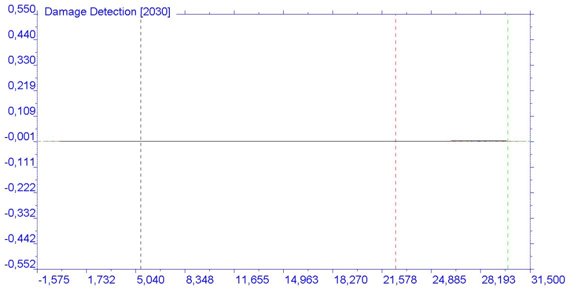
(γ′) Αισθητήρας x_a=1300



(δ′) Αισθητήρας x_a=1300

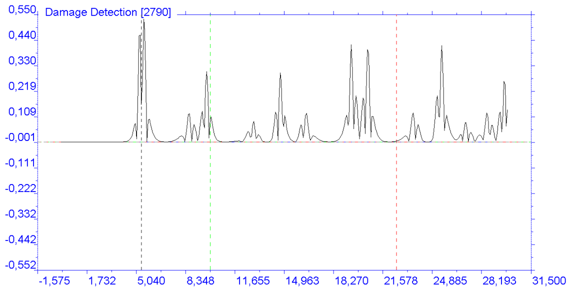


(ε′) Αισθητήρας x_a=1950

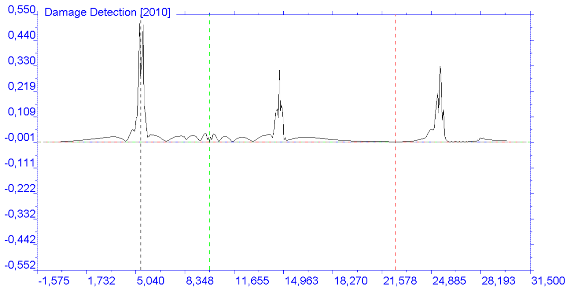


(ϛ′) Αισθητήρας x_a=1950

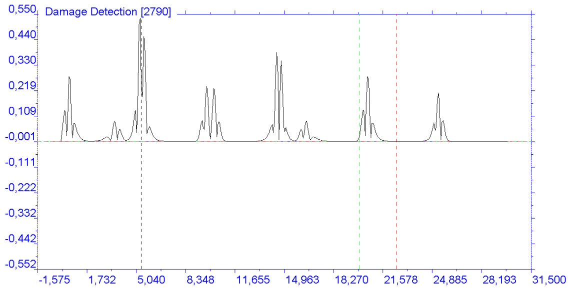
Σχήμα 4.12: Απεικόνιση της Ευκλίδειας νόρμας των μετατοπίσεων για τον εντοπισμό της αλλοιωμένης περιοχής. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, η γκρί δείχνει την τοποθεσία της φθοράς, ενώ η κόκκινη την τοποθεσία της δύναμης.



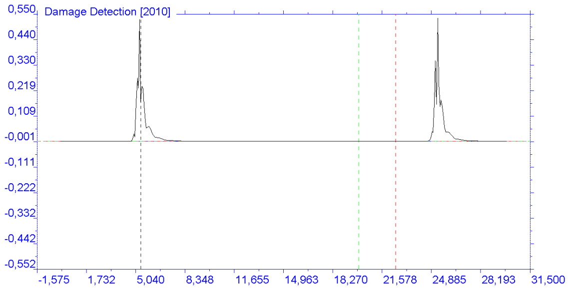
(α′) Αισθητήρας x_a=650



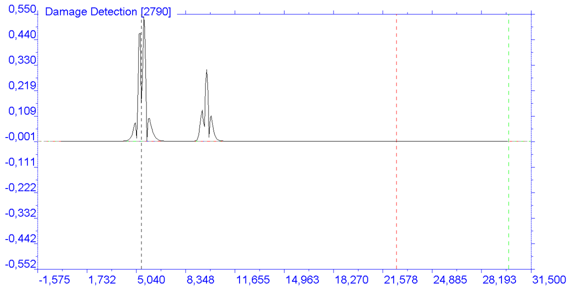
(β′) Αισθητήρας x_a=650



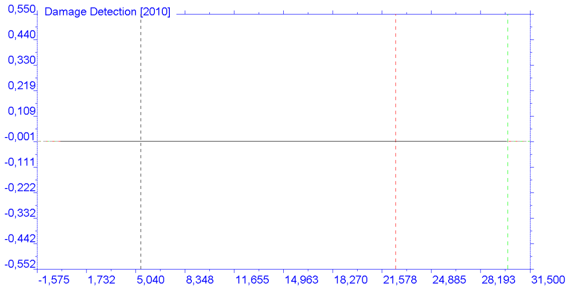
(γ′) Αισθητήρας x_a=1300



(δ′) Αισθητήρας x_a=1300



(ε′) Αισθητήρας x_a=1950



(ϛ′) Αισθητήρας x_a=1950

Σχήμα 4.13: Απεικόνιση της Ευκλίδειας νόρμας των ταχυτήτων για τον εντοπισμό της αλλοιωμένης περιοχής. Η πράσινη διακεκκομένη γραμμή δείχνει την τοποθεσία του αισθητήρα x_a, η γκρί δείχνει την τοποθεσία της φθοράς, ενώ η κόκκινη την τοποθεσία της δύναμης.

4.3 Μελέτη του λόγου σήμα-προς-θόρυβο (Signal to Noise Ratio)

Σημαντικό κομμάτι της παρούσας εργασίας είναι η μελέτη του λόγου σήμα-προς-θόρυβο (Signal to Noise Ratio (SNR)) για διάφορες θέσεις και διευθύνσεις των αισθητήρων. Στόχος της εν λόγω μελέτης είναι να βρεθεί η βέλτιστη διάταξη των αισθητήρων που θα πρέπει να τοποθετηθεί στην κατασκευή ώστε να έχουμε όσο το δυνατόν πιο ποιοτικές καταγραφές.

Ο *Πίνακας 4.1* παρουσιάζει τον λόγο σήμα-προς-θόρυβο για τέσσερις διαφορετικές διατάξεις αισθητήρων των αξονικών μετατοπίσεων και ταχυτήτων με την προσθήκη ή χωρίς Γκαουσιανού θορύβου. Σε κάθε διάταξη οι αισθητήρες τοποθετούνταν διαδοχικά. Δηλαδή, για παράδειγμα στην διάταξη 1 (Αρ. σετ 1), πρώτος και μοναδικός αισθητήρας ήταν ο x_a=650, στη συνέχεια τοποθετήθηκε και ο x_a=1300, ενώ ως τρίτος ορίστικε ο x_a=1950. Επίσης, τα σύμβολα SNR_{DC} και SNR_{DN} δηλώνουν τον λόγο υπολογισμένο από τις αποκρίσεις των μετακινήσεων χωρίς Γκαουσιανό θόρυβο και με Γκαουσιανό θόρυβο, αντίστοιχα. Ακριβώς ίδιας λογικής είναι και τα σύμβολα SNR_{VC} και SNR_{VN}, με τη διαφορά ότι έχουν ληφθεί υπόψη οι αποκρίσεις των ταχυτήτων.

Αρ. σετ	1	2	3	4
Αισθ.	$x_{650}, x_{1300}, x_{1950}$	$x_{650}, x_{1950}, x_{1300}$	$x_{1300}, x_{650}, x_{1950}$	$x_{1300}, x_{1950}, x_{650}$
$\mathrm{SNR_{DC}}$	1.958, 31.826, 63.653	1.958, 5.875, 63.653	2.000, 31.826, 63.653	2.000, 6.000, 63.653
$\mathrm{SNR_{DN}}$	1.767, 23.539, 11.652	1.767, 3.920, 11.652	1.933, 23.539, 11.652	1.933, 6.697, 11.652
$\mathrm{SNR_{VC}}$	1.919, 16.227, 32.453	1.919, 5.756, 32.453	2.000, 16.227, 32.453	2.000, 6.000, 32.453
$\mathrm{SNR_{VN}}$	1.774, 9.771, 6.631	1.774, 4.476, 6.631	1.908, 9.771, 6.631	1.908, 4.491, 6.631

Πίνακας 4.1: SNR εντοπισμού της δύναμης διέγερσης από αξονικές καταγραφές κυμάτων των αισθητήρων, ο δείκτης DC συμβολίζει τις απεικονίσεις των μετατοπίσεων χωρίς θόρυβο, ο DN τις απεικονίσεις των μετατοπίσεων με θόρυβο, ενώ οι VC και VN είναι αντίστοιχα καθορισμένοι αλλά για τις απεικονίσεις των ταχυτήτων.

Μια πρώτη παρατήρηση που μπορεί να γίνει στον *Πίνακα 4.1* αφορά τις τιμές του λόγου. Για παράδειγμα, στα σετ 1 και 3 για τις «καθαρές» τιμές παρατηρούμε ότι χρησιμοποιώντας έναν αισθητήρα η τιμή του σήματος στην πηγή είναι 2 φορές μεγαλύτερη από αυτή του θορύβου (ghost). Τοποθετώντας ακόμη έναν αισθητήρα, βλέπουμε ότι η τιμή γίνεται περίπου 16 φορές μεγαλύτερη. Αυτό συμβαίνει όχι μόνο επειδή το σήμα στην πηγή διέγερσης ενισχύεται αλλά και ο θόρυβος εξασθενεί. Η παραπάνω παρατήρηση μπορεί να εξακριβωθεί και σχηματικά στο *Σχήμα 4.6α΄*. Τελικά, με την προσθήκη του αισθητήρα x_a=1950, η τιμή του λόγου είναι διπλάσια της προηγούμενης. Στην προκειμένη περίπτωση, η τοποθέτηση του αισθητήρα στο ελεύθερο άκρο της δοκού ενισχύει μόνο το σήμα στην δύναμη διέγερσης γιατί ο θόρυβος (ghost) που δημιουργείται από τον συγκεκριμένο αισθητήρα ενισχύει κι αυτός την ίδια την πηγή. Γι΄ αυτό και δεν παρατηρείται γραμμικότητα στον λόγο σήμα-προς-θόρυβο. Επιπλέον, μια γενική παρατήρηση είναι ότι σε κάθε περίπτωση του SNR και σε κάθε διάταξη αισθητήρων καταλήγουμε στην ίδια τιμή του λόγου. Για παράδειγμα, στον *Πίνακα 4.2* και για το SNR_{VC} καταλήγουμε στην τιμή 5.363 για οποιαδήποτε από τις τέσσερις πιθανές διατάξεις.

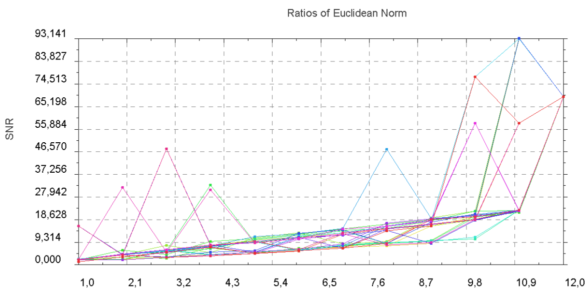
Η βέλτιστη θέση και διεύθυνση καταγραφής για τους αισθητήρες μπορεί να βρεθεί από τις τιμές του λόγου σήμα-προς-θόρυβο. Ας υποθέσουμε ότι υπάρχει δυνατότητα για χρήση δύο αισθητήρων, από τον *Πίνακα 4.2* και από το SNR_{DC} συμπεραίνουμε ότι ο βέλτιστος συνδυασμός αισθητήρων είναι αυτός των σετ 5 και 7. Δηλαδή, επιλέγουμε να τοποθετήσουμε πάνω

Αρ. σετ	5	6	7	8
Αισθ.	$y_{650}, y_{1300}, y_{1950}$	$y_{650}, y_{1950}, y_{1300}$	$y_{1300}, y_{650}, y_{1950}$	$y_{1300}, y_{1950}, y_{650}$
SNR_{DC}	1.790, 4.543, 5.128	1.790, 3.909, 5.128	1.998, 4.543, 5.128	1.998, 3.757, 5.128
SNR_{DN}	1.867, 3.014, 1.015	1.867, 1.155, 1.015	1.795, 3.014, 1.015	1.795, 1.431, 1.015
SNR_{VC}	1.428, 2.979, 5.363	1.428, 3.319, 5.363	1.721, 2.979, 5.363	1.721, 5.245, 5.363
SNR_{VN}	1.416, 2.591, 1.414	1.416, 1.590, 1.414	1.223, 2.591, 1.414	1.223, 1.028, 1.414

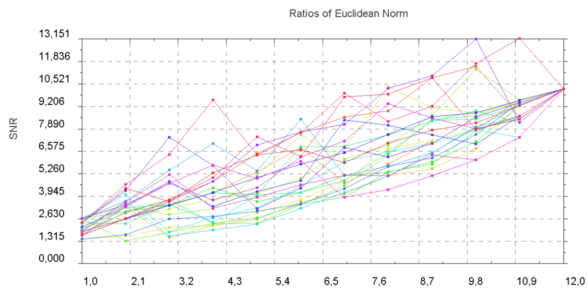
Πίνακας 4.2: SNR εντοπισμού της δύναμης διέγερσης από εγκάρσιες καταγραφές κυμάτων των αισθητήρων, ο δείκτης DC συμβολίζει τις απεικονίσεις των μετατοπίσεων χωρίς θόρυβο, ο DN τις απεικονίσεις των μετατοπίσεων με θόρυβο, ενώ οι VC και VN είναι αντίστοιχα καθορισμένοι αλλά για τις απεικονίσεις των ταχυτήτων.

στη δοκό τους αισθητήρες y_{650} και y_{1300}. Επίσης, να σημειωθεί εδώ πως στην υπολογιστική διαδικασία που ακολουθούμε δεν μας ενδιαφέρει η σειρά τοποθέτησης των αισθητήρων.

Επιπροσθέτως, πραγματοποιήθηκε μια ακόμη μελέτη για 12 αισθητήρες ώστε να εξακριβώσουμε αν υπάρχει κάποια γραμμική σχέση του λόγου με τον αριθμό των αισθητήρων. Στα *Σχήματα 4.14α΄* και *4.14β΄* παρουσιάζεται ο λόγος σήμα-προς-θόρυβο για διάφορους τυχαίους συνδυασμούς των 12 αισθητήρων. Στο *Σχήμα 4.14α΄* απεικονίζεται ο λόγος όπως υπολογίστηκε χρησιμοποιώντας μόνο αισθητήρες που καταγράφουν αξονικές μετατοπίσεις, ενώ στο *Σχήμα 4.14β΄* φαίνεται ο λόγος από τις καταγραφές εγκάρσιων και περιστροφικών αισθητήρων κίνησης. Και στα δύο σχήματα παρατηρείται η γραμμικότητα του λόγου σήμα-προς-θόρυβο, οπότε μπορούμε να συμπεράνουμε ότι η χρήση περισσότερων αισθητήρων βοηθά στον διαυγή εντοπισμό της πηγής διέγερσης ή/και του σημείου φθοράς.



(α΄)



(β΄)

Σχήμα 4.14: SNR για διάφορα σετ αισθητήρων (α΄) χρησιμοποιώντας τις αξονικές αποκρίσεις και (β΄) χρησιμοποιώντας τις αποκρίσεις της εγκάρσιας και περιστροφικής κίνησης.

Κεφάλαιο 5

Πρόβλημα διάδοσης αξονικών και εγκάρσιων κυμάτων σε διαδιάστατη πλαισιακή κατασκευή

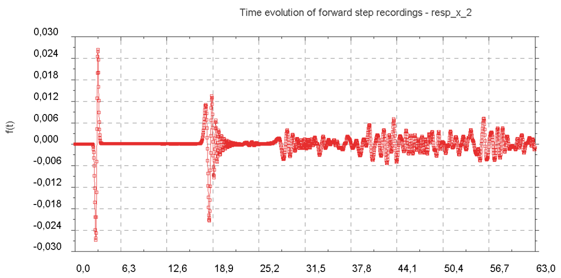
Στο συγκεκριμένο κεφάλαιο παρουσιάζονται με ακρίβεια τα αποτελέσματα για τα δύο βασικά προβλήματα της παρούσας εργασίας στην περίπτωση της δισδιάστατης κατασκευής. Αρχικά, στην *Ενότητα* 5.1 σχολιάζονται τα διαγράμματα τα οποία οδηγούν σε συμπεράσματα για το πρόβλημα του εντοπισμού της πηγής διέγερσης και στη συνέχεια, στην *Ενότητα* 5.2 παρουσιάζεται η μελέτη εντοπισμού μιας αλλοίωσης ή φθοράς πάνω στην κατασκευή. Επιπλέον, το τελευταίο ενδιαφέρον κομμάτι της εργασίας είναι η βελτιστοποίηση του αριθμού των αισθητήρων που θα χρησιμοποιηθούν. Αυτό, μπορεί να επιτευχθεί, αναλύοντας τις τιμές και τα διαγράμματα του λόγου σήμα-προς-θόρυβο (*Ενότητα* 5.3).

5.1 Εντοπισμός πηγής διέγερσης (Source Localization)

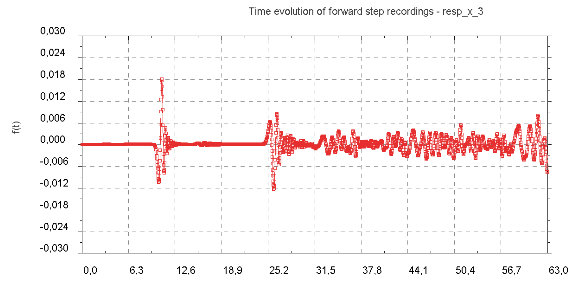
Για την εύρεση της τοποθεσίας της πηγής διέγερσης στη δισδιάστατη περίπτωση χρησιμοποιήθηκαν αισθητήρες-διεγέρτες σε 6 κομβικά σημεία της κατασκευής (βλ. *Σχήμα* 2.5). Τοποθετήθηκαν αισθητήρες οριζόντιας και κατακόρυφης διεύθυνσης, άρα συνολικά χρησιμοποιήθηκαν 12 αισθητήρες. Η διαδικασία που ακολουθήθηκε ήταν όμοια με αυτή της μονοδιάστατης κατασκευής, η οποία περιγράφηκε στην *Ενότητα* 4.1.

5.1.1 Αποτελέσματα επίλυσης του ευθέος προβλήματος (Forward step)

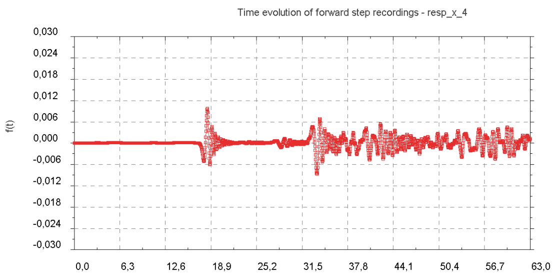
Η ευθεία επίλυση αποτελεί το πρωταρχικό στάδιο για το πρόβλημα του εντοπισμού της πηγής διέγερσης. Στην παρούσα υποενότητα παρουσιάζονται οι καταγραφές για τις μετακινήσεις και τις ταχύτητες όλων των οριζόντιων αισθητήρων. Στο *Σχήμα* 5.1 απεικονίζονται οι τιμές των οριζόντιων μετατοπίσεων συναρτήσει του χρόνου όπως καταγράφηκαν από κάθε αισθητήρα. Παρατηρούμε ότι τα κύματα περνούν αρκετές φορές από τους κόμβους που είναι τοποθετημένοι οι αισθητήρες και στο *Σχήμα* 5.1α΄ διακρίνουμε την έντονη διέγερση του κόμβου με id ίσο με 2. Αυτό συμβαίνει γιατί στον συγκεκριμένο κόμβο δεν υπάρχει μόνο ο αισθητήρας x_a=2 αλλά είναι και η τοποθεσία της δύναμης. Αντίστοιχα σχήματα (βλ. *Σχήμα* 5.2) παρουσιάζονται και παρακάτω με την διαφορά ότι μπορούμε να δούμε τις οριζόντιες αποκρίσεις των ταχυτήτων στους κόμβους των αισθητήρων για κάθε χρονική στιγμή εξέλιξης του φαινομένου.



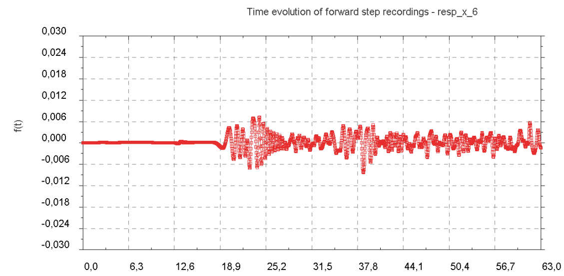
(α′) Αισθητήρας x_a=2



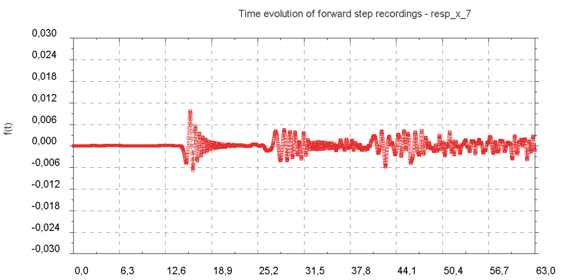
(β′) Αισθητήρας x_a=3



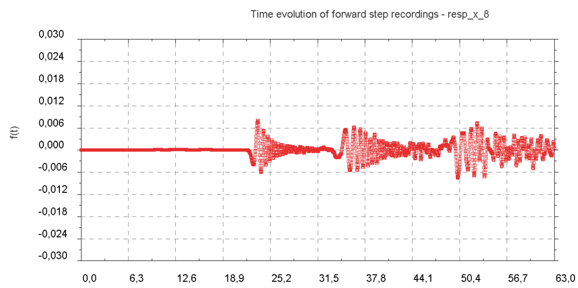
(γ′) Αισθητήρας x_a=4



(δ′) Αισθητήρας x_a=6



(ε′) Αισθητήρας x_a=7



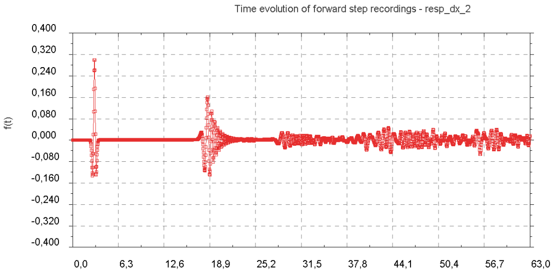
(ϛ′) Αισθητήρας x_a=8

Σχήμα 5.1: Απεικόνιση των μετατοπίσεων κατά την οριζόντια διεύθυνση συναρτήσει του χρόνου όπως καταγράφηκαν από τους αισθητήρες με id ίσο με 2, 3, 4, 6, 7, 8. Μονάδες μέτρησης: άξονας y $[m]$, άξονας x $[s]$.

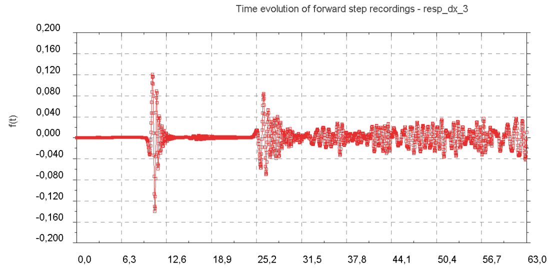
5.1.2 Αποτελέσματα επίλυσης του αντίστροφου προβλήματος (Backward step)

Κατά την αντίστροφη διαδικασία επίλυσης υπολογίστηκε η χρονική στιγμή που αναμένεται ο εντοπισμός της πηγής διέγερσης. Δηλαδή, γνωρίζουμε ότι η δύναμη δρά στην αρχική χρονική στιγμή t_0=3s και το φαινόμενο εξελίσσεται για 63s, άρα ο χρόνος που θα χρησιμοποιηθεί για την απεικόνιση της δύναμης είναι $T-t_0$=60s, το οποίο μεταφράζεται στο χρονικό διακριτό βήμα 1949.

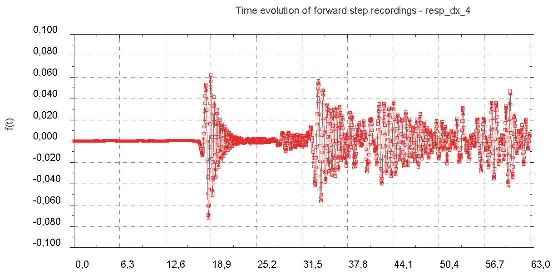
Στα *Σχήματα 5.3* απεικονίζεται η Ευκλίδεια νόρμα των μετατοπίσεων για καθένα από τους αισθητήρες με διεύθυνση καταγραφής την οριζόντια (ΒΕ). Από τα *Σχήματα 5.3α′, 5.3γ′, 5.3δ′* και *5.3ϛ′* συμπεραίνουμε πως η δύναμη διέγερσης προέρχεται από τον δεύτερο κύριο κόμβο (id=2) του κάτω επιπέδου της πλαισιακής κατασκευής, ενώ στα *Σχήματα 5.3β′* και *5.3ϵ′* παρατηρούμε ότι η τοποθεσία της πηγής φαίνεται να είναι όχι μόνο ο κόμβος με id=2, αλλά και ο κόμβος με id=4. Δηλαδή, θα μπορούμε να ισχυριστούμε πως εμφανίζεται ένα δευτερεύον συμμετρικό σήμα (ghost) στον κόμβο 4, το οποίο μας δυσκολεύει στην αναγνώριση της πηγής διέγερσης.



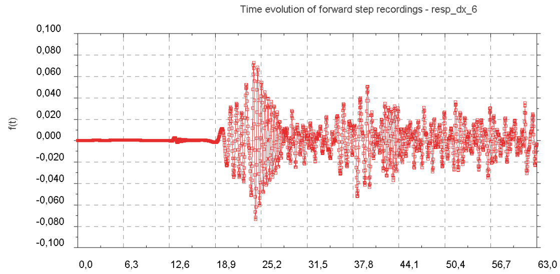
(α′) Αισθητήρας x_a=2



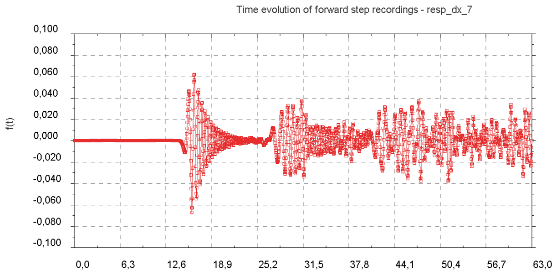
(β′) Αισθητήρας x_a=3



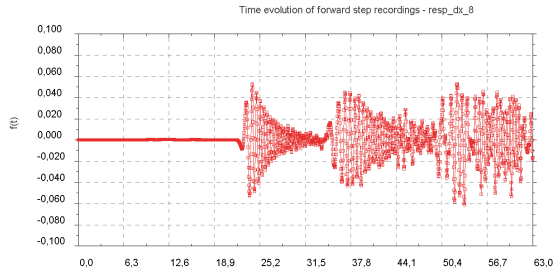
(γ′) Αισθητήρας x_a=4



(δ′) Αισθητήρας x_a=6



(ε′) Αισθητήρας x_a=7



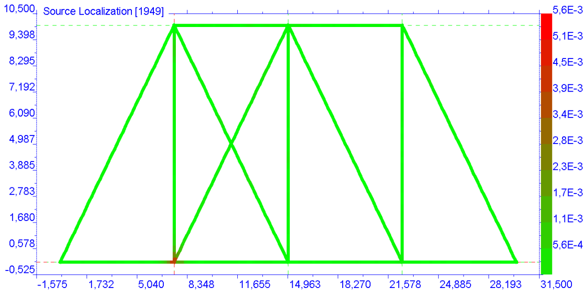
(ϛ′) Αισθητήρας x_a=8

Σχήμα 5.2: Απεικόνιση των ταχυτήτων κατά την οριζόντια διεύθυνση συναρτήσει του χρόνου όπως καταγράφηκαν από τους αισθητήρες με id ίσο με 2, 3, 4, 6, 7, 8. Μονάδες μέτρησης: άξονας y $[m/s]$, άξονας x $[s]$.

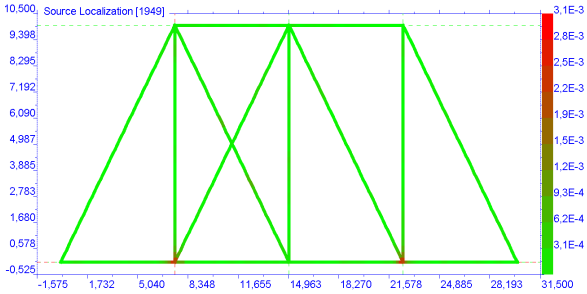
Συνεχίζοντας, τα *Σχήματα* 5.4 παρουσιάζουν με το βαθύ κόκκινο χρώμα την τοποθεσία του σημείου διέγερσης χρησιμοποιώντας τις κατακόρυφες αποκρίσεις. Παρατηρείται ότι εμφανίζεται το ίδιο δευτερεύον συμμετρικό σήμα στα *Σχήματα* 5.4β′ και 5.4ϵ′ όπως είδαμε και στα παραπάνω σχήματα. Συνεπώς, αν τοποθετηθεί αισθητήρας είτε οριζόντιας είτε κατακόρυφης διεύθυνσης καταγραφής στους βασικούς κόμβους 3 και 7, δεν μπορούμε να συμπεράνουμε με ευκολία και ακρίβεια την τοποθεσία εισόδου του παλμού Ricker.

Στο *Σχήμα* 5.5α′ χρησιμοποιήθηκαν 12 αισθητήρες, οριζόντιας και κατακόρυφης διεύθυνσης, με σκοπό να απεικονιστεί με ακρίβεια το σημείο όπου εφαρμόζεται η δύναμη διέγερσης χρησιμοποιώντας ως μεταβλητή απεικόνισης την Ευκλίδεια νόρμα (βλ. *Εξ.* 2.14) των μετακινήσεων. Παρατηρούμε ότι στον κόμβο 2 η τιμή του σήματος ισούται με 0.021, το οποίο επιβεβαιώνεται και στο *Σχήμα* 5.5β′. Εδώ, θα πρέπει να σημειωθεί πως το τελευταίο σχήμα δείχνει την εξέλιξη των μέγιστων τιμών της Ευκλείδιας νόρμας σε κάθε χρονικό διακριτό βήμα.

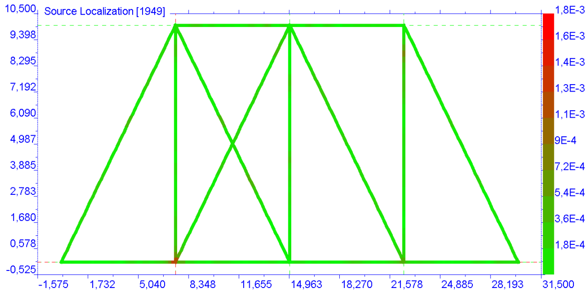
Για την απεικόνιση του σημείου της πηγής διέγερσης χρησιμοποιήθηκαν, επίσης, και οι αποκρίσεις των ταχυτήτων, όπως αυτές καταγράφηκαν στέλνοντας κάθε φορά τις ανεστραμμένες τιμές από κάποιο οριζόντιο αισθητήρα. Στα *Σχήματα* 5.6 διακρίνεται η πηγή έχοντας χρησιμοποιήσει ως μεταβλητή απεικόνισης την Ευκλείδια νόρμα των ταχυτήτων. Ομοίως, και



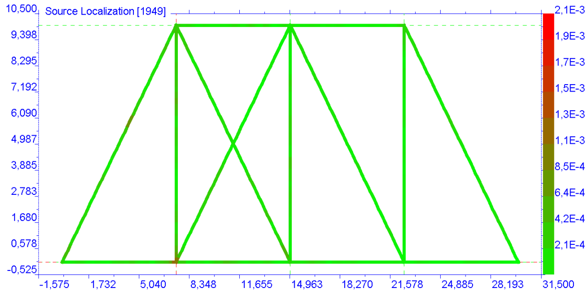
(α′) Αισθητήρας x_a=2



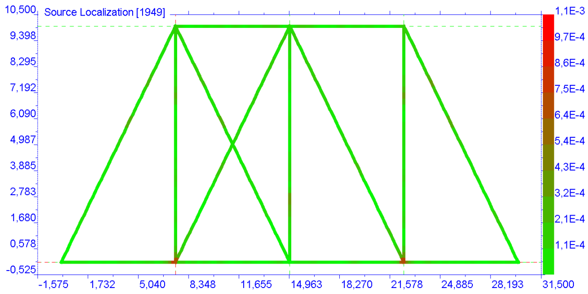
(β′) Αισθητήρας x_a=3



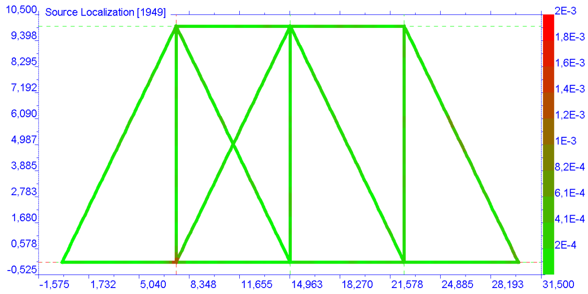
(γ′) Αισθητήρας x_a=4



(δ′) Αισθητήρας x_a=6



(ε′) Αισθητήρας x_a=7



(ϛ′) Αισθητήρας x_a=8

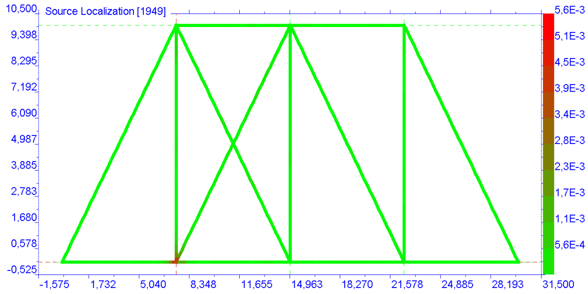
Σχήμα 5.3: Εντοπισμός πηγής διέγερσης απεικονίζοντας την Ευκλίδεια νόρμα των μετατοπίσεων από κάθε οριζόντιας διεύθυνσης καταγραφής αισθητήρα. Μονάδες μέτρησης: άξονας y [m], άξονας x [m], χρωματική κλίμακα [m].

εδώ βλέπουμε ότι αν χρησιμοποιήσουμε τους αισθητήρες 3 και 7, εμφανίζεται το δευτερεύον συμμετρικό σήμα στον κόμβο 4, ενώ στα *Σχήματα* 5.6α′, 5.6γ′, 5.6δ′ και 5.6ϛ′ φαίνεται αναμφίβολα ο κόμβος εφαρμογής της δύναμης.

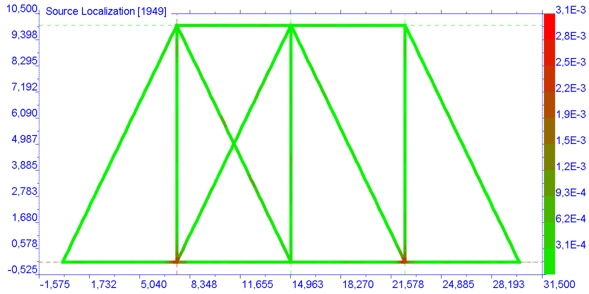
Επιπλέον, στο *Σχήμα* 5.7α′ δίνεται η Ευκλείδια νόρμα των ταχυτήτων χρησιμοποιώντας και τους 12 αισθητήρες. Η μέγιστη τιμή της νόρμας φαίνεται στο *Σχήμα* 5.7β′ τη χρονική διακριτή στιγμή 1954. Συνεπώς, με τη χρήση μεγάλου πλήθους αισθητήρων έχουμε καλύτερα και πιο ευκρινή αποτελέσματα για το πρόβλημα εντοπισμού της πηγής διέγερσης. Ακόμα, αξίζει να σημειωθεί ότι η χρονική διακριτή στιγμή απεικόνισης που χρησιμοποιήθηκε στα σχήματα των ταχυτήτων διαφέρει κατά 5 χρονικά διακριτά βήματα με αυτή των μετατοπίσεων.

5.2 Εντοπισμός αλλοίωσης/φθοράς στη δισδιάστατη περίπτωση (Defect identification)

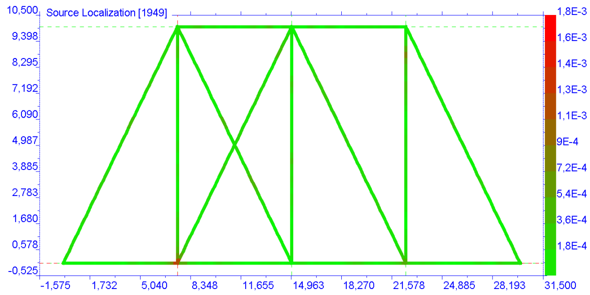
Το πρόβλημα εντοπισμού κάποιας φθοράς ή αλλοίωσης στην δισδιάστατη κατασκευή ακολουθεί τη λογική που αναλύθηκε στην μονοδιάστατη περίπτωση της δοκού. Όμως, μια



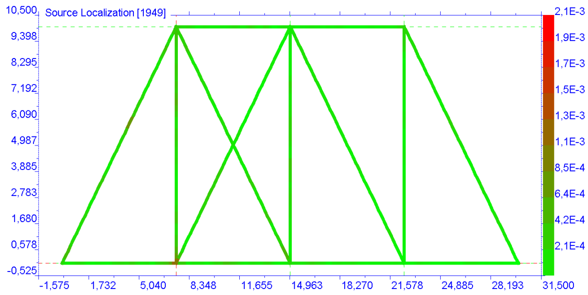
(α′) Αισθητήρας x_a=2



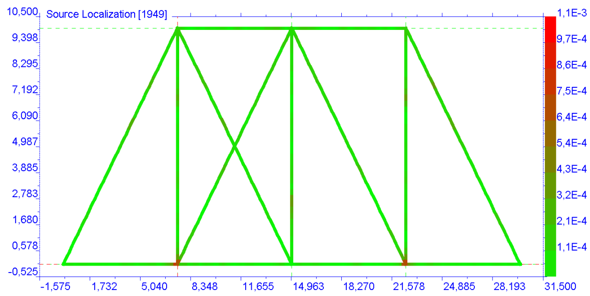
(β′) Αισθητήρας x_a=3



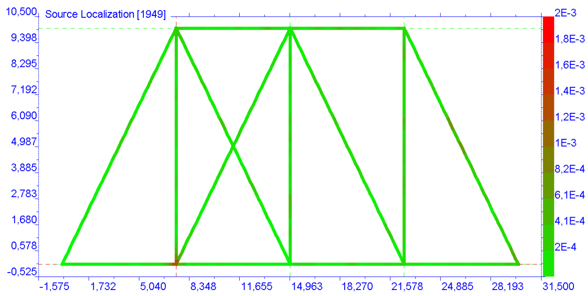
(γ′) Αισθητήρας x_a=4



(δ′) Αισθητήρας x_a=6

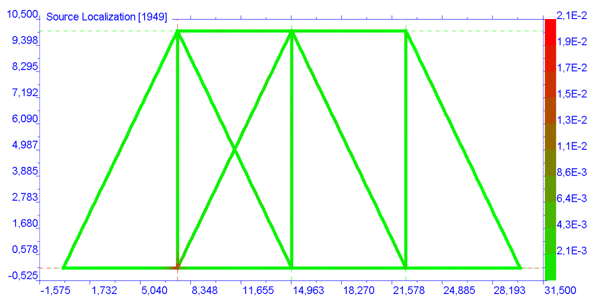


(ε′) Αισθητήρας x_a=7

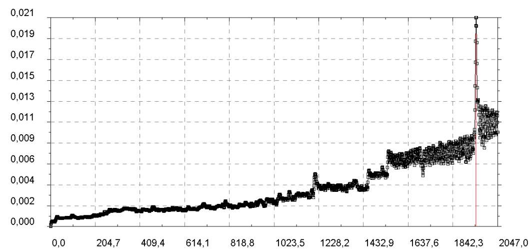


(ϛ′) Αισθητήρας x_a=8

Σχήμα 5.4: Εντοπισμός πηγής διέγερσης απεικονίζοντας την Ευκλίδεια νόρμα των μετατοπίσεων από κάθε κατακόρυφης διεύθυνσης καταγραφής αισθητήρα. Μονάδες μέτρησης: άξονας y [m], άξονας x [m], χρωματική κλίμακα [m].



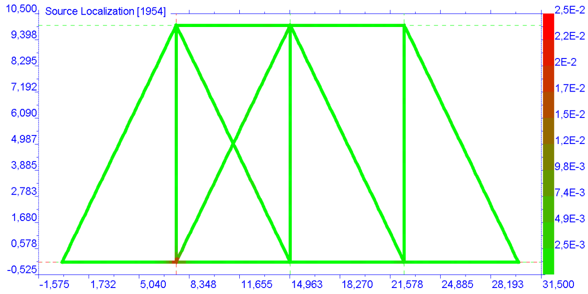
(α′)



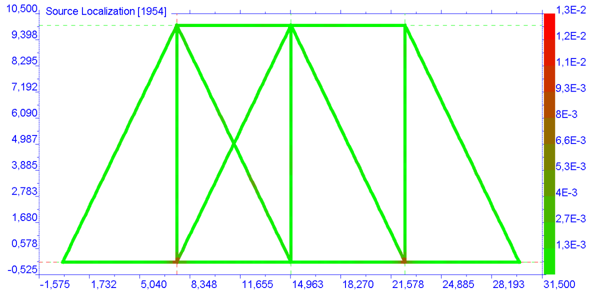
(β′)

Σχήμα 5.5: (α′) Εντοπισμός πηγής διέγερσης απεικονίζοντας την Ευκλίδεια νόρμα των μετατοπίσεων από 12 αισθητήρες και (β′) η χρονική εξέλιξη της μέγιστης τιμής της επιλεγμένης μεταβλητής.

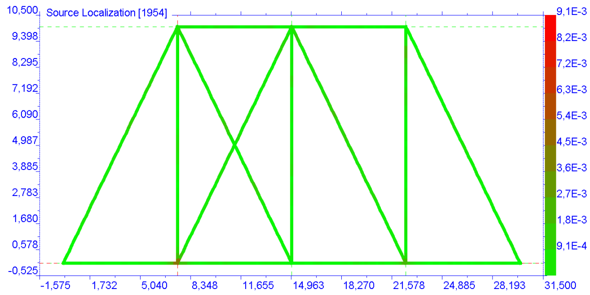
σημαντική διαφορά είναι ο υπολογισμός του χρονικού διακριτού βήματος που κάποιο κύμα χρειάζεται για να διανύσει την απόσταση από την πηγή διέγερσης έως τον κοντινότερο κόμβο της περιοχής αλλοίωσης. Η μελέτη για ανίχνευση κάποιας φθοράς πάνω στην κατασκευή



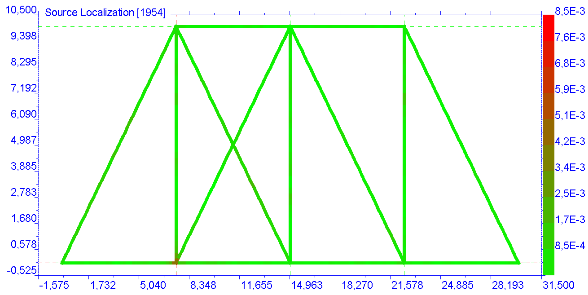
(α′) Αισθητήρας x_a=2



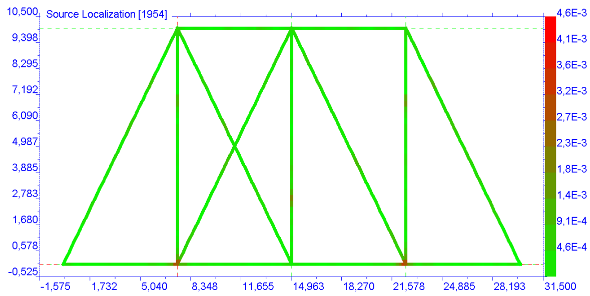
(β′) Αισθητήρας x_a=3



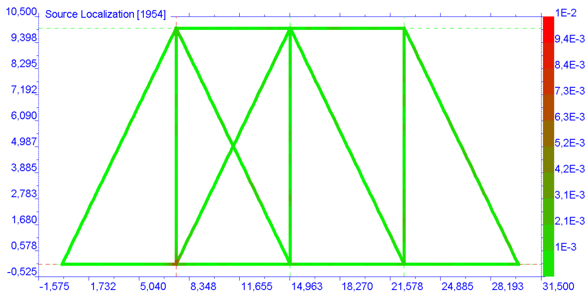
(γ′) Αισθητήρας x_a=4



(δ′) Αισθητήρας x_a=6

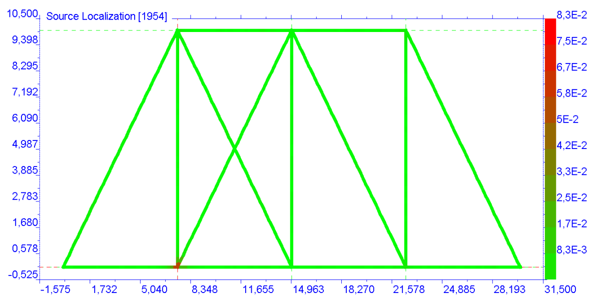


(ε′) Αισθητήρας x_a=7

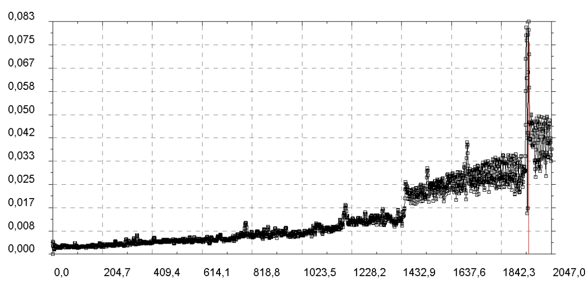


(ϛ′) Αισθητήρας x_a=8

Σχήμα 5.6: Εντοπισμός πηγής διέγερσης απεικονίζοντας την Ευκλίδεια νόρμα των ταχυτήτων από κάθε οριζόντιας διεύθυνσης καταγραφής αισθητήρα. Μονάδες μέτρησης: άξονας y $[m]$, άξονας x $[m]$, χρωματική κλίμακα $[m/s]$.



(α′)



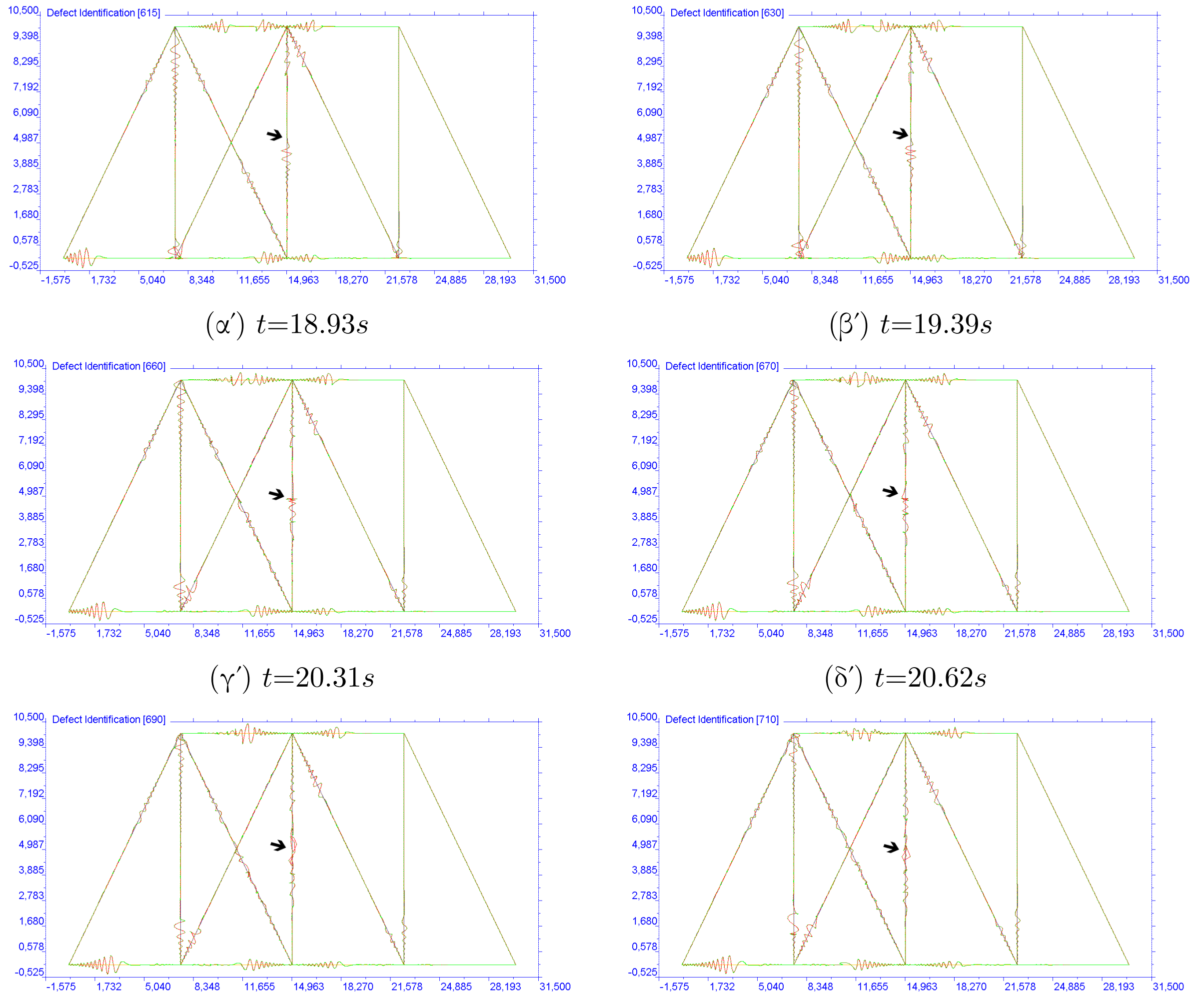
(β′)

Σχήμα 5.7: (α′) Εντοπισμός πηγής διέγερσης απεικονίζοντας την Ευκλίδεια νόρμα των ταχυτήτων από 12 αισθητήρες και (β′) η χρονική εξέλιξη της μέγιστης τιμής της επιλεγμένης μεταβλητής.

παρουσιάζεται σχηματικά και αναλύεται στις *Υποενότητες* 5.2.1 και 5.2.2.

5.2.1 Αποτελέσματα επίλυσης του ευθέος προβλήματος (Forward step)

Στην παρούσα υποενότητα παρουσιάζονται τα σχήματα που δείχνουν προσεγγιστικά την χρονική στιγμή που το σήμα της πηγής φτάνει στο πρώτο σύνορο της αλλοιωμένης περιοχής. Παρόλο που στην *Ενότητα 2.5* δίνεται με ακρίβεια το χρονικό διακριτό βήμα στο οποίο εντοπίζονται τα στοιχεία που έχουν υποστεί φθορά, αξίζει να το εντοπίσουμε και σχηματικά παρατηρώντας τα αξονικά και καμπτικά κύματα. Στο *Σχήμα 5.8α΄*, το μαύρο βέλος δείχνει την αλλοιωμένη περιοχή και διακρίνεται η άφιξη του ταχύτερου κύματος, το οποίο πέρασε πρώτα από τον κύριο κόμβο με id=3. Στη συνέχεια, το κύμα συνεχίζει την πορεία του κατά μήκος της κάθετης δοκού και όπως παρατηρούμε τη χρονική στιγμή t=21.24s (βλ. *Σχήμα 5.8ϵ*) ξεκινά η δημιουργία νέων κυμάτων προς όλες τις κατευθύνσεις, ενώ μερικά κλάσματα του δευτερολέπτου αργότερα το διερχόμενο κύμα από τον κόμβο 7 φτάνει στο άνω σύνορο της περιοχής με τη φθορά. Άρα, μέσω των συνεχόμενων στιγμιοτύπων μπορεί να γίνει μια προσέγγιση της χρονικής στιγμής άφιξης του κύματος στην αλλοιωμένη περιοχή.



(α΄) t=18.93s (β΄) t=19.39s

(γ΄) t=20.31s (δ΄) t=20.62s

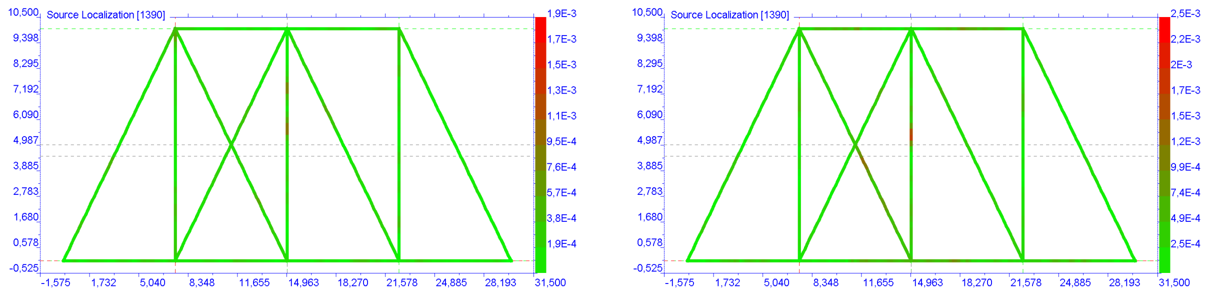
(ε΄) t=21.24s (ϛ΄) t=21.85s

Σχήμα 5.8: Διάδοση κύματος στην αλλοιωμένη δισδιάστατη κατασκευή κατά την ευθεία επίλυση. Μονάδες μέτρησης: άξονας y $[m]$, άξονας x $[m]$. Η παρακολούθηση σχετικού βίντεο μπορεί να γίνει στο ακόλουθο link.

5.2.2 Αποτελέσματα επίλυσης του αντίστροφου προβλήματος (Backward step)

Κατά την επίλυση του αντίστροφου προβλήματος στάλθηκαν οι χρονικά ανεστραμμένες αποκρίσεις από καθένα αισθητήρα. Συνολικά χρησιμοποιήθηκαν 12 αισθητήρες, 6 οριζόντιας καταγραφής και 6 κατακόρυφης. Στο *Σχήμα 5.9α΄* απεικονίζεται το αποτέλεσμα της Ευκλείδιας νόρμας (βλ. *Εξ. 2.14*) των μετακινήσεων χρησιμοποιώντας μόνο τους οριζόντιους αισθητήρες, με αποτέλεσμα να εμφανίζεται ένα ασθενές σήμα πολύ κοντά στην περιοχή αλλοίωσης. Το ίδιο συμβαίνει και στο επόμενο σχήμα (*Σχήμα 5.9β΄*), με τη διαφορά οτι λαμβάνονται υπόψη μόνο οι αισθητήρες κατακόρυφης διεύθυνσης καταγραφής. Και στις δύο περιπτώσεις το σήμα είναι σχεδόν στο ίδιο σημείο και οι τιμές τους έχουν πολύ μικρή διαφορά, με υψηλότερη αυτή του *Σχήματος 5.9β΄*.

Στο *Σχήμα 5.10α΄* μπορούμε να παρατηρήσουμε ότι χρησιμοποιώντας την Ευκλείδια νόρμα από όλους τους αισθητήρες, ένα έντονο κόκκινο χρώμα υπάρχει στο άνω σύνορο της αλλοιωμένης περιοχής. Αυτό σημαίνει ότι κάνοντας χρήση της συγκεκριμένης μεταβλητής απεικόνισης πιθανόν να μην μπορούμε να εντοπίσουμε με απόλυτη ακρίβεια την αλλοιωμένη περιοχή. Επίσης, για να αντιληφθούμε καλύτερα την τιμή του κόκκινου σήματος στο προηγούμενο σχήμα, μπορούμε να βασιστούμε στο *Σχήμα 5.10β΄*, το οποίο δείχνει την χρονική εξέλιξη των μέγιστων τιμών της επιλεγμένης μεταβλητής απεικόνισης. Η κάθετη κόκκινη γραμμή ορίζει την διακριτή χρονική στιγμή που χρησιμοποιήσαμε στην απεικόνιση και η τιμή της Ευκλίδειας νόρμας είναι περίπου $0,0025$.



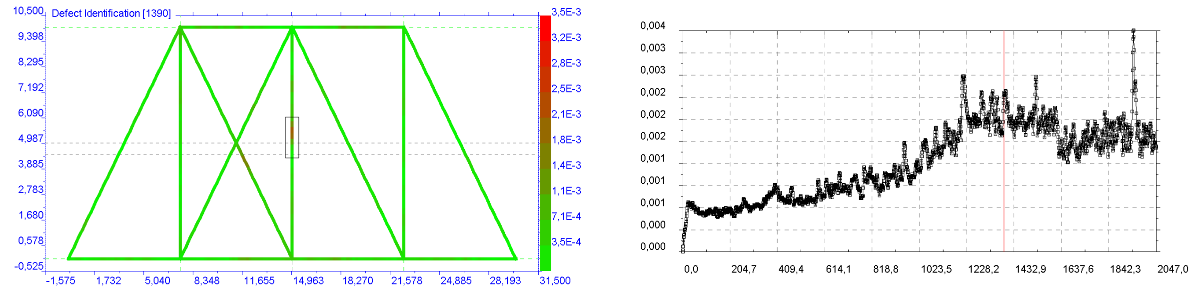
(α΄) (β΄)

Σχήμα 5.9: Απεικόνιση της Ευκλίδειας νόρμας των μετατοπίσεων λαμβάνοντας υπόψη τον συνολικό αριθμό (α΄) των οριζόντιων και (β΄) των κατακόρυφων αισθητήρων. Οι γκρι γραμμές οριοθετούν την αλλοιωμένη περιοχή. Μονάδες μέτρησης: άξονας y $[m]$, άξονας x $[m]$, χρωματική κλίμακα $[m/s]$.

5.3 Μελέτη του λόγου σήμα-προς-θόρυβο (Signal to Noise Ratio)

Στην παρούσα ενότητα παρουσιάζονται τα αποτελέσματα της μελέτης του λόγου σήμα-προς-θόρυβο για τη δισδιάστατη κατασκευή. Όπως αναφέρθηκε στην *Ενότητα 4.3*, στόχος της συγκεκριμένης μελέτης είναι η εύρεση του βέλτιστου αριθμού των αισθητήρων που θα πρέπει να χρησιμοποιηθούν στην κατασκευή ώστε να εξαλειφθούν όσο το δυνατόν περισσότερο τα δευτερεύοντα κύματα (ghosts).

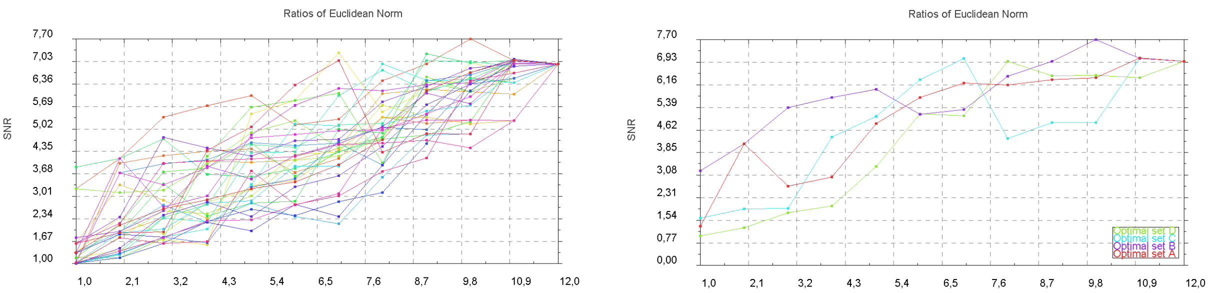
Τα *Σχήματα 5.11α΄* και *5.12α΄* απεικονίζουν τη συμπεριφορά του λόγου σήμα-προς-θόρυβο για τις αποκρίσεις των μετατοπίσεων χωρίς θόρυβο και με θόρυβο, αντίστοιχα. Επίσης,



(α′) (β′)

Σχήμα 5.10: (α′) Απεικόνιση της Ευκλίδειας νόρμας των μετατοπίσεων λαμβάνοντας υπόψη τον συνολικό αριθμό των αισθητήρων και (β′) η χρονική εξέλιξη της μέγιστης τιμής της επιλεγμένης μεταβλητής απεικόνισης. Οι γκρι γραμμές οριοθετούν την αλλοιωμένη περιοχή.

λήφθηκαν υπόψη 31 τυχαίες διατάξεις αισθητήρων και στις δύο περιπτώσεις και ο μέγιστος αριθμός κάθε σετ ήταν 12 αισθητήρες. Μια σημαντικη παρατήρηση που μπορεί να γίνει είναι ότι ο λόγος προσεγγίζει μία γραμμική συμπεριφορά με την διαδοχική αύξηση των αισθητήρων και τελικά, στην περίπτωση των 12 αισθητήρων ο λόγος έχει ακριβώς την ίδια τιμή για όλες τις διατάξεις.

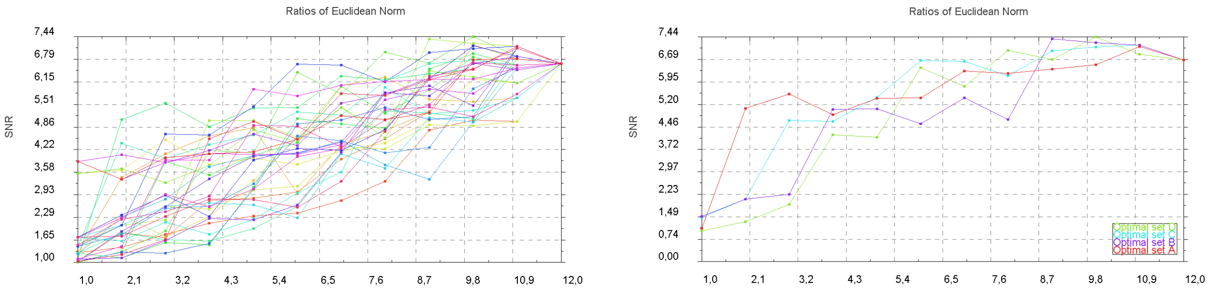


(α′) (β′)

Σχήμα 5.11: (α′) SNR για 31 τυχαία σετ αισθητήρων. (β′) Τα βέλτιστα σετ αισθητήρων, το A είναι για 2, το B είναι για 4 και 10, το C είναι για 6 και το D είναι για 8 αισθητήρες.

Από τα σχήματα που σχολιάστηκαν παραπάνω, μπορούμε να εντοπίσουμε τα βέλτιστα σετ για οποιοδήποτε πλήθος αισθητήρων που μελετήθηκαν. Είναι προφανές πως για 12 αισθητήρες δεν μπορεί να βγει κάποιο συμπέρασμα μιας και η τιμή του λόγου είναι ίδια για όλες τις διατάξεις. Έτσι, επιλέγουμε να εντοπίσουμε τα κατάλληλα σετ για τις δοατάξεις των 2, 4, 6, 8 και 10 αισθητήρων. Στα *Σχήματα* 5.11β′ και 5.12β′ φαίνονται οι διατάξεις που μπορούν να χρησιμοποιηθούν σε μια κατασκευή ώστε να ληφθούν οι πιο αξιόπιστες αποκρίσεις. Επιπλέον, μπορούμε να παρατηρήσουμε πως κάποια σετ είναι ίδια. Για παράδειγμα, στο *Σχήμα* 5.11β′ το σετ B είναι κοινό για 4 και για 10 αισθητήρες. Το ίδιο συμβαίνει όταν προστεθεί Γκαουσιανός θόρυβος στις τιμές, αλλά για 8 και 10 αισθητήρες (βλ. *Σχήμα* 5.12β′).

Επιπρόσθετα, ο *Πίνακας* 5.1 μας βοηθάει να κατανοήσουμε καλύτερα τα σχήματα που σχολιάστηκαν πιο πάνω, δίνοντας με ακρίβεια την θέση και τη διεύθυνση (ΒΕ) που θα πρέπει να τοποθετηθούν οι αισθητήρες των βέλτιστων σετ. Δηλαδή, αν θέλουμε να καταγράψουμε τις καθαρές τιμές της δισδιάστατης κατασκευής και υπάρχει η δυνατότητα για χρήση 4 αισθητήρων, τότε θα πρέπει να τοποθετήσουμε τρεις κατακόρυφους στους κόμβους με id ίσο με 2, 3 και 8 και έναν οριζόντιο στον κόμβο 8. Επίσης, η ίδια διάταξη θα χρησιμοποιηθεί και για



(α′) (β′)

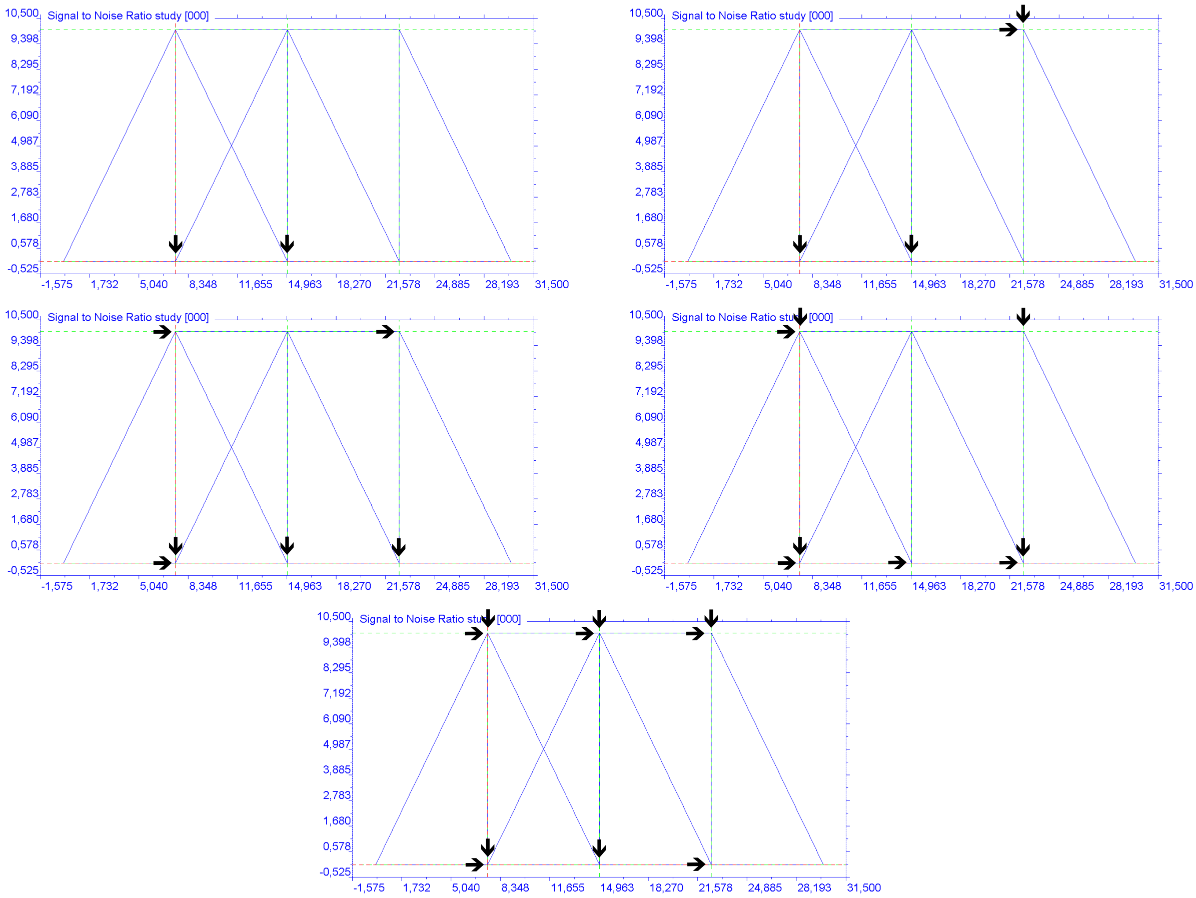
Σχήμα 5.12: (α′) SNR για 31 τυχαία σετ αισθητήρων με την προσθήκη θορύβου. (β′) Τα βέλτιστα σετ αισθητήρων, το A είναι για 2, το B είναι για 4, το C είναι για 6 και το D είναι για 8 και 10 αισθητήρες.

10 αισθητήρες μόνο που προστίθενται ακόμα έξι αισθητήρες. Πιο συγκεκριμένα, οι επιπλεόν αισθητήρες θα είναι οι $x_7, y_6, x_4, x_2, y_7, x_6$.

Στο *Σχήμα 5.13* απεικονίζονται οι θέσεις και οι διευθύνσεις καταγραφής (ΒΕ) των αισθητήρων για την βέλτιστη καταγραφή αποτελεσμάτων εν απουσία Γκαουσιανού θορύβου. Ουσιαστικά, έχουν σχεδιαστεί τα αποτελέσματα της στήλης «Αποκρίσεις χωρίς θόρυβο» του *Πίνακα 5.1*. Επίσης, τα βέλη κατά μήκος του άξονα x δηλώνουν την οριζόντια διεύθυνση κάποιου αισθητήρα, ενώ τα βέλη κατά τον y αφορούν την κατακόρυφη διεύθυνση.

	Αποκρίσεις χωρίς θόρυβο					Αποκρίσεις με θόρυβο				
Αισθ./Βελ. σετ	*A*	*B*	*C*	*D*	*B*	*A*	*B*	*C*	*D*	*D*
2	y_3, y_2					y_8, y_2				
4	y_3, y_2	y_8, x_8				x_8, y_6	y_8, y_2			
6	x_8, y_4	y_3, y_2	x_6, x_2			y_6, x_8	x_2, x_7	y_2, y_8		
8	x_3, y_4	x_6, y_8	x_4, y_2	y_6, x_2		x_3, y_4	y_7, x_2	y_8, y_2	x_8, y_6	
10	y_3, y_2	y_8, x_8	x_7, y_6	x_4, x_2	y_7, x_6	x_3, y_4	y_7, x_2	y_8, y_2	x_8, y_6	x_6, y_3

Πίνακας 5.1: Βέλτιστες διατάξεις αισθητήρων.



Σχήμα 5.13: Βέλτιστη λύση για τις θέσεις και τις κατευθύνσεις καταγραφής (BE) για διάφορα πλήθη αισθητήρων. Μονάδες μέτρησης: άξονας y $[m]$, άξονας x $[m]$.

Κεφάλαιο 6

Συμπεράσματα και πιθανές προεκτάσεις

Συγκεντρωτικά, θα μπορούσαμε να πούμε πως στη συγκεκριμένη διπλωματική εργασία αναλύθηκαν θέματα δυναμικής των κατασκευών και εφαρμόστηκαν μέθοδοι αριθμητικής ολοκλήρωσης τόσο στη μονοδιάστατη δοκό όσο και σε μια δισδιάστατη πλαισιακή κατασκευή.

Τα δυο βασικά ζητήματα που μελετήθηκαν, αφορούν τον εντοπισμό της πηγής διέγερσης και την ανίχνευση κάποιας φθοράς/αλλοίωσης στην κατασκευή. Και στα δυο, καταλήξαμε σε αξιόλογα συμπεράσματα με τα οποία μπορεί κανείς να πειραματιστεί περισσότερο. Ακόμη, η προσθήκη μικρού ποσοστού Γκαουσιανού θορύβου στις τιμές, δεν επηρέασε αισθητά τον εντοπισμό της δύναμης. Τέλος, το τρίτο ενδιαφέρον κομμάτι της εργασίας ήταν η μελέτη του λόγου σήμα-προς-θόρυβο (SNR) στο πρόβλημα εντοπισμού της διεγείρουσας δύναμης. Η μελέτη των δυο περιπτώσεων -της δοκού και της πλαισιακής κατασκευής- κατέληξαν σε αποτελέσματα, τα οποία μπορούν να βοηθήσουν στην βέλτιστη τοποθέτηση των αισθητήρων πάνω σε μια κατασκευή.

Μια πιθανή και αρκετά ενδιαφέρουσα προέκταση της συγκεκριμένης εργασίας είναι η μελέτη του λόγου σήμα-προς-θόρυβο (SNR) για την περίπτωση της αλλοιωμένης κατασκευής. Θεωρούμε ότι θα είχε μεγάλο ενδιαφέρον να αναζητήσουμε τη βέλτιστη διάταξη αισθητήρων σε μια κατασκευή με στόχο την καλύτερη και πιο ευκρινή απεικόνιση των αποτελεσμάτων στο συγκεκριμένο πρόβλημα.

Τέλος, ολόκληρη η εργασία και το οπτικό υλικό της (σχήματα, βίντεο, κώδικες) διατίθενται ελεύθερα στο διαδίκτυο στον ακόλουθο σύνδεσμο:
https://gitlab.com/Mavrikis/computational-study-of-imaging-techniques-based-on-elastic-wave-reversibility.

Παράρτημα Α΄

Κώδικες επίλυσης

Listing A.1: CantileverTransientForwardStep

```
//Forward Step

//------------------Path------------------
//Please define the folder's path where user will save the results
//i.e "C:\\Users\\pshr\\Desktop\\MARIOS\\FinalCodesAndResults_28-6-18\\FS_RESULTS\\FS_1950_4095\\"
path="Type here..."
//----------------------------------------------------

import jfem.*;
theUniverse.cls()
aDomain=theUniverse.FEMDomain();

// geometric data
L = 30.0
Nx = 1950

// material data
Elast=1    //[Pa]
dens=1    //[kg/m^3]
Poisson=0.25 //Poisson
wave_prop_vel=sqrt(Elast/dens)
dx=L/(Nx-1); //distance between nodes

// temporal data
init = 3 //initial time
Tot = init + 2*L/wave_prop_vel //total time
Nt=2**12-1  //discrete time steps (is defined by user)
dt=Tot/Nt as double

//-------------just to check----------
dtd=dx/wave_prop_vel
println("dt/dtd ="+dt/dtd+" (?)<= 1.0")
println("suggested Nx<= "+(L/(dt*wave_prop_vel)))
println ("number of discrete time steps = "+Nt+", while total time =  "+Tot)
println("times that the input signal will pass the whole medium "+(Tot/(wave_prop_vel*L)))
//------------------------------------------------

// sources at value*L
sources=[]
sourcesDOF=[]
```

```
loc_of_Load = 0.75
sources.add(loc_of_Load)
sourcesDOF.add(1) //axial
sourcesDOF.add(2) //transverse

// receivers at value*L
Nrec=3
receivers=[]
(1..Nrec).each{
  receivers.add(it/Nrec)
}


//define material
aMat= new ElasticMaterial(1,Elast,Poisson)
aMat.setDensity(dens)

A=1
As=5*A/6
Imi=1.0*1.0**3/12.0
aSect= new CrossSection(1, A, As, Imi)



file_data = new File(path+"dataReadme.txt")
file_data.createNewFile()
file_data.text = ""

file_data.append "Nx = " <<Nx+"\n"
file_data.append "discrete time steps = " <<Nt+"\n"
file_data.append "L = " <<L+"\n"
file_data.append "init = " <<init+"\n"
file_data.append "Total time = " <<Tot+"\n"
file_data.append "The location of Load is " <<loc_of_Load+" of L \n"
file_data.append "Elasticity = " <<Elast+"\n"
file_data.append "density = " <<dens+"\n"
file_data.append "Poisson = " <<Poisson+"\n"
file_data.append "wave_prop_velocity = " <<wave_prop_vel+"\n"
file_data.append "distance between nodes(dx) = " <<dx+"\n"
file_data.append "dt = " <<dt+"\n"
file_data.append "Total number of receivers = " <<Nrec+"\n"
file_data.append "A = " <<A+"\n"
file_data.append "As = " <<As+"\n"
file_data.append "Imi = " <<Imi+"\n"
(0..<receivers.size()).each{
  file_data.append "the location of receiver "+(it+1)+" is "+receivers[it]+
    " of L \n"
}
//-----------------------------------------------------------------------

// Locate nodes
id=0
(1..Nx).each{
  aNode = new Node(++id,it*L/Nx)
  aDomain.putNode(aNode)
}


// elements
xcord=[]
```

```
ycord=[]
(1..(Nx-1)).each{
  elem =new EBeam2d(it , aDomain.getNode(it),aDomain.getNode(it+1),aMat,
    aSect)
  aDomain.putElement(elem)
}
aDomain.activateConsMass()

//----Plotting the location of the Load-----
LoadPlot= new PlotFrame()
lf=[]
tf=[]
//-------------------------------------------
id=0
(0..<sources.size).each{
  LC=new LoadCase(++id,Nt)
  amplit = 1000
  sigma = 1.5
  coords = new double[2]; coords[1]=0.0; coords[0]=sources[it]*L
  dof=sourcesDOF[it]
  nid=aDomain.find(coords)
  println "the location of Load is the node: "+nid
  file_data.append "the location of Load is the node: " <<nid+"\n"

  jj=it //Plotting the location of the Load
  (0..Nt).each{
    val = amplit*(1.0-2.0*PI*PI*sigma*sigma*(it*dt-init)*(it*dt-init))*exp
    (-PI*PI*sigma*sigma*(it*dt-init)*(it*dt-init));
    aLoad = new Load(val, nid, 1, it)
    LC.putLoad(aLoad)
    aLoad = new Load(val, nid, 2, it)
    LC.putLoad(aLoad)
    //---------Plotting the location of the Load-----
    if(jj==0){
      lf.add(val)
      tf.add(dt*it)
    }
    //-------------------------------------------
  }
  aDomain.putLoadCase(LC)
}
//---------Plotting the location of the Load-----
LoadPlot.addFunction(new plotfunction(tf as double[], lf as double[]))
LoadPlot.show()
//-------------------------------------------


// boundary conditions
mc=aDomain.getMaxKMcomponent();
Ak=10.0e20; Am=Ak*mc[1]/mc[0]
id=0

aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 1)
aCE.setAk(Ak); aCE.setAm(Am)
aDomain.putConstraintElement(aCE)
aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 2)
aCE.setAk(Ak); aCE.setAm(Am)
aDomain.putConstraintElement(aCE)
aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 6)
aCE.setAk(Ak); aCE.setAm(Am)
```

```
aDomain.putConstraintElement(aCE)

theAnalysis = new BetaNewmarkAnalysis(aDomain, dt, 0.25, 0.5)
//if(damping)theAnalysis.setRayleighDamping(a0, a1)
theAnalysis.analyse();
theAnalysis.endAnalysis();


//————————————————saving results————————————
tsteps=(int)Math.floor(Tot/dt)
println("the tsteps are: "+tsteps)

//———————saving values of the displacements and velocities for chosen DOFs
    ——————————
receivers.each{
  coords = new double[2]
  coords[0]=it*L
  coords[1]=0.0
  nid=aDomain.find(coords)
  println("receiver node with id = "+nid+" is located at the "+it+" of L")
  file_data.append "receivers_id = " <<nid+"\n" //it's for "dataReadme.txt"
     file


  fback_x=new double[tsteps+1]
  fback_dx=new double[tsteps+1]
  fback_y=new double[tsteps+1]
  fback_dy=new double[tsteps+1]
  fback_rz=new double[tsteps+1]
  fback_drz=new double[tsteps+1]

  file1 = new File(path+"resp_x_"+nid+".txt")
  file2 = new File(path+"resp_dx_"+nid+".txt")
  file3 = new File(path+"resp_y_"+nid+".txt")
  file4 = new File(path+"resp_dy_"+nid+".txt")
  file5 = new File(path+"resp_rz_"+nid+".txt")
  file6 = new File(path+"resp_drz_"+nid+".txt")
  file1.text = ""
  file2.text = ""
  file3.text = ""
  file4.text = ""
  file5.text = ""
  file6.text = ""

  for(int i=0;i<=tsteps;i++){
    file1<< aDomain.getNode(nid).getLoadCaseDisps(1, i)[0]+"\n"
    file2<< aDomain.getNode(nid).getLoadCaseVelcs(1, i)[0]+"\n"
    file3<< aDomain.getNode(nid).getLoadCaseDisps(1, i)[1]+"\n"
    file4<< aDomain.getNode(nid).getLoadCaseVelcs(1, i)[1]+"\n"
    file5<< aDomain.getNode(nid).getLoadCaseDisps(1, i)[5]+"\n"
    file6<< aDomain.getNode(nid).getLoadCaseVelcs(1, i)[5]+"\n"
  }
}



ex_time=(int)init/dt
DisTimeStepOfExcitation=Nt-ex_time
println("excitation time : "+ex_time+" corresponding to the "+
    DisTimeStepOfExcitation+" discrete time step"+"\n")
```

```
file_data.append "excitation time = " <<ex_time+"\n"
file_data.append "discrete time step of excitation = " <<
    DisTimeStepOfExcitation+"\n"


// plotting in Graphical Panel
theGP.stop()
scale=0.001
theGP.plotDeform(1,scale,0,tsteps,1)
```

Listing A.2: CantileverTransientBackwardStep

```
//Backward Step

//-------paths-------
//Please define the folder's path where user has saved the "dataReadme.txt"
    file
//i.e "C:\\Users\\pshr\\Desktop\\MARIOS\\FinalCodesAndResults_28-6-18\\
    FS_RESULTS\\FS_1950_4095\\"
path="Type here..."

//Please define the folder's path where user will save the results
//i.e "C:\\Users\\pshr\\Desktop\\MARIOS\\FinalCodesAndResults_28-6-18\\
    BS_RESULTS\\BS_1950_4095\\Input--resp_rz_1950\\"
path1="Type here..."
//-------------------

import jfem.*;
theUniverse.cls()
aDomain=theUniverse.FEMDomain();


//----------------------------------------------
ReceiverIDArray = [1950] //defined by user
//String[] DoFArray = ["resp_x_", "resp_y_", "resp_z_", "resp_rx_", "
    resp_ry_", "resp_rz_"] corresponding to DoF [1, 2, 3, 4, 5, 6]
DoFString = ["resp_x_"] //Displacements (defined by user)

if(ReceiverIDArray.size()!=DoFString.size()){
   println("ReceiverIDArray.size()!=DoFString.size()")
   logger("ReceiverIDArray.size()!=DoFString.size()")
}

println "Input: "+DoFString[0]+ReceiverIDArray[0]+"\n"
//----------------------------------------------


// geometric data
L = 30.0
Nx = 1950  //it should be equal with the value from Forward Step

// material data
Elast=1    //[Pa]
dens=1    //[kg/m^3]
Poisson=0.25 //Poisson
wave_prop_vel=sqrt(Elast/dens)
dx=L/(Nx-1); //distance between nodes


// temporal data
init = 3 //initial time
```

```
Tot = init + 2*L/wave_prop_vel //total time
Nt=2**12-1  //discrete time steps (it should be equal with Nt value from
    Forward Step)
dt=Tot/Nt as double
dtd=dx/wave_prop_vel

//-----------just to check----------
println("dt/dtd ="+dt/dtd+" (?)<= 1.0")
println("suggested Nx<= "+(L/(dt*wave_prop_vel)))
println ("number of discrete time steps = "+Nt+", while total time =  "+Tot
    )
println("times that the input signal will pass the whole medium "+(Tot/(
    wave_prop_vel*L)))
//----------------------------------

// imagers
ima=(int)Nx/5
imagers=[]
(0..ima).each{
  imagers.add(it/ima)
}

// material data
Elast=1    //[Pa]
dens=1    //[kg/m^3]
Poisson=0.25 //Poisson
wave_prop_vel=sqrt(Elast/dens)
dx=L/(Nx-1); //distance between nodes


//-----------save aboved data in file----------
file_data = new File(path1+"ImagersDataReadme.txt")
file_data.createNewFile()
file_data.text = ""

file_data.append "Input = " <<DoFString[0]+ReceiverIDArray[0]+"\n"
file_data.append "Nx = " <<Nx+"\n"
file_data.append "discrete time steps = " <<Nt+"\n"
file_data.append "L = " <<L+"\n"
file_data.append "Total time = " <<Tot+"\n"
file_data.append "dt = " <<dt+"\n"
//---------------------------------------------

//Locate nodes
id=0
(1..Nx).each{
  aNode = new Node(++id,it*L/Nx)
  aDomain.putNode(aNode)
}

//define material
aMat= new ElasticMaterial(1,Elast,Poisson)
aMat.setDensity(dens)

A=1
As=5*A/6
Imi=1.0*1.0**3/12.0
aSect= new CrossSection(1, A, As, Imi)

// elements
```

```
xcord=[]
ycord=[]
(1..(Nx-1)).each{
   elem =new EBeam2d(it , aDomain.getNode(it),aDomain.getNode(it+1),aMat,
     aSect)
   aDomain.putElement(elem)
}
aDomain.activateConsMass()

// boundary conditions
mc=aDomain.getMaxKMcomponent();
Ak=10.0e20; Am=Ak*mc[1]/mc[0]
id=0

aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 1)
aCE.setAk(Ak); aCE.setAm(Am)
aDomain.putConstraintElement(aCE)
aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 2)
aCE.setAk(Ak); aCE.setAm(Am)
aDomain.putConstraintElement(aCE)
aCE = new ConstraintElement(++id,1.0, aDomain.getNode(1), 6)
aCE.setAk(Ak); aCE.setAm(Am)
aDomain.putConstraintElement(aCE)

// process for inverse problem
aLoadCase = new LoadCase(1,Nt);
aDomain.putLoadCase(aLoadCase);

(0..<ReceiverIDArray.size()).each{
   fback=new double[Nt+1]
   Respfile=new File (path+DoFString[it]+ReceiverIDArray[it]+".txt")

   i=0
   Respfile.withReader { reader ->
      while ((line=reader.readLine())!=null) {
         fback[i++]=Double.parseDouble(line)
      }
   }

   dof=-1
   if(DoFString[it].equals("resp_x_"))dof=1
   if(DoFString[it].equals("resp_y_"))dof=2
   if(DoFString[it].equals("resp_rz_"))dof=6

   for(int i=0;i<Nt;i++){
      aLoad =new Load(fback[Nt-i], ReceiverIDArray[it], dof, i);
      aLoadCase.putLoad(aLoad);
   }
}



theAnalysis = new BetaNewmarkAnalysis(aDomain, dt, 0.25, 0.5)
//if(damping)theAnalysis.setRayleighDamping(a0, a1)
theAnalysis.analyse();
theAnalysis.endAnalysis();


//----------saving results----------
tsteps=(int)Math.floor(Tot/dt)
```

```
println("the tsteps are: "+tsteps)


//---------saving values of the displacements and velocities for chosen DOFs---------

DoFStringImagers = ["resp_x_", "resp_y_", "resp_rz_"]
DoFStringVelcsImagers = ["resp_dx_", "resp_dy_", "resp_drz_"]
Integer [] DoFsImagers = [0, 1, 5]

imagers.each{
  coords = new double[2]
  coords[0]=it*L
  coords[1]=0.0
  nid=aDomain.find(coords)
  println("imager node with id = "+nid+" is located at the "+it+" of L")
  file_data.append "imager_id = "<<nid+"\n" //it's for "ImagersDataReadme.txt.txt" file


  for(int k=0;k<DoFStringImagers.size();k++){
    fback1=new double[tsteps+1]
    fback2=new double[tsteps+1]
    file1=new File(path1+DoFStringImagers[k]+nid+".txt")
    file2=new File(path1+DoFStringVelcsImagers[k]+nid+".txt")
    file1.text = ""
    file2.text = ""

    for(int i=0;i<=tsteps;i++){
      file1 << aDomain.getNode(nid).getLoadCaseDisps(1, i)[DoFsImagers[k]]+"\n"
      file2 << aDomain.getNode(nid).getLoadCaseVelcs(1, i)[DoFsImagers[k]]+"\n"
    }
  }

}


ex_time=(int)init/dt
DisTimeStepOfExcitation=Nt-ex_time
println("excitation time : "+ex_time+" corresponding to the "+DisTimeStepOfExcitation+" discrete time step"+"\n")
file_data.append "excitation time = " <<ex_time+"\n"
file_data.append "discrete time step of excitation = " << DisTimeStepOfExcitation+"\n"

// Plotting in Graphical Panel
theGP.stop()
scale=1
theGP.plotDeform(1,scale,0,tsteps,1)
```

Listing A.3: ImagingGroovyFile

```
import climax.contraption
import java.awt.Graphics2D
import java.awt.BasicStroke;
import java.awt.Stroke;

class ImagingFrame implements contraption{
  int id
```

```
  Color SelfColor = Color.blue
  Color MotionColor = Color.black
  def xcords=[]
  def ycords=[]
  def val=[]
  def sources=[]
  def receivers=[]
  def scatterers=[]
  def connectivity=[]
  def mode=0 //motion, 0: as diagram, 1: as color distribution
  double dataRangeMax
  double dataRangeMin
  def theGP

  public int getID(){
    return id
  }

  public void SelfPortrait(Graphics2D g2, double ymin, double ymax, int ye, int ys, double xmin, double xmax, int xe, int xs){
    def transX = {double x ->
      return ((int) ((int) ((-xmin * xe + x * (xe - xs) + xmax * xs) / (xmax - xmin))))
    }

    def transY = {double y ->
      return ((int) ((int) ((-ymin * ye + y * (ye - ys) + ymax * ys) / (ymax - ymin))))
    }
    Color origColor=g2.getColor()
    Stroke defaultStroke = g2.getStroke()
    g2.setColor(SelfColor)
    g2.setStroke(new BasicStroke(1))

    if(connectivity.isEmpty()){}else{
      connectivity.each{
        int ni=it[0]-1; int nj=it[1]-1
        g2.drawLine(transX(xcords[ni]), transY(ycords[ni]), transX(xcords[nj]), transY(ycords[nj]));
      }
    }

    float[] fl = new float[1]; fl[0]=9
    Stroke dashed = new BasicStroke(0.5, BasicStroke.CAP_BUTT, BasicStroke.JOIN_BEVEL, 0, fl, 0);
          g2.setStroke(dashed);

          g2.setColor(Color.green)
          if(receivers.isEmpty()){}else{
            receivers.each{
              double x=it[0]; double y=it[1];
        g2.drawLine(transX(x),ys,transX(x),ye);
        g2.drawLine(xs,transY(y),xe,transY(y));
            }
          }
    fl[0]=8
    dashed = new BasicStroke(1.0, BasicStroke.CAP_BUTT, BasicStroke.JOIN_BEVEL, 0, fl, 0);
          g2.setStroke(dashed);
          g2.setColor(Color.red)
```

```
          if(sources.isEmpty()){}else{
            sources.each{
              double x=it[0]; double y=it[1];
        g2.drawLine(transX(x),ys,transX(x),ye);
        g2.drawLine(xs,transY(y),xe,transY(y));
            }
          }
          fl[0]=7
    dashed = new BasicStroke(1.0, BasicStroke.CAP_BUTT, BasicStroke.
  JOIN_BEVEL, 0, fl, 0);
          g2.setStroke(dashed);
          g2.setColor(Color.gray)
          if(scatterers.isEmpty()){}else{
            scatterers.each{
              double x=it[0]; double y=it[1];
        g2.drawLine(transX(x),ys,transX(x),ye);
        g2.drawLine(xs,transY(y),xe,transY(y));
            }
          }
    g2.setColor(origColor)
    g2.setStroke(defaultStroke)
  }

  public void Motion(Graphics2D g2, double ymin, double ymax, int ye, int
   ys, double xmin, double xmax, int xe, int xs, int step, double scale){
    switch(mode){
      case 0:
      def transX = {double x ->
        return ((int) ((int) ((-xmin * xe + x * (xe - xs) + xmax * xs) / (
  xmax - xmin))))
      }

      def transY = {double y ->
        return ((int) ((int) ((-ymin * ye + y * (ye - ys) + ymax * ys) / (
  ymax - ymin))))
      }

      Color origColor=g2.getColor()
      g2.setColor(MotionColor)
      if(connectivity.isEmpty()){}else{connectivity.each{
          int n1=it[0]-1; int n2=it[1]-1
          double[] normal=getNormal(it)
          normal[0]=-normal[0];normal[1]=-normal[1];normal[2]=-normal[2]
                int xp1 = transX(normal[0]*scale*val[n1][step]+xcords[n1]);
          int yp1 = transY(normal[1]*scale*val[n1][step]+ycords[n1]);
          int xp2 = transX(normal[0]*scale*val[n2][step]+xcords[n2]);
          int yp2 = transY(normal[1]*scale*val[n2][step]+ycords[n2]);
          g2.drawLine(xp1, yp1, xp2, yp2)
        }
      }

      g2.setColor(origColor)
      break;
      case 1:
      theGP.dataRangeMax=dataRangeMax
      theGP.dataRangeMin=dataRangeMin
      theGP.colorCount=64
      theGP.colorArray = new Color[theGP.colorCount];
            theGP.setupColors();
      def transX = {double x ->
```

```
        return ((int) ((int) ((-xmin * xe + x * (xe - xs) + xmax * xs) / (
  xmax - xmin))))
      }

      def transY = {double y ->
        return ((int) ((int) ((-ymin * ye + y * (ye - ys) + ymax * ys) / (
  ymax - ymin))))
      }

      Color origColor=g2.getColor()
      Stroke defaultStroke = g2.getStroke()
      g2.setStroke(new BasicStroke(7))

      if(connectivity.isEmpty()){}else{
        connectivity.each{
          int n1=it[0]-1; int n2=it[1]-1
          double dataValue=(val[n1][step]+val[n2][step])/2.0
          double colorLocation =((theGP.colorCount -1.0)*(dataValue -
  dataRangeMin))/(dataRangeMax - dataRangeMin);
          int colorIndex = (int)colorLocation;
          if(colorIndex >= theGP.colorCount) colorIndex = theGP.colorCount
  -1;
          if(colorIndex < 0) colorIndex = 0;
          g2.setColor(theGP.colorArray[colorIndex]);
          g2.drawLine(transX(xcords[n1]), transY(ycords[n1]), transX(xcords
  [n2]), transY(ycords[n2]));
        }
      }
      g2.setColor(origColor)
      g2.setStroke(defaultStroke)
      break;
      default:
      println("Motion not yet defined for mode:"+mode)
    }
  }

  public double maximum_X_coordinate(){if(xcords.isEmpty()){0.0}else{xcords
   .max()}}
  public double minimum_X_coordinate(){if(xcords.isEmpty()){0.0}else{xcords
   .min()}}
  public double maximum_Y_coordinate(){if(ycords.isEmpty()){0.0}else{ycords
   .max()}}
  public double minimum_Y_coordinate(){if(ycords.isEmpty()){0.0}else{ycords
   .min()}}

  def addCoord={x,y->
    xcords.add(x as double)
    ycords.add(y as double)
  }

  def addVal={
    val.add(it)
  }

  def addSource={
    sources.add(it)
  }

  def addReceiver={
    receivers.add(it)
```

```
  }

  def addConnection={
    connectivity.add(it)
  }

  def addScatterer={
    scatterers.add(it)
  }

  def getNormal={
    double[] normal = new double[3];
    double[] vxsi = new double[3];
    double[] veta = new double[3];
    (0..2).each{normal[it]=0.0;vxsi[it]=0.0;veta[it]=0.0;}
    vxsi[0]=xcords[it[1]-1] -xcords[it[0]-1];
    vxsi[1]=ycords[it[1]-1] -ycords[it[0]-1];
    vxsi[2]=0.0;
    veta[0]=0.0;
    veta[1]=0.0;
    veta[2]=1.0;
    normal[0]=vxsi[1]*veta[2]-vxsi[2]*veta[1];
    normal[1]=vxsi[2]*veta[0]-vxsi[0]*veta[2];
    normal[2]=vxsi[0]*veta[1]-vxsi[1]*veta[0];
    double n=Math.sqrt(
            normal[0]*normal[0]+
            normal[1]*normal[1]+
            normal[2]*normal[2]
            );
    normal[0]/=n;
    normal[1]/=n;
    normal[2]/=n;
    return normal;
  }

  def findLimitVals(){
    dataRangeMax=Double.NEGATIVE_INFINITY
    dataRangeMin=Double.POSITIVE_INFINITY
    val.each{
      it.each{
        if(dataRangeMax<it)dataRangeMax=it
        if(dataRangeMin>it)dataRangeMin=it
      }
    }
    theGP.dataRangeMax=dataRangeMax
    theGP.dataRangeMin=dataRangeMin
  }
}
```

Listing A.4: ImagingClimaxFile

```
theUniverse.cls()
ImFr = new ImagingFrame()

theUniverse.putContraption(ImFr)

// ===========================================================
// To read the responses

// Incident Field
path="In order to study incident field, please enter the path of responses
```

```
    here ... "


// Scattered Field
// path="In order to study the scattered field problem, please enter the
    path of responses here ... "

// ==================================================================

// To choose the receiver (InputRec = [" DoF_id "]) which used for Input in
     Inverse problem
InputRec=["x_2","x_3","y_6","y_7","y_8"]

path1 = []
for(i=0;i<InputRec.size();i++){
  path1[i]=path+"Input—resp_"+InputRec[i]+"\\"
}


pathN="In order to add Gaussian noise, please enter the path of noisy
    responses here ..."
path2 = []
for(i=0;i<InputRec.size();i++){
  path2[i]=pathN+"Input—resp_"+InputRec[i]+"\\"
  //println path2[i]
}


// To choose one scenario
disps_scenario=6  // -1: if you don't use displacements or velocities,
velcs_scenario=-1 // 0: axial values,
            // 1: transverse values,
            // 2: rotational values,
            // 3: absolute axial values,
            // 4: absolute transverse values,
            // 5: absolute rotational values,
            // 6: norm


// Discrete time steps
Nt=2**11-1

// coefficient
coefr=1.0


// ==================================================================
GeomFile = new File(path+"GeomFileImagers.txt")
lines = GeomFile.readLines()
readelems=false

map=[]
lines.each{
  if(readelems){
    clms = it.split(" ");
    n1=Integer.parseInt(clms[0])
    n2=Integer.parseInt(clms[1])
    //println("connectivity: "+n1+" "+n2)
    ImFr.addConnection([n1,n2])
  }else{
```

```
      if(it.equals(""))readelems=true
      if(readelems==false){
        clms = it.split(" ");
        x=Double.parseDouble(clms[0])
        y=Double.parseDouble(clms[1])
        map.add(Integer.parseInt(clms[2]))
        //println("coordinates: "+x+" "+y)
        ImFr.addCoord(x,y)
      }
    }
}

ImFr.addSource([7.5,0.0])
ImFr.addReceiver([ImFr.xcords[1],ImFr.ycords[1]])
ImFr.addReceiver([ImFr.xcords[2],ImFr.ycords[2]])
ImFr.addReceiver([ImFr.xcords[3],ImFr.ycords[3]])
ImFr.addReceiver([ImFr.xcords[5],ImFr.ycords[5]])
ImFr.addReceiver([ImFr.xcords[6],ImFr.ycords[6]])
ImFr.addReceiver([ImFr.xcords[7],ImFr.ycords[7]])
ImFr.addScatterer([10.125,3.5])
ImFr.addScatterer([10.2,3.6])
// ====================================================



// Add Gaussian Noise
noise=false // noise --> true/false
if(noise){
  ArrayMaxVal = []
  (0..<map.size()).each{
    lines_r1 = []
    lines_r2 = []
    lines_r3 = []
    lines_dr1 = []
    lines_dr2 = []
    lines_dr3 = []
    for(i=0;i<path2.size();i++){
      ValArray=[]

      // ----------Displacements----------
      switch(disps_scenario){
        case 0:
        lines_r1[i] = (new File (path1[i]+"resp_x_"+map[it]+".txt") ).
    readLines()   //axial DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r1[i][it])))
        }
        break;

        case 1:
        lines_r2[i] = (new File (path1[i]+"resp_y_"+map[it]+".txt") ).
    readLines()   //transverse DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r2[i][it])))
        }
        break;

        case 2:
        lines_r3[i] = (new File (path1[i]+"resp_rz_"+map[it]+".txt") ).
    readLines()  //rotational DoF
```

```
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r3[i][it])))
        }
        break;

        case 3:
        lines_r1[i] = (new File (path1[i]+"resp_x_"+map[it]+".txt") ).readLines()    //axial DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r1[i][it])))
        }
        break;

        case 4:
        lines_r2[i] = (new File (path1[i]+"resp_y_"+map[it]+".txt") ).readLines()     //transverse DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r2[i][it])))
        }
        break;

        case 5:
        lines_r3[i] = (new File (path1[i]+"resp_rz_"+map[it]+".txt") ).readLines()  //rotational DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r3[i][it])))
        }
        break;

        case 6:
        lines_r1[i] = (new File (path1[i]+"resp_x_"+map[it]+".txt") ).readLines()    //axial DoF
        lines_r2[i] = (new File (path1[i]+"resp_y_"+map[it]+".txt") ).readLines()     //transverse DoF
        lines_r3[i] = (new File (path1[i]+"resp_rz_"+map[it]+".txt") ).readLines()  //rotational DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_r1[i][it])))
          ValArray.add(abs(Double.parseDouble(lines_r2[i][it])))
          ValArray.add(abs(Double.parseDouble(lines_r3[i][it])))
        }
        break;
      }

      switch(velcs_scenario){
        case 0:
        lines_dr1[i] = (new File (path1[i]+"resp_dx_"+map[it]+".txt") ).readLines()    //axial DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr1[i][it])))
        }
        break;

        case 1:
        lines_dr2[i] = (new File (path1[i]+"resp_dy_"+map[it]+".txt") ).readLines()     //transverse DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr2[i][it])))
        }
        break;
```

```

        case 2:
        lines_dr3[i] = (new File (path1[i]+"resp_drz_"+map[it]+".txt") ).
    readLines()  //rotational DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr3[i][it])))
        }
        break;

        case 3:
        lines_dr1[i] = (new File (path1[i]+"resp_dx_"+map[it]+".txt") ).
    readLines()   //axial DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr1[i][it])))
        }
        break;

        case 4:
        lines_dr2[i] = (new File (path1[i]+"resp_dy_"+map[it]+".txt") ).
    readLines()    //transverse DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr2[i][it])))
        }
        break;

        case 5:
        lines_dr3[i] = (new File (path1[i]+"resp_drz_"+map[it]+".txt") ).
    readLines()  //rotational DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr3[i][it])))
        }
        break;

        case 6:
        lines_dr1[i] = (new File (path1[i]+"resp_dx_"+map[it]+".txt") ).
    readLines()   //axial DoF
        lines_dr2[i] = (new File (path1[i]+"resp_dy_"+map[it]+".txt") ).
    readLines()    //transverse DoF
        lines_dr3[i] = (new File (path1[i]+"resp_drz_"+map[it]+".txt") ).
    readLines()  //rotational DoF
        (0..Nt).each{
          ValArray.add(abs(Double.parseDouble(lines_dr1[i][it])))
          ValArray.add(abs(Double.parseDouble(lines_dr2[i][it])))
          ValArray.add(abs(Double.parseDouble(lines_dr3[i][it])))
        }
        break;
      }

      ArrayMaxVal.add(ValArray.max())
    }
  }

  MaxVal = ArrayMaxVal.max()
  println MaxVal
  coefn=10*MaxVal/100
}else{
  coefn=0.0
}


```

```
// ------------------------ Read & add the imaging values in ImagingFrame
   ------------------------
ImFr.val=[]
if(InputRec.size()==1){

  for(i=0;i<InputRec.size();i++){

    (0..<map.size()).each{
      val2video=[]

      lines_ux = []
      lines_uy = []
      lines_urz = []
      lines_dux = []
      lines_duy = []
      lines_durz = []

      nlines_ux = []
      nlines_uy = []
      nlines_urz = []
      nlines_dux = []
      nlines_duy = []
      nlines_durz = []

      r1 = []
      r2 = []
      r3 = []

      dr1 = []
      dr2 = []
      dr3 = []


      lines_ux[i]=(new File (path1[i]+"resp_x_"+map[it]+".txt") ).readLines
   ()
      lines_uy[i]=(new File (path1[i]+"resp_y_"+map[it]+".txt") ).readLines
   ()
      lines_urz[i]=(new File (path1[i]+"resp_rz_"+map[it]+".txt") ).
   readLines()

      lines_dux[i]=(new File (path1[i]+"resp_dx_"+map[it]+".txt") ).
   readLines()
      lines_duy[i]=(new File (path1[i]+"resp_dy_"+map[it]+".txt") ).
   readLines()
      lines_durz[i]=(new File (path1[i]+"resp_drz_"+map[it]+".txt") ).
   readLines()

      if(noise){
        nlines_ux[i]=(new File (path2[i]+"resp_x_"+map[it]+".txt") ).
   readLines()
        nlines_uy[i]=(new File (path2[i]+"resp_y_"+map[it]+".txt") ).
   readLines()
        nlines_urz[i]=(new File (path2[i]+"resp_rz_"+map[it]+".txt") ).
   readLines()

        nlines_dux[i]=(new File (path2[i]+"resp_dx_"+map[it]+".txt") ).
   readLines()
        nlines_duy[i]=(new File (path2[i]+"resp_dy_"+map[it]+".txt") ).
   readLines()
        nlines_durz[i]=(new File (path2[i]+"resp_drz_"+map[it]+".txt") ).
```

```
readLines()
   }


   switch(disps_scenario){
   case 0:
   (0..Nt).each{
     if(noise){
       r1.add(coefr*Double.parseDouble(lines_ux[i][it])+coefn*Double.parseDouble(nlines_ux[i][it]))

     }else{
       r1.add(Double.parseDouble(lines_ux[i][it]))
     }
     val2video.add( r1[it] )
   }
   break;

   case 1:
   (0..Nt).each{
     if(noise){
       r2.add(coefr*Double.parseDouble(lines_uy[i][it])+coefn*Double.parseDouble(nlines_uy[i][it]))

     }else{
       r2.add(Double.parseDouble(lines_uy[i][it]))
     }
     val2video.add( r2[it] )
   }
   break;

   case 2:
   (0..Nt).each{
     if(noise){
       r3.add(coefr*Double.parseDouble(lines_urz[i][it])+coefn*Double.parseDouble(nlines_urz[i][it]))

     }else{
       r3.add(Double.parseDouble(lines_urz[i][it]))
     }
     val2video.add( r3[it] )
   }
   break;

   case 3:
   (0..Nt).each{
     if(noise){
       r1.add(coefr*Double.parseDouble(lines_ux[i][it])+coefn*Double.parseDouble(nlines_ux[i][it]))

     }else{
       r1.add(Double.parseDouble(lines_ux[i][it]))
     }
     val2video.add( abs(r1[it]) )
   }
   break;

   case 4:
   (0..Nt).each{
     if(noise){
```

```
          r2.add(coefr*Double.parseDouble(lines_uy[i][it])+coefn*Double.
    parseDouble(nlines_uy[i][it]))

        }else{
          r2.add(Double.parseDouble(lines_uy[i][it]))
        }
        val2video.add( abs(r2[it]) )
      }
      break;

      case 5:
      (0..Nt).each{
        if(noise){
          r3.add(coefr*Double.parseDouble(lines_urz[i][it])+coefn*Double.
    parseDouble(nlines_urz[i][it]))

        }else{
          r3.add(Double.parseDouble(lines_urz[i][it]))
        }
        val2video.add( abs(r3[it]) )
      }
      break;

      case 6:
      (0..Nt).each{
        if(noise){
          r1.add(coefr*Double.parseDouble(lines_ux[i][it])+coefn*Double.
    parseDouble(nlines_ux[i][it]))
          r2.add(coefr*Double.parseDouble(lines_uy[i][it])+coefn*Double.
    parseDouble(nlines_uy[i][it]))
          r3.add(coefr*Double.parseDouble(lines_urz[i][it])+coefn*Double.
    parseDouble(nlines_urz[i][it]))

        }else{
          r1.add(Double.parseDouble(lines_ux[i][it]))
          r2.add(Double.parseDouble(lines_uy[i][it]))
          r3.add(Double.parseDouble(lines_urz[i][it]))
        }
        val2video.add( sqrt(r1[it]*r1[it]+r2[it]*r2[it]+r3[it]*r3[it]) )
      }
      break;

      //default: println "Read only the values of velocities "+
    disps_scenario
      //break;
      }

      switch(velcs_scenario){
      case 0:
      (0..Nt).each{
        if(noise){
          dr1.add(coefr*Double.parseDouble(lines_dux[i][it])+coefn*Double.
    parseDouble(nlines_dux[i][it]))

        }else{
          dr1.add(Double.parseDouble(lines_dux[i][it]))
        }
        val2video.add( dr1[it] )
      }
      break;
```

```

        case 1:
        (0..Nt).each{
          if(noise){
            dr2.add(coefr*Double.parseDouble(lines_duy[i][it])+coefn*Double.
      parseDouble(nlines_duy[i][it]))

          }else{
            dr2.add(Double.parseDouble(lines_duy[i][it]))
          }
          val2video.add( dr2[it] )
        }
        break;

        case 2:
        (0..Nt).each{
          if(noise){
            dr3.add(coefr*Double.parseDouble(lines_durz[i][it])+coefn*Double.
      parseDouble(nlines_durz[i][it]))

          }else{
            dr3.add(Double.parseDouble(lines_durz[i][it]))
          }
          val2video.add( dr3[it] )
        }
        break;

        case 3:
        (0..Nt).each{
          if(noise){
            dr1.add(coefr*Double.parseDouble(lines_dux[i][it])+coefn*Double.
      parseDouble(nlines_dux[i][it]))

          }else{
            dr1.add(Double.parseDouble(lines_dux[i][it]))
          }
          val2video.add( abs(dr1[it]) )
        }
        break;

        case 4:
        (0..Nt).each{
          if(noise){
            dr2.add(coefr*Double.parseDouble(lines_duy[i][it])+coefn*Double.
      parseDouble(nlines_duy[i][it]))

          }else{
            dr2.add(Double.parseDouble(lines_duy[i][it]))
          }
          val2video.add( abs(dr2[it]) )
        }
        break;

        case 5:
        (0..Nt).each{
          if(noise){
            dr3.add(coefr*Double.parseDouble(lines_durz[i][it])+coefn*Double.
      parseDouble(nlines_durz[i][it]))

          }else{
```

```
          dr3.add(Double.parseDouble(lines_durz[i][it]))
        }
        val2video.add( abs(dr3[it]) )
      }
      break;

      case 6:
      (0..Nt).each{
        if(noise){
          dr1.add(coefr*Double.parseDouble(lines_dux[i][it])+coefn*Double.
    parseDouble(nlines_dux[i][it]))
          dr2.add(coefr*Double.parseDouble(lines_duy[i][it])+coefn*Double.
    parseDouble(nlines_duy[i][it]))
          dr3.add(coefr*Double.parseDouble(lines_durz[i][it])+coefn*Double.
    parseDouble(nlines_durz[i][it]))
        }else{
          dr1.add(Double.parseDouble(lines_dux[i][it]))
          dr2.add(Double.parseDouble(lines_duy[i][it]))
          dr3.add(Double.parseDouble(lines_durz[i][it]))
        }
        val2video.add( sqrt(dr1[it]*dr1[it]+dr2[it]*dr2[it]+dr3[it]*dr3[it
    ]) )
      }
      break;

      //default: println "Read only the values of displacements "+
    velcs_scenario
      //break;
      }
      ImFr.addVal(val2video)
    }
  }
}else{
  (0..<map.size()).each{
    val2video=[]

    values1=[]
    values2=[]
    values3=[]

    dvalues1=[]
    dvalues2=[]
    dvalues3=[]

    for(i=0;i<InputRec.size();i++){
      lines_ux = []
      lines_uy = []
      lines_urz = []
      lines_dux = []
      lines_duy = []
      lines_durz = []

      nlines_ux = []
      nlines_uy = []
      nlines_urz = []
      nlines_dux = []
      nlines_duy = []
      nlines_durz = []
```

```
    r1 = []
    r2 = []
    r3 = []

    dr1 = []
    dr2 = []
    dr3 = []

    lines_ux[i]=(new File (path1[i]+"resp_x_"+map[it]+".txt") ).readLines()
    lines_uy[i]=(new File (path1[i]+"resp_y_"+map[it]+".txt") ).readLines()
    lines_urz[i]=(new File (path1[i]+"resp_rz_"+map[it]+".txt") ).readLines()

    lines_dux[i]=(new File (path1[i]+"resp_dx_"+map[it]+".txt") ).readLines()
    lines_duy[i]=(new File (path1[i]+"resp_dy_"+map[it]+".txt") ).readLines()
    lines_durz[i]=(new File (path1[i]+"resp_drz_"+map[it]+".txt") ).readLines()

    if(noise){
      nlines_ux[i]=(new File (path2[i]+"resp_x_"+map[it]+".txt") ).readLines()
      nlines_uy[i]=(new File (path2[i]+"resp_y_"+map[it]+".txt") ).readLines()
      nlines_urz[i]=(new File (path2[i]+"resp_rz_"+map[it]+".txt") ).readLines()

      nlines_dux[i]=(new File (path2[i]+"resp_dx_"+map[it]+".txt") ).readLines()
      nlines_duy[i]=(new File (path2[i]+"resp_dy_"+map[it]+".txt") ).readLines()
      nlines_durz[i]=(new File (path2[i]+"resp_drz_"+map[it]+".txt") ).readLines()
    }


    (0..Nt).each{
      if(noise){
        r1.add(coefr*Double.parseDouble(lines_ux[i][it])+coefn*Double.parseDouble(nlines_ux[i][it]))
        r2.add(coefr*Double.parseDouble(lines_uy[i][it])+coefn*Double.parseDouble(nlines_uy[i][it]))
        r3.add(coefr*Double.parseDouble(lines_urz[i][it])+coefn*Double.parseDouble(nlines_urz[i][it]))

        dr1.add(coefr*Double.parseDouble(lines_dux[i][it])+coefn*Double.parseDouble(nlines_dux[i][it]))
        dr2.add(coefr*Double.parseDouble(lines_duy[i][it])+coefn*Double.parseDouble(nlines_duy[i][it]))
        dr3.add(coefr*Double.parseDouble(lines_durz[i][it])+coefn*Double.parseDouble(nlines_durz[i][it]))
      }else{
        r1.add(Double.parseDouble(lines_ux[i][it]))
        r2.add(Double.parseDouble(lines_uy[i][it]))
        r3.add(Double.parseDouble(lines_urz[i][it]))

        dr1.add(Double.parseDouble(lines_dux[i][it]))
```

```
          dr2.add(Double.parseDouble(lines_duy[i][it]))
          dr3.add(Double.parseDouble(lines_durz[i][it]))
        }
      }
      values1[i]=r1
      values2[i]=r2
      values3[i]=r3

      dvalues1[i]=dr1
      dvalues2[i]=dr2
      dvalues3[i]=dr3

    }


    switch(disps_scenario){
    case 0:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values1[it][tt]
      }
      val2video.add(tval)
    }
    break;

    case 1:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values2[it][tt]
      }
      val2video.add(tval)
    }
    break;

    case 2:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values3[it][tt]
      }
      val2video.add(tval)
    }
    break;

    case 3:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values1[it][tt]
      }
```

```
      val2video.add(abs(tval))
    }
    break;

    case 4:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values2[it][tt]
      }
      val2video.add(abs(tval))
    }
    break;

    case 5:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=values3[it][tt]
      }
      val2video.add(abs(tval))
    }
    break;

    case 6:
    (0..Nt).each{
      sz=InputRec.size()
      tval1=0.0
      tval2=0.0
      tval3=0.0
      tt=it
      (0..<sz).each{
        tval1+=values1[it][tt]
        tval2+=values2[it][tt]
        tval3+=values3[it][tt]
      }
      val2video.add(sqrt(tval1*tval1+tval2*tval2+tval3*tval3))
    }
    break;

    //default: println "Read only the values of velocities "+disps_scenario
    //break;
    }


    switch(velcs_scenario){
    case 0:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues1[it][tt]
      }
      val2video.add(tval)
    }
```

```
    break;

    case 1:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues2[it][tt]
      }
      val2video.add(tval)
    }
    break;

    case 2:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues3[it][tt]
      }
      val2video.add(tval)
    }
    break;

    case 3:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues1[it][tt]
      }
      val2video.add(abs(tval))
    }
    break;

    case 4:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues2[it][tt]
      }
      val2video.add(abs(tval))
    }
    break;

    case 5:
    (0..Nt).each{
      sz=InputRec.size()
      tval=0.0
      tt=it
      (0..<sz).each{
        tval+=dvalues3[it][tt]
      }
      val2video.add(abs(tval))
    }
```

```
      break;

      case 6:
      PlotArray=[]
      (0..Nt).each{
        sz=InputRec.size()
        tval1=0.0
        tval2=0.0
        tval3=0.0
        tt=it
        (0..<sz).each{
          tval1+=dvalues1[it][tt]
          tval2+=dvalues2[it][tt]
          tval3+=dvalues3[it][tt]
        }
        val2video.add(sqrt(tval1*tval1+tval2*tval2+tval3*tval3))
      }
      break;

      //default: println "Read only the values of displacements "+
     velcs_scenario
      //break;
      }
      ImFr.addVal(val2video)
    }
}



// ———————————— the GP settings ————————————
ImFr.theGP=theGP
ImFr.findLimitVals()
ImFr.mode=1

theGP.GlobalMinMax=false

theGP.stop()

theGP.gridClr=theGP.getBackground()

theGP.fontClr=Color.blue
theGP.setMargin_x(100)
theGP.setMargin_y(50)

theGP.colormode=1

theGP.colorinv=false

//theGP.fontClr=Color.pink
//theGP.FontSize=2

theGP.title="Source Localization"
//theGP.title="Defect Identification"
//theGP.titlestep=false

import java.text.DecimalFormat;
theGP.FontSize=23
theGP.gridnumfx=new DecimalFormat("0.000");
theGP.gridnumfy=new DecimalFormat("0.000");
theGP.cbarnumf=new DecimalFormat("0.#E0");
```

```
// Incident Field
//ex_time=(int)(Nt-97.47619047619048)

// Scattered Field
//ex_time=(int)(Nt-129)

scale=1200.0
theGP.plotDeform(scale,ex_time)
theGP.showcolorbar(true)
//theGP.stop()


// ========== Plots ==========


// check plot for time evolution of max norm vals
import java.awt.Dimension;
import java.text.*;

pl=[]
println ImFr.val.size()
(0..Nt).each{
  ts=it
  maxv=Double.NEGATIVE_INFINITY
  ImFr.val.each{
    if(it[ts]>maxv)maxv=it[ts]
  }
  pl.add(maxv)
}
thePlot.clear()

thePlot.setPreferredSize(new Dimension(1372, 732)); // Set dimensions of
    images
// Margins
thePlot.setMargin_x(180)
thePlot.setMargin_y(90)

// Limits
thePlot.activate_ylimit_min(true)
//thePlot.activate_xlimit_min(true)
//thePlot.setxlimit_min()
thePlot.setylimit_min(0.0)

thePlot.setTexfontsize(22)
thePlot.setFormatXAxis(new DecimalFormat("0.0"))
thePlot.setFormatYAxis(new DecimalFormat("0.000"))

thePlot.vline(ex_time, java.awt.Color.red)

thePlot.addFunction(new plotfunction(pl as double[]))
thePlot.setMarker(true)
thePlot.show()
println Nt
//-----------------------------------------------------------------


//-----------------------------------------------------------------
```

```
PeakVal=[]
xcoord=[]
ycoord=[]

// Find each value (in exact discrete time step) from Imaging Frame
for(i=0;i<ImFr.val.size();i++){
  x=ImFr.val[i][ex_time]
  PeakVal.add(x)

  // Keep info(value, id, xcoord, ycoord) about "PeakVal" values
  xcoord.add(ImFr.getXcords()[i])
  ycoord.add(ImFr.getYcords()[i])
}

MaxVal=PeakVal[PeakVal.indexOf(PeakVal.max())]
println "(x,y)= "+"("+ImFr.getXcords()[PeakVal.indexOf(PeakVal.max())]+", "
    +ImFr.getYcords()[PeakVal.indexOf(PeakVal.max())]+")"
println MaxVal
//----------------------------------------------------------------------
```

Βιβλιογραφία

[1] Χρήστος Γ. Παναγιωτόπουλος. Ανάλυση πλαισιακής κατασκευής, με επίπεδα στοιχεία πλήρωσης, σε στατικό και δυναμικό φορτίο. http://thales.iacm.forth.gr/~pchr/CA2017.pdf, Εαρινό Εξάμηνο 2017. Αντίγραφο: 2019-02-30.

[2] G.E. Stavroulakis. Inverse and identification problems in mechanics, 2000.

[3] Graff, Karl F. *Wave motion in elastic solids*. Oxford University Press, 1973.

[4] Χριστόφορος Γ. Προβατίδης. *Πεπερασμένα Στοιχεία στην Ανάλυση Κατασκευών*. 2016.

[5] C.G. Panagiotopoulos. Symplegma, a JVM implementation for numerical methods in computational mechanics, Heraklion, October 13-14, 2018.

[6] I. Petromichelakis , C. Tsogka , C.G. Panagiotopoulos . Signal-to-Noise Ratio analysis for time-reversal based imaging techniques in bounded domains. Wave Motion, 2018.

[7] C.G. Panagiotopoulos, M.I. Mavrikis and G.E. Stavroulakis. Computational Study of Imaging Techniques based on Elastic Wave Reversibility in Beams. September 2019.

[8] Γεώργιος Δ. Μανώλης Παναγιώτης Κ. Κολιόπουλος, Χρήστος Γ. Παναγιωτόπουλος. *Δυναμική των κατασκευών*. http://repository.kallipos.gr/handle/11419/2465, ISBN: 978-960-603-074-1, "ΣΕΑΒ", 2015.

[9] Χρήστος Γ. Παναγιωτόπουλος. Δυναμική κατασκευαστικών και μηχανικών συστημάτων. Σημειώσεις μαθήματος: ΜΠΔ 432 - Δυναμική, ταλαντώσεις και έλεγχος κατασκευών.

[10] S.C. Chapra, R.P. Canale. *Αριθμητικές Μέθοδοι για Μηχανικούς*. 2014.

[11] S. Koo. Subsurface elastic wave energy focusing based on a time reversal concept, 2017.

[12] D. Givoli . Time Reversal as a Computational Tool in Acoustics and Elastodynamics. Journal of Computational Acoustics, 2014.