



ΕΛΛΗΝΙΚΟ ΑΝΟΙΚΤΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ
ΣΕΙΣΜΙΚΗ ΜΗΧΑΝΙΚΗ ΚΑΙ ΑΝΤΙΣΕΙΣΜΙΚΕΣ ΚΑΤΑΣΚΕΥΕΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
ΑΝΑΓΝΩΡΙΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ ΥΣΤΕΡΗΤΙΚΟΥ
ΜΟΝΤΕΛΟΥ ΒΟΥC-WEN

ΧΑΛΕΠΑΙΔΟΥ ΜΕΡΟΠΗ

ΕΠΙΒΛΕΠΩΝ
ΠΑΝΑΓΙΩΤΟΠΟΥΛΟΣ ΧΡΗΣΤΟΣ

Πάτρα, Ιούνιος 2026

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία της φοιτήτριας Χαλεπλίδου Μερόπης που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης η συγγραφέας εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση της συγγραφέα. Η συγγραφέας διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.

Διπλωματική εργασία

**ΑΝΑΓΝΩΡΙΣΗ ΤΩΝ ΠΑΡΑΜΕΤΡΩΝ ΤΟΥ
ΥΣΤΕΡΗΤΙΚΟΥ ΜΟΝΤΕΛΟΥ ΒΟΥC - WEN**

ΧΑΛΕΠΑΙΔΟΥ ΜΕΡΟΠΗ

Επιτροπή Διπλωματικής Εργασίας

Επιβλέπων Καθηγητής:
ΠΑΝΑΓΙΩΤΟΠΟΥΛΟΣ ΧΡΗΣΤΟΣ
Δρ. ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ

Συν-Επιβλέπουσα Καθηγήτρια:
ΚΑΤΣΙΜΠΙΝΗ ΠΑΝΑΓΙΩΤΑ
Δρ. ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ

Πάτρα, Ιούνιος, 2026

ΠΡΟΛΟΓΟΣ

Η παρούσα διπλωματική εργασία εκπονήθηκε στα πλαίσια της ολοκλήρωσης των μεταπτυχιακών μου σπουδών στο πρόγραμμα Σεισμική Μηχανική και Αντισεισμικές κατασκευές της σχολής θετικών επιστημών και τεχνολογίας του Ελληνικού Ανοικτού Πανεπιστημίου (ΕΑΠ). Έχει ως θέμα την «Αναγνώριση των παραμέτρων του υστερητικού μοντέλου Bouc-Wen».

Στο σημείο αυτό θα ήθελα να ευχαριστήσω όλους όσους συνέβαλλαν στην ολοκλήρωση αυτής της εργασίας.

Αρχικά θα ήθελα να ευχαριστήσω τον κ. Παναγιωτόπουλο Χρήστο για την επίβλεψη της εργασίας, την εμπιστοσύνη που μου έδειξε και την επιστημονική του καθοδήγηση καθ' όλη τη διάρκεια εκπόνησης της εργασίας.

Επιπλέον, ευχαριστώ τους φίλους μου για την πορεία μου μαζί τους τα χρόνια των σπουδών.

Τέλος, ένα μεγάλο ευχαριστώ οφείλω στην οικογένεια μου για την στήριξη τους όλα αυτά τα χρόνια.

Μερόπη Χαλεπλίδου

Πάτρα, Ιούνιος 2026

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΣΑΓΩΓΗ	9
1.1 Αντικείμενο και στόχος της εργασίας	9
1.2 Διάρθρωση εργασίας	9
2. ΤΟ ΥΣΤΕΡΗΤΙΚΟ ΜΟΝΤΕΛΟ BOUC - WEN	10
2.1 Η υστέρηση	10
2.2 Γενική περιγραφή μοντέλου.....	10
2.3 Μαθηματική διατύπωση.....	11
2.4 Φυσική σημασία παραμέτρων.....	12
3. Η ΧΡΗΣΗ ΤΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ SYMPLEGMATOS DEVELOPMENT ENVIRONMENT (SDE) ΓΙΑ ΤΗΝ ΕΠΙΛΥΣΗ ΕΝΟΣ ΣΥΣΤΗΜΑΤΟΣ ΤΑΛΑΝΤΩΣΗΣ.....	14
3.1 Γενική παρουσίαση του περιβάλλοντος.....	14
3.2 Μέθοδοι επίλυσης	14
3.2.1. Άμεσες Μέθοδοι Ολοκλήρωσης (Direct Integration Methods)	15
3.2.2. Μέθοδοι της Οικογένειας Runge-Kutta.....	15
3.2.3. Αριθμητική Επίλυση Μη Γραμμικών Συστημάτων: Ο Ταλαντωτής Duffing στο SDE.....	16
3.2.4. Αριθμητική Υλοποίηση Ιδανικά Ελαστοπλαστικού Μοντέλου SDOF στο SDE	17
4. ΑΝΑΓΝΩΡΙΣΗ ΠΑΡΑΜΕΤΡΩΝ	21
4.1 Γενικά.....	21
4.2 Μέθοδοι βελτιστοποίησης.....	21
4.2.1. Διαφορική Εξέλιξη (Differential Evolution):	21
4.2.2. Βελτιστοποίηση Σμήνους Σωματιδίων (Particle Swarm Optimization - PSO)	22
4.3 Υπολογιστική Υλοποίηση και Παραμετροποίηση των Scripts στο SDE	22
4.3.1. Διαδικασία ταυτοποίησης από μοντέλο Bouc - Wen σε μοντέλο Bouc - Wen	22
4.3.2. Διαδικασία ταυτοποίησης από ελαστοπλαστικό μοντέλο σε μοντέλο Bouc - Wen.....	23
5. ΠΑΡΑΔΕΙΓΜΑΤΑ ΕΦΑΡΜΟΓΗΣ.....	24
5.1 Γενικά.....	24

5.2	Μονοβάθμιο σύστημα (SDOF).....	24
5.2.1.	Σενάριο 1: σεισμική Διέγερση.....	24
5.2.2.	Σενάριο 2: Δυναμικό πλήγμα.....	31
5.3	Πολυβάθμιο σύστημα	33
5.3.1.	Σενάριο 1: Σεισμική Διέγερση.....	34
5.3.2.	Σενάριο 2: Δυναμικό Πλήγμα (Blast).....	37
6.	ΣΥΜΠΕΡΑΣΜΑΤΑ	42
6.1	Σύνοψη εργασίας.....	42
6.2	Κύρια ευρήματα	42
6.3	Περιορισμοί και προτάσεις για μελλοντική έρευνα.....	43
6.4	Γενικά συμπεράσματα.....	43
	ΒΙΒΛΙΟΓΡΑΦΙΑ.....	45
	ΠΑΡΑΡΤΗΜΑ.....	46

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Εικόνα 1: Επίδραση της παραμέτρου η στη μορφή του υστερητικού βρόγχου.	12
Εικόνα 2: Διάγραμμα φάσης ($u-v$) του ταλαντωτή Duffing.....	17
Εικόνα 3: Διάγραμμα υστερητικού βρόγχου ($u-F_s$) ελαστοπλαστικού SDOF.....	19
Εικόνα 4: Χρονοϊστορία μετατόπισης $u(t)$ ελαστοπλαστικού SDOF.....	19
Εικόνα 5: Χρονοϊστορία ταχύτητας $v(t)$ ελαστοπλαστικού SDOF.	20
Εικόνα 6: Σεισμικό επιταχυνσιογράφημα διέγερσης.	25
Εικόνα 7: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=1, N=4000.....	25
Εικόνα 8: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=1, N=4000.....	26
Εικόνα 9: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=1, N=4000	26
Εικόνα 10: Σεισμικό επιταχυνσιογράφημα διέγερσης.....	27
Εικόνα 11: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=4000.....	27
Εικόνα 12: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=4000.....	28
Εικόνα 13: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=4000	28
Εικόνα 14: Σεισμικό επιταχυνσιογράφημα διέγερσης.....	29
Εικόνα 15: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.....	29
Εικόνα 16: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.....	30
Εικόνα 17: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000	30
Εικόνα 18: Τριγωνικός παλμός διέγερσης (blast).	32
Εικόνα 19: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.....	32
Εικόνα 20: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.....	33
Εικόνα 21: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.....	33
Εικόνα 22: Επιταχυνσιογράφημα σεισμικής διέγερσης MDOF.....	34
Εικόνα 23: Υστερητικοί βρόγχοι – Σεισμός MDOF (level =1, N= 4000, 1 ^{ος} όροφος).	35
Εικόνα 24: Χρονοϊστορία μετατόπισης – Σεισμός MDOF (level =1, N= 4000, 1 ^{ος} όροφος).....	35
Εικόνα 25: Υστερητικοί βρόγχοι – Σεισμός MDOF (level =1, N= 4000, 2 ^{ος} όροφος).	36
Εικόνα 26: Χρονοϊστορία μετατόπισης – Σεισμός MDOF (level =1, N= 4000, 2 ^{ος} όροφος).....	36

Εικόνα 27: Υστερητικοί βρόγχοι – Σεισμός MDOF (level =1, N= 4000, 3 ^{ος} όροφος).	37
Εικόνα 28: Χρονοϊστορία μετατόπισης – Σεισμός MDOF (level =1, N= 4000, 3 ^{ος} όροφος).....	37
Εικόνα 29: Τριγωνικός παλμός διέγερσης (Blast) MDOF.	38
Εικόνα 30: : Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 1 ^{ος} όροφος).	39
Εικόνα 31: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 1 ^{ος} όροφος).....	39
Εικόνα 32: Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 2 ^{ος} όροφος)..	40
Εικόνα 33: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 2 ^{ος} όροφος).....	40
Εικόνα 34: Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 3 ^{ος} όροφος)..	41
Εικόνα 35: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 3 ^{ος} όροφος).....	41

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πίνακας 1: Αποτελέσματα αναγνώρισης για Level 1, N=4000	25
Πίνακας 2: Αποτελέσματα αναγνώρισης για Level 3, N=4000	27
Πίνακας 3: Βέλτιστα αποτελέσματα αναγνώρισης για Level 3, N=8000	29
Πίνακας 4: Αποτελέσματα αναγνώρισης – Δυναμικό πλήγμα (Blast).	31
Πίνακας 5: Αποτελέσματα αναγνώρισης – Σεισμική Διέγερση MDOF (3 όροφοι). ..	34
Πίνακας 6: Αποτελέσματα αναγνώρισης – τριγωνικός παλμός (blast) MDOF (3 όροφοι).	38
Πίνακας 7: Βέλτιστες παράμετροι αναγνώρισης - Σεισμική διέγερση (level = 1, N=4000), MDOF.	43
Πίνακας 8: Βέλτιστες παράμετροι αναγνώρισης – Τριγωνικό πλήγμα (level =3, N=10000), MDOF.	43

ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία πραγματεύεται τη μοντελοποίηση και την ανάλυση της υστερητικής (μη γραμμικής) συμπεριφοράς των υλικών στο πεδίο της ανελαστικής δυναμικής ανάλυσης κατασκευών. Το φαινόμενο της υστέρησης εμφανίζεται έντονα κατά τη διάρκεια δυναμικών καταπονήσεων, όπως οι σεισμοί, όπου η απόκριση ενός δομικού στοιχείου εξαρτάται άμεσα από την ιστορία της παραμόρφωσής του. Κύριος στόχος της εργασίας είναι η αποφυγή της πολυπλοκότητας των αναλυτικών φυσικών μηχανισμών μέσω της χρήσης του φαινομενολογικού μοντέλου Bouc – Wen, το οποίο επιτρέπει τη ρεαλιστική και υπολογιστικά γρήγορη αναπαράσταση των υστερητικών βρόγχων.

Η ανάπτυξη και η αριθμητική επίλυση των συστημάτων ταλάντωσης πραγματοποιήθηκε στο εξειδικευμένο υπολογιστικό περιβάλλον Symplegma Development Environment (SDE), το οποίο βασίζεται στη γλώσσα προγραμματισμού Java. Στο πλαίσιο αυτό, εξετάστηκαν και εφαρμόστηκαν μέθοδοι αριθμητικής ολοκλήρωσης, όπου χρησιμοποιήθηκαν άμεσες μέθοδοι ολοκλήρωσης καθώς και αλγόριθμοι της οικογένειας Runge – Kutta για την επίλυση μη γραμμικών συστημάτων. Επιπλέον, χρησιμοποιήθηκαν αλγόριθμοι αναγνώρισης συστημάτων και βελτιστοποίησης. Για τον προσδιορισμό των άγνωστων μηχανικών παραμέτρων του μοντέλου Bouc -Wen, το πρόβλημα διατυπώθηκε ως πρόβλημα μαθηματικής βελτιστοποίησης με ελαχιστοποίηση μιας αντικειμενικής συνάρτησης σφάλματος. Ως σύστημα αναφοράς χρησιμοποιήθηκε ο ιδανικά ελαστοπλαστικός ταλαντωτής, του οποίου η απόκριση προσομοιώθηκε με τον αλγόριθμο Newmark.

Η αξιολόγηση και η αποτελεσματικότητα της μεθοδολογίας ελέγχθηκε μέσω παραδειγμάτων εφαρμογής σε μονοβάθμια (SDOF) και πολυβάθμια συστήματα υπό την επίδραση δυναμικών φορτίσεων, όπως σεισμικά επιταχυνσιογραφήματα και τριγωνικά πλήγματα. Η παρούσα εργασία αναδεικνύει την υψηλή αξιοπιστία και τη χαμηλή υπολογιστική απαίτηση του μοντέλου Bouc – Wen στην προσομοίωση της ανελαστικής δυναμικής συμπεριφοράς κατασκευών. Παράλληλα, επιβεβαιώνεται η καταλληλότητα του περιβάλλοντος SDE και των εξελκτικών αλγορίθμων στην επιτυχή αναγνώριση και ταυτοποίηση των κρίσιμων παραμέτρων του μοντέλου.

Λέξεις κλειδιά: ελαστοπλαστικό, Bouc – Wen, επιταχυνσιογράφημα, τριγωνικό πλήγμα, Symplegma Development Environment, αναγνώριση παραμέτρων, υστέρηση.

IDENTIFICATION OF THE PARAMETERS OF THE BOUC-WEN HYSTERETIC MODEL MEROPI CHALEPLIDOU

ABSTRACT

This thesis focuses on the modeling and examination of the hysteretic (nonlinear) behavior of materials within the context of inelastic dynamic analysis of structures. The hysteresis phenomenon is pronounced during dynamic stresses, like earthquakes, where a structural element's response is closely linked to its deformation history. The primary aim of the study is to eliminate the intricacy of analytical physical mechanisms by employing the Bouc-Wen phenomenological model, which facilitates a realistic and computationally efficient depiction of hysteretic loops.

The creation and numerical resolution of the oscillation systems were performed in the specialized computing environment Symplegma Development Environment (SDE), which utilizes the Java programming language. In this context, numerical integration techniques were studied and utilized, employing both direct integration methods and Runge – Kutta family algorithms to tackle nonlinear systems. Furthermore, algorithms for system identification and optimization were utilized. To identify the unknown mechanical parameters of the Bouc-Wen model, the issue was established as a mathematical optimization problem focused on minimizing an objective error function. The reference system employed was the ideally elastoplastic oscillator, whose response was simulated using the Newmark algorithm.

The methodology's evaluation and effectiveness were assessed using application examples in single-degree-of-freedom (SDOF) and multi-degree-of-freedom systems subjected to dynamic loads, including seismic accelerograms and triangular impacts. This study emphasizes the excellent reliability and minimal computational demands of the Bouc-Wen model in simulating the inelastic dynamic response of structures. Concurrently, the appropriateness of the SDE environment and evolutionary algorithms in effectively recognizing and identifying the model's critical parameters is validated.

Keywords: elastoplastic, Bouc – Wen, accelerogram, triangular impact, Symplegma Development Environment, parameter identification, hysteresis.

1. ΕΙΣΑΓΩΓΗ

1.1 Αντικείμενο και στόχος της εργασίας

Αντικείμενο της παρούσας διπλωματικής εργασίας είναι η ανάλυση και μοντελοποίηση της υστερητικής συμπεριφοράς των υλικών. Η υστέρηση αποτελεί ένα πολύπλοκο φαινόμενο, στο οποίο η απόκριση ενός δομικού στοιχείου εξαρτάται από το ιστορικό της παραμόρφωσης του ανεξάρτητα από τον ρυθμό εφαρμογής του φορτίου.

Στόχος της παρούσας εργασίας είναι η αναγνώριση και βελτιστοποίηση των παραμέτρων του μοντέλου Bouc -Wen για την απεικόνιση της ανελαστικής απόκρισης των υλικών και σύνθετων δομικών συστημάτων. Η μέθοδος που χρησιμοποιείται βασίζεται σε μια φαινομενολογική προσέγγιση, η οποία επιτρέπει την προσομοίωση των υστερητικών βρόγχων με περιορισμένο υπολογιστικό κόστος. Με τη μέθοδο αυτή επιτυγχάνεται η αποφυγή της πολυπλοκότητας των αναλυτικών μηχανικών μοντέλων και διασφαλίζεται η αξιοπιστία στον καθορισμό της δυναμικής απόκρισης της κατασκευής.

1.2 Διάρθρωση εργασίας

Στο δεύτερο κεφάλαιο παρουσιάζεται η θεωρητική βάση και η μαθηματική διατύπωση του μοντέλου Bouc-Wen. Αναλύεται η φυσική σημασία των παραμέτρων του και οι περιορισμοί που απαιτούνται για να παρουσιάζει το προσομοίωμα συνεπή μηχανική συμπεριφορά.

Στο τρίτο κεφάλαιο γίνεται αναλυτική παρουσίαση του υπολογιστικού περιβάλλοντος Symplegma Development Environment (SDE), το οποίο βασίζεται στη γλώσσα προγραμματισμού Java. Εξηγούνται οι μέθοδοι επίλυσης συστημάτων ταλάντωσης που ενσωματώνει το λογισμικό.

Στο τέταρτο κεφάλαιο εξετάζονται μέθοδοι βελτιστοποίησης που χρησιμοποιούνται για την ταυτοποίηση των παραμέτρων του μοντέλου με βάση πειραματικά δεδομένα.

Στο πέμπτο κεφάλαιο πραγματοποιείται εφαρμογή των προτεινόμενων αλγορίθμων σε μονοβάθμια (SDOF) και πολυβάθμια (MDOF) συστήματα. Γίνεται προσομοίωση της απόκρισης δομικών συστημάτων υπό σεισμική διέγερση και άλλους τύπους διέγερσης και αξιολογούνται τα αποτελέσματα.

Στο έκτο κεφάλαιο συνοψίζονται τα βασικά ευρήματα της έρευνας, αξιολογείται η απόδοση των αλγορίθμων ταυτοποίησης στο περιβάλλον SDE και διατυπώνονται προτάσεις για μελλοντική έρευνα.

2. ΤΟ ΥΣΤΕΡΗΤΙΚΟ ΜΟΝΤΕΛΟ BOUC - WEN

2.1 Η υστέρηση

Ο όρος «υστέρηση» χρησιμοποιήθηκε στην επιστήμη για να περιγράψει τη χρονική καθυστέρηση που παρουσιάζει η απόκριση ενός συστήματος σε σχέση με τη φόρτιση που δέχεται. Αυτό είναι ένα πολύπλοκο φαινόμενο της μη γραμμικής δυναμικής, του οποίου βασικό χαρακτηριστικό είναι η «μνήμη» που αποκτά το υλικό. Αυτό σημαίνει ότι η απόκριση του υλικού δεν εξαρτάται αποκλειστικά από την τρέχουσα φόρτιση, αλλά διαμορφώνεται καθοριστικά από το ιστορικό των προηγούμενων καταπονήσεών του. Στον τομέα της σεισμικής μηχανικής και της ανελαστικής ανάλυσης των κατασκευών, η κατανόηση της υστέρησης είναι βασική. Μέσω αυτού ερμηνεύεται ο μηχανισμός με τον οποίο τα δομικά στοιχεία απορροφούν και καταναλώνουν ενέργεια, καθώς και ο τρόπος που οδηγούνται σε μόνιμες (ανελαστικές) παραμορφώσεις κατά τη διάρκεια έντονων δυναμικών φορτίσεων (π.χ. σεισμών).

Τα βασικά χαρακτηριστικά που χαρακτηρίζουν το φαινόμενο είναι:

- Ανεξαρτησία από την ταχύτητα φόρτισης (Rate Independence): Η υστερητική απόκριση διατηρείται αμετάβλητη ανεξάρτητα από την ταχύτητα με την οποία εφαρμόζεται το φορτίο. Ως εκ τούτου, ο καθυστερημένος βρόχος (το διάγραμμα εισόδου-εξόδου) διατηρεί την μορφή του και προσδιορίζεται αποκλειστικά από την γεωμετρική πορεία της φόρτισης.
- Ικανότητα της Μνήμης: Λόγω της επιρροής του παρελθόντος, το σύστημα δεν παρέχει την ίδια έξοδο αν η διέγερση επιστρέψει σε μια προγενέστερη τιμή, διότι η νέα του κατάσταση εξαρτάται από τη διαδικασία που διασχίστηκε.
- Μαθηματική Αναπαράσταση: Στο διεθνές λογοτεχνικό υλικό, βασισμένο στο έργο των Krasnoselskii και Pokrovskii, η υστέρηση δίνεται μαθηματικά με τη βοήθεια ειδικών υστερητικών τελεστών, οι οποίοι πληρούν τις βασικές προϋποθέσεις της συνέχειας και της αιτιότητας.

Αν και στα πραγματικά υλικά η υστέρηση συνοδεύεται από φαινόμενα που εξαρτώνται χρονικά (ιξώδη), είναι ο κυρίαρχος μηχανισμός στις περισσότερες πρακτικές εφαρμογές, ειδικά υπό συνθήκες αργής φόρτισης. Αξιοσημείωτο είναι ότι, πέρα από τη δομοστατική μηχανική, το φαινόμενο παρατηρείται σε πολλές φυσικές διεργασίες, όπως στην τριβή, στη πλαστική συμπεριφορά των υλικών, στον μαγνητισμό, καθώς και σε χημεία και τη βιολογία.

2.2 Γενική περιγραφή μοντέλου

Στον τομέα της υπολογιστικής μηχανικής, το μοντέλο Bouc-Wen αποτελεί σημαντικό εργαλείο για την περιγραφή συστημάτων με υστέρηση, καθώς θεωρείται ένα από τα πιο αξιόπιστα φαινομενολογικά μοντέλα. Η κατηγοριοποίησή του ως «φαινομενολογικό» αναδεικνύει την κεντρική του φιλοσοφία: το μοντέλο δεν ασχολείται με τα εσωτερικά φυσικά αίτια ή τις μικροδομικές μεταβολές των υλικών που προκαλούν την υστέρηση. Αντιθέτως, επικεντρώνεται αποκλειστικά στην εξωτερική, μακροσκοπική καταγραφή της δράσης τους. Με αυτόν τον τρόπο, καταφέρνει να προσομοιώνει με μεγάλη ακρίβεια τους

κλειστούς βρόχους των διαγραμμάτων φορτίσεων-παραμορφώσεων, ενώ διατηρεί το υπολογιστικό κόστος σε πολύ χαμηλά επίπεδα. Έτσι, στην παρούσα εργασία, το μοντέλο Bouc - Wen χρησιμοποιείται για την φαινομενολογική αναπαράσταση της ελαστοπλαστικής συμπεριφοράς δομικών στοιχείων.

Από άποψη μαθηματικής δομής, το εν λόγω μοντέλο λειτουργεί σαν ένας μη γραμμικός τελεστής με μνημονικές ιδιότητες (τελεστής Volterra). Ουσιαστικά, αυτό υποδηλώνει ότι θεωρεί ως δεδομένα εισόδου την ιστορική πορεία της μετατόπισης και, μέσω της επίλυσης ενός συστήματος διαφορικών εξισώσεων, υπολογίζει τη στιγμιαία υστερητική πίεση. Η εκτενής μαθηματική του ευελιξία επιτρέπει τη δημιουργία μιας ποικιλίας υστερητικών μορφών, καθιστώντας το ιδανικό για πολλές εφαρμογές: από τη μοντελοποίηση δομικού χάλυβα και οπλισμένου σκυροδέματος, μέχρι την ανάλυση ειδικών αντισεισμικών συστημάτων, όπως οι αποσβεστήρες ενέργειας και τα ελαστομερή εφέδρανα σεισμικής μόνωσης.

2.3 Μαθηματική διατύπωση

Κατά την ανάλυση ενός τυπικού μονοβάθμιου συστήματος (SDOF), η συνολική αντίσταση $F(t)$ που δημιουργείται, αντιπροσωπεύεται ως το γραμμικό σύνολο δύο παράλληλων μηχανισμών: μιας καθαρής ελαστικής συνιστώσας και μιας συνιστώσας που υπόκειται στο φαινόμενο της υστέρησης.

Η μαθηματική σχέση που περιγράφει αυτή τη λειτουργία είναι:

$$F(t) = aK_0u(t) + (1-a) K_0 z(t)$$

Στην παραπάνω εξίσωση:

- Η μεταβλητή $u(t)$ αντιπροσωπεύει τη μετατόπιση του συστήματος σε δεδομένο χρόνο t .
- Η παράμετρος K_0 εκφράζει την αρχική ελαστική δυσκαμψία του στοιχείου.
- Ο αδιάστατος συντελεστής a ορίζει τη σχέση ανάμεσα στη μετελαστική δυσκαμψία και την αρχική δυσκαμψία K_0 (με τιμές μεταξύ 0 και 1). Όταν ο συντελεστής ισούται με τη μονάδα ($a=1$), το σύστημα αντιδρά με απόλυτα γραμμικό τρόπο, ενώ όταν εκμηδενίζεται ($a=0$), το μοντέλο αναπαριστά μια ιδανική ελαστοπλαστική απόκριση.
- Η μεταβλητή $z(t)$ αποτελεί μια εσωτερική, αδιάστατη μεταβλητή κατάστασης, η οποία είναι υπεύθυνη για την καταγραφή του ιστορικού των παραμορφώσεων («μνήμη» του συστήματος).

Η χρονική διαφορική εξέλιξη της υστερητικής μεταβλητής $z(t)$ ορίζεται από την ακόλουθη μη γραμμική διαφορική εξίσωση πρώτης τάξης:

$$\dot{z}(t) = \frac{\dot{u}(t)}{u_y} [A - |z(t)|^n (\gamma + \beta \operatorname{sgn}(\dot{u}(t)z(t)))]$$

Όπου ο όρος $\dot{u}(t)$ εκφράζει την ταχύτητα του συστήματος, η παράμετρος u_y τη μετατόπιση στο σημείο της διαρροής, η συνάρτηση $\operatorname{sgn}(\bullet)$ το πρόσημο του μεγέθους, ενώ οι σταθερές A , β , γ και n αποτελούν τις παραμέτρους που διαμορφώνουν τη συμπεριφορά του προσομοιώματος.

2.4 Φυσική σημασία παραμέτρων

Παρότι οι σταθερές A , β , γ και n δεν συνδέονται με άμεσα μετρήσιμα γεωμετρικά ή φυσικά μεγέθη, ο συνδυασμός τους καθορίζει τη γεωμετρία, το πλάτος και την κλίση των παραγόμενων υστερητικών βρόχων. Η μηχανική και μαθηματική τους ανάλυση συνοψίζεται στα παρακάτω σημεία:

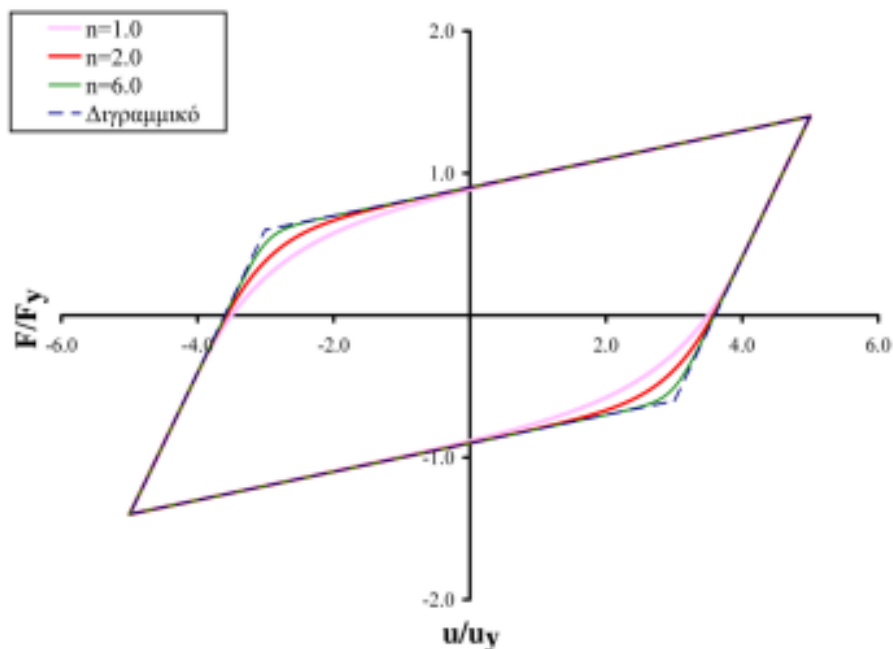
Παράμετρος A : Προσδιορίζει την κλίμακα της αρχικής αντοχής κατά την αρχή της φόρτισης. Σύμφωνα με τη σύγχρονη βιβλιογραφία, για να είναι το μοντέλο φυσικά συνεπές και η μακροσκοπική δυσκαμψία του βρόχου να ταυτίζεται με την πραγματική αρχική δυσκαμψία K_0 , είναι απαραίτητο να ορίζεται $A = 1$.

Παράμετροι β και γ : Αυτές οι δύο σταθερές ρυθμίζουν τη μορφή του βρόχου στις φάσεις φόρτισης, αποφόρτισης και επαναφόρτισης, καθορίζοντας αν το σύστημα εμφανίζει τάσεις κράτησης ή μάλαξης. Για να εξασφαλίζεται η θερμοδυναμική εγκυρότητα του μοντέλου (ώστε να μην δημιουργείται αυθαίρετη ενέργεια κατά τη διάρκεια των κύκλων), πρέπει να πληρείται η συνθήκη:

$$\beta + \gamma = 1$$

Στην περίπτωση όπου $\beta = \gamma = 0.5$, οι αποφορτιζόμενοι κλάδοι του συστήματος γίνονται ευθείες γραμμές.

Παράμετρος n : Η εκθετική παράμετρος n ρυθμίζει τη σταδιακή μετάβαση από την ελαστική στην πλαστική ζώνη. Προσεγγίζει το ιδανικά ελαστικό (διγραμμικό) μοντέλο, όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 1: Επίδραση της παραμέτρου n στη μορφή του υστερητικού βρόχου.

Από το παραπάνω σχήμα παρατηρείται ότι για $n=1$ ο βρόγχος είναι πλήρως στρογγυλεμένος, ενώ για $n=6$ πλησιάζει έντονα τη διγραμμική ελαστική συμπεριφορά.

Περιορισμοί και Εσωτερικές Αδυναμίες:

Κατά την υλοποίηση του μοντέλου Bouc-Wen, είναι σημαντικό να εξετάζονται ορισμένες μαθηματικές και φυσικές ασυνέπειες της αρχικής του μορφής: Απουσία καθαρά ελαστικής ζώνης: Το μοντέλο προκαλεί μικρές μόνιμες παραμορφώσεις και απώλεια ενέργειας ακόμη και σε πολύ χαμηλά επίπεδα ενεργοποίησης, όπου τα πραγματικά υλικά συμπεριφέρονται με εντελώς γραμμικό και ελαστικό τρόπο.

Παράβαση των κριτηρίων πλαστικότητας: Σε περιπτώσεις κυκλικών φορτίσεων με μικρούς εσωτερικούς κύκλους στην περιοχή απελευθέρωσης, το μοντέλο παρουσιάζει αφύσικα φαινόμενα, όπως τεχνητή συγκέντρωση παραμορφώσεων χωρίς ουσιαστική μεταβολή του φορτίου. Για να ξεπεραστούν αυτά τα ζητήματα, στην πράξη συνήθως εφαρμόζονται κατάλληλα προσαρμοσμένες παραλλαγές του μοντέλου.

3. Η ΧΡΗΣΗ ΤΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ SYMPLEGMMA DEVELOPMENT ENVIROMENT (SDE) ΓΙΑ ΤΗΝ ΕΠΙΛΥΣΗ ΕΝΟΣ ΣΥΣΤΗΜΑΤΟΣ ΤΑΛΑΝΤΩΣΗΣ

3.1 Γενική παρουσίαση του περιβάλλοντος

Το περιβάλλον υπολογισμού Symplegma Development Environment (SDE) είναι μια πλήρης και εξειδικευμένη πλατφόρμα λογισμικού, που στοχεύει στην επίλυση σύνθετων ζητημάτων της δομικής δυναμικής, των μη γραμμικών ταλαντώσεων και της υπολογιστικής σεισμικής μηχανικής. Δημιουργήθηκε με κύριο στόχο να δρα ως σύνδεσμος ανάμεσα στη θεωρητική έρευνας και την πρακτική εφαρμογή, επιτρέποντας ρεαλιστική προσομοίωση κατασκευαστικών φορέων που δέχονται ισχυρές δυναμικές καταπονήσεις, όπως οι σεισμικές διεγέρσεις.

Το SDE είναι ένα περιβάλλον βασισμένο σε Java, το οποίο χρησιμοποιεί τη γλώσσα Groovy ως κύρια γλώσσα για τη συγγραφή σεναρίων (scripting). Η Groovy είναι μια γλώσσα προγραμματισμού αντικειμενοστραφούς χαρακτήρα που λειτουργεί άμεσα στην εικονική μηχανή Java (Java Virtual Machine).

Η συγκεκριμένη επιλογή παρέχει στην πλατφόρμα SDE την άμεση πρόσβαση σε βιβλιοθήκες: Το λογισμικό εκμεταλλεύεται το ευρύ οικοσύστημα της Java, ενσωματώνοντας πολύ γρήγορους αλγόριθμους γραμμικής άλγεβρας που είναι κρίσιμοι για την αναστροφή μητρών δυσκαμψίας και τον υπολογισμό ιδιοτιμών μεγάλων δομικών συστημάτων.

Στο περιβάλλον SDE, η δημιουργία ενός μοντέλου για ένα σύστημα ταλάντωσης γίνεται μέσω του αντικειμενοστραφούς καθορισμού των μηχανικών του χαρακτηριστικών (μάζα, απόσβεση, ελαστική δυσκαμψία). Ταυτόχρονα, η εφαρμογή διευκολύνει την ενσωμάτωση εξωτερικών δεδομένων διέγερσης με τη μορφή χρονοϊστοριών επιτάχυνσης (επιταχυνσιογραφήματα).

Το κυριότερο πλεονέκτημα του SDE, που εκμεταλλεύεται πλήρως στην παρούσα μελέτη, είναι η χρήση ειδικών κατηγοριών για την ενσωμάτωση διαφορικών χειριστών μνήμης. Αυτό επιτρέπει τη σύνδεση των εξισώσεων κίνησης του φορέα με τη μη γραμμική διαφορική εξίσωση του υστερητικού προτύπου Bouc-Wen. Η εν λόγω αντικειμενοστραφής δομή διασφαλίζει εξαιρετικά χαμηλό υπολογιστικό κόστος και μεγάλη ταχύτητα επεξεργασίας δεδομένων, χαρακτηριστικά που καθιστούν το SDE ιδανικό μέσο για τις απαιτητικές επαναληπτικές διαδικασίες βελτιστοποίησης και ταυτοποίησης των παραμέτρων του μοντέλου.

3.2 Μέθοδοι επίλυσης

Η διαφορική εξίσωση κίνησης ενός μη γραμμικού μονοβάθμιου συστήματος (SDOF) υπό την επίδραση σεισμικής επιτάχυνσης εδάφους $\ddot{u}_g(t)$ διατυπώνεται ως:

$$m\ddot{u}(t) + c\dot{u}(t) + F(t) = -m\ddot{u}_g(t)$$

Όπου m αντιπροσωπεύει τη μάζα, c τον συντελεστή απόσβεσης και $F(t)$ την συνολική δύναμη επαναφοράς, η οποία, σύμφωνα με το μοντέλο Bouc-Wen, εξαρτάται από την μετατόπιση u και την εσωτερική μεταβλητή κατάστασης z . Εξαιτίας της έντονα μη γραμμικής φύσης του όρου $F(t)$, η εξίσωση αυτή αδυνατεί να λυθεί με αναλυτικές (κλειστές) μαθηματικές προσεγγίσεις. Κατά συνέπεια, το SDE χρησιμοποιεί μεθόδους ολοκλήρωσης αριθμητικής βηματικής στο χρονικό πεδίο (time-stepping integration methods).

Οι διαθέσιμοι αλγόριθμοι στο SDE χωρίζονται σε δύο κύριες κατηγορίες, καθεμία από τις οποίες εξυπηρετεί διαφορετικές υπολογιστικές ανάγκες.

3.2.1. Άμεσες Μέθοδοι Ολοκλήρωσης (Direct Integration Methods)

Οι μέθοδοι αυτοί βασίζονται στην ικανοποίηση της εξίσωσης κίνησης σε διακριτές χρονικές στιγμές t_i , με σταθερό χρονικό βήμα $\Delta t = t_{i+1} - t_i$.

- **Μέθοδος Newmark (Newmark's β Method):** Βασίζεται στην παραδοχή ότι η ταχύτητα και η μετατόπιση στο επόμενο χρονικό βήμα ($i+1$) μπορούν να εκφραστούν συναρτήσει των μεγεθών του παρόντος βήματος (i) και της επιτάχυνσης, μέσω των σχέσεων:

$$\dot{u}_{i+1} = \dot{u}_i + [(1 - \gamma)\Delta t]\ddot{u}_i + (\gamma\Delta t)\ddot{u}_{i+1}$$

$$u_{i+1} = u_i + \Delta t\dot{u}_i + \left[\left(\frac{1}{2} - \beta\right)\Delta t^2\right]\ddot{u}_i + (\beta\Delta t^2)\ddot{u}_{i+1}$$

Στο SDE, εφαρμόζεται κυρίως η παραλλαγή της Μέσης Επιτάχυνσης ($\gamma = 0.5$ και $\beta = 0.25$). Αυτή η επιλογή κάνει τη μέθοδο έμμεση και εξαιρετικά αξιόπιστη. Αυτό υποδηλώνει ότι ο αλγόριθμος δεν θα παρεκκλίνει αριθμητικά ποτέ, ανεξάρτητα από το μέγεθος του χρονικού βήματος Δt , διασφαλίζοντας σταθερότητα ακόμη και σε έντονες μη γραμμικότητες.

- **Μέθοδος της Κεντρικής Διαφοράς (Central Difference Method):** Πρόκειται για μια άμεση μέθοδο, όπου οι παραγωγοί της μετατόπισης (ταχύτητα και επιτάχυνση) προσεγγίζονται μέσω πεπερασμένων διαφορών:

$$\dot{u}_i = \frac{u_{i+1} - u_{i-1}}{2\Delta t}, \quad \ddot{u}_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta t^2}$$

Η διαδικασία αυτή είναι ιδιαίτερα ταχεία ανά επανάληψη, καθώς δεν χρειάζεται την επίλυση συστήματος πινάκων στο βήμα $i+1$. Ωστόσο, παρουσιάζει περιορισμένη ακρίβεια. Για να αποφευχθούν μαθηματικές εκρήξεις και εσφαλμένα αποτελέσματα, το χρονικό βήμα θα πρέπει να είναι αυστηρά μικρότερο από μια κρίσιμη τιμή ($\Delta t < \Delta t_{cr} = T_n/\pi$, όπου T_n η φυσική περίοδος του συστήματος).

3.2.2. Μέθοδοι της Οικογένειας Runge-Kutta

Η εισαγωγή του μοντέλου Bouc-Wen απαιτεί την ταυτόχρονη επίλυση μιας επιπλέον διαφορικής εξίσωσης πρώτης τάξης για την υστερητική μεταβλητή $\dot{z} = f(u, \dot{u}, z)$. Για αυτό το λόγο οι κλασικές μέθοδοι δομοστατικής μηχανικής (όπως ο Newmark) συχνά συνδυάζονται ή αντικαθίστανται στο SDE από αλγόριθμους Runge-Kutta.

- **Κλασικός αλγόριθμος Runge-Kutta 4ης Τάξης:** Αυτός ο αλγόριθμος εκτιμά την τιμή της μεταβλητής στο επόμενο βήμα, υπολογίζοντας τέσσερις ξεχωριστές κλίσεις (k_1, k_2, k_3, k_4) μέσα στο διάστημα Δt και χρησιμοποιώντας έναν σταθμισμένο μέσο όρο αυτών.

Προβλέπει εξαιρετική ακρίβεια. Αποτελεί την κατάλληλη επιλογή για την παρακολούθηση της προόδου της μεταβλητής $z(t)$, καθώς αποτρέπει το σφάλμα «ολίσθησης» εκτός των ορίων του υστερητικού βρόχου.

- **Προσαρμοστικές Μέθοδοι (Adaptive Runge-Kutta-Fehlberg):** Το SDE περιλαμβάνει επίσης προχωρημένους αλγόριθμους μεταβλητής χρονικής διάρκειας. Οι αλγόριθμοι αυτοί εξετάζουν τα αποτελέσματα μιας μεθόδου 4ης και 5ης τάξης για να εκτιμήσουν το στιγμιαίο υπολογιστικό λάθος. Εάν το σφάλμα ξεπεράσει μια καθορισμένη ανοχή, το λογισμικό αυτόματα μειώνει το χρονικό βήμα Δt σε περιοχές όπου ο υστερητικός βρόχος παρουσιάζει απότομη κλίση (όπως σε σημεία διαρροής ή αναστροφής της φόρτισης), ενώ το αυξάνει στην ελαστική περιοχή, βελτιώνοντας έτσι τον συνολικό χρόνο ανάλυσης.
- Στην επίλυση του SDE, η επιλογή του σωστού χρονικού βήματος ολοκλήρωσης (Δt) είναι συνεχώς σημαντική. Ένα πολύ μεγάλο βήμα μπορεί να προκαλέσει αριθμητική αστάθεια ή τεχνητή απώλεια ενέργειας (αριθμητική απόσβεση), παραμορφώνοντας τη μορφή των υστερητικών βρόχων, ενώ ένα υπερβολικά μικρό βήμα αυξάνει αδικαιολόγητα τον χρόνο υπολογισμού, κυρίως κατά την εκτέλεση επαναληπτικών αλγορίθμων βελτιστοποίησης και ταυτοποίησης.

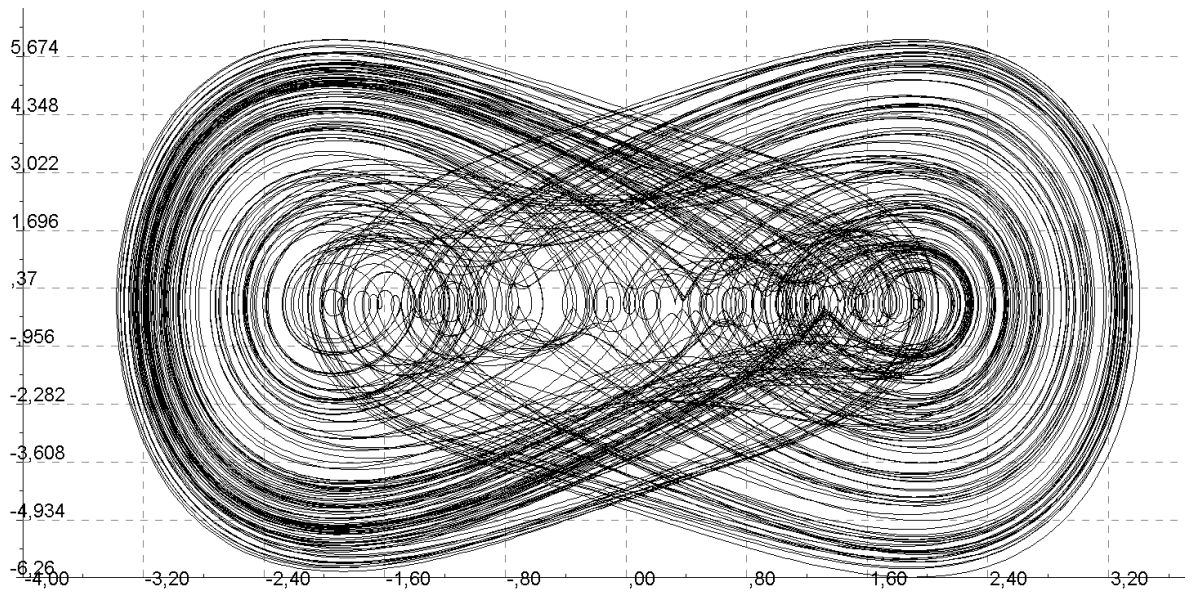
3.2.3. Αριθμητική Επίλυση Μη Γραμμικών Συστημάτων: Ο Ταλαντωτής Duffing στο SDE

Για να διαπιστωθεί η ικανότητα του περιβάλλοντος SDE στη διαχείριση και επίλυση μη γραμμικών δυναμικών προβλημάτων, πρώτα μελετάται ο κλασικός μη γραμμικός ταλαντωτής Duffing. Η διαφορική εξίσωση που περιγράφει την κίνηση του συστήματος με κυβικό όρο δυσκαμψίας μ διατυπώνεται ως εξής:

$$\ddot{u} + \frac{c}{m}\dot{u} + \frac{k}{m}u + \frac{\mu}{m}u^3 = \frac{f(t)}{m}$$

Όπου m η μάζα, c ο συντελεστής απόσβεσης, k η γραμμική δυσκαμψία, μ η παράμετρος της μη γραμμικής κυβικής δυσκαμψίας και $f(t) = \gamma \cos(\omega t)$ η εξωτερική αρμονική διεγερτική δύναμη.

Για την αριθμητική ολοκλήρωση στο SDE, η μη γραμμική διαφορική εξίσωση 2ης τάξης μετασχηματίζεται σε ένα ισοδύναμο σύστημα δύο διαφορικών εξισώσεων 1ης τάξης, μέσω της εισαγωγής των μεταβλητών κατάστασης $y[0] = u$ (μετατόπιση) και $y[1] = \dot{u}$ (ταχύτητα). Το σύστημα αποκτά τη μορφή που φαίνεται στην παράγραφο Α του Παραρτήματος.



Εικόνα 2: Διάγραμμα φάσης (u-v) του ταλαντωτή Duffing.

Το παραπάνω διάγραμμα αποτυπώνει τη σχέση μεταξύ της μετατόπισης και της ταχύτητας ενός μη γραμμικού ταλαντωτή Duffing που επιλύθηκε αριθμητικά με τη μέθοδο Runge-Kutta 4ης τάξης. Στον οριζόντιο άξονα (X) οι τιμές της μετατόπισης κυμαίνονται από -4.00 έως +3.20. Η αρχική μετατόπιση του συστήματος στον κώδικα έχει οριστεί ίση με $u_0 = 3.1$. Στον κατακόρυφο άξονα (Y) οι τιμές της ταχύτητας κυμαίνονται από -6.26 έως +5.674. Η αρχική ταχύτητα του συστήματος στον κώδικα έχει οριστεί ίση με $\dot{u}_0 = 4.1$.

3.2.4. Αριθμητική Υλοποίηση Ιδανικά Ελαστοπλαστικού Μοντέλου SDOF στο SDE

Εκτός από τις διαρκείς μη γραμμικότητες, το περιβάλλον SDE χρησιμοποιείται για την προσομοίωση συστημάτων με ανελαστική συμπεριφορά καθυστέρησης, όπως το ιδανικό ελαστοπλαστικό μοντέλο μονοβάθμιου ταλαντωτή (SDOF). Η αριθμητική επίλυση των εξισώσεων κίνησης για το συγκεκριμένο ζήτημα στηρίζεται στη μέθοδο Newmark, η οποία εξετάστηκε θεωρητικά στην ενότητα 3.2.1. Για την υπολογιστική εφαρμογή του μοντέλου χρησιμοποιούνται οι τυπικές παράμετροι σταθερής επιτάχυνσης $\gamma = 1/2$ και $\beta = 1/4$.

Σε κάθε χρονικό βήμα Δt , η ενεργός δυσκαμψία k_{ef} και το ενεργό φορτίο f_{ef} του συστήματος διαμορφώνονται σύμφωνα με τις σχέσεις:

$$k_{ef} = k_{ep} + \frac{m}{\beta \Delta t^2} + \frac{c\gamma}{\beta \Delta t}$$

$$f_{ef} = f(t_i) - f(t_{i-1}) + m \left(\frac{1}{\beta \Delta t} \dot{u}_{i-1} + \frac{1}{2\beta} \ddot{u}_{i-1} \right) + c \left(\frac{\gamma}{\beta} \dot{u}_{i-1} + \Delta t \left(\frac{\gamma}{2\beta} - 1 \right) \ddot{u}_{i-1} \right)$$

Όπου k_{ep} είναι η στιγμιαία ελαστοπλαστική δυσκαμψία του συστήματος. Ο αλγόριθμος ελέγχει τη συνθήκη διαρροής (yield) σε κάθε βήμα:

$$yield = k_h * u_i + (k - k_h) * u_y * sgn(\dot{u}_i)$$

- **Ελαστική Περιοχή:** Όσο το σύστημα παραμένει εντός των ορίων διαρροής, η στιγμιαία δυσκαμψία ισούται με την αρχική ελαστική δυσκαμψία ($k_{ep} = k$).

- **Πλαστική Περιοχή:** Με την υπέρβαση του ορίου διαρροής, το σύστημα εισέρχεται σε κατάσταση πλαστικότητας, η στιγμιαία δυσκαμψία μεταπίπτει στη μεθελαστική δυσκαμψία ($k_{ep} = k_h = 0.25k$), ενώ υπολογίζεται η πλαστική παραμόρφωση.

Η προαναφερόμενη αριθμητική διαδικασία επιτρέπει την καταγραφή του διαγράμματος υστέρησης, αναδεικνύοντας τις ξαφνικές αλλαγές της δυσκαμψίας κατά την αποφόρτιση και επαναφόρτιση, κάτι που καθιστά αναγκαία την ενσωμάτωση πιο προχωρημένων, ομαλών μοντέλων υστέρησης, όπως το μοντέλο Bouc-Wen που μελετάται στη παρούσα εργασία.

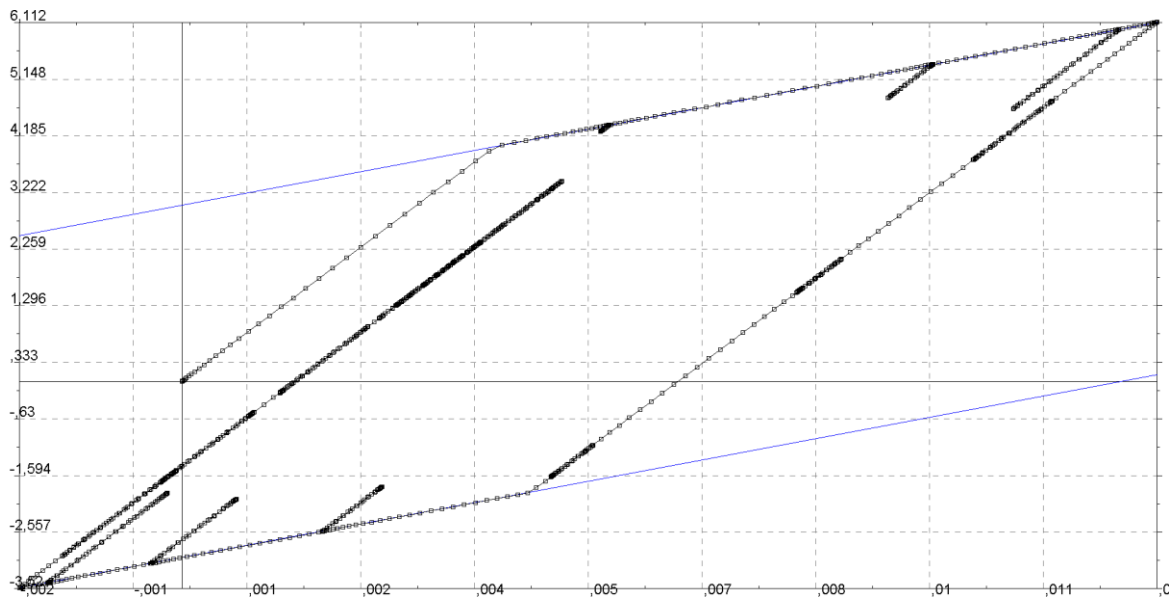
Η υπολογιστική αποτύπωση της παραπάνω αριθμητικής διαδικασίας στο περιβάλλον SDE πραγματοποιείται με τη συγγραφή κατάλληλου σεναρίου κώδικα (Groovy Script). Στον κώδικα ορίζεται η χρονική διάρκεια της ανάλυσης μέσω $N=1000$ βημάτων με σταθερό βήμα ολοκλήρωσης $dt = 0.001$ sec, καθώς και η εξωτερική ημιτονοειδής δυναμική διέγερση $f(t) = 100 * \sin(10.5\sqrt{k/m} * t)$

Για την παρακολούθηση της ελαστοπλαστικής συμπεριφοράς, εισάγεται η λογική μεταβλητή ελέγχου plastic, η οποία καθορίζει τη μετάπτωση της δυσκαμψίας από την ελαστική τιμή (k) στη μεθελαστική (k_h). Κατά τη φάση της αποφόρτισης, η αλλαγή προσήμου στην ταχύτητα ($\dot{u}[it] * \dot{u}[it-1] < 0$) σηματοδοτεί την επιστροφή του συστήματος σε ελαστική συμπεριφορά. Στο τέλος της αριθμητικής διαδικασίας, ενεργοποιούνται οι εγγενείς κλάσεις σχεδίασης του SDE (PlotFrame και plotfunction) για την παραγωγή του διαγράμματος υστερητικού βρόχου ($u - F$) και των χρονοϊστοριών μετατόπισης και ταχύτητας.

Η πλήρης υπολογιστική δομή του αλγορίθμου Newmark για το ιδανικά ελαστοπλαστικό SDOF σύστημα παρατίθεται στον κώδικα που βρίσκεται στην παράγραφο Β του Παραρτήματος.

Το παρακάτω διάγραμμα παράγεται από την εντολή `ufPlot.show()` και δείχνει τη σχέση Μετατόπισης - Δύναμης Επαναφοράς. Το διάγραμμα αυτό στον οριζόντιο άξονα (X) απεικονίζει την μετατόπιση, της οποίας οι τιμές κυμαίνονται από -0.002 m έως $+0.011$ m. Στον κατακόρυφο Άξονα (Y) αντιπροσωπεύει τη δύναμη (F_s) του ελαστηρίου, με τιμές από -3.65 kN έως $+6.11$ kN. Οι δύο μπλε λοξές γραμμές απεικονίζουν τη μεθελαστική κράτυνση του συστήματος με κλίση $k_h = 0.25 * k = 250$ kN/m.

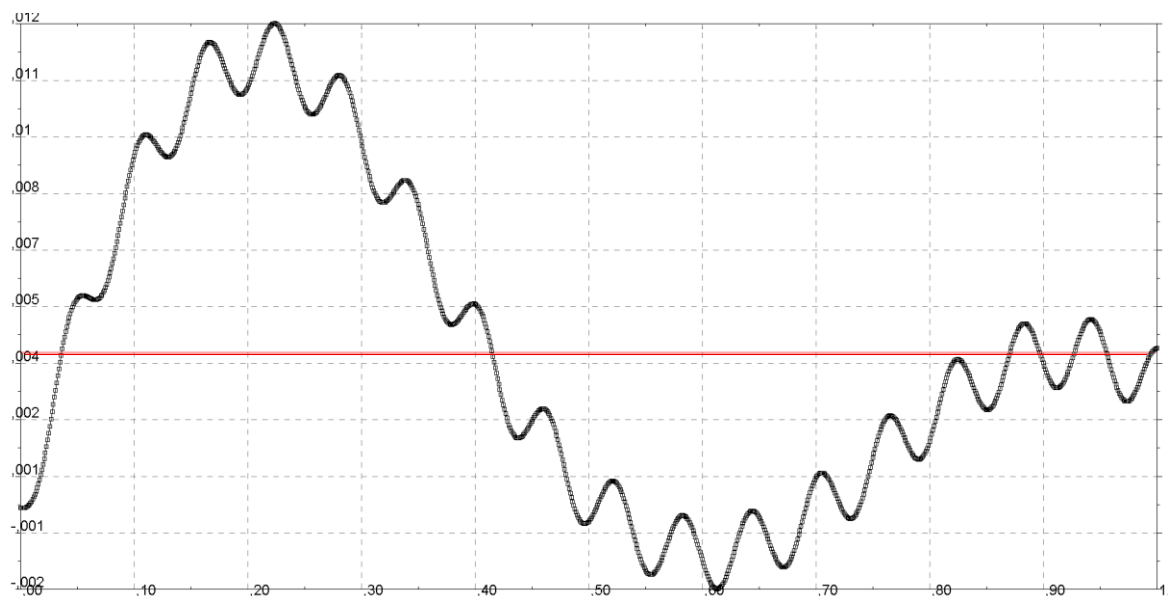
Από το παρακάτω διάγραμμα μπορούμε να διακρίνουμε τη γραμμική τμηματική ελαστοπλαστική συμπεριφορά. Το σύστημα ξεκινάει ελαστικά από το (0,0) με μεγάλη κλίση $k=1000$. Μόλις η δύναμη φτάσει στο $F_{sy}= 4.0$ kN (και η μετατόπιση στο $u_y = 0.004$), το σύστημα διαρρέει και η κλίση του πέφτει απότομα, ακολουθώντας τη μπλε γραμμή της κράτυνσης (k_h). Όταν η ταχύτητα μηδενίζεται και αλλάζει πρόσημο, το σύστημα κάνει αποφόρτιση παράλληλα προς την αρχική ελαστική κλίση k , μέχρι να ξανά διαρρεύσει προς την αντίθετη κατεύθυνση.



Εικόνα 3: Διάγραμμα υστερητικού βρόγχου (u - F_s) ελαστοπλαστικού SDOF.

Το παρακάτω διάγραμμα παράγεται από την εντολή `tuPlot.show()` και απεικονίζει τη χρονοϊστορία μετατόπισης του συστήματος. Στον οριζόντιο άξονα (X) απεικονίζεται ο χρόνος με τιμές από 0.0 έως 1.0 δευτερόλεπτο. Ο κατακόρυφος άξονας (Y) δείχνει τη μετατόπιση της μάζας σε μέτρα, με τιμές από -0.002 m έως +0.012 m. Η κόκκινη οριζόντια γραμμή αντιπροσωπεύει το θετικό όριο ελαστικής διαρροής του υλικού. Βρίσκεται ακριβώς στο $u_y = \pm 0.004$ m.

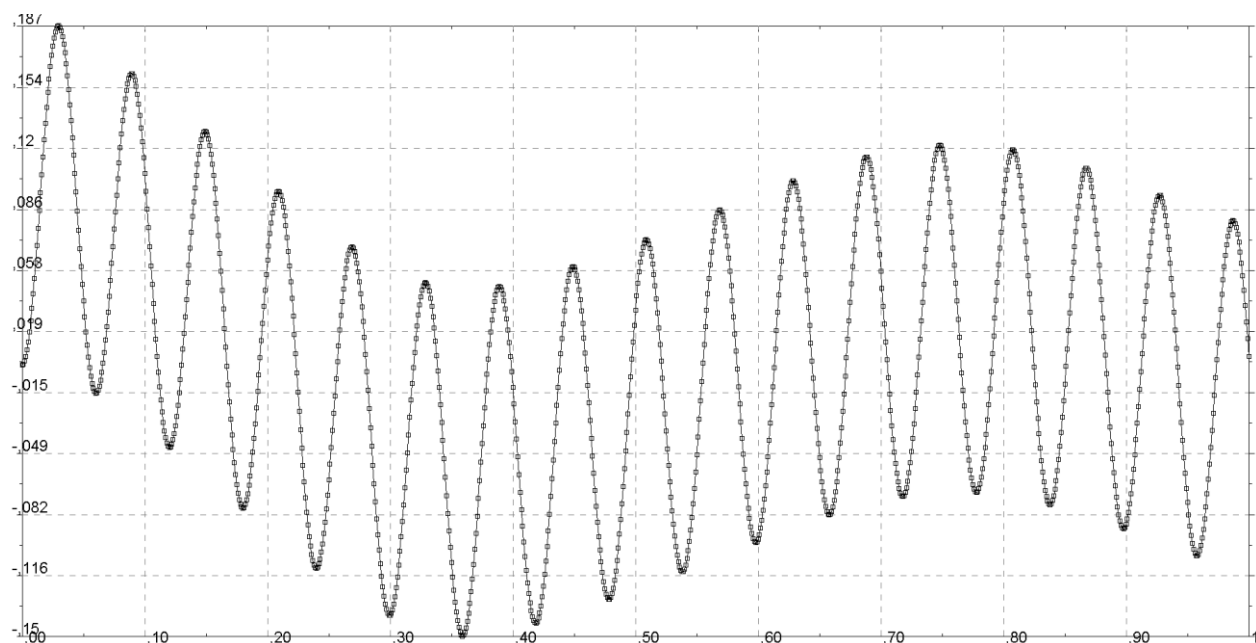
Η μαύρη καμπύλη ξεπερνά το όριο διαρροής στα πρώτα βήματα της ανάλυσης, φτάνοντας μέγιστη τιμή περίπου +0.012 m. Στη συνέχεια το σύστημα αποκρίνεται με φθίνουσα ταλάντωση γύρω από μία μόνιμη πλαστική παραμόρφωση περίπου +0.004 m, όπου σταθεροποιείται στο τέλος της ανάλυσης ($t=1.0$ sec).



Εικόνα 4: Χρονοϊστορία μετατόπισης $u(t)$ ελαστοπλαστικού SDOF.

Το επόμενο διάγραμμα είναι η χρονοϊστορία ταχύτητας του συστήματός κατά τη διάρκεια της δυναμικής του ταλαντωτή. Δείχνει πώς μεταβάλλεται η ταχύτητα της μάζας του μονοβάθμιου (SDOF) ελαστοπλαστικού ταλαντωτή σε συνάρτηση με τον χρόνο. Το γράφημα αυτό παράγεται από την εντολή `tvPlot.show()` και στον οριζόντιο άξονα (X) όπως και στο προηγούμενο διάγραμμα της μετατόπισης, ο χρόνος εκτείνεται από 0 έως 1,0 δευτερόλεπτο. Στον κατακόρυφο άξονα (Y), ο οποίος αντιπροσωπεύει την ταχύτητα κίνησης της μάζας, εκφρασμένη σε μέτρα ανά δευτερόλεπτο (m/sec). Οι τιμές της κυμαίνονται από περίπου -0.150 m/sec (αρνητική μέγιστη ταχύτητα) έως +0.187 m/sec (θετική μέγιστη ταχύτητα).

Κάθε φορά που η μαύρη καμπύλη τέμνει την οριζόντια γραμμή του μηδενός ($v = 0$), σημαίνει ότι η μάζα σταματά στιγμιαία για να αλλάξει κατεύθυνση κίνησης (αναστροφή φόρτισης). Στο ελαστοπλαστικό μοντέλο, αυτά τα σημεία είναι κρίσιμα, διότι η αλλαγή προσήμου στην ταχύτητα ενεργοποιεί την αποφόρτιση του συστήματος, επαναφέροντας τη δυσκαμψία από την πλαστική τιμή (k_h) στην αρχική ελαστική τιμή (k).



Εικόνα 5: Χρονοϊστορία ταχύτητας $v(t)$ ελαστοπλαστικού SDOF.

Τα παραπάνω διαγράμματα των εικόνων 3, 4 και 5 απεικονίζουν την πλήρη κινηματική εικόνα της απόκρισης του φορέα που επιλύθηκε με τη μέθοδο Newmark.

Η οπτικοποίηση των αποτελεσμάτων μέσω του παραπάνω κώδικα επιτρέπει την άμεση αξιολόγηση της ανελαστικής συμπεριφοράς του συστήματος, αποτυπώνοντας τις χαρακτηριστικές απότομες μεταβολές της δυσκαμψίας. Τα αποτελέσματα αυτά καταδεικνύουν την ανάγκη μετάβασης σε πιο σύνθετα και συνεχή μοντελοποιήματα υστέρησης, όπως το προσομοίωμα Bouc-Wen,. Η θεωρητική ανάλυση του εν λόγω μοντέλου παρουσιάστηκε αναλυτικά στο προηγούμενο κεφάλαιο (κεφάλαιο 2), ενώ η υπολογιστική αναγνώριση και βελτιστοποίηση των παραμέτρων αποτελεί το αντικείμενο του επόμενου κεφαλαίου (κεφάλαιο 4).

4. ΑΝΑΓΝΩΡΙΣΗ ΠΑΡΑΜΕΤΡΩΝ

4.1 Γενικά

Η διαδικασία αναγνώρισης συστημάτων (System Identification) στο μοντέλο Bouc-Wen περιλαμβάνει τον καθορισμό ενός διανύσματος αδιευκρίνιστων αριθμητικών παραμέτρων, στηριζόμενο σε διαθέσιμα δεδομένα απόκρισης (είτε πειραματικά είτε συνθετικά αναφοράς). Λόγω του φαινομενολογικού χαρακτήρα του μοντέλου, οι παράμετροι αυτοί δεν προέρχονται από γεωμετρικές μετρήσεις, αλλά πρέπει να υπολογιστούν έτσι ώστε η αριθμητική ανάλυση να προσεγγίζει όσο το δυνατόν καλύτερα την πραγματική συμπεριφορά του υλικού.

Για αυτήν την ανάγκη, το ζήτημα διατυπώνεται ως πρόβλημα μαθηματικής βελτιστοποίησης. Ορίζεται μια αντικειμενική συνάρτηση σφάλματος $J(p)$, η οποία αντιπροσωπεύει το μέσο τετραγωνικό σφάλμα ανάμεσα στη χρονοϊστορία μετατόπισης αναφοράς u_{ref} και στην αριθμητική μετατόπιση u_{num} που υπολογίζεται με το λογισμικό SDE για ένα συγκεκριμένο διάνυσμα παραμέτρων p :

$$J(p) = \frac{1}{N} \sum_{i=1}^N [u_{ref}(t_i) - u_{num}(t_i, p)]^2$$

Όπου N αντιπροσωπεύει το συνολικό αριθμό των διακριτών χρονικών βημάτων στην ανάλυση δυναμικής. Το διάνυσμα των παραμέτρων που θα ταυτοποιηθούν περιλαμβάνει τις πέντε κύριες μεταβλητές του υστερητικού τελεστή:

$$p = [\alpha, A, \beta, n, \gamma]^T$$

Σε συγκεκριμένες υπολογιστικές εφαρμογές της παρούσας μελέτης, η σταθερά A εκλαμβάνεται ως σταθερή και ίση με τη μονάδα ($A=1.0$) ώστε να διασφαλιστεί η φυσική συνέπεια της αρχικής δυσκαμψίας, περιορίζοντας το ζήτημα σε τέσσερις παραμέτρους.

$$p = [\alpha, \beta, n, \gamma]^T$$

Η μείωση αυτή περιορίζει την περιοχή αναζήτησης, διευκολύνει την σύγκλιση του αλγόριθμου και μειώνει σημαντικά το υπολογιστικό κόστος.

4.2 Μέθοδοι βελτιστοποίησης

Για να ελαχιστοποιηθεί η συνάρτηση σφάλματος $J(p)$, η κλασική μαθηματική ανάλυση (μέθοδοι που στηρίζονται στην παράγωγο) συχνά αποτυγχάνει εξαιτίας της έντονης μη γραμμικότητας και της ύπαρξης πολλών τοπικών ελαχίστων στην επιφάνεια σφάλματος του Bouc-Wen.

4.2.1. Διαφορική Εξέλιξη (Differential Evolution):

Ένας εξαιρετικά δυνατός αλγόριθμος εξέλιξης που στηρίζεται σε πληθυσμούς. Μέσω διαδικασιών μετάλλαξης, διασταύρωσης και επιλογής, ο αλγόριθμος ερευνά τον χώρο των παραμέτρων, επιδιώκοντας το παγκόσμιο ελάχιστο χωρίς να χρειάζεται πληροφορίες παραγώγων. Αναλυτικά:

- Σε κάθε μετάλλαξη, το διάνυσμα-στόχος του πληθυσμού, παράγει ένα μεταλλαγμένο διάνυσμα προσθέτοντας τη σταθμισμένη διαφορά δύο τυχαίων διανυσμάτων σε ένα τρίτο.
- Στο στάδιο της διασταύρωσης, τα δεδομένα του μεταλλαγμένου διανύσματος συνδυάζονται με το διάνυσμα-στόχο και δημιουργούν ένα πειραματικό διάνυσμα.
- Στο τέλος της διαδικασίας επιλογής, το δοκιμαστικό διάνυσμα εισάγεται ως παράμετρος στον αριθμητικό επιλυτή Runge-Kutta του SDE, που πραγματοποιεί τη χρονοϊστορία και επιστρέφει την τιμή του σφάλματος $J(p)$. Αν η καινούργια δοκιμαστική λύση έχει μικρότερο σφάλμα από την τρέχουσα, την αντικαθιστά στην επόμενη γενιά, διαφορετικά απορρίπτεται.

4.2.2. Βελτιστοποίηση Σμήνους Σωματιδίων (Particle Swarm Optimization - PSO)

Αυτή η μέθοδος αναπαριστά τη συλλογική συμπεριφορά και τη δυναμική επικοινωνία κοινωνικών ομάδων στη φύση, όπου κάθε «σωματίδιο» συμβολίζει μια υποψήφια λύση (με θέση x_i και ταχύτητα v_i) που μετακινείται στο χώρο των παραμέτρων, επηρεαζόμενο από την προσωπική του βέλτιστη θέση και τη συνολική βέλτιστη θέση του σμήνους.

4.3 Υπολογιστική Υλοποίηση και Παραμετροποίηση των Scripts στο SDE

Στην τρέχουσα ενότητα εξετάζεται η δομή των κωδίκων σεναρίων (Groovy Scripts) που δημιουργήθηκαν στο υπολογιστικό περιβάλλον SDE για την εφαρμογή της μεθοδολογίας αναγνώρισης. Αυτά τα σενάρια διακρίνονται σε δύο κύριες υπολογιστικές μεθόδους, ανάλογα με τη φύση του ζητήματος ταυτοποίησης και αποτελούν τον υπολογιστικό πυρήνα, στον οποίο θεμελιώθηκαν τα πρακτικά παραδείγματα εφαρμογής του επόμενου κεφαλαίου.

4.3.1. Διαδικασία ταυτοποίησης από μοντέλο Bouc - Wen σε μοντέλο Bouc - Wen

Η μέθοδος αυτή είναι μια ελεγχόμενη μαθηματική εξέταση, στην οποία ως δεδομένα αναφοράς για τις αποκρίσεις χρησιμοποιούνται συνθετικά δεδομένα που έχουν παραχθεί από ένα γνωστό εκ των προτέρων υστερητικό σύστημα Bouc - Wen. Ο στόχος της διαδικασίας είναι η εκτίμηση της μαθηματικής σύγκλισης και της ακρίβειας των αλγορίθμων βελτιστοποίησης του SDE, ελέγχοντας αν ο μηχανισμός ελαχιστοποίησης μπορεί να αναγνωρίσει τις πραγματικές «κρυμμένες» παραμέτρους. Μέσα σε αυτό το πλαίσιο αναλύονται δύο υπολογιστικές παραλλαγές:

- Η καθολική αναγνώριση 5 παραμέτρων, όπου καθορίζεται ένα ολοκληρωμένο διάνυσμα πέντε αγνώστων $p = [\alpha, A, \beta, n, \gamma]^T$. Η αντικειμενική συνάρτηση σφάλματος Ra διαμορφώνεται με τρόπο που να υπολογίζει ταυτόχρονα το τετράγωνο σφάλματος της μετατόπισης u και της ταχύτητας v , εισάγοντας συντελεστές βαρύτητας.

$$J(p) = \sum_{i=1}^N \omega_u [u_{ref}(t_i) - u_{num}(t_i, p)]^2 + \omega_v [v_{ref}(t_i) - v_{num}(t_i, p)]^2$$

Η πλήρη υπολογιστική δομή του κώδικα αυτού φαίνεται στην παράγραφο Γ του Παραρτήματος.

- Η περιορισμένη αναγνώριση 4 παραμέτρων με στόχο τη μείωση της μαθηματικής αβεβαιότητας και την αποφυγή της μη-μοναδικότητας των λύσεων, σχεδιάστηκε η παραλλαγή τεσσάρων παραμέτρων $p = [\alpha, \beta, n, \gamma]^T$, όπου η σταθερά A ορίζεται στη μονάδα ($A=1$). Στον παρόντα κώδικα, η αντικειμενική συνάρτηση R_a απλουστεύεται υπολογίζοντας το τετραγωνικό σφάλμα μόνο πάνω στο σήμα της μετατόπισης u .

Η πλήρη υπολογιστική δομή του κώδικα αυτού παρουσιάζεται στην παράγραφο Δ του Παραρτήματος.

4.3.2. Διαδικασία ταυτοποίησης από ελαστοπλαστικό μοντέλο σε μοντέλο Bouc – Wen

Στη συγκεκριμένη υπολογιστική μέθοδο, το σύστημα αναφοράς (στόχος) δεν είναι ένα ομαλό υστερητικό μοντέλο, αλλά ένας διγραμμικός ιδανικά ελαστοπλαστικός φορέας, ο οποίος επιδεικνύει απότομες και ασυνεχείς μεταβολές δυσκαμψίας κατά τη διαδικασία της διαρροής και της αποφόρτισης. Σκοπός της παρούσας διαδικασίας είναι να εξετάσει την ικανότητα του λείου φαινομενολογικού μοντέλου Bouc-Wen να προσομοιώσει με ακρίβεια αυτές τις έντονες ασυνέχειες και να αντικαταστήσει υπολογιστικά τον ελαστοπλαστικό φορέα, αναπαράγοντας πιστά την κινηματική διαδρομή της κατασκευής.

Η υπολογιστική ροή περιλαμβάνει στην αρχή τη φάση της προσομοίωσης, κατά την οποία χρησιμοποιείται ο αλγόριθμος Newmark για τον προσδιορισμό της ελαστοπλαστικής μετατόπισης αναφοράς (u_{ref}) υπό την καθορισμένη δυναμική φόρτιση. Ακολούθως, πραγματοποιείται η αντίστροφη ανάλυση και αναγνώριση μέσω του εξελικτικού αλγορίθμου Διαφορικής Εξέλιξης (Differential Evolution) για τον καθορισμό τεσσάρων παραμέτρων: $s = [\alpha, n, \beta, \gamma]^T$.

Σε αυτό το στάδιο, η παράμετρος A σταθεροποιείται και ορίζεται σε μία μονάδα ($A=1.0$). Αυτή η παραδοχή εξασφαλίζει τη διατήρηση της αρχικής ελαστικής κλίσης του φορέα και την πλήρη αντιστοιχία της με την ελαστική δυσκαμψία k του αρχικού συστήματος, ενώ ταυτόχρονα εξαλείφει θέματα μαθηματικού πλεονασμού.

Η αντικειμενική συνάρτηση R_a που ελαχιστοποιείται κατά τη διαδικασία της βελτιστοποίησης καθορίζεται μέσω της εξής συνάρτησης:

$$J(p) = \int_0^T |u_{ref}(t) - u_{num}(t, p)| dt$$

Στην παράγραφο Ε του παραρτήματος απεικονίζεται η υπολογιστική διάρθρωση του κώδικα, όπως αναπτύχθηκε στο περιβάλλον SDE για την εκτέλεση αυτής της ταυτοποίησης.

5. ΠΑΡΑΔΕΙΓΜΑΤΑ ΕΦΑΡΜΟΓΗΣ

5.1 Γενικά

Στο παρόν κεφάλαιο πραγματοποιείται η εφαρμογή και η αξιολόγηση της υπολογιστικής μεθόδου αναγνώρισης παραμέτρων που αναπτύχθηκε στο περιβάλλον Symplegma Development Environment (SDE). Η διαδικασία της αντίστροφης ανάλυσης εστιάζει στην πλήρη αντικατάσταση της απόκρισης ενός ιδανικά ελαστοπλαστικού μονοβάθμιου συστήματος από το μοντέλο Bouc – Wen.

Για την αξιολόγηση του διαφορικού εξελικτικού αλγορίθμου, το σύστημα εκτίθεται σε δύο μορφές δυναμικής φόρτισης, στη σεισμική διέγερση μέσω επιταχυνσιογραφήματος και σε δυναμικό πλήγμα με τριγωνικό παλμό.

Στη συνέχεια, θα παρουσιαστούν οι τιμές των φυσικών παραμέτρων που προήλθαν από τις υπολογιστικές αναλύσεις, μαζί με τα αντίστοιχα διαγράμματα απόκρισης, τονίζοντας την ικανότητα του μοντέλου Bouc -Wen να αναπαριστά με ακρίβεια την ανελαστική συμπεριφορά του φορέα.

5.2 Μονοβάθμιο σύστημα (SDOF)

Στην παρούσα ενότητα αναλύεται η αποδοτικότητα της μεθοδολογίας αναγνώρισης παραμέτρων σε ένα μονοβάθμιο σύστημα (SDOF) βάσει του υπολογιστικού πυρήνα που δημιουργήθηκε στο περιβάλλον SDE. Το μοντέλο αναφοράς αποτελεί ο διγραμμικός ιδανικά ελαστοπλαστικός ταλαντωτής με απότομες μεταβολές δυσκαμψίας. Για την αναγνώριση χρησιμοποιείται η υπολογιστική εκδοχή των τεσσάρων παραμέτρων $p=[\alpha, \beta, n, \gamma]^T$, με την παράμετρο A να παραμένει σταθερή και ίση με τη μονάδα ($A=1.0$), διασφαλίζοντας τη διατήρηση της αρχικής ελαστικής κλίσης k του φορέα. Η ελαχιστοποίηση της αντικειμενικής συνάρτησης σφάλματος $J(p)$ επιτυγχάνεται με τη μέθοδο Διαφορικής Εξέλιξης (Differential Evolution - DEminimizer). Η συνεκτικότητα και η αξιοπιστία του αλγορίθμου εξετάζονται παρακάτω σε δύο διαφορετικά σενάρια δυναμικής φόρτισης.

5.2.1. Σενάριο 1: σεισμική Διέγερση

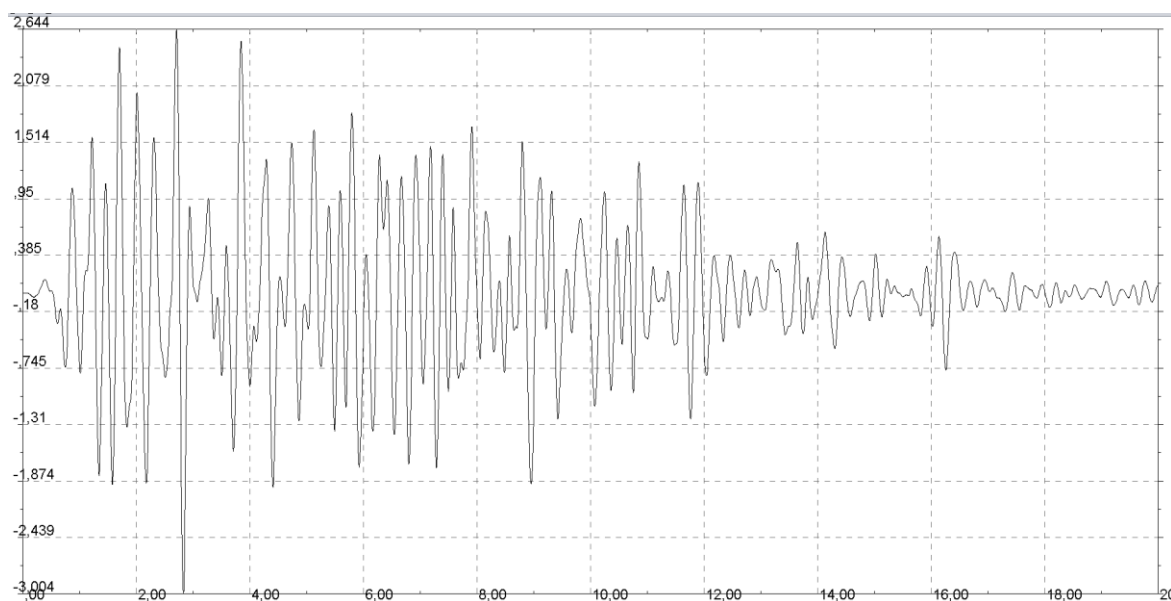
Το πρώτο σενάριο δυναμικής φόρτισης αφορά σεισμική διέγερση που παράγεται μέσω συνθετικού επιταχυνσιογραφήματος διάρκειας 20 sec, με χρήση της κλάσης synthaccel του SDE. Οι παράμετροι του μονοβάθμιου συστήματος ορίζονται ως εξής: μάζα $m=10$ kg, δυσκαμψία $k=1000$ N/m, συντελεστής απόσβεσης $c=1$ N*s/m, δύναμη διαρροής $f_{sy}=10$ N και μετατόπιση διαρροής $u_y=f_{sy}/k=0.01$ m. Η ελαστοπλαστική μετατόπιση αναφοράς u_{ref} υπολογίζεται με τον αλγόριθμο Newmark ($\gamma=1/2$, $\beta=1/4$) με $N=8000$ βήματα ($dt=0,0025$ sec), ενώ η μεθελαστική δυσκαμψία ορίζεται $k_h=0.1*k$.

Η βελτιστοποίηση εκτελείται με τον αλγόριθμο Διαφορικής Εξέλιξης (DE) για τον προσδιορισμό του διανύσματος παραμέτρων $p=[\alpha, n, \beta, \gamma]^T$, με αντικειμενική συνάρτηση το ολοκλήρωμα της απόλυτης διαφοράς μετατόπισης $J(p) = \int |u_{ref}(t) - u_{mun}(t,p)dt|$. Τα όρια αναζήτησης ορίζονται ως $\alpha \in [0,0, 0,30]$, $n \in [1,0, 25,0]$, $\beta \in [0,0, 2,0]$, $\gamma \in [0,0, 2,0]$.

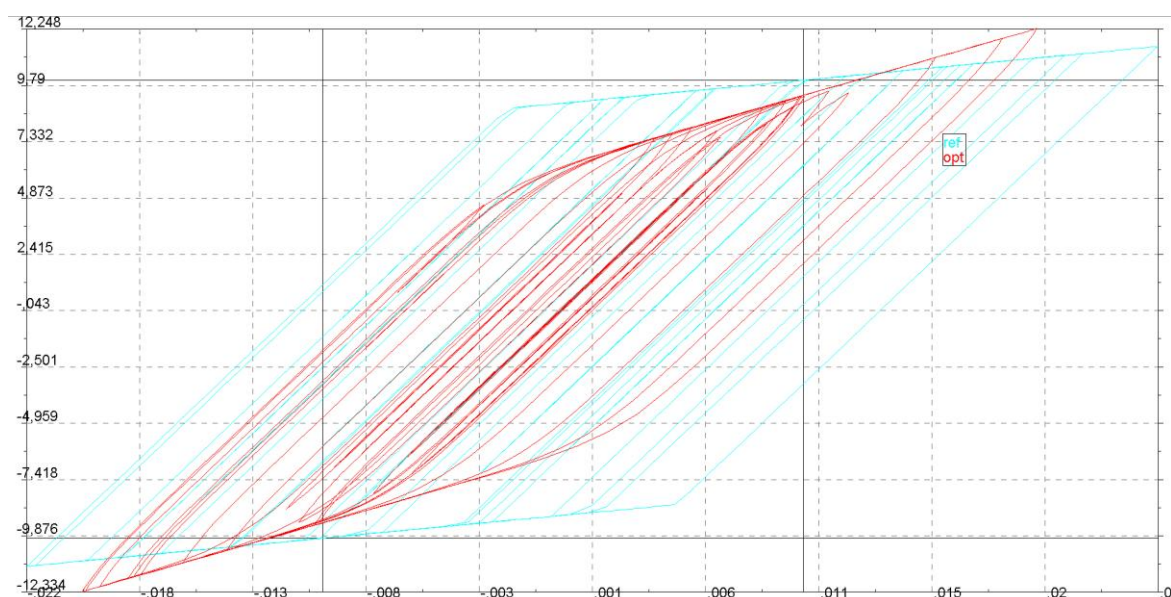
Αρχικά εφαρμόστηκε ο αλγόριθμος με χαμηλές υπολογιστικές απαιτήσεις (level 1, 40 μέλη πληθυσμού, 100 επαναλήψεις) και χρονικό βήμα ολοκλήρωσης ($N=4000$ βήματα, $dt=0,005$ sec). Τα αποτελέσματα της πρώτης εκτέλεσης παρουσιάζονται στον παρακάτω πίνακα και στις αντίστοιχες εικόνες.

Πίνακας 1: Αποτελέσματα αναγνώρισης για Level 1, $N=4000$

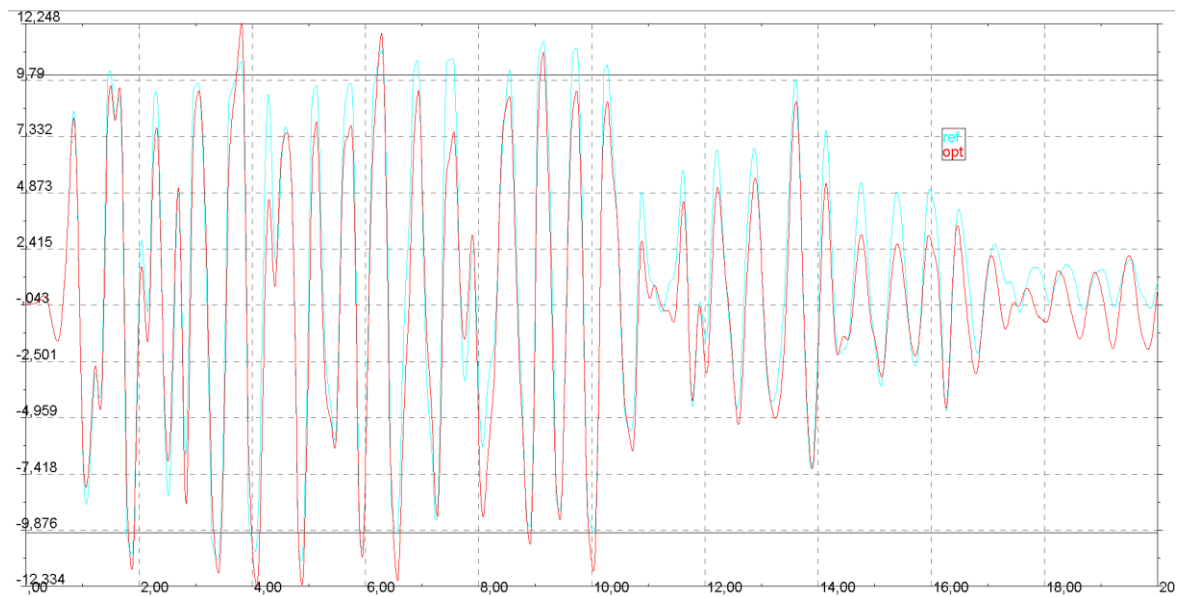
α	n	β	γ	$J(p)$
0,30	6,905	0,000191	1,984	0,0843



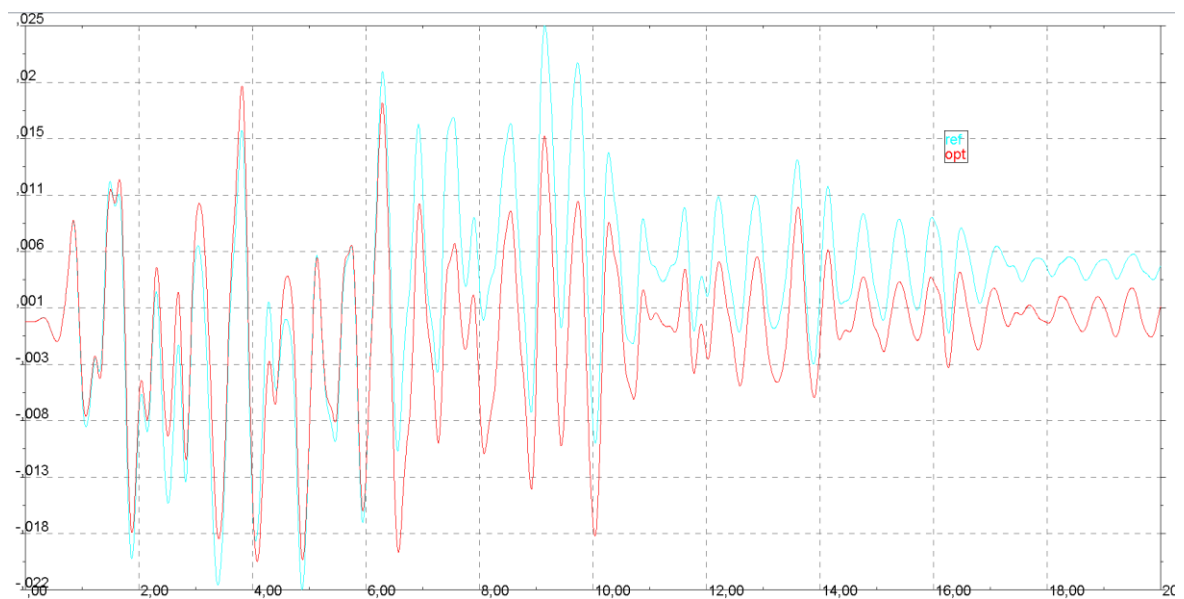
Εικόνα 6: Σεισμικό επιταχυνσιογράφημα διέγερσης.



Εικόνα 7: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt).
Level=1, $N=4000$



Εικόνα 8: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=1, N=4000

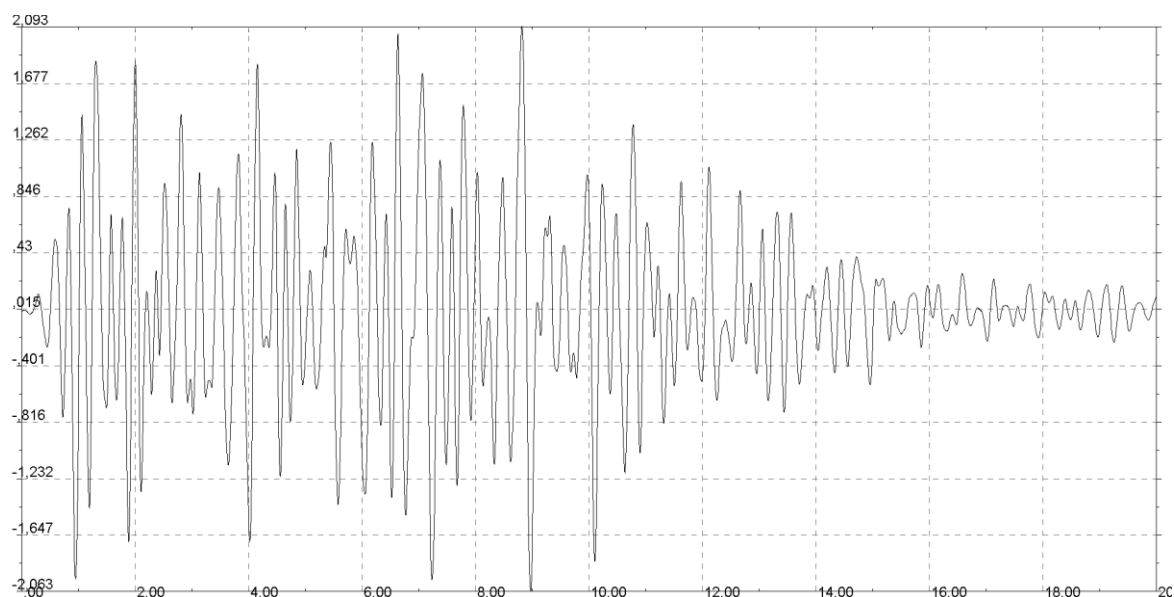


Εικόνα 9: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=1, N=4000

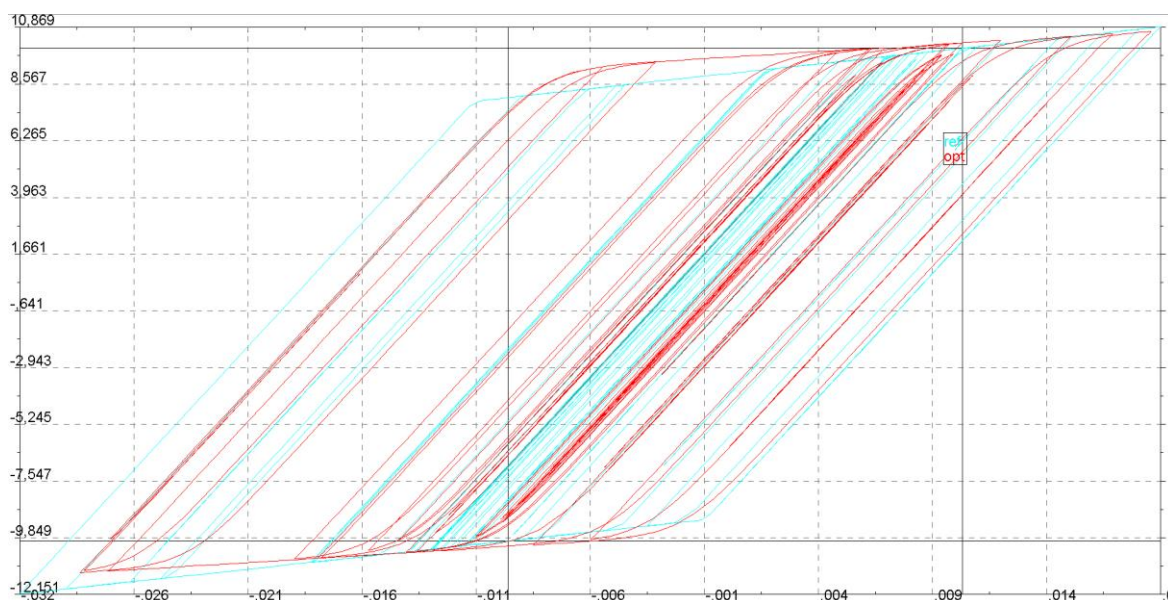
Τα αποτελέσματα της πρώτης εκτέλεσης δεν κρίνονται ικανοποιητικά, καθώς το σφάλμα $J(p) = 0,0843$ είναι υψηλό και η σύγκλιση των καμπύλων ref και opt δεν είναι οπτικά αποδεκτή, ιδιαίτερα στον υστερητικό βρόγχο. Ως εκ τούτου, επαναλήφθηκε η ανάλυση με αυξημένες υπολογιστικές απαιτήσεις (level 3, 120 μέλη πληθυσμού, 300 επαναλήψεις), διατηρώντας το ίδιο βήμα ολοκλήρωσης ($N=4000$). Όπως φαίνεται στον πίνακα 2 και στις εικόνες 10, 11, 12 και 13.

Πίνακας 2: Αποτελέσματα αναγνώρισης για Level 3, $N=4000$

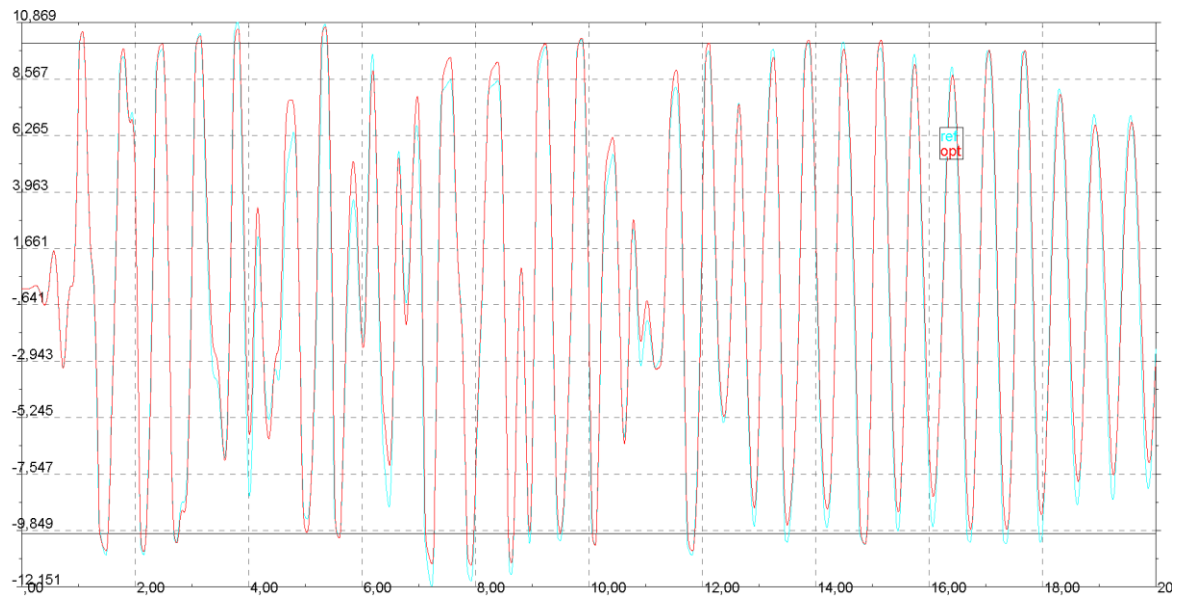
α	n	β	γ	$J(p)$
0,057	9,741	0,435	0,359	0,0297



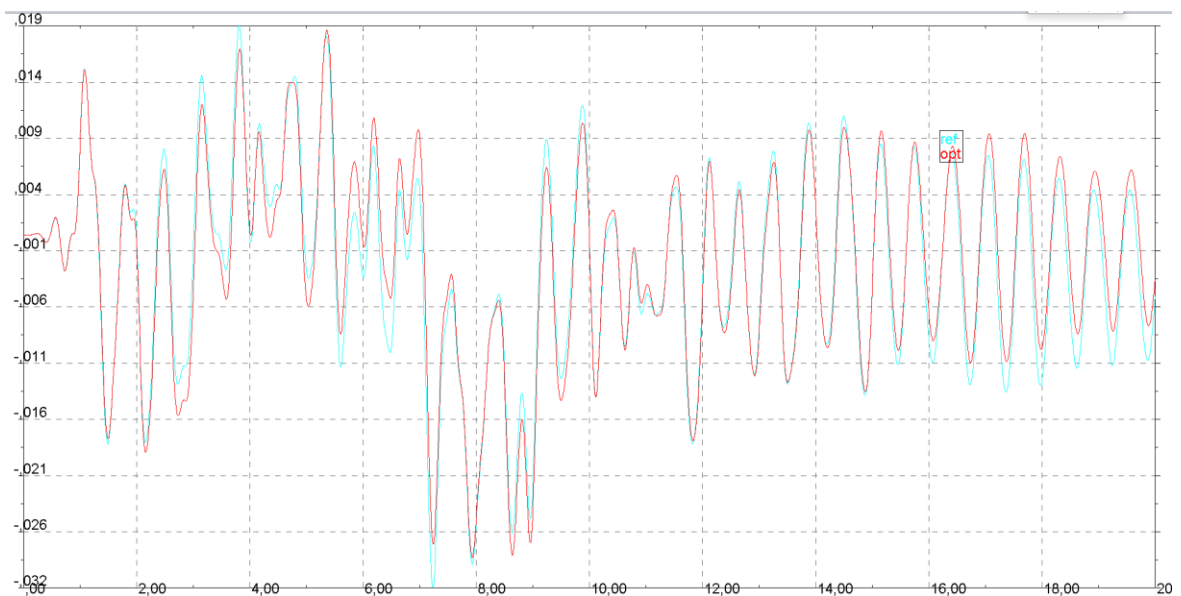
Εικόνα 10: Σεισμικό επιταχυνσιογράφημα διέγερσης.



Εικόνα 11: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt).
Level=3, $N=4000$



Εικόνα 12: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=4000

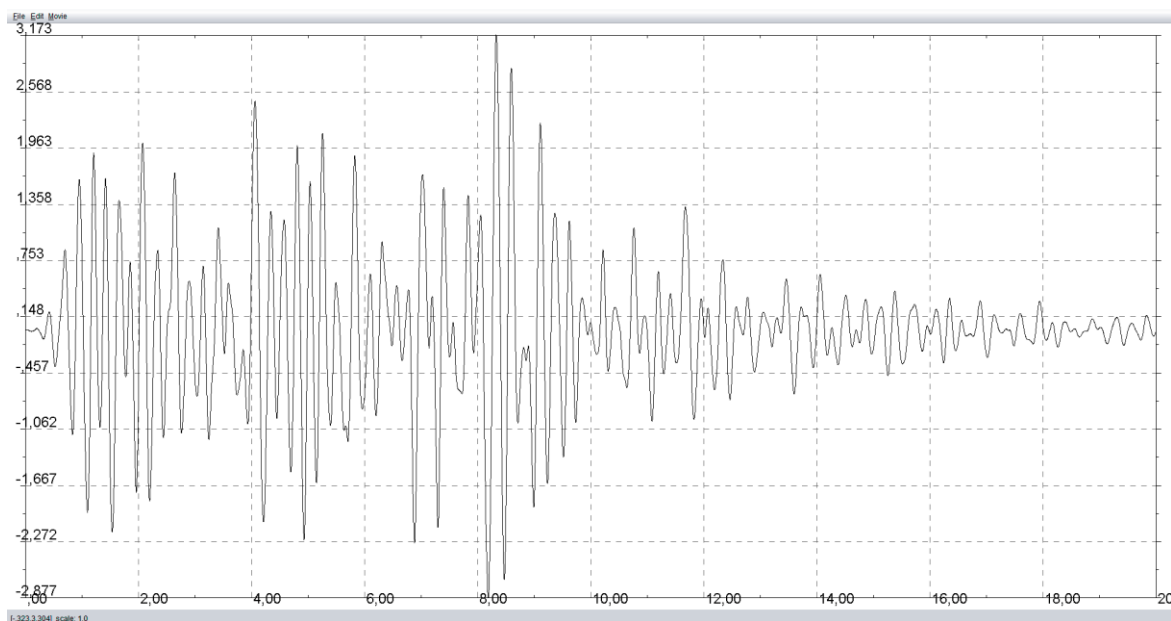


Εικόνα 13: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=4000

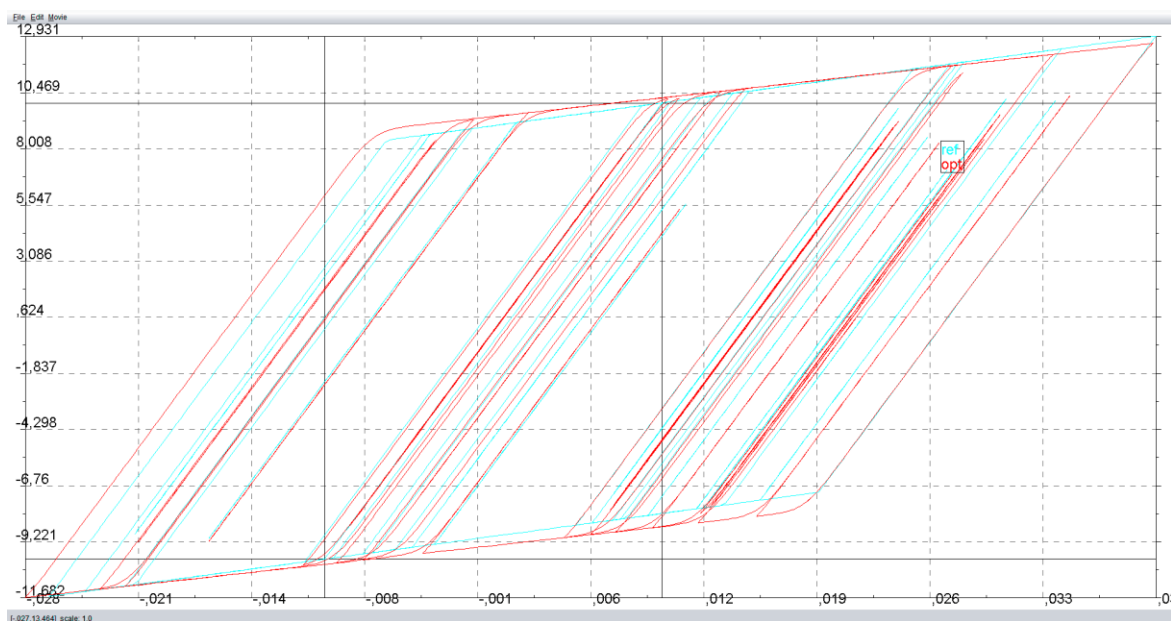
Παρά τη σημαντική βελτίωση (μείωση σφάλματος κατά 65% περίπου), ο υστερητικός βρόγχος εξακολουθούσε να παρουσιάζει αισθητή απόκλιση μεταξύ ref και opt. για να αντιμετωπιστεί αυτό, επαναλήφθηκε η ανάλυση με διπλάσια βήματα ολοκλήρωσης ($N=8000$, $dt=0,0025$ sec), διατηρώντας level 3. Η αύξηση της ακρίβειας της αριθμητικής ολοκλήρωσης, σε συνδυασμό με τον στοχαστικό χαρακτήρα του αλγορίθμου Διαφορικής Εξέλιξης που επιτρέπει διαφορετικές διαδρομές αναζήτησης σε κάθε εκτέλεση, οδήγησε στα βέλτιστα αποτελέσματα. Αυτά φαίνονται στον πίνακα 3 και στις εικόνες 14, 15, 16 και 17.

Πίνακας 3: Βέλτιστα αποτελέσματα αναγνώρισης για Level 3, $N=8000$

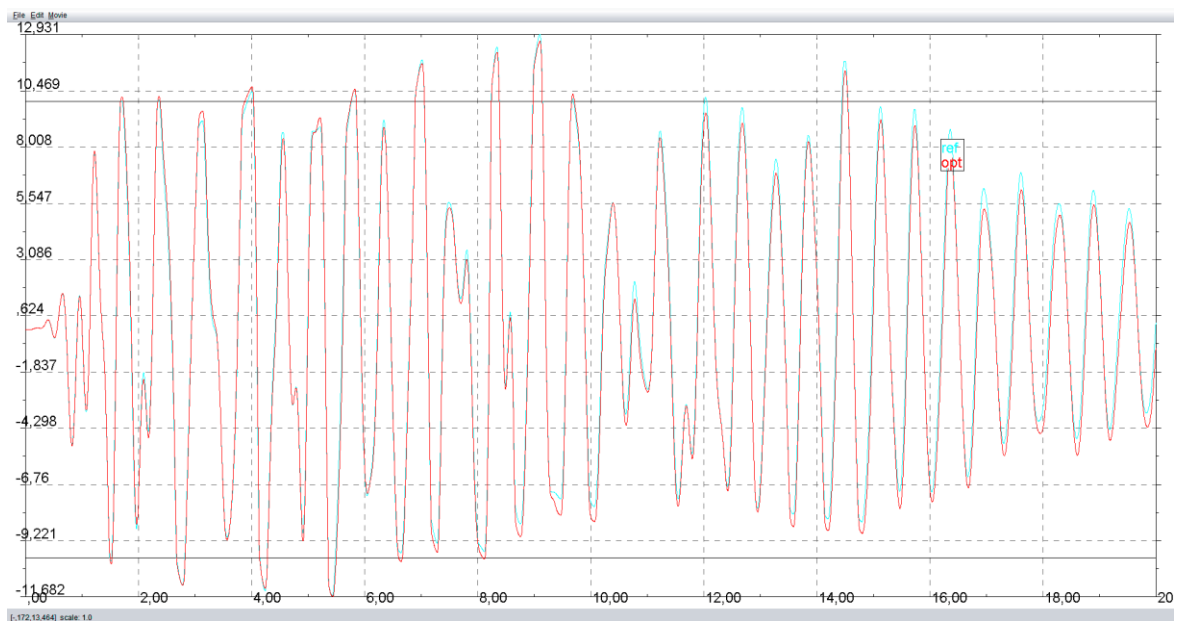
α	n	β	γ	$J(p)$
0,082	18,106	0,407	0,238	0,0092



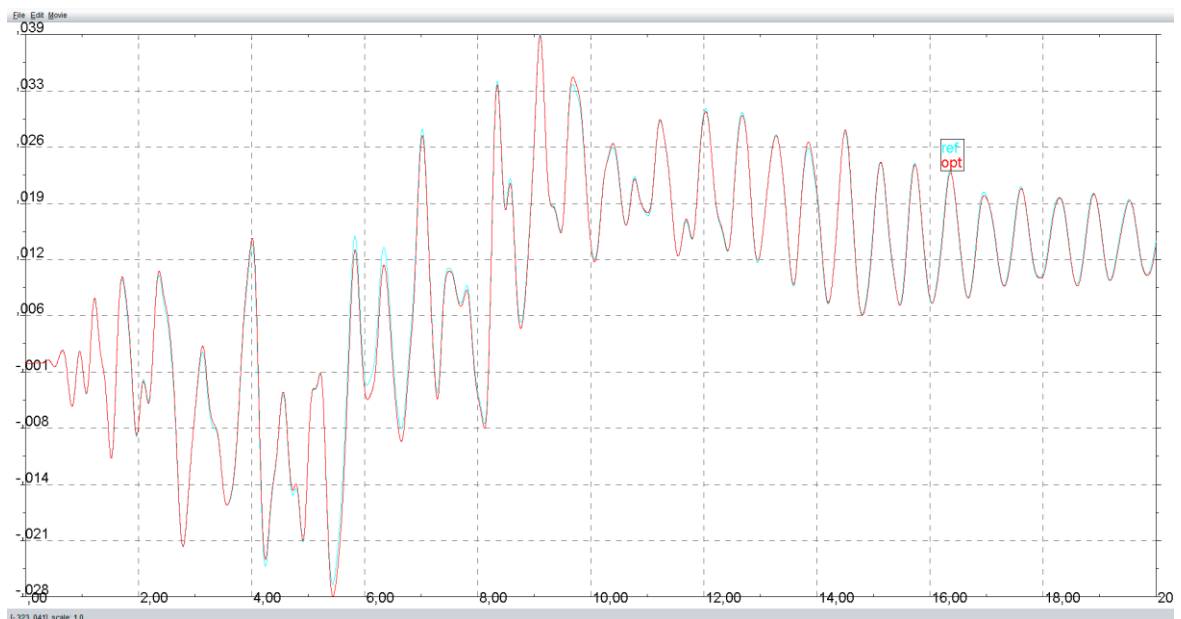
Εικόνα 14: Σεισμικό επιταχυνσιογράφημα διέγερσης.



Εικόνα 15: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt).
Level=3, $N=8000$



Εικόνα 16: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000



Εικόνα 17: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000

Η τελική τιμή του σφάλματος $J(p) = 0,0092$ αντιπροσωπεύει μείωση 89% περίπου σε σχέση με την αρχική εκτέλεση (level 1). Η βέλτιστη τιμή της παραμέτρου $n=18,106$ αντιστοιχεί σε απότομη μετάβαση από ελαστική σε πλαστική συμπεριφορά, συμβατή με τον χαρακτήρα του ελαστοπλαστικού συστήματος αναφοράς. Η μικρή τιμή του $\alpha=0,082$ υποδηλώνει κυρίως πλαστική συμπεριφορά μετά τη διαρροή, ενώ οι τιμές $\beta=0,407$ και $\gamma=0,238$ διασφαλίζουν την ευστάθεια του υστερητικού τελεστή ($\beta+\gamma>0$).

Από την σύγκριση των παραπάνω περιπτώσεων συμπεραίνεται ότι η χρονοϊστορία μετατόπισης (εικόνα 17) δείχνει πολύ καλή σύγκλιση μεταξύ ελαστοπλαστικού και Bouc –

Wen από $t=8$ sec και μετά όπου οι καμπύλες σχεδόν ταυτίζονται. Ο υστερητικός βρόγχος (εικόνα 15) αποτυπώνει την ικανότητα του μοντέλου Bouc -Wen να αναπαράγει την ανελαστική συμπεριφορά του ελαστοπλαστικού φορέα, αν και η ομαλή φύση του μοντέλου δεν επιτρέπει την πλήρη αναπαραγωγή των απότομων γωνιών του διγραμμικού βρόγχου. Τέλος, η χρονοϊστορία της εσωτερικής δύναμης (εικόνα 16) επιβεβαιώνει την αξιοπιστία της ταυτοποίησης.

5.2.2. Σενάριο 2: Δυναμικό πλήγμα

Το δεύτερο σενάριο δυναμικής φόρτισης αφορά δυναμικό πλήγμα τύπου blast, που προσομοιώνεται ως τριγωνικός παλμός με μέγιστη τιμή $P_{so}=10$ N και διάρκεια $t_d=1,0$ sec. Οι φυσικές παράμετροι του συστήματος είναι ίδιοι με το σενάριο 1 ($m=10$ kg, $k=1000$ N/m, $c=1$ N*s/m, $f_{sy}=10$ N, $u_y=0,01$ m). Η συνολική διάρκεια ανάλυσης είναι 20 sec.

Αξίζει να σημειωθεί ότι ο τριγωνικός παλμός έχει μικρότερη διάρκεια ($t_d=1$ sec) σε σχέση με τη συνολική διάρκεια ανάλυσης. Αυτό σημαίνει ότι μετά το $t=1$ sec το σύστημα ταλαντώνεται ελεύθερα με απόσβεση, χωρίς νέες αντιστροφές φόρτισης. Αυτό έχει ως αποτέλεσμα ο υστερητικός βρόγχος να εμφανίζεται ανοιχτός και όχι κλειστός, γεγονός που αντικατοπτρίζει τη φύση της απλής αυτής διέγερσης, εν αντιθέσει με το πλούσιο υστερητικό βρόγχο που παράγεται από τη σεισμική διέγερση του επιταχυνσιογραφήματος.

Ακολουθώντας την ίδια μεθοδολογία με το σενάριο 1, εκτελέστηκαν τρεις δοκιμές του αλγορίθμου διαφορικής εξέλιξης με αυξανόμενες υπολογιστικές απαιτήσεις. Τα αποτελέσματα δίνονται στον παρακάτω πίνακα (πίνακας 4).

Πίνακας 4: Αποτελέσματα αναγνώρισης – Δυναμικό πλήγμα (Blast).

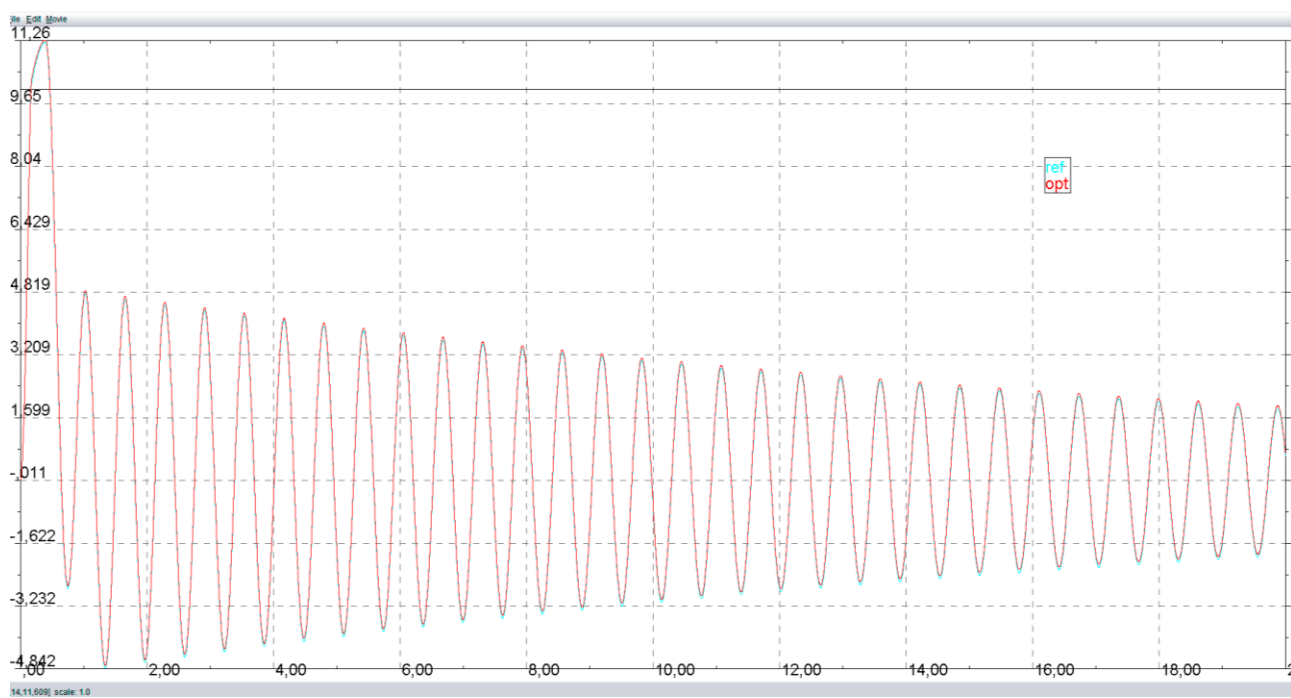
level	N	α	n	β	γ	J(p)	βελτίωση
1	4000	0,0997	13,097	0,4407	0,5134	$7,08 \cdot 10^{-4}$	-
3	4000	0,0836	21,418	0,3752	0,2592	$3,91 \cdot 10^{-4}$	-45%
3	8000	0,0927	24,986	0,3676	0,3174	$9,82 \cdot 10^{-5}$	-75%

Αξίζει να επισημανθεί ότι τα σφάλματα είναι γενικά πολύ μικρότερα σε σύγκριση με το σενάριο 1 (σεισμός). Αυτό οφείλεται στην απλούστερη μορφή της διέγερσης. Ο τριγωνικός παλμός χαρακτηρίζεται από καλά καθορισμένη και προβλέψιμη χρονική εξέλιξη σε σχέση με τη σεισμική διέγερση, γεγονός που διευκολύνει την ταυτοποίηση. Συνολικά, η επιλογή της διέγερσης επηρεάζει καθοριστικά την ποιότητα της ταυτοποίησης του υστερητικού μοντέλου.

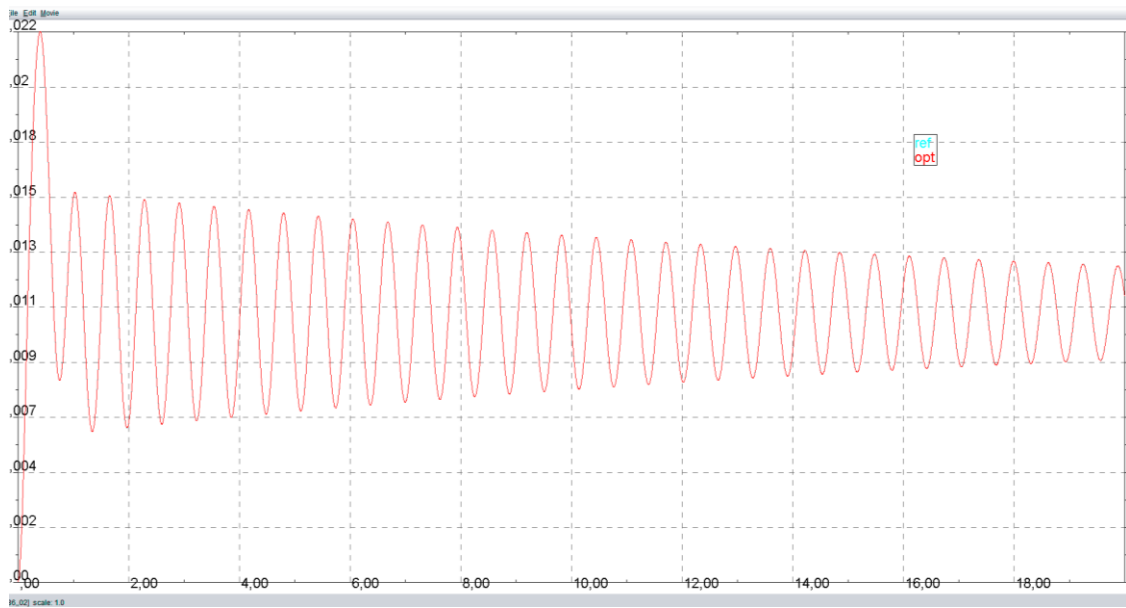
Τα διαγράμματα της βέλτιστης εκτέλεσης (level 3, $N=8000$) παρουσιάζονται παρακάτω. Η χρονοϊστορία μετατόπισης (εικόνα 20) εμφανίζει καλή σύγκλιση μεταξύ ελαστοπλαστικού (ref) και Bouc – Wen (opt). Ο υστερητικός βρόγχος (εικόνα 21) εμφανίζεται ανοιχτός, καθώς η μικρή διάρκεια του παλμού δεν επιτρέπει την ανάπτυξη πλούσιου κλειστού βρόγχου. Η χρονοϊστορία εσωτερικής δύναμης (εικόνα 19) επιβεβαιώνει την καλή σύγκλιση των αποκρίσεων.



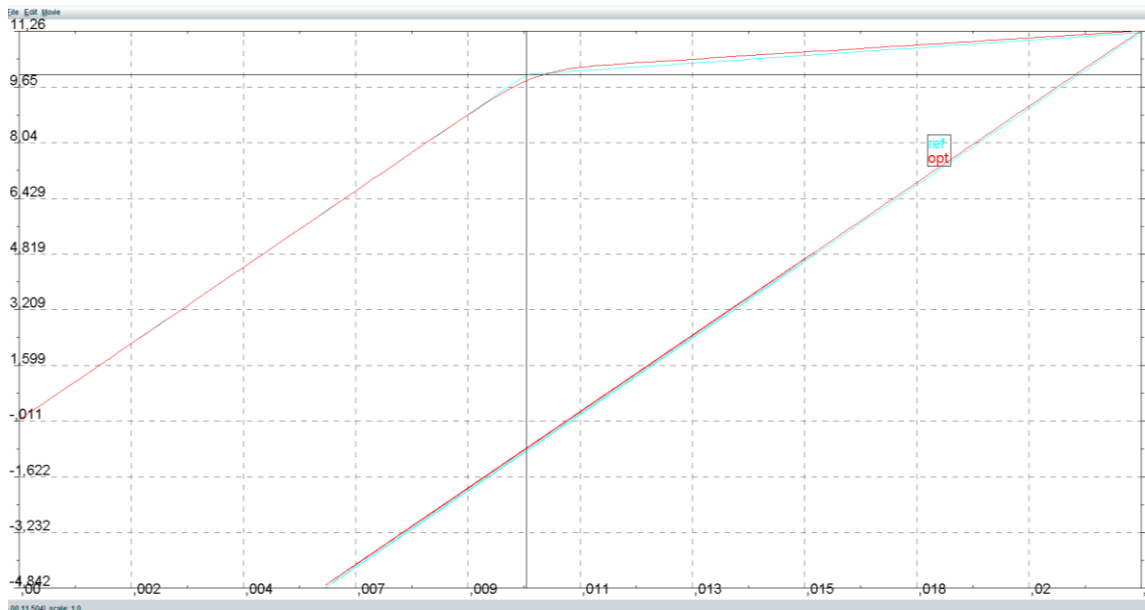
Εικόνα 18: Τριγωνικός παλμός διέγερσης (blast).



Εικόνα 19: Χρονοϊστορία εσωτερικής δύναμης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000



Εικόνα 20: Χρονοϊστορία μετατόπισης – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.



Εικόνα 21: Υστερητικός βρόγχος – σύγκριση ελαστοπλαστικού (ref) και Bouc – Wen (opt). Level=3, N=8000.

5.3 Πολυβάθμιο σύστημα

Στην ενότητα αυτή επεκτείνεται η μεθοδολογία ταυτοποίησης σε πολυβάθμιο σύστημα (MDOF) τριών ορόφων. Το σύστημα αναφοράς αποτελείται από τρεις ορόφους με τις εξής ιδιότητες ανά όροφο: μάζες $m = [10, 8, 6]$ kg, δυσκαμψίες $k = [1000, 800, 600]$ N/m, συντελεστές απόσβεσης $c = [0,50, 0,40, 0,30]$ N*s/m και δυνάμεις διαρροής $f_{sy} = [12, 10, 8]$ N. Η ελαστοπλαστική απόκριση αναφοράς υπολογίζεται με τον αλγόριθμο Newmark ($\gamma=1/2$, $\beta=1/4$), ενώ η μεθελαστική δυσκαμψία ορίζεται $k_h=0,1*k$ για κάθε όροφο. Η βελτιστοποίηση εκτελείται για 12 συνολικά παραμέτρους (4 ανά όροφο: α_i , n_i , β_i , γ_i), με αντικειμενική συνάρτηση το άθροισμα τετραγώνων σφαλμάτων μετατόπισης (SSE) σε

όλους τους ορόφους. Εξετάζονται δύο σενάρια φόρτισης: σεισμική διέγερση και δυναμικό πλήγμα (blast).

5.3.1. Σενάριο 1: Σεισμική Διέγερση

Εφαρμόστηκε συνθετικό σεισμικό επιταχυνσιογράφημα διάρκειας 20 sec ως διέγερση βάσης, με $N = 4000$ βήματα ολοκλήρωσης ($dt = 0,005$ sec). Ο αλγόριθμος Διαφορικής εξέλιξης εκτελέστηκε με χαμηλές υπολογιστικές απαιτήσεις (level=1, 40 μέλη πληθυσμού, 100 επαναλήψεις). Τα αποτελέσματα παρουσιάζονται στον παρακάτω πίνακα (πίνακας 5).

Πίνακας 5: Αποτελέσματα αναγνώρισης – Σεισμική Διέγερση MDOF (3 όροφοι).

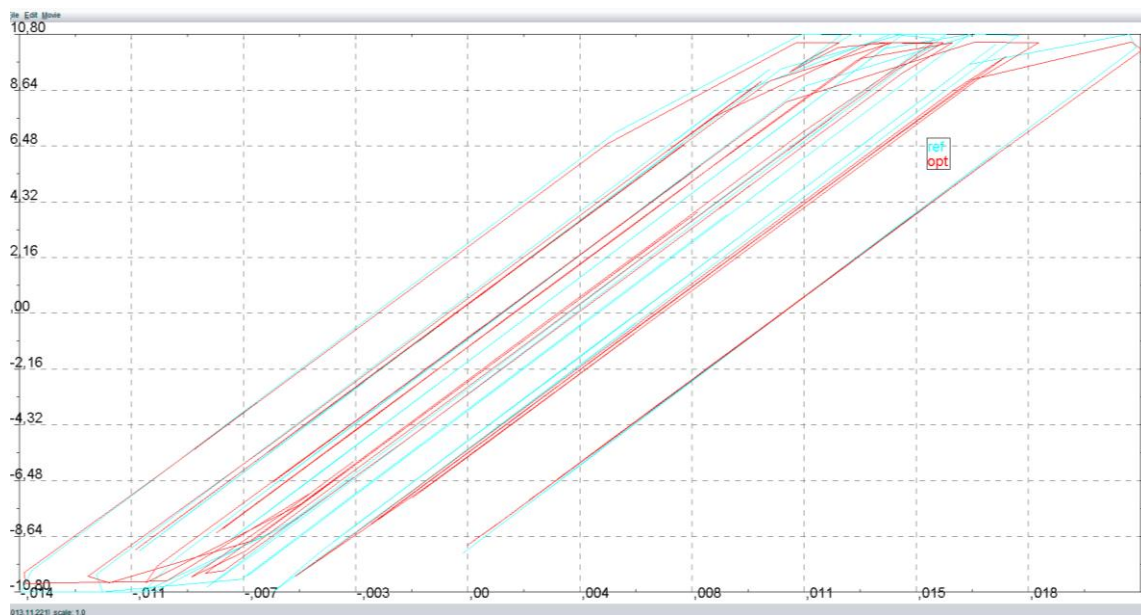
Level	A ₁	n ₁	A ₂	n ₂	A ₃	n ₃	RMSE
1	0,113	23,47	0,083	20,60	0,131	24,04	$5,91 \cdot 10^{-4}$

Τα αποτελέσματα της εκτέλεσης με level=1 και $N = 4000$ ($dt = 0,005$ sec) παρουσιάζουν πολύ ικανοποιητική σύγκλιση με τελική τιμή $RMSE = 5,91 \cdot 10^{-4}$. Οι τιμές n είναι υψηλές ($n \geq 20$), γεγονός συνεπές με τη φύση του ελαστοπλαστικού συστήματος αναφοράς που χαρακτηρίζεται από απότομη μετάβαση ελαστικής/ πλαστικής συμπεριφοράς. Οι τιμές a κυμαίνονται μεταξύ 0,083 και 0,131, υποδηλώνοντας κυρίως ανελαστική συμπεριφορά μετά τη διαρροή. Οι χρονοϊστορίες μετατόπισης εμφανίζουν ικανοποιητική σύγκλιση μεταξύ ελαστοπλαστικού (ref) και Bouc – Wen (opt) και στους τρεις ορόφους, ενώ οι υστερητικοί βρόγχοι αναπαράγουν ικανοποιητικά την ανελαστική συμπεριφορά.

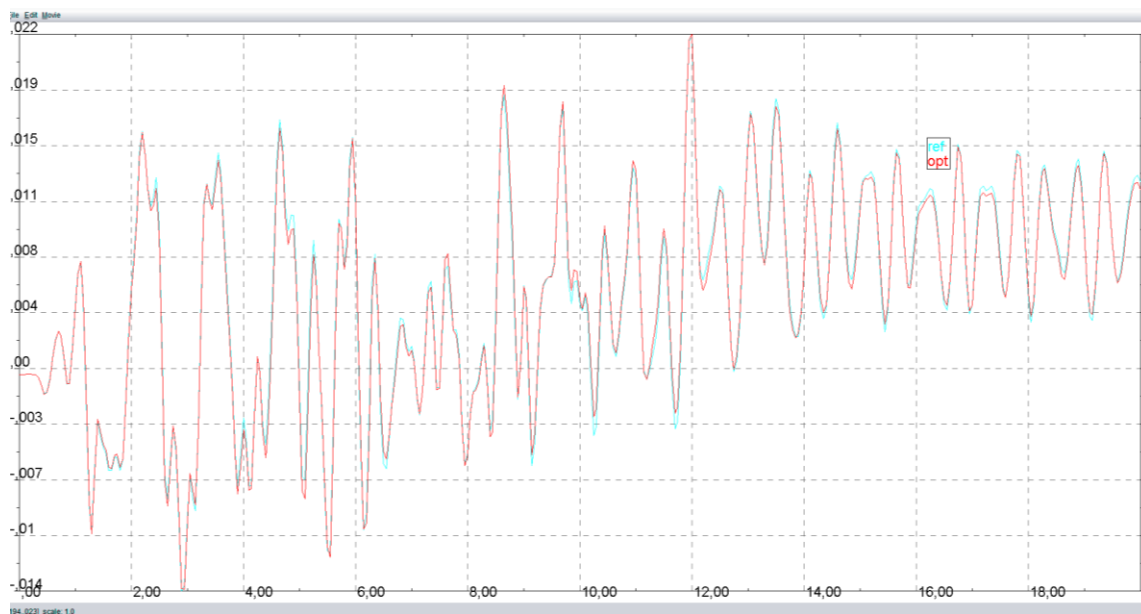
Αξίζει να επισημανθεί ότι οι τιμές n είναι σημαντικά υψηλότερες σε σύγκριση με την βιβλιογραφία ($n = 3-10$), γεγονός που είναι συνεπές με τη φύση του ελαστοπλαστικού συστήματος αναφοράς, το οποίο χαρακτηρίζεται από απότομη μετάβαση ελαστικής/ πλαστικής συμπεριφοράς. Οι τιμές a κυμαίνονται μεταξύ 0,08 και 0,16 υποδηλώνοντας κυρίως ανελαστική συμπεριφορά μετά τη διαρροή. Οι χρονοϊστορίες μετατόπισης εμφανίζουν ικανοποιητική σύγκλιση μεταξύ ελαστοπλαστικού (ref) και Bouc – Wen (opt) και στους τρεις ορόφους.



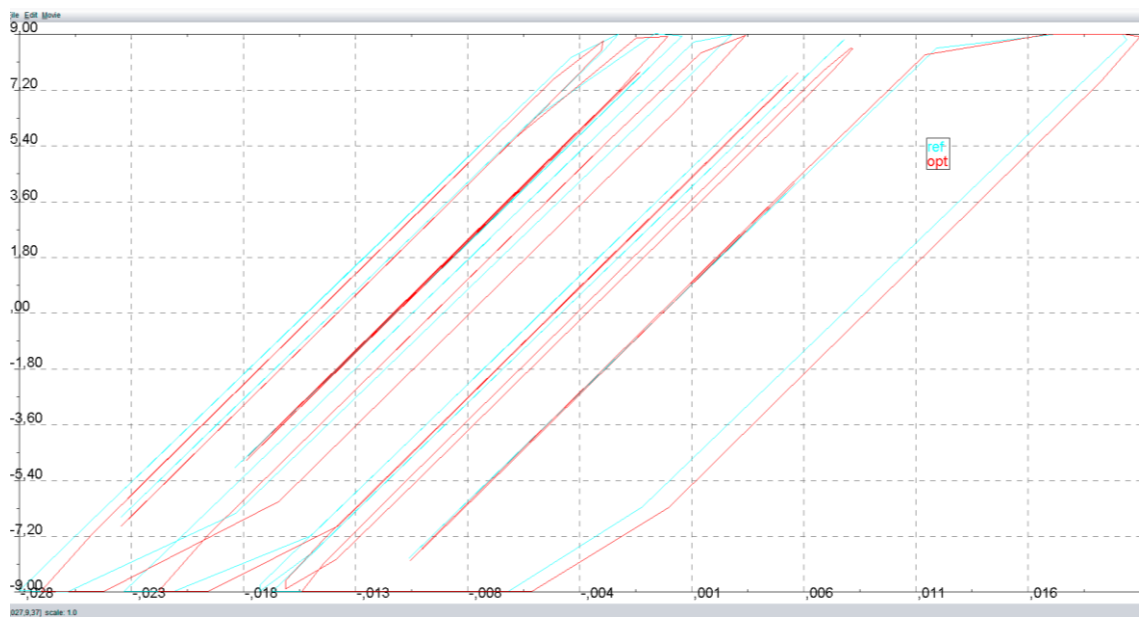
Εικόνα 22: Επιταχυνσιογράφημα σεισμικής διέγερσης MDOF.



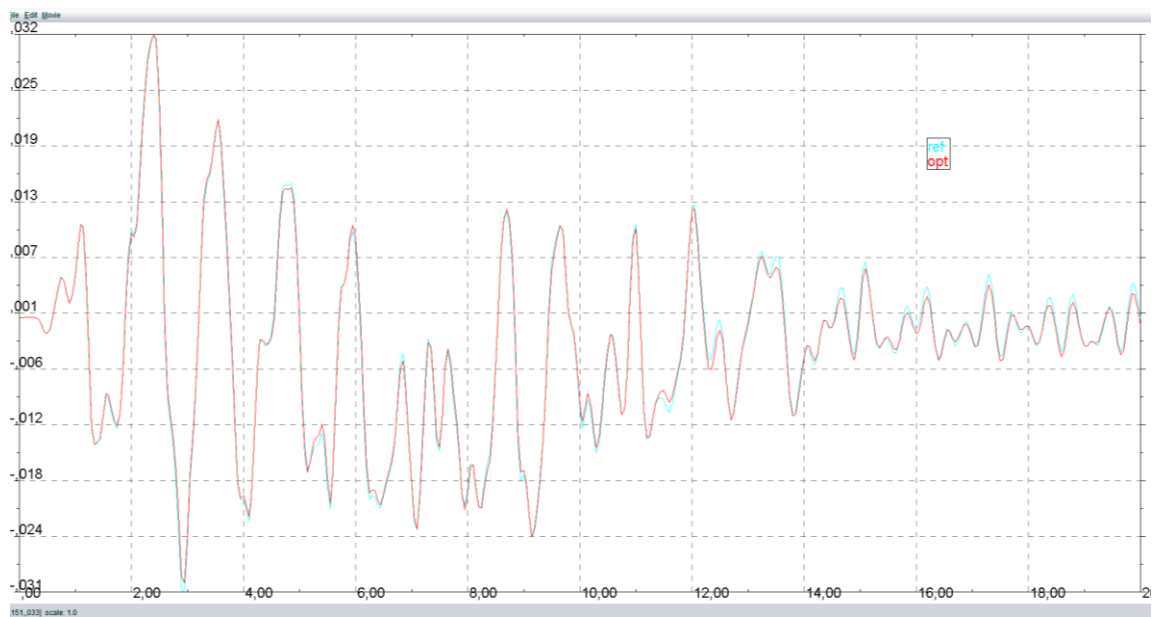
Εικόνα 23: Υστερητικοί βρόγχοι – Σεισμός MDOF ($level = 1$, $N = 4000$, $1^{ος}$ όροφος).



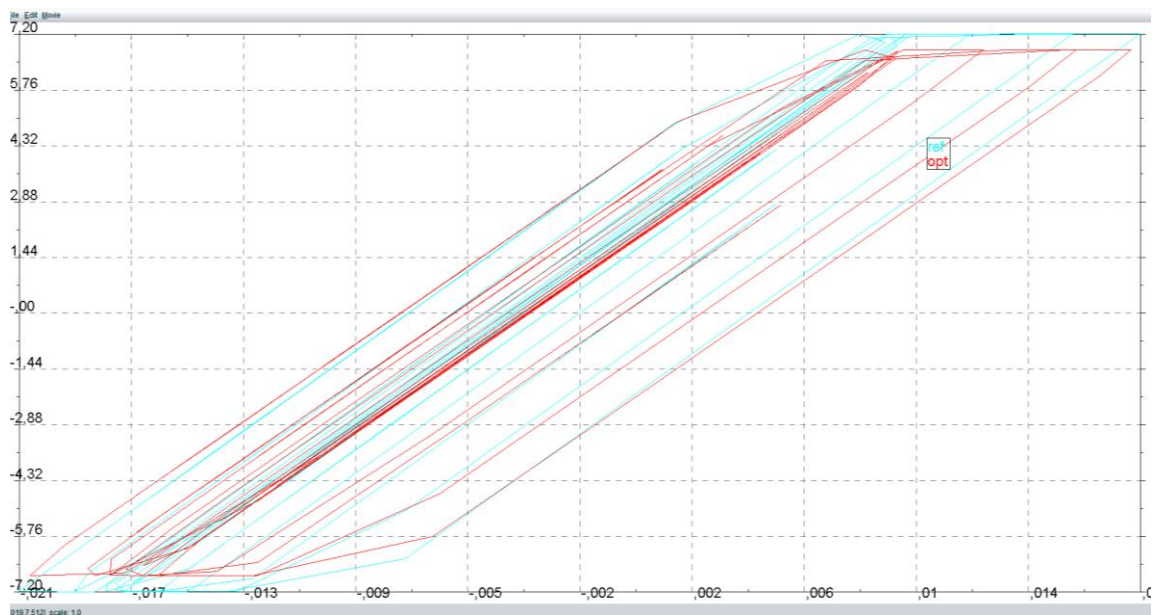
Εικόνα 24: Χρονοϊστορία μετατόπισης – Σεισμός MDOF ($level = 1$, $N = 4000$, $1^{ος}$ όροφος).



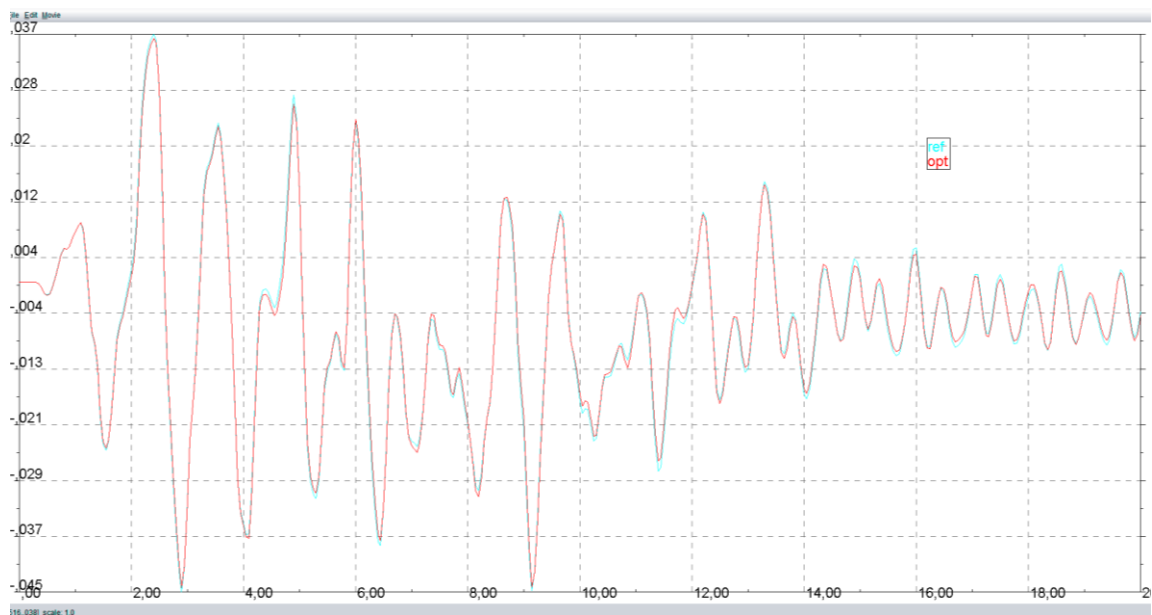
Εικόνα 25: Υστερητικοί βρόγχοι – Σεισμός MDOF ($level = 1$, $N = 4000$, $2^{ος}$ όροφος).



Εικόνα 26: Χρονοϊστορία μετατόπισης – Σεισμός MDOF ($level = 1$, $N = 4000$, $2^{ος}$ όροφος).



Εικόνα 27: Υστερητικοί βρόγχοι – Σεισμός MDOF ($level = 1$, $N = 4000$, $3^{ος}$ όροφος).



Εικόνα 28: Χρονοϊστορία μετατόπισης – Σεισμός MDOF ($level = 1$, $N = 4000$, $3^{ος}$ όροφος).

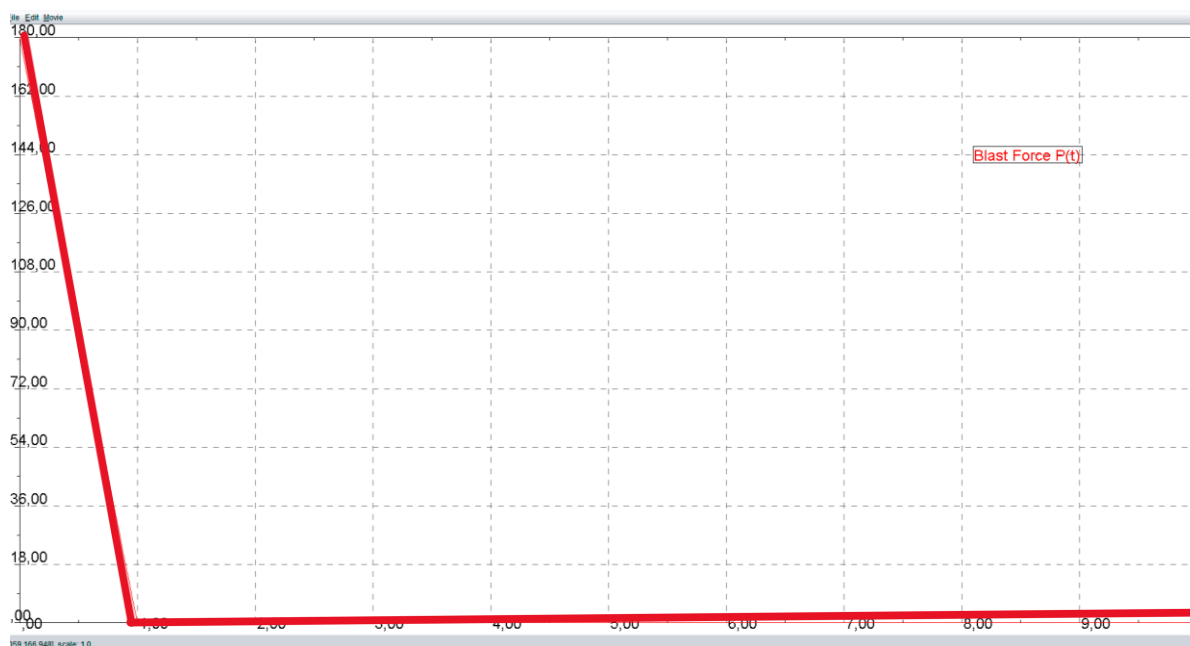
5.3.2. Σενάριο 2: Δυναμικό Πλήγμα (Blast)

Για την παρούσα διέγερση χρησιμοποιήθηκε τριγωνικός παλμός $P_{so} = 180 \text{ N}$ και $t_d = 1,0 \text{ sec}$, εφαρμοζόμενος μόνο στον $1^{ο}$ όροφο. Η συνολική διάρκεια ανάλυσης είναι 10 sec με $N = 10000$ βήματα ($dt = 0.001 \text{ sec}$). Ο αλγόριθμος διαφορικής εξίσωσης εκτελέστηκε με αυξημένες υπολογιστικές απαιτήσεις ($level = 3$, 90 μέλη πληθυσμού, 300 επαναλήψεις). Τα αποτελέσματα παρουσιάζονται στον παρακάτω πίνακα (πίνακα 6).

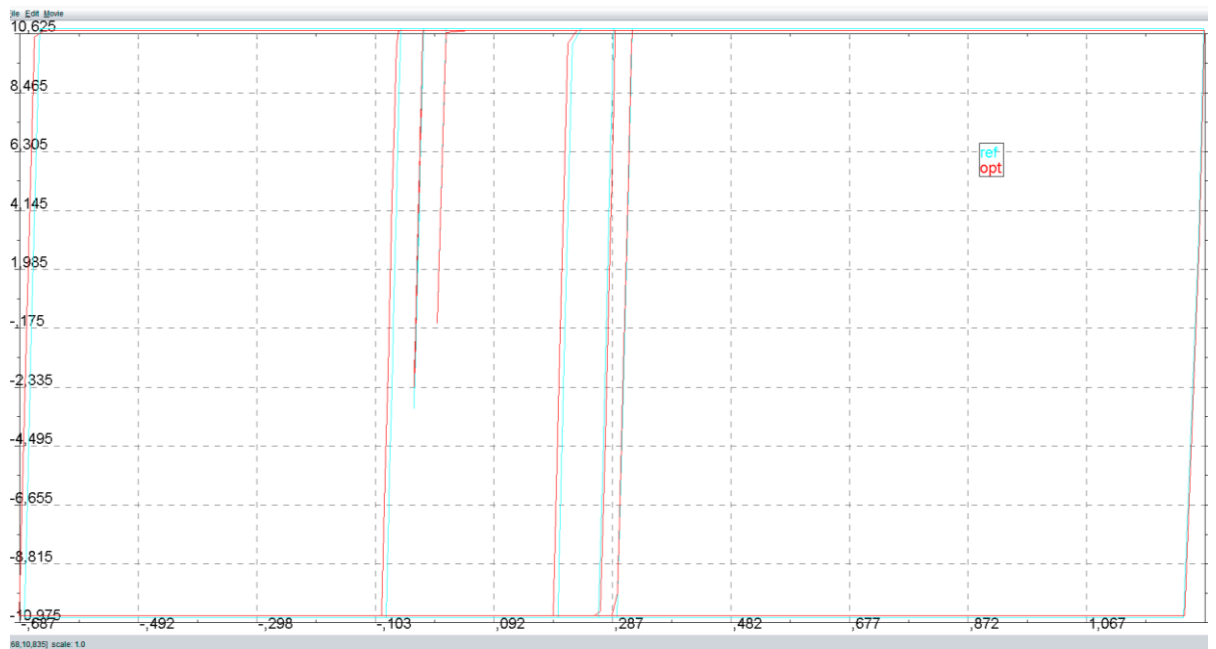
Πίνακας 6: Αποτελέσματα αναγνώρισης – τριγωνικός παλμός (blast) MDOF (3 όροφοι).

Level	A ₁	n ₁	A ₂	n ₂	A ₃	n ₃	RMSE
3	0,10	14,92	0,10	14,02	0,10	14,71	5,26*10 ⁻³

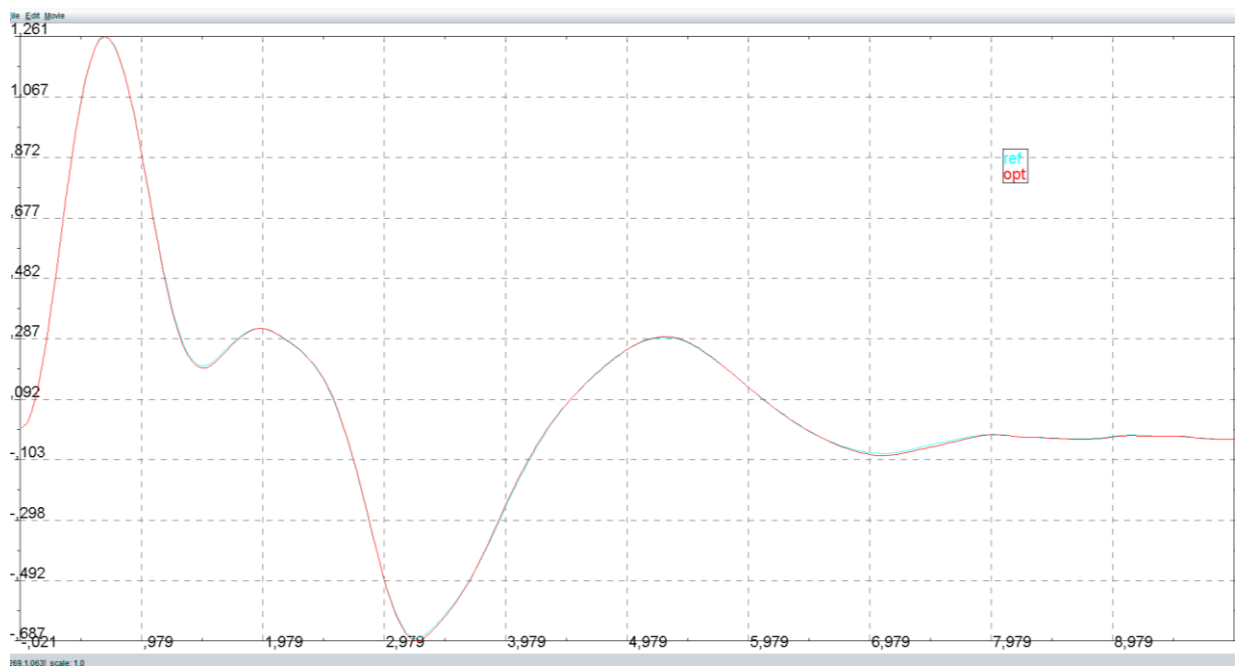
Τα αποτελέσματα της εκτέλεσης με level = 3 και N= 10000 βήματα (dt=0.001 sec) παρουσιάζουν ικανοποιητική σύγκλιση με τελική τιμή RMSE = 5,26*10⁻³. Αξίζει να επισημανθεί ότι οι τιμές α =0,10 είναι ίδιες και στους τρεις ορόφους, γεγονός που δείχνει την ομοιόμορφη ελαστοπλαστική φύση του συστήματος αναφοράς. Οι τιμές n κυμαίνονται μεταξύ 14 και 15, αντιπροσωπεύοντας σχετικά απότομη μετάβαση από ελαστική σε πλαστική συμπεριφορά. Οι υστερητικοί βρόγχοι εμφανίζονται ανοιχτοί, γεγονός αναμενόμενο λόγω της απλής μορφής της διέγερσης (τριγωνικός παλμός), αντίστοιχα με το SDOF σενάριο 2 (ενότητα 5.2). Οι χρονοϊστορίες μετατόπισης εμφανίζουν ικανοποιητική σύγκλιση μεταξύ ελαστοπλαστικού (ref) και Bouc – Wen (opt) και στους τρεις ορόφους.



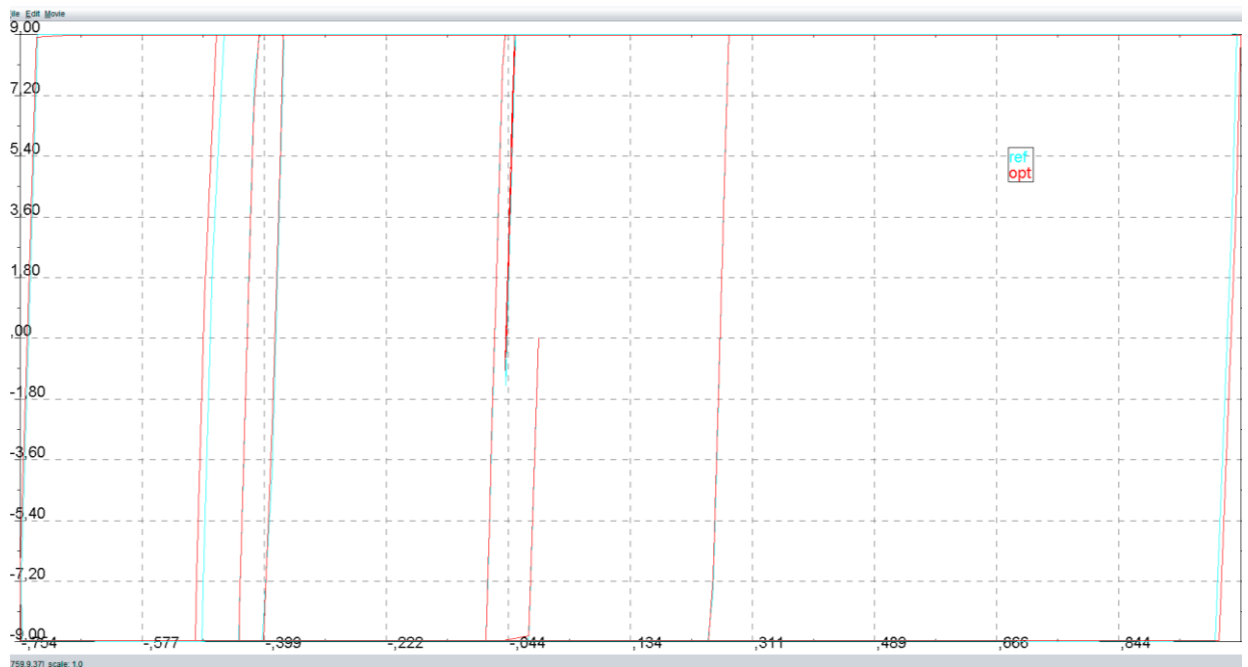
Εικόνα 29: Τριγωνικός παλμός διέγερσης (Blast) MDOF.



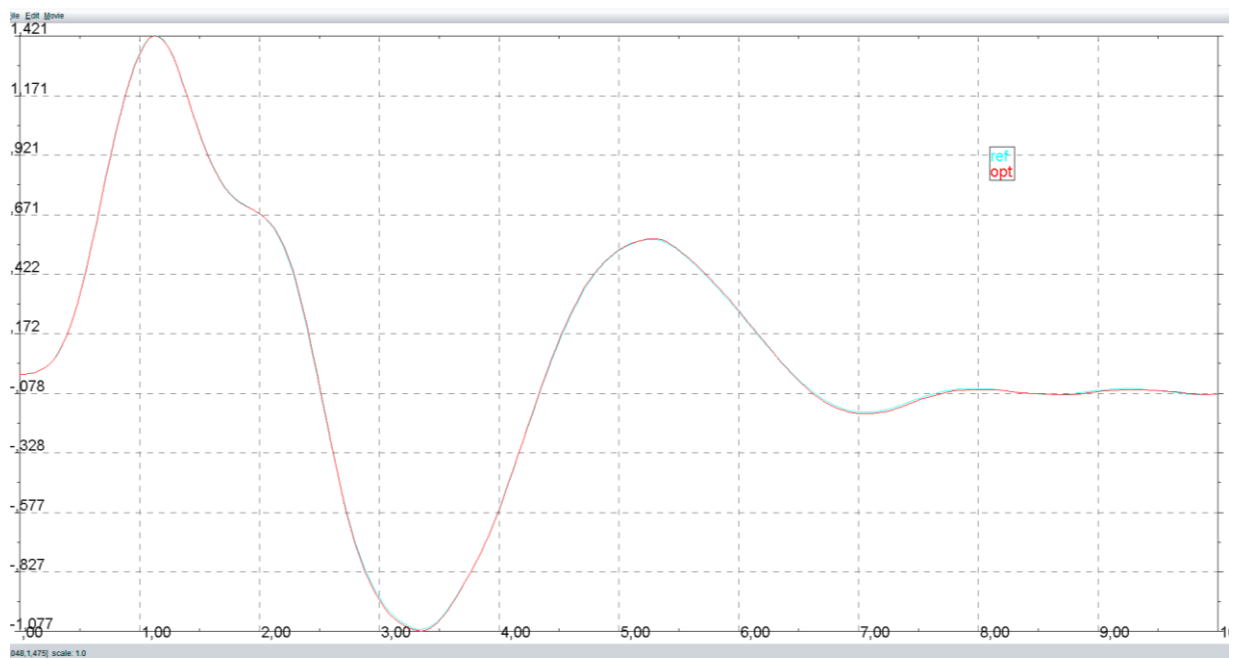
Εικόνα 30: : Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 1^{ος} όροφος).



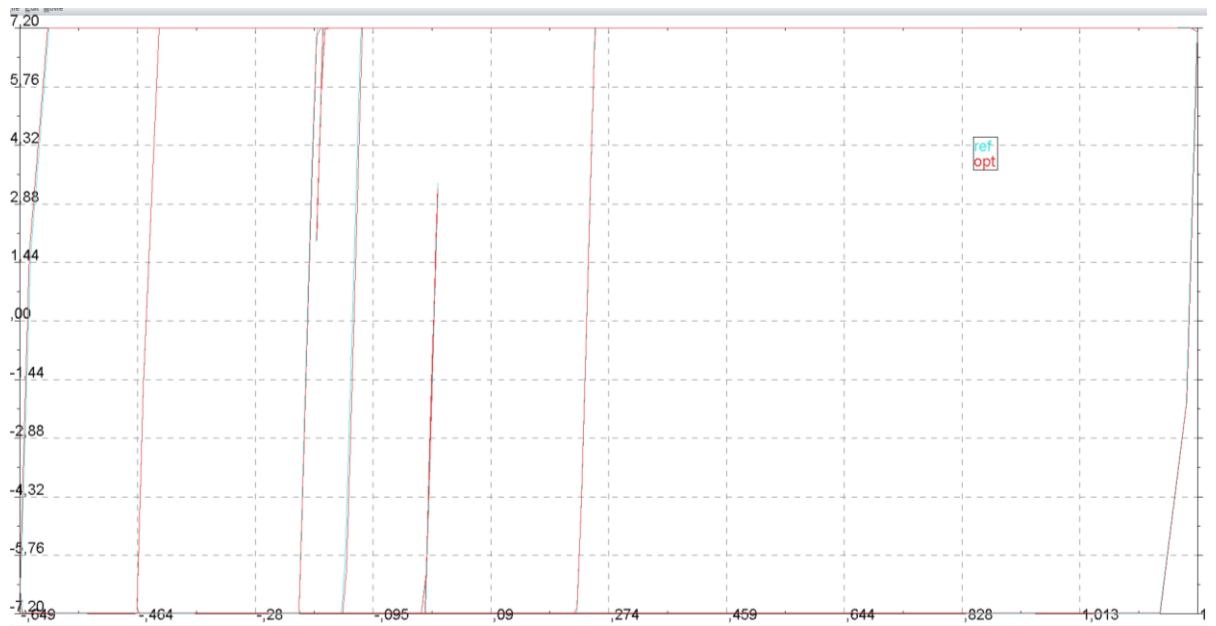
Εικόνα 31: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 1^{ος} όροφος).



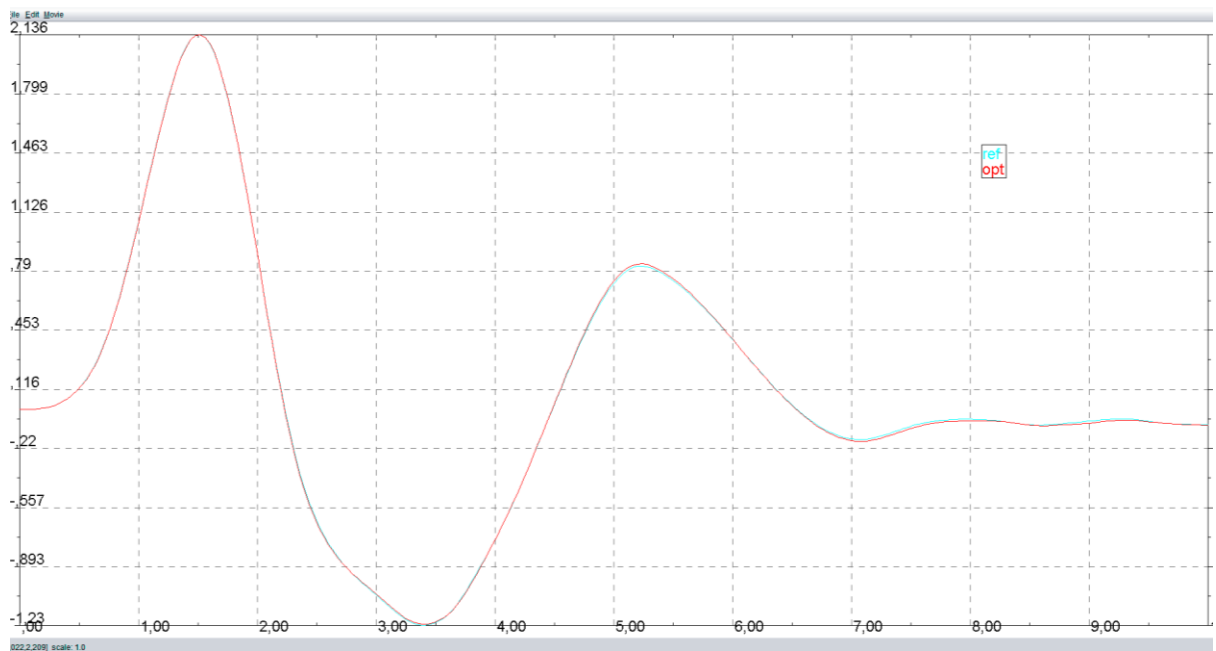
Εικόνα 32: Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 2^{ος} όροφος).



Εικόνα 33: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 2^{ος} όροφος).



Εικόνα 34: Υστερητικοί βρόγχοι – Blast MDOF (level =3, N= 10000, 3^{ος} όροφος).



Εικόνα 35: Χρονοϊστορία μετατόπισης – Blast MDOF (level =3, N= 10000, 3^{ος} όροφος).

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

6.1 Σύνοψη εργασίας

Στην παρούσα διπλωματική εργασία αναπτύχθηκε και αξιολογήθηκε μία ολοκληρωμένη υπολογιστική μεθοδολογία αναγνώρισης παραμέτρων του υστερητικού μοντέλου Bouc -Wen, με εφαρμογή σε μονοβάθμια (SDOF) και πολυβάθμια (MDOF) δυναμικά συστήματα εκτεθειμένα σε σεισμική διέγερση και δυναμικό πλήγμα τύπου blast. Η μεθοδολογία βασίστηκε στη χρήση του αλγορίθμου Διαφορικής Εξέλιξης (Differential Evolution - DE) για την ελαχιστοποίηση της απόκλισης μεταξύ της απόκρισης αναφοράς (ελαστοπλαστικός φορέας – αλγόριθμος Newmark) και της αριθμητικής απόκρισης του μοντέλου Bouc – Wen. Ο αλγόριθμος υλοποιήθηκε και εκτελέστηκε στο περιβάλλον Symplegma Development Environment (SDE).

6.2 Κύρια ευρήματα

Τα κύρια ευρήματα της εν λόγω διπλωματικής εργασίας μπορούν να συνοψιστούν ως εξής:

- Αποτελεσματικότητα του μοντέλου Bouc -Wen: το μοντέλο Bouc -Wen αποδείχθηκε ικανό να αναπαράγει την ανελαστική συμπεριφορά ελαστοπλαστικών φορέων με υψηλή ακρίβεια, τόσο σε μονοβάθμια όσο και σε πολυβάθμια συστήματα. Η ομαλή φύση του μοντέλου, ωστόσο, δεν επιτρέπει την πλήρη αναπαραγωγή των απότομων γωνιών του διγραμμικού ελαστοπλαστικού βρόγχου.
- Αποδοτικότητα του αλγορίθμου Διαφορικής Εξέλιξης: ο αλγόριθμος Διαφορικής Εξέλιξης απέδειξε υψηλή ικανότητα σύγκλισης σε πολύπλοκα προβλήματα βελτιστοποίησης πολλαπλών παραμέτρων. Στο μονοβάθμιο σύστημα, η αύξηση των υπολογιστικών απαιτήσεων (level=3, N=8000) οδήγησε σε μείωση του σφαλματος $J(p)$ κατά 89% περίπου σε σχέση με την αρχική εκτέλεση. Στο πολυβάθμιο σύστημα, εκτελέστηκε επιτυχώς η ταυτόχρονη αναγνώριση 12 παραμέτρων (4 ανά όροφο) με αποτελεσματική σύγκλιση.
- Σύγκριση σεναρίων φόρτισης: η σεισμική διέγερση παράγει πυκνούς, κλειστούς υστερητικούς βρόγχους που παρέχουν πληρέστερη πληροφορία για την αναγνώριση των παραμέτρων. Αντίθετα, το δυναμικό πλήγμα (blast), λόγω της σύντομης διάρκειάς του, παράγει ανοιχτούς υστερητικούς βρόγχους γεγονός που επιβεβαιώνει πως η επιλογή της διέγερσης επηρεάζει καθοριστικά την ποιότητα και πληρότητα της αναγνώρισης.
- Πολυβάθμιο (MDOF) – Σεισμική διέγερση (level = 1, N=4000): τα αποτελέσματα για το τριώροφο σύστημα υπό σεισμική διέγερση έδωσαν $RMSE = 5,91 \cdot 10^{-4}$, επιδεικνύοντας υψηλή ακρίβεια σύγκλισης με μόλις 4000 βήματα ολοκλήρωσης και χαμηλό επίπεδο υπολογιστικών απαιτήσεων (level = 1). Οι βέλτιστες παράμετροι αναγνώρισης για κάθε όροφο ήταν οι εξής:

Πίνακας 7: Βέλτιστες παράμετροι αναγνώρισης - Σεισμική διέγερση (level = 1, N=4000), MDOF.

	α	n	β	γ
Όροφος 1	0,113	23,47	1,095	0,357
Όροφος 2	0,083	20,60	0,151	1,343
Όροφος 3	0,131	24,04	0,385	1,382

- Πολυβάθμιο (MDOF) – Τριγωνικός πλήγμα (blast) (level= 3, N=10000): τα αποτελέσματα για το τριώροφο σύστημα υπό δυναμικό πλήγμα έδωσαν RMSE = $5,26 \cdot 10^{-3}$, με αυξημένες υπολογιστικές απαιτήσεις (level =3) και μεγάλο αριθμό βημάτων ολοκλήρωσης (N=10000). Οι βέλτιστες παράμετροι αναγνώρισης για κάθε όροφο ήταν οι εξής:

Πίνακας 8: Βέλτιστες παράμετροι αναγνώρισης – Τριγωνικό πλήγμα (level =3, N=10000), MDOF.

	α	n	β	γ
Όροφος 1	0,10	14,92	0,603	0,508
Όροφος 2	0,10	14,02	0,335	0,729
Όροφος 3	0,10	14,71	0,663	0,482

- Φυσική ερμηνεία παραμέτρων: σε όλες τις αναλύσεις, οι αναγνωρισμένες τιμές της παραμέτρου n είναι σημαντικά υψηλότερες από τις τυπικές τιμές της βιβλιογραφίας (n=3-10), γεγονός που συνάδει με το ελαστοπλαστικό σύστημα αναφοράς που χαρακτηρίζεται από απότομη μετάβαση ελαστικής/ πλαστικής συμπεριφοράς. Οι τιμές $\alpha < 0,15$ υποδηλώνουν κυρίως ανελαστική συμπεριφορά μετά τη διαρροή, ενώ οι τιμές β και γ ικανοποιούν σε κάθε περίπτωση τη συνθήκη ευστάθειας $\beta + \gamma > 0$.

6.3 Περιορισμοί και προτάσεις για μελλοντική έρευνα

Παρά τα ικανοποιητικά αποτελέσματα, η παρούσα εργασία παρουσιάζει ορισμένες περιορισμούς που αποτελούν ταυτόχρονα κατευθύνσεις για μελλοντική έρευνα:

- Το μοντέλο Bouc -Wen στη βασική του μορφή δεν αναπαράγει με πλήρη ακρίβεια τη συμπεριφορά συστημάτων με σκλήρυνση ή μαλάκωση μετά τη διαρροή. Η χρήση εκτεταμένων εκδοχών του μοντέλου θα μπορούσε να βελτιώσει την αναπαράσταση πραγματικών κατασκευών.
- Η αξιολόγηση βασίστηκε αποκλειστικά σε συνθετικά δεδομένα (αριθμητικός ταλαντωτής αναφοράς). Η επέκταση σε πειραματικά δεδομένα από τράπεζα δοκιμών ή πραγματικές κατασκευές θα επιβεβαίωνε την εφαρμοσιμότητα της μεθόδου σε πραγματικές συνθήκες.
- Η ανάλυση ευαισθησίας ως προς τον αριθμό των παραμέτρων, τα όρια αναζήτησης και τον αριθμό των σωματιδίων της Διαφορικής Εξέλιξης αποτελεί σημαντική κατεύθυνση για περαιτέρω έρευνα, προκειμένου να βελτιστοποιηθεί η αποτελεσματικότητα της μεθόδου για διαφορετικούς τύπους φορτίσεων και φορέων.

6.4 Γενικά συμπεράσματα

Συνολικά, η παρούσα εργασία αποδεικνύει ότι ο συνδυασμός του μοντέλου Bouc -Wen με τον αλγόριθμο Διαφορικής Εξέλιξης αποτελεί ένα αξιόπιστο και αποτελεσματικό

εργαλείο για την αντίστροφη ανάλυση και αναγνώριση παραμέτρων υστερητικών συστημάτων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- [1] George D. Manolis and Christos G. Panagiotopoulos, 2026. Vibrations of Structural Systems. SynthesisLectures on Mechanical Engineering, Springer Nature.
- [2] R. Bouc, “Forced Vibration of Mechanical Systems with hysteresis,” Proceedings of 4th Conference Nonlinear Oscillation, Prague, 5-9 September 1967.
- [3] Χαραλαμπίδης Α. ε., 2009. Ανελαστική Δυναμική Ανάλυση Φορέων με προσομοίωση κατά Bouc -Wen. Αθήνα: Διδακτορική Διατριβή, Σχολή Πολιτικών Μηχανικών, Εθνικό Μετσόβιο Πολυτεχνείο (Ε.Μ.Π.).
- [4] A. E. Charalampakis, C. K. Dimou, 2011. Comparison of Differential Evolution, Particle Swarm Optimization and Genetic Algorithms for the identification of Bouc-Wen hysteretic systems. Proc. 2nd International Conference on Soft Computing Technology in Civil, Structural and Environmental Engineering, Chania, Greece 2011.
- [5] Aristotelis E. Charalampakis, 2010. Parameters of Bouc-Wen hysteretic model revisited. 9th HSTAM International Congress on Mechanics, Limassol, Cyprus, 12 – 14 July, 2010
- [6] Mohammed Ismail, Fayçal Ikhoulane, José Rodellar, 2009. The Hysteresis Bouc-Wen Model, a Survey. Arch Comput Methods Eng.
- [7] Michael Heredia-Pérez, Diego A. Alvarez, Daniel Bedoya-Ruiz. 2025. A State-of-the-Art Review of the Bouc-Wen Class Model of Hysteresis: Origin, Evolution and Current State. Archives of Computational Methods in Engineering. 1 April 2025.
- [8] K. Vlachas, K. Tatsis, K. Agathos, A.R. Brink & E. Chatzi, 2021. Two-story frame with Bouc-Wen hysteretic links as a multidegree of freedom nonlinear response simulator. Sandia National Laboratories.

ΠΑΡΑΡΤΗΜΑ

Α. Παράμετροι και αριθμητική επίλυση Duffing

```
import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel
class duffing implements FirstOrderDifferentialEquations {
    Closure f;
    double m, c, k, mu;
    duffing(double m, double k, double mu, double c, Closure f) {
        this.f = f;
        this.m = m;
        this.k = k;
        this.mu = mu;
        this.c = c;
    }
    int getDimension() {
        return 2;
    }
    void computeDerivatives(double t, double[] y, double[] yDot) {
        yDot[0] = y[1];
        yDot[1] = -c*y[1]/m-k*y[0]/m-mu*y[0]*y[0]*y[0]/m + f(t)/m;
    }
}
dt=0.01
CRK = new ClassicalRungeKuttaIntegrator(dt)
gamma=7.5
omega=1.0
f={gamma*cos(omega*t)}
ode= new duffing(1.0,0.0,1.0,0.05,f);
double [] y= [3.1, 4.1] // initial state
tracker = new ContinuousOutputModel()
CRK.addStepHandler(tracker);
tot=250.0*PI
CRK.integrate(ode,0.0,y,tot,y)
u=[]
v=[]
(0..tot/dt).each {
    tracker.setInterpolatedTime(it*dt)
    res=tracker.getInterpolatedState()
    u.add(res[0])
    v.add(res[1])
}
plot(); plot(u,v) [1]
```

B. Ελαστοπλαστικό SDOF σύστημα του αλγορίθμου Newmark

```

k=1000.0; kh=0.25*k; m=10.0; c=1.0
fsy=4.0; uy=fsy/k
u0=0.0; v0=0.0// initial displacement u0 should be lower than uy
upl=0.0
f={100.0*sin(10.5*sqrt(k/m)*it)}

g=1.0/2.0 // gamma
b=1.0/4.0 // beta

dt=0.001
N=1000

u=new double[N+1]; u[0]=u0
v=new double[N+1]; v[0]=v0
a=new double[N+1]; a[0]=(f(0)-c*v[0]-k*u[0])/m
fs=new double[N+1]; fs[0]=k*u0

plastic=false
kep=k
// iterative computations
(1..N) each {
    // K_effective
    kef=kep+m/(b*dt*dt)+c*g/(b*dt)

    // effective load
    fef=f(it*dt)-f((it-1)*dt)+m*(v[it-1]/(b*dt)+a[it-1]/(2.0*b))+c*(v[it-1]*g/b+a[it-1]*dt*(g/(2.0*b)-1.0))

    // solve for current displacement step
    du=fef/kef
    u[it]=u[it-1]+du

    // current acceleration and velocity
    v[it]=v[it-1]+du*g/(b*dt)-v[it-1]*g/b+a[it-1]*g*(1.0-g/(2.0*b))
    a[it]=a[it-1]+( f(it*dt)-f((it-1)*dt) - c*(v[it]-v[it-1]) - kep*(u[it]-u[it-1]))/m

    // check yielding
    yield=kh*u[it]+(k-kh)*uy*signum(v[it])
    if(!plastic){
        if(v[it]>0.0 && kep*(u[it]-upl)>yield){
            kep=kh
            upl=(1.0-k/kh)*uy
            plastic=true
            alter=true
        }
        if(v[it]<0.0 && kep*(u[it]-upl)<-yield){
            kep=kh

```

```

        upl=(k/kh-1.0)*uy
        plastic=true
        alter=true
    }
} else {
    if(v[it]*v[it-1]<0){
        kep=k
        upl=u[it]-(u[it]-upl)*kh/k
        plastic=false
        alter=true
    }
}
fs[it]=kep*(u[it]-upl)
}
ufPlot = new PlotFrame()
ufPlot.addFunction(new plotfunction(u.fs))
ufPlot.vline(0.0)
ufPlot.hline(0.0)
ufPlot.incline(kh uy fsy Color blue)
ufPlot.incline(kh uy -fsy Color blue)
ufPlot.setMarker(true)
ufPlot.show()

tuPlot = new PlotFrame()
tuPlot.addFunction(new plotfunction(dt.u))
tuPlot.hline(uy.Color red)
tuPlot.hline(-uy.Color red)
tuPlot.setMarker(true)
tuPlot.show()

tvPlot = new PlotFrame()
tvPlot.addFunction(new plotfunction(dt.v))
tvPlot.setMarker(true)
tvPlot.show()
uref=u

```

Γ. Αναγνώριση πέντε παραμέτρων υστερητικού μοντέλου Bouc-Wen με Διαφορική Εξέλιξη (Bouc-Wen σε Bouc-Wen).

```

import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel

class BoucWen implements FirstOrderDifferentialEquations{

    Closure f;
    double m,c,k,alpha,A,beta,gamma,n;

```

```

BoucWen(double m, double k, double c, double alpha, double A, double beta,
double n, double gamma, Closure f) {
    this.f = f;
    this.m = m;
    this.k = k;
    this.c = c;
    this.alpha = alpha;
    this.A = A;
    this.beta = beta;
    this.gamma = gamma;
    this.n = n;
}

int getDimension() {
    return 3;
}

void computeDerivatives(double t, double[] y, double[] yDot) {
    yDot[0] = y[1]
    yDot[1] = -c*y[1]/m-alpha*k*y[0]/m-(1.0-alpha)*k*y[2]/m+f(t)/m
    yDot[2] = A*y[1]-beta*y[2]*abs(y[1])*abs(y[2])**n-
gamma*y[1]*abs(y[2])**n
}

}

dt=0.01

m = 1.0
k = 10.0
c = 1.0
alpha = 0.2
A = 0.1
beta = 0.1
n = 3.5
gamma = 0.1
F0 = 30.0
omega = 1.0

CRK = new ClassicalRungeKuttaIntegrator(dt)
f={F0*cos(omega*it)}

tot=20.0

solve=f
    CRK = new ClassicalRungeKuttaIntegrator(dt)

    ode = new BoucWen(m,k,c,0,0,0,0,0,0,f);
    double[] y = [0.0, 0.0, 0.0]; // initial state

```

```

tracker = new ContinuousOutputModel()

CRK.addStepHandler(tracker);

CRK.integrate(ode, 0.0, y, tot, y);
u=[]
v=[]
z=[]
felst=[]
fhyst=[]
(0..tot/dt).each{
    tracker.setInterpolatedTime(it*dt)
    res=tracker.getInterpolatedState()
    u.add(res[0])
    v.add(res[1])
    z.add(res[2])
    felst.add(ode.k*ode.alpha*res[0])
    fhyst.add(ode.k*(1.0-ode.alpha)*res[2])
}
return [u,v,z,fhyst]
}

xref=[alpha A beta n gamma]
println "reference params= "+xref

result=solve(xref)
uref=result[0]
vref=result[1]
zref=result[2]
fhyst_ref=result[3]
plot();
thePlot.setAutoColor(true)
//plot(u,felst);
plot(uref,fhyst_ref)

/// MINIMIZATION
Ra={params->
    result=solve(params)
    u=result[0]
    v=result[1]

    cf=0.0
    wu = 1.0
    wv = 1.0
    u.size().times{
        cf+=wu*(u[it]-uref[it])**2+wv*(v[it]-vref[it])**2
    }
}

```

```

    }
    return cf
}

dim=5; level=3
minv=[0.0,0.0,0.0,1.0,0.1] as double[]
maxv=[1.0,1.0,1.0,10.0,1.0] as double[]

demin=new DFminimizer(Ra as DoubleFunction, dim, minv, maxv)
demin.setMaxData(10*level) // default -> 30*Dimensions (30*4)
demin.setMaxIterations(100*level) // default -> 300
sol= demin.Minimize()
println "solution for params= "+sol+", with error= "+Ra(sol)

/*
minres=solve(sol)
usol=minres[0]
vsol=minres[1]
zsol=minres[2]
fhyst_sol=minres[3]
plot(usol,fhyst_sol)
*/

```

Δ. Αναγνώριση τεσσάρων παραμέτρων υστερητικού μοντέλου Bouc-Wen με Διαφορική Εξέλιξη (Bouc-Wen σε Bouc-Wen).

```

import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel

class BoucWen implements FirstOrderDifferentialEquations {
    Closure p
    double m, c, k, a, n, b, gamma, uy, Fy

    BoucWen(double m, double c, double k, double a, double n, double b, double
gamma, double uy, double Fy, Closure p) {
        this.m = m
        this.c = c
        this.k = k
        this.a = a
        this.n = n
        this.b = b
        this.gamma = gamma
        this.uy = uy
        this.Fy = Fy
        this.p = p
    }

    int getDimension() {

```

```

    return 3 // u, u_dot, z
}

void computeDerivatives(double t, double[] y, double[] yDot) {
    double u = y[0]
    double v = y[1]
    double z = y[2]

    double dz = (1.0 / uy) * (1.0 - Math.pow(Math.abs(z), n) * (gamma *
Math.signum(v * z) + b)) * v

    double F = a * k * u + (1.0 - a) * Fy * z

    yDot[0] = v
    yDot[1] = (p(t) - c * v - F) / m
    yDot[2] = dz
}
}

dt = 0.001
m = 10.0
k = 1000.0
a = 0.1
uy = 0.008
Fy = 8.0
n = 2.0
b = 0.5
gamma = 0.5
c = 0.0
amp = 100.0
freq = 10.5*sqrt(k/m)

f_input = { double t -> amp * Math.sin( freq * t) }

solve={
    CRK = new ClassicalRungeKuttaIntegrator(dt)
    ode = new BoucWen(m, c, k, it[0], it[1], it[2], it[3], uy, Fy, f_input)
    double[] y_initial = [0.0, 0.0, 0.0] // u=0, v=0, z=0
    tracker = new ContinuousOutputModel()
    CRK.addStepHandler(tracker)

    double totalTime = 1.0
    CRK.integrate(ode, 0.0, y_initial, totalTime, y_initial)

    u_vals = []
    force_vals = []

    (0..(totalTime/dt)).each {
        double currentTime = it * dt

```

```

    tracker.setInterpolatedTime(currentTime)
    double[] state = tracker.getInterpolatedState()

    double u = state[0]
    double v = state[1]
    double z = state[2]

    double totalF = a * k * u + (1.0 - a) * Fy * z

    u_vals.add(u)
    force_vals.add(totalF)
  }
  return [u_vals, force_vals]
}

//result=solve([0.2999453706935031, 1.998999430516851, 1.498120505692241,
0.9989645829582822])
//uref=result[0]
//plot()
//plot(result[0],result[1])

/// MINIMIZATION
Ra={params->
  result=solve(params)
  u=result[0]

  cf=0.0
  u.size() times{
    cf+=(u[it]-uref[it])**2
  }
  return cf
}

dim=4; level=1
minv=[0.0,1.0,0.0,0.0] as double[]
maxv=[1.0,10.0,2.0,1.0] as double[]

demin=new DMinimizer(Ra as DF, dim, minv, maxv)
demin.setMaxData(10*level) // default -> 30*Dimensions (30*4)
demin.setMaxIterations(100*level) // default -> 300
sol= demin.Minimize()
println "solution for params= "+sol+", with error= "+Ra(sol)
if(false){
  result=solve([0.26408582537861947,8.367232645838829,
0.0307796200511148, 0.36162593798477705])
  //ufPlot=new PlotFrame()
  ufPlot.addFunction ( new plotfunction (result[0], result[1]))
  ufPlot.setMarker (true)
  ufPlot.setAutoColor (true)
}

```

```

ufPlot.show()

//tuPlot=new PlotFrame()
tuPlot.addFunction( new plotfunction( dt_result[0]))
tuPlot.setAutoColor( true )
tuPlot.setMarker( true )
tuPlot.show()
}

```

Ε. Αναγνώριση τεσσάρων παραμέτρων υστερητικού μοντέλου Bouc-Wen από ελαστοπλαστική απόκριση αναφοράς υπό σεισμική διέγερση με Διαφορική Εξέλιξη.

```

import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel
import courses.structuraldynamics.synthaccel

// --- ΒΗΜΑ 1: ΟΡΙΣΜΟΣ ΔΙΕΓΕΡΣΗΣ (ΣΕΙΣΜΟΣ) ---
duration = 20.0; steps = 8000; dt = duration/steps
if(true){
    t1=1.0; t2=9.0
    So=70.0
    om2=3.0*2.0*pi // K-T
    xsi2=0.6000
    om1=0.75*2.0*pi // H-P
    xsi1=1.0
    om3=4.2*2.0*pi // L-P
    xsi3=0.2

    synthacc=new synthaccel(So, om1, xsi1, om2, xsi2, om3, xsi3)
    synthacc.duration=duration
    synthacc.t1=t1
    synthacc.t2=t2
    synthacc.Calc(steps)
    acc = (synthacc.getSynthAcc()) collect { it / 100.0 }
}
accPlot=new PlotFrame()
accPlot.addFunction(new plotfunction(dt acc))
accPlot.show()

m = 10.0; k = 1000.0; c = 1.0; fsy = 10.0; uy = fsy/k
f_input = { double t ->
    int idx = (int)(t / dt)
    return (idx < acc.size()) ? -m * acc[idx] : 0.0
}

// --- ΒΗΜΑ 2: ΠΑΡΑΓΩΓΗ "ΠΡΑΓΜΑΤΙΚΗΣ" ΑΠΟΚΡΙΣΗΣ (ΕΛΑΣΤΟΠΛΑΣΤΙΚΟ) ---
g=1.0/2.0 // gamma

```

```

b=1.0/4.0 // beta
upl=0.0
f=f_input
kh=0.1*k
N = steps
u_ref=new double[N+1]; u_ref[0]=0.0
v=new double[N+1]; v[0]=0.0
a=new double[N+1]; a[0]=(f(0)-c*v[0]-k*u_ref[0])/m
ft_ref=new double[N+1]; ft_ref[0]=0.0

plastic=false
kep=k
// iterative computations
(1..N).each{
    // K_effective
    kef=kep+m/(b*dt*dt)+c*g/(b*dt)

    // effective load
    fef=f(it*dt)-f((it-1)*dt)+m*(v[it-1]/(b*dt)+a[it-1]/(2.0*b))+c*(v[it-1]*g/b+a[it-1]*dt*(g/(2.0*b)-1.0))

    // solve for current displacement step
    du=fef/kef
    u_ref[it]=u_ref[it-1]+du

    // current acceleration and velocity
    v[it]=v[it-1]+du*g/(b*dt)-v[it-1]*g/b+a[it-1]*g*(1.0-g/(2.0*b))
    a[it]=a[it-1]+( f(it*dt)-f((it-1)*dt) - c*(v[it]-v[it-1]) - kep*(u_ref[it]-u_ref[it-1])
)/m

    // check yielding
    yield=kh*u_ref[it]+(k-kh)*uy*signum(v[it])
    if(!plastic){
        if(v[it]>0.0 && kep*(u_ref[it]-upl)>yield){
            kep=kh
            upl=(1.0-k/kh)*uy
            plastic=true
            alter=true
        }
        if(v[it]<0.0 && kep*(u_ref[it]-upl)<=yield){
            kep=kh
            upl=(k/kh-1.0)*uy
            plastic=true
            alter=true
        }
    }
    else{
        if(v[it]*v[it-1]<0){
            kep=k
            upl=u_ref[it]-(u_ref[it]-upl)*kh/k

```

```

        plastic=false
        alter=true
    }
}
ft_ref[it]=kep*(u_ref[it]-upl)
}

// --- BHMA 3: MONTEAO BOUC-WEN ---
class BoucWen implements FirstOrderDifferentialEquations {
    Closure p; double m, c, k, a, n, b, gamma, uy, Fy
    BoucWen(double m, double c, double k, double a, double n, double b, double
gamma, double uy, double Fy, Closure p) {
        this.m=m; this.c=c; this.k=k; this.a=a; this.n=n; this.b=b; this.gamma=gamma;
this.uy=uy; this.Fy=Fy; this.p=p
    }
    int getDimension() { return 3 }
    void computeDerivatives(double t, double[] y, double[] yDot) {
        double dz = (1.0/uy)*(1.0 - Math.pow(Math.abs(y[2]),
n)*(gamma*Math.signum(y[1]*y[2]) + b))*y[1]
        yDot[0] = y[1]
        yDot[1] = (p(t) - c*y[1] - (a*k*y[0] + (1.0-a)*Fy*y[2]))/m
        yDot[2] = dz
    }
}

// --- BHMA 4: ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ---
Ra = { params ->
    CRK = new ClassicalRungeKuttaIntegrator(dt)
    ode = new BoucWen(m, c, k, params[0], params[1], params[2], params[3], uy, fsy,
f, input)
    double[] y = [0.0, 0.0, 0.0]; tracker = new ContinuousOutputModel()
    CRK.addStepHandler(tracker); CRK.integrate(ode, 0.0, y, duration, y)
    err = 0.0
    (0..N).each {
        tracker.setInterpolatedTime(it*dt);
        err1=abs((tracker.getInterpolatedState()[0] - u_ref[it]))
        tracker.setInterpolatedTime((it+1)*dt);
        err2=abs((tracker.getInterpolatedState()[0] - u_ref[(it+1)]))
        err += 0.5*(err1+err2)*dt
    }
    return err
}

minBound=[0.00, 1.0, 0.0, 0.0]
maxBound=[0.3, 25.0, 2.0, 2.0]
demin = new DEminimizer(Ra as DF, 4, minBound as double[], maxBound as
double[])
level=3
demin.setMaxData(10*level)

```

```
demin.setMaxIterations(100*level)
sol = demin.Minimize()

// --- ΒΗΜΑ 5: ΕΚΤΥΠΩΣΗ & ΟΠΤΙΚΗ ΕΠΑΛΗΘΕΥΣΗ ΚΑΙ ΔΙΑΓΡΑΜΜΑΤΑ
---

println "-----"
println "DE parameters level="+level
println "DE minBound="+minBound
println "DE maxBound="+maxBound
println "OPTIMAL PARAMETERS (sol): " + sol
println "FINAL ERROR (Ra): " + Ra(sol)
println "-----"

// 5α. Επανεκτέλεση με τη βέλτιστη λύση
CRK_final = new ClassicalRungeKuttaIntegrator(dt)
ode_final = new BoucWen(m, c, k, sol[0], sol[1], sol[2], sol[3], uy, fsy, f_input)
double[] y_final = [0.0, 0.0, 0.0]
tracker_final = new ContinuousOutputModel()
CRK_final.addStepHandler(tracker_final)
CRK_final.integrate(ode_final, 0.0, y_final, duration, y_final)

u_final = []; ft_final = []
(0..N).each {
    tracker_final.setInterpolatedTime(it * dt)
    double[] state = tracker_final.getInterpolatedState()
    u_final.add(state[0])
    ft_final.add(k*sol[0]* state[0]+(1.0 - sol[0]) * fsy * state[2])
}

// 5β. Διάγραμμα Χρονοϊστορίας (Σύγκριση Ref vs Opt)
compPlot = new PlotFrame()
compPlot.setTitle("Validation: Displacement Time History")
pf=new plotfunction(dt, u_ref as double[])
pf.Name="ref"
compPlot.addFunction(pf)
pf=new plotfunction(dt, u_final as double[])
pf.Name="opt"
compPlot.addFunction(pf)
compPlot.makeLegend(true)
compPlot.setAutoColor(true)
compPlot.show()

// 5γ. Διάγραμμα Υστερητικού Βρόχου (Σύγκριση)
hPlot = new PlotFrame()
hPlot.setTitle("Total Internal Force (Validation)")
pf=new plotfunction(u_ref as double[], ft_ref as double[]); pf.Name="ref"
hPlot.addFunction(pf)
pf=new plotfunction(u_final as double[], ft_final as double[]); pf.Name="opt"
hPlot.addFunction(pf)
```

```

hPlot.vline(uy)
hPlot.hline(fsy)
hPlot.vline(-uy)
hPlot.hline(-fsy)
hPlot.makeLegend(true)
hPlot.setAutoColor(true)
hPlot.show()

// 5δ. Διάγραμμα Χρονοϊστορίας Συνολικής Εσωτερικής Δύναμης (Σύγκριση Ref vs
Opt)
forcePlot = new PlotFrame()
forcePlot.setTitle("Validation: Total Internal Force Time History")
pf=new plotfunction(dt, ft_ref as double[])
pf.Name="ref"
forcePlot.addFunction(pf)
pf=new plotfunction(dt, ft_final as double[])
pf.Name="opt"
forcePlot.addFunction(pf)
forcePlot.hline(fsy)
forcePlot.hline(-fsy)
forcePlot.makeLegend(true)
forcePlot.setAutoColor(true)
forcePlot.show()

```

ΣΤ. Αναγνώριση τεσσάρων παραμέτρων υστερητικού μοντέλου Bouc-Wen από ελαστοπλαστική απόκριση αναφοράς υπό δυναμικό πλήγμα με Διαφορική Εξέλιξη.

```

import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel
import courses.structuraldynamics.synthaccel

// --- BHMA 1: ΟΡΙΣΜΟΣ ΔΙΕΓΕΡΣΗΣ (ΠΛΗΓΜΑ) ---
duration = 20.0; steps = 8000; dt = duration/steps
td=1.0
Pso=10.0d

Triangularpulse={
  t=it
  val=Pso*(1.0d-t/td)
  if(t<0 || t>td)val=0.0d
  return val
}
time=[]; steps.times{time.add(it*dt)}
blastPlot=new PlotFrame()
blastPlot.addFunction(new plotfunction(time as double[],Triangularpulse as DF))
blastPlot.show()

```

```

m = 10.0; k = 1000.0; c = 1.0; fsy = 10.0; uy = fsy/k
f_input = Triangularpulse

// --- ΒΗΜΑ 2: ΠΑΡΑΓΩΓΗ "ΠΡΑΓΜΑΤΙΚΗΣ" ΑΠΟΚΡΙΣΗΣ (ΕΛΑΣΤΟΠΛΑΣΤΙΚΟ) ---

g=1.0/2.0 // gamma
b=1.0/4.0 // beta
upl=0.0
f=f_input
kh=0.1*k
N = steps
u_ref=new double[N+1]; u_ref[0]=0.0
v=new double[N+1]; v[0]=0.0
a=new double[N+1]; a[0]=(f(0)-c*v[0]-k*u_ref[0])/m
ft_ref=new double[N+1]; ft_ref[0]=0.0

plastic=false
kep=k
// iterative computations
(1..N).each{
    // K_effective
    kef=kep+m/(b*dt*dt+c*g/(b*dt))

    // effective load
    fef=f(it*dt)-f((it-1)*dt)+m*(v[it-1]/(b*dt)+a[it-1]/(2.0*b))+c*(v[it-1]*g/b+a[it-1]*dt*(g/(2.0*b)-1.0))

    // solve for current displacement step
    du=fef/kef
    u_ref[it]=u_ref[it-1]+du

    // current acceleration and velocity
    v[it]=v[it-1]+du*g/(b*dt)-v[it-1]*g/b+a[it-1]*g*(1.0-g/(2.0*b))
    a[it]=a[it-1]+( f(it*dt)-f((it-1)*dt) - c*(v[it]-v[it-1]) - kep*(u_ref[it]-u_ref[it-1]))/m

    // check yielding
    yield=kh*u_ref[it]+(k-kh)*uy*signum(v[it])
    if(!plastic){
        if(v[it]>0.0 && kep*(u_ref[it]-upl)>yield){
            kep=kh
            upl=(1.0-k/kh)*uy
            plastic=true
            alter=true
        }
        if(v[it]<0.0 && kep*(u_ref[it]-upl)<-yield){
            kep=kh
            upl=(k/kh-1.0)*uy
            plastic=true
            alter=true
        }
    }
}

```

```

    }
    }else{
        if(v[it]*v[it-1]<0){
            kep=k
            upl=u_ref[it]-(u_ref[it]-upl)*kb/k
            plastic=false
            alter=true
        }
    }
    ft_ref[it]=kep*(u_ref[it]-upl)
}

// --- BHMA 3: MONTEΛO BOUC-WEN ---
class BoucWen implements FirstOrderDifferentialEquations {
    Closure p; double m, c, k, a, n, b, gamma, uy, Fy
    BoucWen(double m, double c, double k, double a, double n, double b, double gamma,
double uy, double Fy, Closure p) {
        this.m=m; this.c=c; this.k=k; this.a=a; this.n=n; this.b=b; this.gamma=gamma;
this.uy=uy; this.Fy=Fy; this.p=p
    }
    int getDimension() { return 3 }
    void computeDerivatives(double t, double[] y, double[] yDot) {
        double dz = (1.0/uy)*(1.0 - Math.pow(Math.abs(y[2]),
n)*(gamma*Math.signum(y[1]*y[2]) + b))*y[1]
        yDot[0] = y[1]
        yDot[1] = (p(t) - c*y[1] - (a*k*y[0] + (1.0-a)*Fy*y[2]))/m
        yDot[2] = dz
    }
}

// --- BHMA 4: ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ---
Ra = { params ->
    CRK = new ClassicalRungeKuttaIntegrator(dt)
    ode = new BoucWen(m, c, k, params[0], params[1], params[2], params[3], uy, fsy,
f_input)
    double[] y = [0.0, 0.0, 0.0]; tracker = new ContinuousOutputModel()
    CRK.addStepHandler(tracker); CRK.integrate(ode, 0.0, y, duration, y)
    err = 0.0
    (0..N).each {
        tracker.setInterpolatedTime(it*dt);
        err1=abs((tracker.getInterpolatedState()[0] - u_ref[it]))
        tracker.setInterpolatedTime((it+1)*dt);
        err2=abs((tracker.getInterpolatedState()[0] - u_ref[(it+1)]))
        err += 0.5*(err1+err2)*dt
    }
    return err // that is the integral in time of the error
}

minBound=[0.00, 1.0, 0.0, 0.0]

```

```

maxBound=[0.3, 25.0, 2.0, 2.0]
demin = new DEminimizer(Ra as DF, 4, minBound as double[], maxBound as double[])
level=3
demin.setMaxData(10*level) // default -> 30*Dimensions (30*4)
demin.setMaxIterations(100*level) // default -> 300
sol = demin.Minimize()

// --- ΒΗΜΑ 5: ΕΚΤΥΠΩΣΗ & ΟΠΤΙΚΗ ΕΠΑΛΗΘΕΥΣΗ ΚΑΙ ΔΙΑΓΡΑΜΜΑΤΑ ---
println "-----"
println "DE parameters level="+level
println "DE minBound="+minBound
println "DE maxBound="+maxBound
println "OPTIMAL PARAMETERS (sol): " + sol
println "FINAL ERROR (Ra): " + Ra(sol)
println "-----"

// 5α. Επανεκτέλεση με τη βέλτιστη λύση
CRK_final = new ClassicalRungeKuttaIntegrator(dt)
ode_final = new BoucWen(m, c, k, sol[0], sol[1], sol[2], sol[3], uy, fsy, f_input)
double[] y_final = [0.0, 0.0, 0.0]
tracker_final = new ContinuousOutputModel()
CRK_final.addStepHandler(tracker_final)
CRK_final.integrate(ode_final, 0.0, y_final, duration, y_final)

u_final = []; ft_final = []
(0..N).each {
    tracker_final.setInterpolatedTime(it * dt)
    double[] state = tracker_final.getInterpolatedState()
    u_final.add(state[0])
    ft_final.add(k*sol[0]* state[0]+(1.0 - sol[0]) * fsy * state[2]) // Συνολική Εσωτερική
    δύναμη
}

// 5β. Διάγραμμα Χρονοϊστορίας (Σύγκριση Ref vs Opt)
compPlot = new PlotFrame()
compPlot.setTitle("Validation: Displacement Time History")
pf=new plotfunction(dt, u_ref as double[])
pf.Name="ref"
compPlot.addFunction(pf) // (Ελαστοπλαστικό)
pf=new plotfunction(dt, u_final as double[])
pf.Name="opt"
compPlot.addFunction(pf) // (Bouc-Wen)
compPlot.makeLegend(true)
compPlot.setAutoColor(true)
compPlot.show()

// 5γ. Διάγραμμα Συνολικής Εσωτερικής Δύναμης (Σύγκριση)
hPlot = new PlotFrame()
hPlot.setTitle("Total Internal force (Validation)")

```

```
pf=new plotfunction(u_ref as double[], ft_ref as double[]); pf.Name="ref"
hPlot.addFunction(pf)
pf=new plotfunction(u_final as double[], ft_final as double[]); pf.Name="opt"
hPlot.addFunction(pf)
hPlot.vline(uy)
hPlot.hline(fsy)
hPlot.vline(-uy)
hPlot.hline(-fsy)
hPlot.makeLegend(true)
hPlot.setAutoColor(true)
hPlot.show()

// 5δ. Διάγραμμα Χρονοϊστορίας Συνολικής Εσωτερικής Δύναμης (Σύγκριση Ref vs Opt)
forcePlot = new PlotFrame()
forcePlot.setTitle("Validation: Displacement Time History")
pf=new plotfunction(dt, ft_ref as double[])
pf.Name="ref"
forcePlot.addFunction(pf) // (Ελαστοπλαστικό)
pf=new plotfunction(dt, ft_final as double[])
pf.Name="opt"
forcePlot.addFunction(pf) // (Bouc-Wen)
forcePlot.hline(fsy)
forcePlot.hline(-fsy)
forcePlot.makeLegend(true)
forcePlot.setAutoColor(true)
forcePlot.show()
```

Z. Αναγνώριση δώδεκα παραμέτρων υστερητικού μοντέλου Bouc-Wen σε πολυβάθμιο σύστημα (MDOF τριών ορόφων) από ελαστοπλαστική απόκριση αναφοράς υπό σεισμική διέγερση με Διαφορική Εξέλιξη (N=4000).

```
import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel
import courses.structuraldynamics.synthaccel

// -- 1. ΕΝΣΩΜΑΤΩΣΗ ΓΕΝΙΚΕΥΜΕΝΗΣ ΚΛΑΣΗΣ BOUCWENMDOF (Για N
ορόφους) --
class BoucWenMDOF implements FirstOrderDifferentialEquations {
    Closure p
    int numFloors
    double[] m, c, k, uy, Fy
    double[] a, n, b, gamma

    BoucWenMDOF(int numFloors, double[] m, double[] c, double[] k, double[] uy,
double[] Fy,
        double[] params, Closure p) {
        this.numFloors = numFloors
```

```

this.m = m; this.c = c; this.k = k; this.uy = uy; this.Fy = Fy; this.p = p

this.a = new double[numFloors]; this.n = new double[numFloors]
this.b = new double[numFloors]; this.gamma = new double[numFloors]

for (int i = 0; i < numFloors; i++) {
    this.a[i] = params[i * 4 + 0]
    this.n[i] = params[i * 4 + 1]
    this.b[i] = params[i * 4 + 2]
    this.gamma[i] = params[i * 4 + 3]
}

int getDimension() { return 3 * numFloors; }

void computeDerivatives(double t, double[] y, double[] yDot) {
    double[] u = new double[numFloors]
    double[] v = new double[numFloors]
    double[] z = new double[numFloors]

    for (int i = 0; i < numFloors; i++) {
        u[i] = y[i]
        v[i] = y[numFloors + i]
        z[i] = y[2 * numFloors + i]
    }

    double[] du = new double[numFloors]; double[] dv = new double[numFloors]
    du[0] = u[0]; dv[0] = v[0]
    for (int i = 1; i < numFloors; i++) {
        du[i] = u[i] - u[i - 1]
        dv[i] = v[i] - v[i - 1]
    }

    double[] dz = new double[numFloors]; double[] F = new double[numFloors]
    for (int i = 0; i < numFloors; i++) {
        dz[i] = (1.0 / uy[i]) * (1.0 - Math.pow(Math.abs(z[i]), n[i]) * (gamma[i] *
Math.signum(dv[i] * z[i] + b[i])) * dv[i]
        F[i] = a[i] * k[i] * du[i] + (1.0 - a[i]) * Fy[i] * z[i]
    }

    // uDot = v
    for (int i = 0; i < numFloors; i++) yDot[i] = v[i]

    // Εξισώσεις Κίνησης (Σεισμική Διέγερση: -m_i * ug_ddot)
    double ug_dot_dot = p(t)

    // Όρος 1
    double totalC_0 = (numFloors > 1) ? (c[0] * v[0] - c[1] * dv[1]) : (c[0] * v[0])
    double totalF_0 = (numFloors > 1) ? (F[0] - F[1]) : F[0]

```

```

yDot[numFloors] = (-m[0] * ug_dot_dot - totalC_0 - totalF_0) / m[0]

// Ενδιάμεσοι Όροφοι
for (int i = 1; i < numFloors - 1; i++) {
    yDot[numFloors + i] = (-m[i] * ug_dot_dot - c[i] * dv[i] + c[i + 1] * dv[i + 1] - F[i] + F[i + 1]) / m[i]
}

// Τελανταίος Όροφος
if (numFloors > 1) {
    int last = numFloors - 1
    yDot[numFloors + last] = (-m[last] * ug_dot_dot - c[last] * dv[last] - F[last]) / m[last]
}

// zDot = dz
for (int i = 0; i < numFloors; i++) yDot[2 * numFloors + i] = dz[i]
}

// --- 2. ΚΥΡΙΩΣ ΠΡΟΓΡΑΜΜΑ ---
// *** ΑΛΛΑΓΗ: steps = 4000, dt = 0.005 sec (αντί 2000 / 0.01) ***
duration = 20.0; steps = 4000; dt = duration / steps
N = steps

// ΟΡΙΣΜΟΣ ΑΡΙΘΜΟΥ ΟΡΟΦΩΝ
numFloors = 3

// --- ΒΗΜΑ 1: ΟΡΙΣΜΟΣ ΔΙΕΓΕΡΣΗΣ (ΣΕΙΣΜΟΣ) ---
synthacc = new synthaccel(70.0, 0.75*2.0*Math.PI, 1.0, 3.0*2.0*Math.PI, 0.6, 4.2*2.0*Math.PI, 0.2)
synthacc.duration = duration
synthacc.t1 = 1.0
synthacc.t2 = 9.0
synthacc.Calc(steps)
acc = (synthacc.getSynthAcc()).collect{ it / 100.0 }

// Διάγραμμα επιταχυνσιολογισμού
accPlot = new PlotFrame()
accPlot.setTitle("Input Ground Motion Time History (Earthquake)")
accPlot.addFunction(new plotfunction(dt, acc as double[]) with { Name = "Ground Acceleration (m/s²)"; it })
accPlot.makeLegend(true); accPlot.setAutoColor(true); accPlot.show()

f_input = { double t ->
    int idx = (int)(t / dt)
    return (idx >= 0 && idx < acc.size()) ? acc[idx] : 0.0
}

```

```
// Δομικές Ιδιότητες ανά όροφο
m_val = [10.0, 8.0, 6.0] as double[]
k_val = [1000.0, 800.0, 600.0] as double[]
c_val = [0.5, 0.4, 0.3] as double[]
fsy_val = [12.0, 10.0, 8.0] as double[]

uy_val = new double[numFloors]
kh_val = new double[numFloors]
for (int i = 0; i < numFloors; i++) {
    uy_val[i] = fsy_val[i] / k_val[i]
    kh_val[i] = 0.1 * k_val[i]
}

gN = 1.0/2.0; bN = 1.0/4.0

// --- ΒΗΜΑ 2: ΠΑΡΑΓΩΓΗ "ΠΡΑΓΜΑΤΙΚΗΣ" ΑΠΟΚΡΙΣΗΣ
(ΕΛΑΣΤΟΠΛΑΣΤΙΚΟ MDOF) ---
u_ref = new double[numFloors][N+1]
v_ref = new double[numFloors][N+1]
a_ref = new double[numFloors][N+1]
z_ref = new double[numFloors][N+1]

plastic = new boolean[numFloors]
kep = k_val.clone()
upl = new double[numFloors]

// Αρχική επιτάχυνση (t=0)
double ug0 = f_input(0)
for (int i = 0; i < numFloors; i++) {
    a_ref[i][0] = -ug0
}

// Αλγόριθμος Newmark για N ορόφους
(1..N).each { step ->
    double t_curr = step * dt; double t_prev = (step - 1) * dt
    double ug_curr = f_input(t_curr); double ug_prev = f_input(t_prev)

    double[] fef = new double[numFloors]
    for (int i = 0; i < numFloors; i++) {
        double dP = -m_val[i] * (ug_curr - ug_prev)
        double inertia = m_val[i] * (v_ref[i][step-1] / (bN * dt) + a_ref[i][step-1] / (2.0
* bN))

        double y_i = v_ref[i][step-1]
        double y_prev_rel = (i == 0) ? y_i : (y_i - v_ref[i-1][step-1])
        double a_i = a_ref[i][step-1]
        double a_prev_rel = (i == 0) ? a_i : (a_i - a_ref[i-1][step-1])
```

```

double damping_curr = c_val[i] * (v_prev_rel * gN / bN + a_prev_rel * dt *
(gN / (2.0 * bN) - 1.0))
double damping_next = 0.0
if (i < numFloors - 1) {
    double v_next_rel = v_ref[i+1][step-1] - v_ref[i][step-1]
    double a_next_rel = a_ref[i+1][step-1] - a_ref[i][step-1]
    damping_next = c_val[i+1] * (v_next_rel * gN / bN + a_next_rel * dt * (gN
/ (2.0 * bN) - 1.0))
}

fef[i] = dP + inertia + damping_curr - damping_next
}

double[][] K_ef = new double[numFloors][numFloors]
double factor_m = 1.0 / (bN * dt * dt)
double factor_c = gN / (bN * dt)

for (int i = 0; i < numFloors; i++) {
    K_ef[i][i] = m_val[i] * factor_m + c_val[i] * factor_c + kep[i]
    if (i < numFloors - 1) {
        K_ef[i][i] += c_val[i+1] * factor_c + kep[i+1]
        K_ef[i][i+1] = -c_val[i+1] * factor_c - kep[i+1]
        K_ef[i+1][i] = -c_val[i+1] * factor_c - kep[i+1]
    }
}

double[] du = fef.clone()
double[][] A = new double[numFloors][numFloors]
for (int i = 0; i < numFloors; i++) A[i] = K_ef[i].clone()

for (int i = 0; i < numFloors - 1; i++) {
    double factor = A[i+1][i] / A[i][i]
    A[i+1][i] = 0.0
    A[i+1][i+1] -= factor * A[i][i+1]
    du[i+1] -= factor * du[i]
}
du[numFloors-1] /= A[numFloors-1][numFloors-1]
for (int i = numFloors - 2; i >= 0; i--) {
    du[i] = (du[i] - A[i][i+1] * du[i+1]) / A[i][i]
}

for (int i = 0; i < numFloors; i++) {
    u_ref[i][step] = u_ref[i][step-1] + du[i]
    v_ref[i][step] = v_ref[i][step-1] + du[i] * gN / (bN * dt) - v_ref[i][step-1] * gN
/ bN + a_ref[i][step-1] * dt * (1.0 - gN / (2.0 * bN))
}

for (int i = 0; i < numFloors; i++) {
    double drift = (i == 0) ? u_ref[i][step] : (u_ref[i][step] - u_ref[i-1][step])

```

```

double y_rel = (i == 0) ? y_ref[i][step] : (y_ref[i][step] - y_ref[i-1][step])
double y_prev = (i == 0) ? y_ref[i][step-1] : (y_ref[i][step-1] - y_ref[i-1][step-1])

1))

double yieldF = kh_val[i] * drift + (k_val[i] - kh_val[i]) * uy_val[i] *
Math.signum(y_rel)

if (!plastic[i]) {
    if (y_rel > 0.0 && kep[i] * (drift - upl[i]) > yieldF) { kep[i] = kh_val[i]; upl[i]
= (1.0 - k_val[i]/kh_val[i])*uy_val[i]; plastic[i] = true }
    if (y_rel < 0.0 && kep[i] * (drift - upl[i]) < yieldF) { kep[i] = kh_val[i]; upl[i]
= (k_val[i]/kh_val[i] - 1.0)*uy_val[i]; plastic[i] = true }
} else if (y_rel * y_prev < 0) {
    kep[i] = k_val[i]
    upl[i] = drift - (drift - upl[i]) * kh_val[i] / k_val[i]
    plastic[i] = false
}

z_ref[i][step] = (kep[i] * (drift - upl[i]) - kh_val[i] * drift) / ((k_val[i] -
kh_val[i]) * uy_val[i])
}

for (int i = 0; i < numFloors; i++) {
    double drift = (i == 0) ? u_ref[i][step] : (u_ref[i][step] - u_ref[i-1][step])
    double y_rel = (i == 0) ? y_ref[i][step] : (y_ref[i][step] - y_ref[i-1][step])
    double F_curr = kep[i] * (drift - upl[i])

    double F_next = 0.0; double y_next_rel = 0.0
    if (i < numFloors - 1) {
        double drift_next = u_ref[i+1][step] - u_ref[i][step]
        F_next = kep[i+1] * (drift_next - upl[i+1])
        y_next_rel = y_ref[i+1][step] - y_ref[i][step]
    }

    double P_eff = -m_val[i] * ug_curr - c_val[i] * y_rel - F_curr
    if (i < numFloors - 1) P_eff += c_val[i+1] * y_next_rel + F_next

    a_ref[i][step] = P_eff / m_val[i]
}
}

// --- BHMA 3: ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ BOUC-WEN ---
solveBoucWenMDOF = { params ->
    def CRK = new ClassicalRungeKuttaIntegrator(dt)
    def ode = new BoucWenMDOF(numFloors, m_val, c_val, k_val, uy_val, fsy_val,
params as double[], f_input)
    double[] y = new double[ode.getDimension()]
    def tracker = new ContinuousOutputModel()
    CRK.addStepHandler(tracker)

```

```

CRK.integrate(ode, 0.0, y, duration, y)

def u_bw = (0..numFloors-1).collect { [] }
def z_bw = (0..numFloors-1).collect { [] }

(0..N).each { step ->
    tracker.setInterpolatedTime(step * dt)
    def state = tracker.getInterpolatedState()
    for (int i = 0; i < numFloors; i++) {
        u_bw[i].add(state[i])
        z_bw[i].add(state[2 * numFloors + i])
    }
}
return [u_bw, z_bw]
}

Ra = { params ->
    def result = solveBoucWenMDOF(params)
    def u_curr = result[0]
    double error = 0.0
    int timesteps = u_curr[0].size()
    timesteps.times { step ->
        for (int i = 0; i < numFloors; i++) {
            error += (u_curr[i][step] - u_ref[i][step]) ** 2
        }
    }
    return error
}

dim = numFloors * 4
level = 1

def minv_list = [], maxv_list = []
numFloors.times {
    minv_list.addAll([0.01, 1.0, 0.0, 0.0])
    maxv_list.addAll([0.50, 25.0, 2.0, 2.0])
}
minv = minv_list as double[]
maxv = maxv_list as double[]

demin = new DEminimizer(Ra as DF, dim, minv, maxv)
demin.setMaxData(10 * level)
demin.setMaxIterations(100 * level)

println "-----"
println "DE parameters level=" + level
println "DE minBound=" + minv_list
println "DE maxBound=" + maxv_list

```

```
println "Executing optimization for ${numFloors}-floor MDOF system
(Earthquake), N=4000..."
sol = demin Minimize()

double totalSSE = Ra(sol)
double totalPoints = (N + 1) * numFloors
double totalRMSE = Math.sqrt(totalSSE / totalPoints)

println "OPTIMAL PARAMETERS (sol): " + sol
println "FINAL RMSE: " + totalRMSE
println "-----"

// --- BHMA 4: ΔΙΑΓΡΑΜΜΑΤΑ ---
def opt_results = solveBoucWenMDOF(sol)
u_opt = opt_results[0]; z_opt = opt_results[1]

time_axis_sparse = []
(0..N).each { if (it % 10 == 0 || it == N) time_axis_sparse.add(it * dt) }

for (int i = 0; i < numFloors; i++) {
    def floorNum = i + 1

    def u_ref_p = []; def u_opt_p = []
    (0..N).each { step ->
        if (step % 10 == 0 || step == N) {
            u_ref_p.add(u_ref[i][step])
            u_opt_p.add(u_opt[i][step])
        }
    }

    def plotFrame = new PlotFrame()
    plotFrame.setTitle("Όροφος ${floorNum}: Χρονοϊστορία Μετατόπισης (ref vs
    opt)")
    plotFrame.addFunction(new plotfunction(time_axis_sparse as double[], u_ref_p
    as double[]) with { Name = "ref"; it })
    plotFrame.addFunction(new plotfunction(time_axis_sparse as double[], u_opt_p
    as double[]) with { Name = "opt"; it })
    plotFrame.makeLegend(true); plotFrame.setAutoColor(true); plotFrame.show()

    def drift_ref = []; def fh_ref = []
    def drift_opt = []; def fh_opt = []

    (0..N).each { step ->
        if (step % 10 == 0 || step == N) {
            double d_ref = (i == 0) ? u_ref[i][step] : (u_ref[i][step] - u_ref[i-1][step])
            double d_opt = (i == 0) ? u_opt[i][step] : (u_opt[i][step] - u_opt[i-1][step])

            drift_ref.add(d_ref); drift_opt.add(d_opt)
        }
    }
}
```

```

        fh_ref.add(z_ref[i][step] * (1.0 - kh_val[i]/k_val[i]) * fsy_val[i])
        fh_opt.add(z_opt[i][step] * (1.0 - sol[i] * 4) * fsy_val[i])
    }
}

def plotFrameH = new PlotFrame()
plotFrameH.setTitle("Όροφος $ {floorNum}: Υστερητικός Βρόχος (ref vs opt)")
plotFrameH.addFunction(new plotfunction(drift_ref as double[], fh_ref as
double[]) with{ Name = "ref"; it })
plotFrameH.addFunction(new plotfunction(drift_opt as double[], fh_opt as
double[]) with{ Name = "opt"; it })
plotFrameH.makeLegend(true); plotFrameH.setAutoColor(true);
plotFrameH.show()
}

```

Η. Αναγνώριση δώδεκα παραμέτρων υστερητικού μοντέλου Bouc-Wen σε πολυβάθμιο σύστημα (MDOF τριών ορόφων) από ελαστοπλαστική απόκριση αναφοράς υπό δυναμικό πλήγμα με Διαφορική Εξέλιξη (N=10000).

```

import org.apache.commons.math3.ode.FirstOrderDifferentialEquations
import org.apache.commons.math3.ode.nonstiff.ClassicalRungeKuttaIntegrator
import org.apache.commons.math3.ode.ContinuousOutputModel

// --- 1. ΕΝΣΩΜΑΤΩΣΗ ΓΕΝΙΚΕΥΜΕΝΗΣ ΚΛΑΣΗΣ BOUCWENMDOF (Για N
ορόφους) ---
class BoucWenMDOF implements FirstOrderDifferentialEquations {
    Closure p
    int numFloors
    double[] m, c, k, uy, Fy
    double[] a, n, b, gamma

    BoucWenMDOF(int numFloors, double[] m, double[] c, double[] k, double[] uy,
double[] Fy,
        double[] params, Closure p) {
        this.numFloors = numFloors
        this.m = m; this.c = c; this.k = k; this.uy = uy; this.Fy = Fy; this.p = p

        this.a = new double[numFloors]; this.n = new double[numFloors]
        this.b = new double[numFloors]; this.gamma = new double[numFloors]

        for (int i = 0; i < numFloors; i++) {
            this.a[i] = params[i * 4 + 0]
            this.n[i] = params[i * 4 + 1]
            this.b[i] = params[i * 4 + 2]
            this.gamma[i] = params[i * 4 + 3]
        }
    }
}

```

```

int getDimension() { return 3 * numFloors; }

void computeDerivatives(double t, double[] y, double[] yDot) {
    double[] u = new double[numFloors]
    double[] v = new double[numFloors]
    double[] z = new double[numFloors]

    for (int i = 0; i < numFloors; i++) {
        u[i] = y[i]
        v[i] = y[numFloors + i]
        z[i] = y[2 * numFloors + i]
    }

    double[] du = new double[numFloors]; double[] dv = new double[numFloors]
    du[0] = u[0]; dv[0] = v[0]
    for (int i = 1; i < numFloors; i++) {
        du[i] = u[i] - u[i - 1]
        dv[i] = v[i] - v[i - 1]
    }

    double[] dz = new double[numFloors]; double[] F = new double[numFloors]
    for (int i = 0; i < numFloors; i++) {
        dz[i] = (1.0 / uv[i]) * (1.0 - Math.pow(Math.abs(z[i]), n[i]) * (gamma[i] *
        Math.signum(dv[i] * z[i]) + b[i])) * dv[i]
        F[i] = a[i] * k[i] * du[i] + (1.0 - a[i]) * Fy[i] * z[i]
    }

    for (int i = 0; i < numFloors; i++) yDot[i] = v[i]

    double totalC_0 = (numFloors > 1) ? (c[0] * v[0] - c[1] * dv[1]) - (c[0] * v[0])
    double totalF_0 = (numFloors > 1) ? (F[0] - F[1]) - F[0]
    yDot[numFloors] = (p(t) - totalC_0 - totalF_0) / m[0]

    for (int i = 1; i < numFloors - 1; i++) {
        yDot[numFloors + i] = (-c[i] * dv[i] + c[i + 1] * dv[i + 1] - F[i] + F[i + 1]) /
        m[i]
    }

    if (numFloors > 1) {
        int last = numFloors - 1
        yDot[numFloors + last] = (-c[last] * dv[last] - F[last]) / m[last]
    }

    for (int i = 0; i < numFloors; i++) yDot[2 * numFloors + i] = dz[i]
}
}

```

// --- 2. ΚΥΡΙΩΣ ΠΡΟΓΡΑΜΜΑ ---

```

duration = 10.0; steps = 10000; dt = duration/steps
N = steps
td = 1.0
Pso = 180.0d

Triangularpulse = { double t ->
    double val = Pso * (1.0d - t / td)
    if (t < 0 || t > td) val = 0.0d
    return val
}
f_input = Triangularpulse

numFloors = 3

m_val = [10.0, 8.0, 6.0] as double[]
k_val = [1000.0, 800.0, 600.0] as double[]
c_val = [0.5, 0.4, 0.3] as double[]
fsy_val = [12.0, 10.0, 8.0] as double[]

uv_val = new double[numFloors]
kh_val = new double[numFloors]
for(int i=0; i<numFloors; i++) {
    uv_val[i] = fsy_val[i] / k_val[i]
    kh_val[i] = 0.1 * k_val[i]
}

g = 1.0/2.0; b = 1.0/4.0

u_ref = new double[numFloors][N+1]
v_ref = new double[numFloors][N+1]
a_ref = new double[numFloors][N+1]
z_ref = new double[numFloors][N+1]

plastic = new boolean[numFloors]
kep = k_val.clone()
upl = new double[numFloors]

a_ref[0][0] = f_input(0) / m_val[0]
for (int i = 1; i < numFloors; i++) {
    a_ref[i][0] = 0.0
}

// --- ΑΛΓΟΡΙΘΜΟΣ NEWMARK ---
(1..N).each { step ->
    double t_curr = step * dt; double t_prev = (step - 1) * dt
    double p1_curr = f_input(t_curr); double p1_prev = f_input(t_prev)

    double[] fef = new double[numFloors]

```

```

for (int i = 0; i < numFloors; i++) {
    double dP = (i == 0) ? (p1_curr - p1_prev) : 0.0
    double inertia = m_val[i] * (y_ref[i][step-1] / (b * dt) + a_ref[i][step-1] / (2.0 *
b))

    double y_i = y_ref[i][step-1]
    double y_prev_rel = (i == 0) ? y_i : (y_i - y_ref[i-1][step-1])
    double a_i = a_ref[i][step-1]
    double a_prev_rel = (i == 0) ? a_i : (a_i - a_ref[i-1][step-1])

    double damping_curr = c_val[i] * (y_prev_rel * g / b + a_prev_rel * dt * (g /
(2.0 * b) - 1.0))
    double damping_next = 0.0
    if (i < numFloors - 1) {
        double y_next_rel = y_ref[i+1][step-1] - y_ref[i][step-1]
        double a_next_rel = a_ref[i+1][step-1] - a_ref[i][step-1]
        damping_next = c_val[i+1] * (y_next_rel * g / b + a_next_rel * dt * (g / (2.0
* b) - 1.0))
    }

    fef[i] = dP + inertia + damping_curr - damping_next
}

double[][] K_eff = new double[numFloors][numFloors]
double factor_m = 1.0 / (b * dt * dt)
double factor_c = g / (b * dt)

for (int i = 0; i < numFloors; i++) {
    K_eff[i][i] = m_val[i] * factor_m + c_val[i] * factor_c + kep[i]
    if (i < numFloors - 1) {
        K_eff[i][i] += c_val[i+1] * factor_c + kep[i+1]
        K_eff[i][i+1] = -c_val[i+1] * factor_c - kep[i+1]
        K_eff[i+1][i] = -c_val[i+1] * factor_c - kep[i+1]
    }
}

double[] du = fef.clone()
double[][] A = new double[numFloors][numFloors]
for (int i = 0; i < numFloors; i++) A[i] = K_eff[i].clone()

for (int i = 0; i < numFloors - 1; i++) {
    double factor = A[i+1][i] / A[i][i]
    A[i+1][i] = 0.0
    A[i+1][i+1] -= factor * A[i][i+1]
    du[i+1] -= factor * du[i]
}
du[numFloors-1] /= A[numFloors-1][numFloors-1]
for (int i = numFloors - 2; i >= 0; i--) {
    du[i] = (du[i] - A[i][i+1] * du[i+1]) / A[i][i]
}

```

```

    }

    for (int i = 0; i < numFloors; i++) {
        u_ref[i][step] = u_ref[i][step-1] + du[i]
        v_ref[i][step] = v_ref[i][step-1] + du[i] * g / (b * dt) - v_ref[i][step-1] * g / b +
        a_ref[i][step-1] * dt * (1.0 - g / (2.0 * b))
    }

    for (int i = 0; i < numFloors; i++) {
        double drift = (i == 0) ? u_ref[i][step] - (u_ref[i-1][step])
        double v_rel = (i == 0) ? v_ref[i][step] - (v_ref[i-1][step])
        double v_prev = (i == 0) ? v_ref[i][step-1] - (v_ref[i-1][step-1])

1)

        double yield = kh_val[i] * drift + (k_val[i] - kh_val[i]) * uv_val[i] *
        Math.signum(v_rel)

        if (!plastic[i]) {
            if (v_rel > 0.0 && kep[i] * (drift - upl[i]) > yield) { kep[i] = kh_val[i]; upl[i]
            = (1.0 - k_val[i]/kh_val[i])*uv_val[i]; plastic[i] = true; }
            if (v_rel < 0.0 && kep[i] * (drift - upl[i]) < -yield) { kep[i] = kh_val[i]; upl[i]
            = (k_val[i]/kh_val[i] - 1.0)*uv_val[i]; plastic[i] = true; }
        } else if (v_rel * v_prev < 0) {
            kep[i] = k_val[i]
            upl[i] = drift - (drift - upl[i]) * kh_val[i] / k_val[i]
            plastic[i] = false
        }

        z_ref[i][step] = (kep[i] * (drift - upl[i]) - kh_val[i] * drift) / ((k_val[i] -
        kh_val[i]) * uv_val[i])
    }

    for (int i = 0; i < numFloors; i++) {
        double drift = (i == 0) ? u_ref[i][step] - (u_ref[i-1][step])
        double v_rel = (i == 0) ? v_ref[i][step] - (v_ref[i-1][step])
        double F_curr = kep[i] * (drift - upl[i])

        double F_next = 0.0
        double v_next_rel = 0.0
        if (i < numFloors - 1) {
            double drift_next = u_ref[i+1][step] - u_ref[i][step]
            F_next = kep[i+1] * (drift_next - upl[i+1])
            v_next_rel = v_ref[i+1][step] - v_ref[i][step]
        }

        double P_eff = ((i == 0) ? p1_curr : 0.0) - c_val[i] * v_rel - F_curr
        if (i < numFloors - 1) P_eff += c_val[i+1] * v_next_rel + F_next

        a_ref[i][step] = P_eff / m_val[i]
    }

```

```

    }
}

// --- 3. ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ BOUC-WEN ---
solveBoucWenMDOF = { params ->
    def CRK = new ClassicalRungeKuttaIntegrator(dt)
    def ode = new BoucWenMDOF(numFloors, m_val, c_val, k_val, uy_val, fsy_val,
params as double[], f_input)
    double[] y = new double[ode.getDimension()]
    def tracker = new ContinuousOutputModel()
    CRK.addStepHandler(tracker)
    CRK.integrate(ode, 0.0, y, duration, y)

    def u_bw = (0..numFloors-1).collect { [] }
    def z_bw = (0..numFloors-1).collect { [] }

    (0..N).each { step ->
        tracker.setInterpolatedTime(step * dt)
        def state = tracker.getInterpolatedState()
        for (int i = 0; i < numFloors; i++) {
            u_bw[i].add(state[i])
            z_bw[i].add(state[2 * numFloors + i])
        }
    }
    return [u_bw, z_bw]
}

Ra = { params ->
    def result = solveBoucWenMDOF(params)
    def u_curr = result[0]
    double error = 0.0
    int timesteps = u_curr[0].size()
    timesteps.times { step ->
        for (int i = 0; i < numFloors; i++) {
            error += (u_curr[i][step] - u_ref[i][step])**2
        }
    }
    return error
}

dim = numFloors * 4
level = 3

def minv_list = [], maxv_list = []
numFloors.times {
    minv_list.addAll([0.00, 1.0, 0.0, 0.0])
    maxv_list.addAll([0.30, 25.0, 2.0, 2.0])
}
minv = minv_list as double[]

```

```

maxv = maxv_list as double[]

demin = new DEminimizer(Ra as DF, dim, minv, maxv)
demin.setMaxData(15 * level)
demin.setMaxIterations(80 * level)

println "-----"
println "DE parameters level=" + level
println "DE minBound=" + minv_list
println "DE maxBound=" + maxv_list
println "Executing optimization for ${numFloors}-floor MDOF system (Blast),
N=10000..."
sol = demin.Minimize()

double totalSSE = Ra(sol)
double totalPoints = (N + 1) * numFloors
double totalRMSE = Math.sqrt(totalSSE / totalPoints)

println "OPTIMAL PARAMETERS (sol): " + sol
println "FINAL RMSE: " + totalRMSE
println "-----"

def opt_results = solveBoucWenMDOF(sol)
u_opt = opt_results[0]; z_opt = opt_results[1]

// --- 4. ΔΙΑΓΡΑΜΜΑΤΑ ---
time_axis = []
(0..N).each { if (it % 40 == 0 || it == N) time_axis.add(it * dt) }

for (int i = 0; i < numFloors; i++) {
    def floorNum = i + 1

    def u_ref_p = []; def u_opt_p = []
    (0..N).each { step ->
        if (step % 40 == 0 || step == N) {
            u_ref_p.add(u_ref[i][step])
            u_opt_p.add(u_opt[i][step])
        }
    }

    def plotFrame = new PlotFrame()
    plotFrame.setTitle("Όροφος ${floorNum}: Χρονοϊστορία Μετατόμισης (ref vs
opt)")
    plotFrame.addFunction(new plotfunction(time_axis as double[], u_ref_p as
double[]) with { Name = "ref"; it })
    plotFrame.addFunction(new plotfunction(time_axis as double[], u_opt_p as
double[]) with { Name = "opt"; it })
    plotFrame.makeLegend(true); plotFrame.setAutoColor(true); plotFrame.show()

```

```

def drift_ref = []; def fh_ref = []
def drift_opt = []; def fh_opt = []

(0..N).each { step ->
    if (step % 40 == 0 || step == N) {
        double d_ref = (i == 0) ? u_ref[i][step] : (u_ref[i][step] - u_ref[i-1][step])
        double d_opt = (i == 0) ? u_opt[i][step] : (u_opt[i][step] - u_opt[i-1][step])

        drift_ref.add(d_ref); drift_opt.add(d_opt)

        fh_ref.add(z_ref[i][step] * (1.0 - 0.1) * fsy_val[i])
        fh_opt.add(z_opt[i][step] * (1.0 - sol[i * 4]) * fsy_val[i])
    }
}

def plotFrameH = new PlotFrame()
plotFrameH.setTitle("Όροφος ${floorNum}: Υστερητικός Βρόχος (ref vs opt)")
plotFrameH.addFunction(new plotfunction(drift_ref as double[], fh_ref as
double[]) with { Name = "ref"; it })
plotFrameH.addFunction(new plotfunction(drift_opt as double[], fh_opt as
double[]) with { Name = "opt"; it })
plotFrameH.makeLegend(true); plotFrameH.setAutoColor(true);
plotFrameH.show()
}

// Γράφημα Κρουστικής Διέγερσης
pulse_p = []
(0..N).each { if (it % 40 == 0 || it == N) pulse_p.add(f_input(it * dt)) }

plotFrame3 = new PlotFrame(); plotFrame3.setTitle("Input Transient Blast Loading
Time History")
plotFrame3.addFunction(new plotfunction(time_axis as double[], pulse_p as
double[]) with { Name = "Blast Force P(t)"; it })
plotFrame3.makeLegend(true); plotFrame3.setAutoColor(true); plotFrame3.show()

```

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.