Ειδική Υπηρεσία Διαχείρισης και Εφαρμογής Δράσεων
στους τομείς Έρευνας, Τεχνολογικής Ανάπτυξης και Καινοτομίας

ΕΝΙΑΙΑ ΔΡΑΣΗ ΚΡΑΤΙΚΩΝ ΕΝΙΣΧΥΣΕΩΝ ΕΡΕΥΝΑΣ, ΤΕΧΝΟΛΟΓΙΚΗΣ ΑΝΑΠΤΥΞΗΣ & ΚΑΙΝΟΤΟΜΙΑΣ

ΕΡΕΥΝΩ – ΔΗΜΙΟΥΡΓΩ – ΚΑΙΝΟΤΟΜΩ

ΕΝΟΤΗΤΑ ΕΡΓΑΣΙΑΣ 2: ΠΑΡΑΜΕΤΡΙΚΗ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΚΑΤΑΣΚΕΥΩΝ

Π2.1 Εφαρμογή της διαδικασίας Παραμετρικής βελτιστοποίησης στο HP-OCP

Κωδικός Έργου:	**Τ1ΕΔΚ-05603**
Τίτλος Έργου:	Ολιστική Υπολογιστική Πλατφόρμα Βελτιστοποίησης Υψηλών Επιδόσεων – HP-OCP
Φορείς:	ΕΜΠ-ISAAR, ACE HELLAS AE, DBC A.E.
Χρόνος Παράδοσης:	Από το πρόγραμμα: M24 Υλοποιήθηκε: M24 (Σεπτέμβριος 2020)

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Ταμείο Περιφερειακής Ανάπτυξης

ΕΠΑνΕΚ 2014-2020
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΑΝΤΑΓΩΝΙΣΤΙΚΟΤΗΤΑ
ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΤΗΤΑ
ΚΑΙΝΟΤΟΜΙΑ

ΕΣΠΑ 2014-2020

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Εισαγωγή

Το συγκεκριμένο παραδοτέο αποτελεί την παρουσίαση της διερεύνησης των δυνατοτήτων παραμετρικής βελτιστοποίησης που έχουν αναπτυχθεί στην παρούσα φάση ως ανεξάρτητο λογισμικό ως μέρος του Π2.1 Εφαρμογή της διαδικασίας παραμετρικής βελτιστοποίησης στο HP-OCP, (M24 ACE HELLAS AE), για να ενσωματωθεί στην συνέχεια ως ενιαίο λογισμικό στο HP-OCP. Πιο συγκεκριμένα γίνεται διερεύνηση αποδοτικότητας του λογισμικού Παραμετρικής Βελτιστοποίησης σε προβλήματα βελτιστοποίησης κατασκευών.

Η διερεύνηση συνοδεύεται από οδηγίες που αφορούν την διασύνδεση του λογισμικού παραμετρικής βελτιστοποίησης, το οποίο είναι δομημένο σε μορφή βιβλιοθήκης API (Application Programming Interface), για την διασύνδεση του με το αντίστοιχο λογισμικό στατικής ανάλυσης-διαστασιολόγησης.

Στον σύνδεσμο αυτό: http://users.ntua.gr/nlagaros/parametric.mp4, παρουσιάζεται σε μορφή video η χρήση του λογισμικού παραμετρικής βελτιστοποίησης, ενώ υπάρχει δυνατότητα παροχής πρόσβασης για δοκιμή σε Η/Υ στον οποίο είναι εγκατεστημένο. Γραφικό περιβάλλον χρήσης (GUI) αναπτύσσεται στα πλαίσια Π5.1 που αφορά την ανάπτυξη του ενιαίου λογισμικού HP-OCP.

ΕΦΑΡΜΟΓΕΣ ΠΑΡΑΜΕΤΡΙΚΗΣ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗΣ ΣΕ ΠΡΟΒΛΗΜΑΤΑ ΚΑΤΑΣΚΕΥΩΝ ΠΟΛΙΤΙΚΟΥ ΜΗΧΑΝΙΚΟΥ

1.1 Εισαγωγή

Ο όρος «βελτιστοποίηση» στα εφαρμοσμένα μαθηματικά αναφέρεται στην αναζήτηση βέλτιστων παραμέτρων ενός συστήματος και μπορεί να οριστεί ως η διαδικασία εξεύρεσης των συνθηκών εκείνων που δίνουν το μέγιστο ή το ελάχιστο μίας συνάρτησης. Η ύπαρξη μεθόδων βελτιστοποίησης εντοπίζεται στις ημέρες των Newton, Lagrange και Cauchy.

Σήμερα, ο όρος "βέλτιστος σχεδιασμός κατασκευών" μπορεί να ερμηνευτεί με πολλούς τρόπους. Ο όρος "κατασκευή" χρησιμοποιείται για να περιγράψει τη διάταξη των στοιχείων ή / και των υλικών προκειμένου να δημιουργηθεί ένα σύστημα ικανό να αναλάβει τα φορτία που επιβάλλονται από τις απαιτήσεις σχεδιασμού. Η διαδικασία που εφαρμόζεται για το σχεδιασμό των κατασκευών είναι μια επαναληπτική διαδικασία με στόχο την επίτευξη του βέλτιστου σχεδιασμού. Η πρόοδος της τεχνολογίας των υπολογιστών δημιούργησε περισσότερες απαιτήσεις. Πλέον, ο σχεδιασμός ενός δομικού συστήματος που ικανοποιεί μόνο τις απαιτήσεις της κατασκευής που σχετίζονται με την ασφάλεια, δεν αρκεί πια.

Στις μέρες μας πια ο βέλτιστος σχεδιασμός ενός δομικού συστήματος είναι ζωτικής σημασίας. Ο όρος "βέλτιστος σχεδιασμός" χρησιμοποιείται για το σχεδιασμό εκείνο ο οποίος όχι μόνο ικανοποιεί τις απαιτήσεις λειτουργικότητας αλλά επίσης ικανοποιεί και κριτήρια όπως την ελαχιστοποίηση του κόστους ή του βάρους του συστήματος. Στόχος του μηχανικού είναι να βρει έναν συνδυασμό ανεξάρτητων μεταβλητών που λαμβάνουν πραγματικές ή ακέραιες τιμές, που ονομάζονται παράμετροι ή μεταβλητές σχεδιασμού, έτσι ώστε να βελτιστοποιηθεί η αντικειμενική συνάρτηση του προβλήματος.

Για τον υπολογισμό του βέλτιστου σχεδιασμού είναι απαραίτητο να εκτελεστούν δύο βασικά βήματα: η μαθηματική διατύπωση του προβλήματος βελτιστοποίησης και η εφαρμογή ενός αλγορίθμου βελτιστοποίησης.

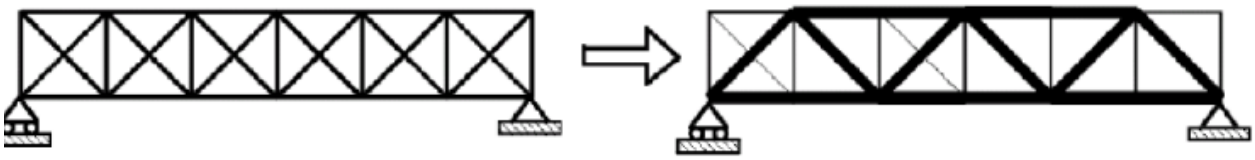
Η ύπαρξη αποτελεσματικών αλγορίθμων βελτιστοποίησης εγγυάται ότι το πρόβλημα του βέλτιστου σχεδιασμού θα αντιμετωπιστεί επιτυχώς. Η εμπειρία του μηχανικού είναι σημαντική παράμετρος για τη σωστή χρήση αυτών των αλγορίθμων. Η διαδικασία σχεδιασμού είναι μια επαναληπτική διαδικασία όπου η επανάληψη θεωρείται ως η διαδοχική δοκιμή υποψήφιων λύσεων και αξιολογεί αν κάθε υποψήφια λύση είναι καλύτερη ή όχι σε σύγκριση με τα προηγούμενα, ενώ ικανοποιεί τους περιορισμούς του προβλήματος. Η συμβατική διαδικασία που χρησιμοποιείται από τους μηχανικούς είναι αυτή της «δοκιμής και διόρθωσης». Φυσικά, με την αύξηση της πολυπλοκότητας και του μεγέθους των προβλημάτων, η χρήση τέτοιων εμπειρικών τεχνικών δεν οδηγεί στη βέλτιστη λύση. Έτσι κατέστη αναγκαία η αυτοματοποίηση του σχεδιασμού των κτιρίων αξιοποιώντας τις εξελίξεις στην τεχνολογία των υπολογιστών και την πρόοδο των αλγορίθμων βελτιστοποίησης.

1.2 Τύποι προβλημάτων βελτιστοποίησης κατασκευών

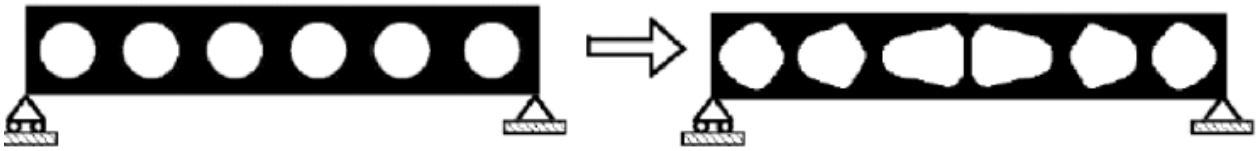
Υπάρχουν τρεις βασικές κατηγορίες προβλημάτων βελτιστοποίησης κατασκευών: μεγέθους (sizing) ή παραμετρική, σχήματος (shape)και τοπολογίας (topology). Στα προβλήματα βελτιστοποίησης μεγέθους ο στόχος είναι συνήθως η ελαχιστοποίηση του βάρους της κατασκευής υπό ορισμένους περιορισμούς στη συμπεριφορά (τάσεις και μετατοπίσεις). Κατά αντιστοιχία, στα προβλήματα βελτιστοποίησης σχήματος ο στόχος είναι να βελτιωθεί η αποτελεσματικότητα της κατασκευής τροποποιώντας τα όριά της.

Αρχικά η βελτιστοποίηση κατασκευών επικεντρώθηκε στην παραμετρική βελτιστοποίηση, όπως η βελτιστοποίηση της διατομής των δοκών και πλαισίων ή το πάχος των πλακών και των κελύφων (Σχ.1). Το επόμενο βήμα ήταν να μελετηθεί η δυνατότητα να βρεθούν τα βέλτιστα όρια μιας κατασκευής και επομένως να βελτιστοποιηθεί το σχήμα της (Σχ.2). Στην πρώτη περίπτωση η κατασκευή έχει σταθερή μορφή, ενώ στην τελευταία περίπτωση δεν είναι σταθερή αλλά έχει μια προκαθορισμένη τοπολογία.

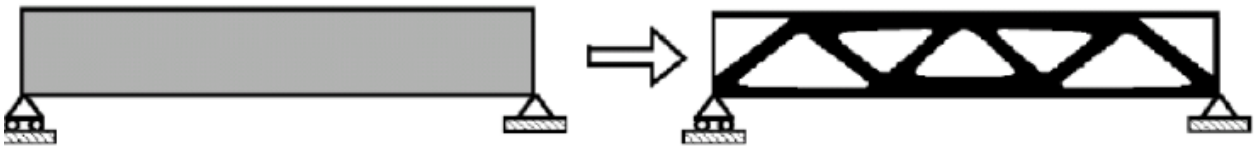
Τελευταία εξέλιξη είναι η βελτιστοποίηση της τοπολογίας, η οποία επιτρέπει στον σχεδιαστή να βελτιστοποιήσει τη διάταξη ή την τοπολογία μιας κατασκευής ανιχνεύοντας και αφαιρώντας το υλικό που παραλαμβάνει μικρή φόρτιση και που στην ουσία δεν χρησιμοποιείται αποτελεσματικά (Σχ.3). Μπορεί να θεωρηθεί ως μια διαδικασία βελτιστοποίησης της ορθολογικής κατανομής του διαθέσιμου υλικού και την εξάλειψη αυτού που δεν χρειάζεται.



Σχ.1 Βελτιστοποίηση μεγέθους (sizing) – παραμετρική βελτιστοποίηση



Σχ.2 Βελτιστοποίηση σχήματος (shape)



Σχ.3 Βελτιστοποίηση τοπολογίας (topology)

1.3 Μέθοδοι / Αλγόριθμοι βελτιστοποίησης

Τα τελευταία χρόνια έχουν αναπτυχθεί αρκετές μέθοδοι βελτιστοποίησης των κατασκευών. Η κατηγοριοποίηση αυτών ποικίλλει μέσα στην υφιστάμενη βιβλιογραφία, ανάλογα με τον τύπο και τα χαρακτηριστικά των αλγορίθμων που χρησιμοποιούν. Οι μέθοδοι βελτιστοποίησης διακρίνονται γενικά σε:

Αιτιοκρατικού ή Προσδιοριστικού ή Ντετερμινιστικού χαρακτήρα**,** δηλαδή στηρίζονται σε αλγόριθμους που κινούνται προς συνεχώς καλύτερες λύσεις με αυστηρά καθορισμένο τρόπο αξιοποιώντας πληροφορία από την «παράγωγο» της αντικειμενικής συνάρτησης (gradient-based).

Πιθανοκρατικού ή Τυχηματικού ή Στοχαστικού χαρακτήρα**,** οι οποίες στηρίζονται στην τεχνική της δημιουργίας και ελέγχου μέσω της τυχαίας αναζήτησης (random search) και δεν εξαρτώνται από την παράγωγο της αντικειμενικής συνάρτησης (derivative-free). Αξιολογούν ένα σύνολο τυχαίων λύσεων του προβλήματος και επιλέγουν την βέλτιστη από αυτές. Δηλαδή δημιουργείται μια τυχαία υποψήφια λύση η οποία ελέγχεται και αν δεν ικανοποιεί κάποια κριτήρια, τότε δημιουργείται νέα υποψήφια λύση. Σε άλλες περιπτώσεις, αντί της μίας τυχαίας υποψήφιας λύσης χρησιμοποιούν έναν «πληθυσμό» (population) υποψήφιων λύσεων του προβλήματος. Οι λύσεις αυτές είναι ανεξάρτητες μεταξύ τους ενώ ο αρχικός πληθυσμός επιλέγεται με τυχαίο τρόπο και στη συνέχεια, ο πληθυσμός αλλάζει κινούμενος σε όλο και καλύτερες περιοχές του χώρου αναζήτησης.

Έτσι, για την επίλυση των προβλημάτων βελτιστοποίησης οι διάφορες μέθοδοι / αλγόριθμοι που έχουν αναπτυχθεί και εξελιχτεί, μπορούν να κατηγοριοποιηθούν σε:

Μαθηματικές μεθόδους, οι οποίες στηρίζονται στις αρχές του μαθηματικού προγραμματισμού, είναι αιτιοκρατικού χαρακτήρα και μπορούν να ταξινομηθούν σε πέντε μεγάλες κατηγορίες:

- Γραμμικός Προγραμματισμός (Linear Programming – LP)
- Μη Γραμμικός Προγραμματισμός Non Linear Programming – NLP)
- Ακέραιος Προγραμματισμός (Integer Programming – IP)
- Δυναμικός προγραμματισμός (Dynamic Programming – DP)
- Γεωμετρικός προγραμματισμός (Geometric Programming – GP).

Τις **ευρεστικές (heuristic) μεθόδους,** οι οποίες είναι εμπειρικές – προσεγγιστικές μέθοδοι τυχηματικού χαρακτήρα και είναι ικανές να εντοπίζουν ικανοποιητικές λύσεις, όχι όμως απαραίτητα βέλτιστες, σε εύλογο χρόνο κυρίως σε προβλήματα μεγάλου μεγέθους ή περίπλοκης δομής. Οι αλγόριθμοι αυτοί πρακτικά σχεδιάζονται ανάλογα με το εκάστοτε πρόβλημα και ως εκ τούτου δεν υπάρχει γενική μορφή.

Στον τομέα της Επιστήμης Υπολογιστών, της τεχνητής νοημοσύνης, και στην Μαθηματική βελτιστοποίηση, ευρεστική καλείται η τεχνική που μπορεί να δώσει την λύση ενός προβλήματος – χωρίς να είναι απαραίτητα η βέλτιστη – αρκετά γρήγορα όταν οι κλασικές μέθοδοι είναι πολύ αργές, ή για την εύρεση λύσης, κατά προσέγγιση, όταν οι κλασικές μέθοδοι αποτύχουν να βρουν οποιαδήποτε ακριβή λύση. Αυτό επιτυγχάνεται βάζοντας ως προτεραιότητα την ταχύτητα σε βάρος της βελτιστοποίησης, της πληρότητας και της ακρίβειας της μεθόδου. Παρόλο που η λύση δεν είναι πάντοτε η βέλτιστη εξακολουθεί να είναι σημαντική, επειδή η εύρεσή της δεν απαιτεί απαγορευτικά μεγάλο χρόνο εκτέλεσης. Χαρακτηριστικό παράδειγμα ευρεστικού αλγορίθμου είναι η «δοκιμή και διόρθωση» (trial and error).

Τους **μετα-ευρεστικούς (metaheuristic) αλγορίθμους,** οι οποίοι είναι επίσης τυχηματικού χαρακτήρα και εμπνευσμένοι από τις βιολογικές και φυσικές διεργασίες. Η αυξημένη πολυπλοκότητα και το μέγεθος κάποιων προβλημάτων έχει καταστήσει τις παραπάνω μεθόδους αναποτελεσματικές και επιτακτική την ανάγκη ανάπτυξης εξελιγμένων αλγορίθμων βελτιστοποίησης.

Σύμφωνα με τους Glover και Sörensen, μετα-ευρεστική μέθοδος είναι ένα υψηλού επιπέδου αλγοριθμικό πλαίσιο που παρέχει ένα πακέτο κατευθυντήριων γραμμών και στρατηγικών με στόχο να αναπτύξει έναν ευρεστικό αλγόριθμο βελτιστοποίησης. Ο όρος χρησιμοποιείται επίσης για να αναφερθεί σε μια εφαρμογή συγκεκριμένου προβλήματος ενός ευρεστικού αλγόριθμου βελτιστοποίησης σύμφωνα με τις κατευθυντήριες γραμμές που εκφράζονται σε ένα τέτοιο πλαίσιο. Οι αλγόριθμοι αυτοί σε αντίθεση με τις μεθόδους αιτιοκρατικού χαρακτήρα, χρησιμοποιούν έναν «πληθυσμό» (population) υποψήφιων λύσεων και δεν εγκλωβίζονται, έτσι, σε τοπικά βέλτιστα με αποτέλεσμα να αυξάνονται οι πιθανότητες εύρεσης καθολικών βέλτιστων λύσεων. Οι σημαντικότεροι **μετα-ευρεστικοί αλγόριθμοι** είναι:

- Μέθοδος Στρατηγικών Εξέλιξης (Evolutionary Strategies – ES)
- Γενετικοί Αλγόριθμοι (Genetic Algorithms – GAs)
- Σμήνος Σωματιδίων (Particle Swarm Optimization – PSO)
- Αποικία Μυρμηγκιών (Ant Colony Optimization – ACO)

1.4 Εφαρμογή αλγορίθμων βελτιστοποίησης στην παραμετρική βελτιστοποίηση

1.4.1 Γενικά

Στην παρούσα εργασία μελετήθηκαν διεξοδικά τα αποτελέσματα βελτιστοποίησης έξι συγκεκριμένων μοντέλων, μετά την εφαρμογή διάφορων αλγορίθμων βελτιστοποίησης. Τα κατασκευαστικά μοντέλα έχουν σχεδιαστεί και αναλυθεί στο πρόγραμμα κατασκευών SAP2000 v.20 της CSI. Η πλατφόρμα βελτιστοποίησης Optimization Computing Platform OCP έχει σχεδιαστεί έτσι ώστε να είναι συμβατή και να μπορεί να λειτουργεί ταυτόχρονα με τα προγράμματα της CSI (SAP2000 και ETABS). Μέσω του OCP, ο χρήστης επιλέγει το

κατασκευαστικό μοντέλο που θέλει να βελτιστοποιήσει επιλέγοντας τις ακόλουθες επιθυμητές παραμέτρους, όπως:

- Αντικειμενική συνάρτηση (Objective function)
- Περιορισμούς (Constraints)
- Κριτήρια Σύγκλισης (Convergence Criteria)
- Όρια Διατομών (Bounds of Section Property Dimensions)

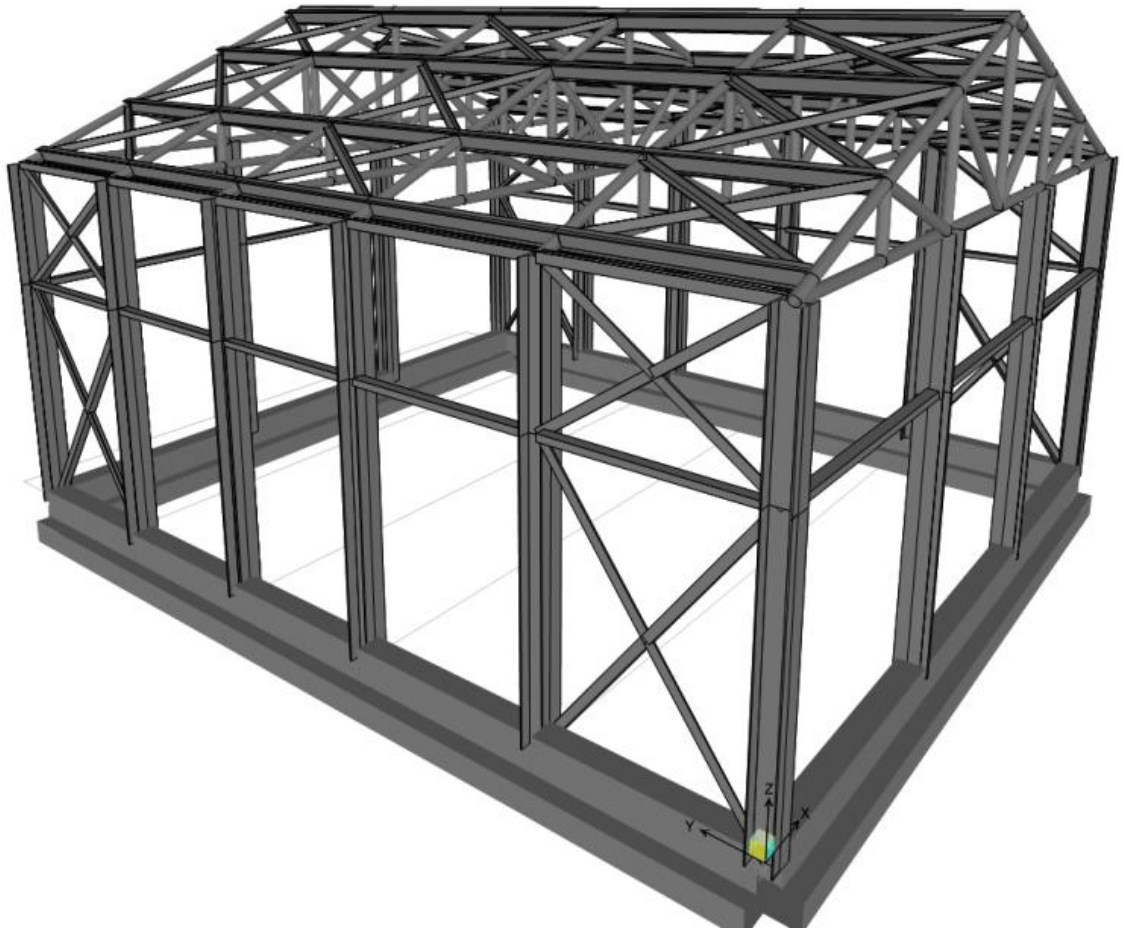
Έτσι, ανάλογα με τον αλγόριθμο που επιλέγεται κάθε φορά, το OCP προσπαθεί να υπολογίσει την βέλτιστη εφικτή λύση μέσω επαναληπτικής διαδικασίας, σύμφωνα με τις παραμέτρους που έχει καθορίσει ο χρήστης, ώστε τελικά να μεγιστοποιήσει ή να ελαχιστοποιήσει την αντικειμενική συνάρτηση. Πρακτικά, το OCP μεταβάλλει σε κάθε επανάληψη (δηλ. σε κάθε ανάλυση του μοντέλου που γίνεται μέσω του SAP2000) τις τιμές των μεταβλητών σχεδιασμού, όπως αυτές προκύπτουν από τον αλγόριθμο που χρησιμοποιείται, προκειμένου να βρει εκείνες τις τιμές που όταν εφαρμοστούν στο μοντέλο θα βελτιστοποιήσουν την αντικειμενική συνάρτηση.

Σε όλες τις περιπτώσεις που μελετήθηκαν, η αντικειμενική συνάρτηση είναι η ελαχιστοποίηση του βάρους της κατασκευής και οι περιορισμοί που έχουν τεθεί είναι οι σχεδιαστικοί (Design Violations) δηλ. οι κατασκευαστικοί και σεισμικοί κώδικες (EC-2, EC-3, EC-8, κτλ.).

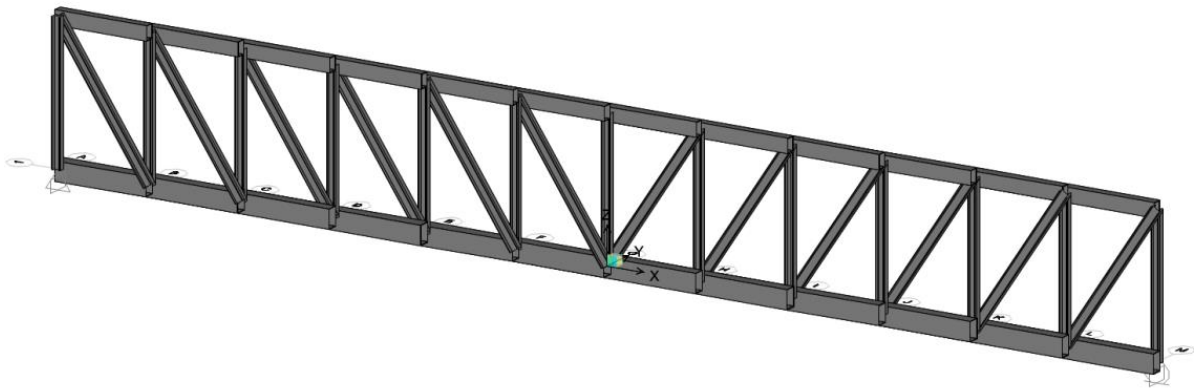
1.4.2 Τύποι μοντέλων διερεύνησης

Τα κατασκευαστικά μοντέλα που χρησιμοποιήθηκαν στο πλαίσιο της παρούσης διπλωματικής εργασίας είναι τα ακόλουθα:

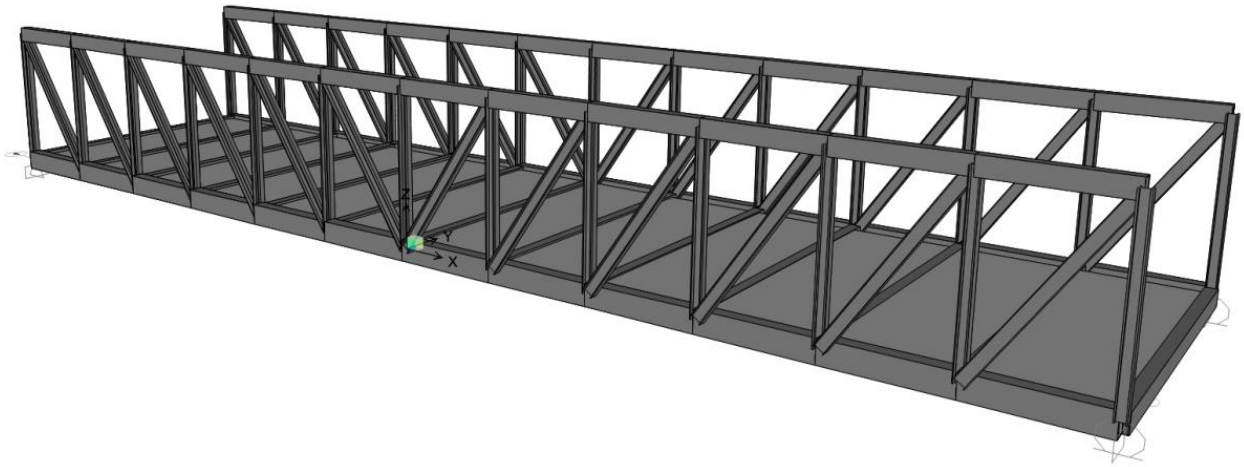
a) Ένα στέγαστρο από χαλύβδινα στοιχεία διαφόρων διατομών και θεμελίωση από οπλισμένο σκυρόδεμα.



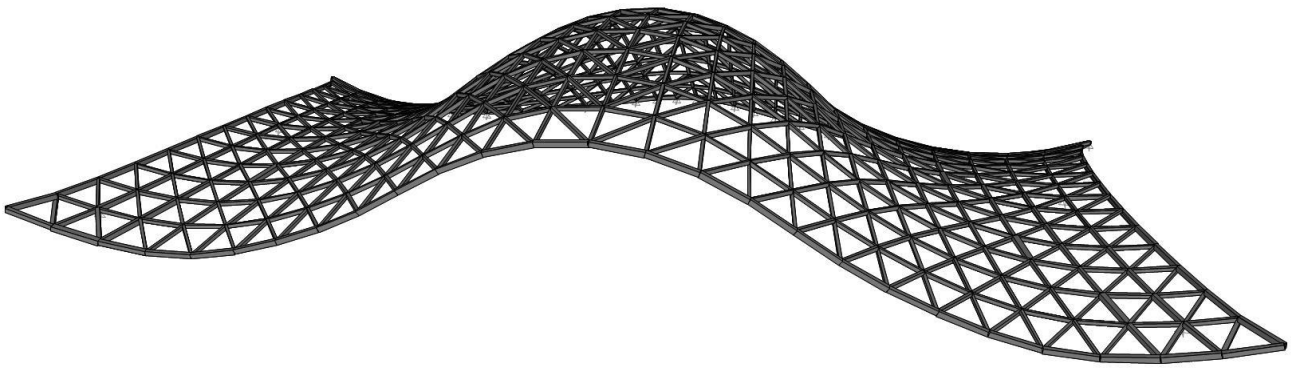
b) Μία 2D δικτυωτή γέφυρα με μεταλλικά στοιχεία διαφόρων διατομών.



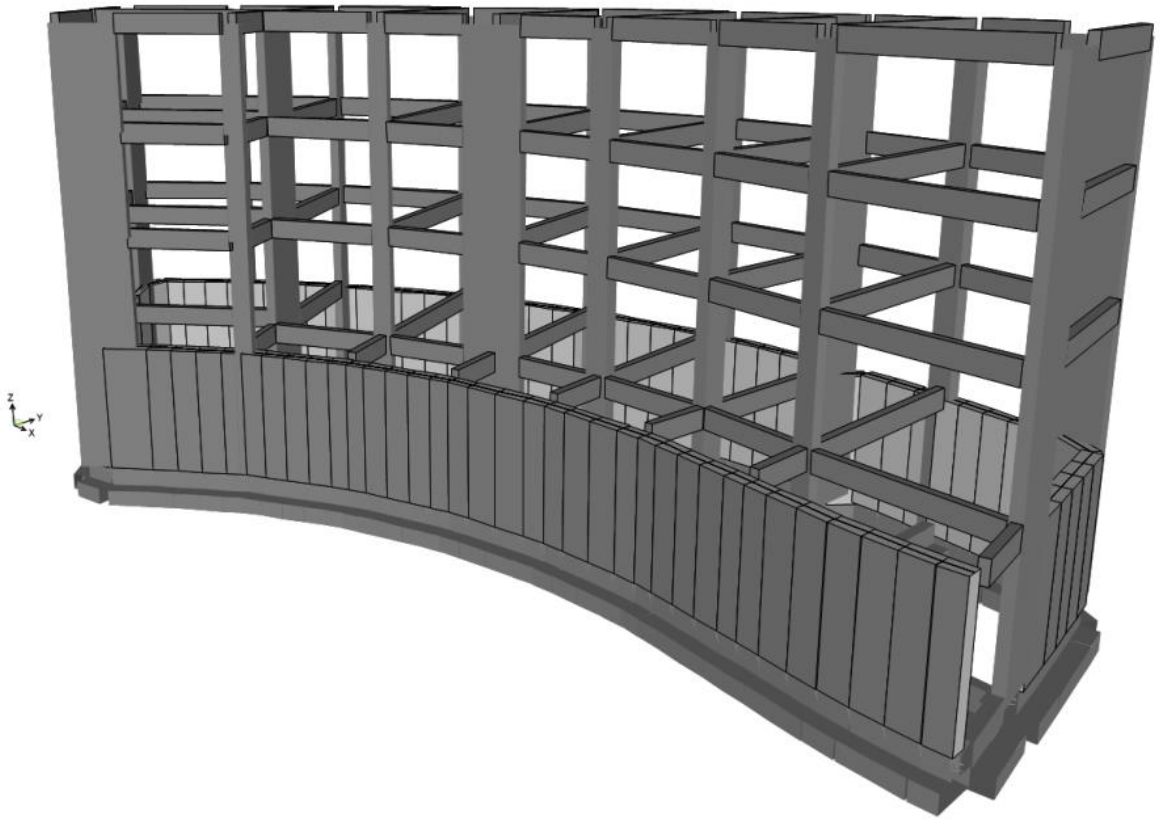
c) Μία 3D σύμμικτη γέφυρα κάτω διάβασης, με κατάστρωμα από οπλισμένο σκυρόδεμα και χαλύβδινες δικτυωτές κύριες δοκούς.



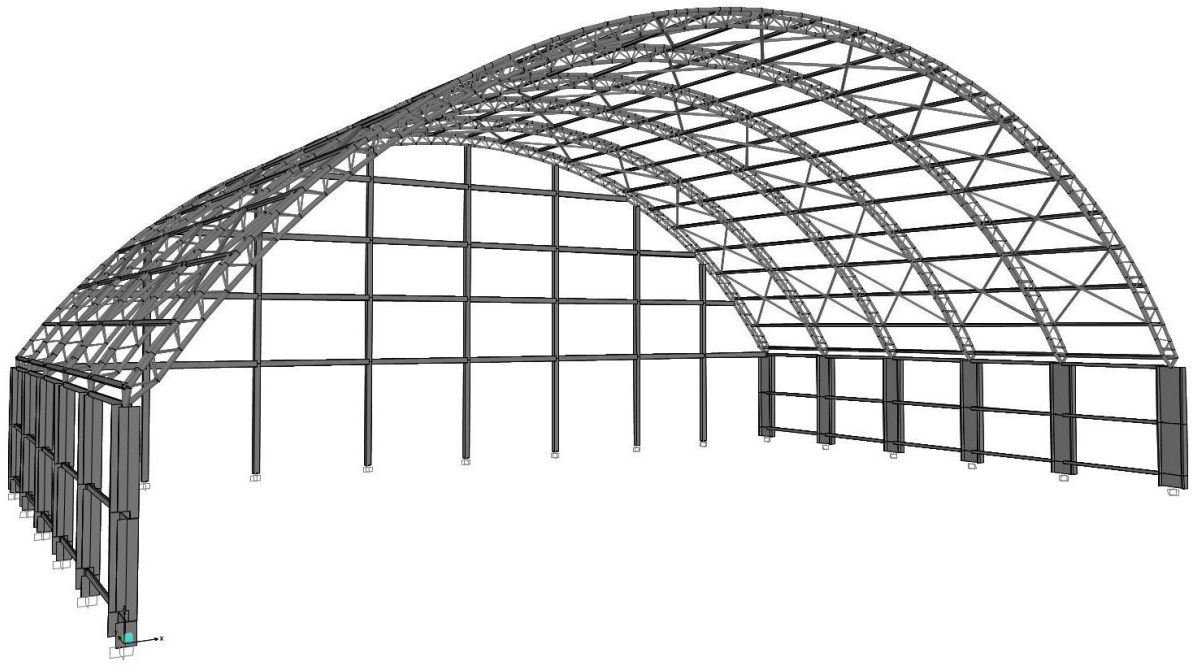
d) Ένα στέγαστρο σχεδιασμένο ως χωροδικτύωμα με χαλύβδινα στοιχεία διαφόρων διατομών.



e) Ένα σχολείο κατασκευασμένο από οπλισμένο σκυρόδεμα.



f) Ένα υπόστεγο από χαλύβδινα στοιχεία διαφόρων διατομών.



1.5 Αλγόριθμοι βελτιστοποίησης διερεύνησης

Για τη διερεύνηση των παραπάνω μοντέλων χρησιμοποιήθηκαν οι ακόλουθοι αλγόριθμοι βελτιστοποίησης:

- PQN Master
- Matsumoto

- Latin Hypercube
- Differential Evolution

Τα αποτελέσματα που προέκυψαν και τα οποία θα αναλυθούν διεξοδικά στη συνέχεια αποδεικνύουν την ντετερμινιστική ή στοχαστική φύση του κάθε αλγορίθμου βελτιστοποίησης. Διαπιστώθηκε ότι, κάθε αλγόριθμος συμπεριφέρεται τελείως διαφορετικά ανάλογα με τα χαρακτηριστικά της κάθε κατασκευής. Έτσι, κάποιοι αλγόριθμοι έδωσαν εξαιρετικά αποτελέσματα όταν εφαρμόστηκαν σε ένα συγκεκριμένο δομικό σύστημα αλλά οι ίδιοι αλγόριθμοι, σε άλλη κατασκευή, είτε εμφάνισαν «κακή» επίδοση είτε δεν κατάφεραν ποτέ να βρουν εφικτές λύσεις κάτω από τους τιθέμενους περιορισμούς της αντικειμενικής συνάρτησης.

1.5.1 Αλγόριθμος PQN Master

Ο αλγόριθμος αυτός βασίζεται στον γνωστό αλγόριθμο Implicit filtering, ο οποίος αποτελεί μία μέθοδο βελτιστοποίησης που δεν κάνει χρήση «παραγώγων» (derivative-free) και έχει αναπτυχθεί τα τελευταία χρόνια. Πρακτικά, είναι ένας συνδυασμός των γνωστών αλγορίθμων Quasi-Newton ή Gauss-Newton που χρησιμοποιούνται για βελτιστοποίηση προβλημάτων που έχουν περιορισμούς (bound constraints), των μη-γραμμικών προβλημάτων ελάχιστων τετραγώνων και ενός ντετερμινιστικού αλγόριθμου αναζήτησης. Τα διαφορικά όμως που απαιτούνται στον αλγόριθμο Quasi-Newton όπως και οι Ιακωβιανοί πίνακες στον Gauss-Newton, εδώ προσεγγίζονται με πεπερασμένες διαφορές.

Ο Implicit filtering είναι κατά βάση μια ντετερμινιστική μέθοδος δειγματοληψίας για βελτιστοποίηση προβλημάτων με οριακές συνθήκες (bound constrained optimization). Οι μέθοδοι δειγματοληψίας, γενικά, ελέγχουν την πρόοδο της βελτιστοποίησης αξιολογώντας τις εφικτές λύσεις της αντικειμενικής συνάρτησης, χωρίς τη χρήση «διαφορικών» (gradients). Σε κάποιες μεθόδους όμως, όπως στη συγκεκριμένη μέθοδο, γίνεται «εκτίμηση» και υπολογισμός της παραγώγου ή/ και άλλων απαιτούμενων πληροφοριών από τη δειγματοληψία.

Μέθοδοι όπως η Implicit filtering έχουν σχεδιαστεί για να ελαχιστοποιούν αντικειμενικές συναρτήσεις που είναι μη ομαλές, πιθανώς ασυνεχείς ή στοχαστικές και οι οποίες μπορεί να μην ορίζονται καν σε όλα τα σημεία του χώρου. Δεν είναι μέθοδοι για «ομαλά» προβλήματα όπου η συνάρτηση που θα ελαχιστοποιηθεί και οι περιορισμοί εκφράζονται σε όρους παραγωγίσιμων συναρτήσεων. Γιατί σε αυτή την περίπτωση υπάρχουν άλλες αποτελεσματικές, αυστηρά ντετερμινιστικές μέθοδοι, που βασίζονται στη χρήση παραγώγων, όπως η μέθοδος μεγίστης καθόδου (Steepest Descent) κτλ. Επίσης, μέθοδοι όπως η Implicit filtering, μπορεί να επιφέρουν σημαντικά οφέλη και σε περιπτώσεις όπου η χρήση άλλων μαθηματικών μεθόδων απαιτούν μεγάλο υπολογιστικό «κόστος» προκειμένου να δημιουργήσουν τις παραγώγους της αντικειμενικής συνάρτησης.

Η Implicit filtering θεωρείται επέκταση της μεθόδου αναζήτησης συντεταγμένων (coordinate search algorithm), την απλούστερη δυνατή προσέγγιση στη βελτιστοποίηση. Διαφέρει όμως από αυτήν, καθώς η Implicit filtering δημιουργεί ένα τοπικό μοντέλο της αντικειμενικής συνάρτησης χρησιμοποιώντας τη μέθοδο Quasi-Newton. Εν ολίγοις, η Implicit filtering επεκτείνει την μέθοδο αναζήτησης συντεταγμένων προσεγγίζοντας ένα διαφορικό με χρήση της μεθόδου παρεμβολής των ελαχίστων τετραγώνων, στη συνέχεια χρησιμοποιεί αυτήν την κατά προσέγγιση «παράγωγο» για να δημιουργήσει ένα νέο απλούστερο μοντέλο και έπειτα αναζητά μία καλύτερη λύση αντλώντας πλέον πληροφορίες από αυτό το μοντέλο. Πρακτικά, εκμεταλλεύεται τα πλεονεκτήματα αφενός των απλών μεθόδων δειγματοληψίας για μη ομαλά προβλήματα ή προβλήματα με ασυνέχειες (derivative-free μέθοδοι) και αφετέρου άλλων αποτελεσματικών μαθηματικών μεθόδων (gradient-based) οι οποίες χρησιμοποιούνται μόνο σε ομαλά και συνεχή προβλήματα και αποφέρουν γρήγορη σύγκλιση.

1.5.2 Αλγόριθμος MATSuMoTo

Ο αλγόριθμος MATSuMoTo (MATLAB Surrogate Model Toolbox) είναι μία μέθοδος βελτιστοποίησης για προβλήματα τα οποία απαιτούν σημαντικό υπολογιστικό κόστος επίλυσης, των οποίων η αντικειμενική συνάρτηση δεν είναι παραγωγίσιμη (derivative-free) και το πεδίο ορισμού των μεταβλητών σχεδιασμού λαμβάνει είτε συνεχείς ή ακέραιες τιμές ή συνδυασμό αυτών.

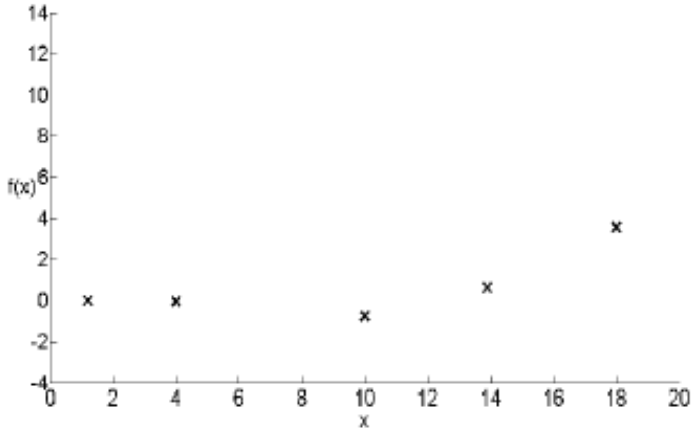
Για την επίλυση των παραπάνω προβλημάτων, ο αλγόριθμος MATSuMoTo χρησιμοποιεί υποκατάστατα μοντέλα (surrogate models) της αρχικής αντικειμενικής συνάρτησης για την

εύρεση βέλτιστων λύσεων, δηλαδή απλούστερες συναρτήσεις που απαιτούν έναν πολύ περιορισμένο αριθμό υπολογισμών. Η προσέγγιση αυτή, δηλαδή η χρήση υποκατάστατου μοντέλου, απαιτεί σε σύγκριση με αλγόριθμους που λειτουργούν με βάση τον «πληθυσμό» υποψήφιων λύσεων (όπως για παράδειγμα οι γενετικοί αλγόριθμοι) σημαντικά λιγότερους υπολογισμούς και επομένως είναι καταλληλότερη για προβλήματα που έχουν υψηλό υπολογιστικό κόστος. Πρακτικά, η αρχική περίπλοκη (άρα και υπολογιστικά «ακριβή») αντικειμενική συνάρτηση (αρχικό μοντέλο) διασπάται σε δύο απλούστερες συναρτήσεις. Η μία προσομοιώνει το υποκατάστατο μοντέλο που απαιτεί μικρότερο υπολογιστικό κόστος επίλυσης και η δεύτερη αφορά τη συνάρτηση διαφοράς μεταξύ του αρχικού και υποκατάστατου μοντέλου.

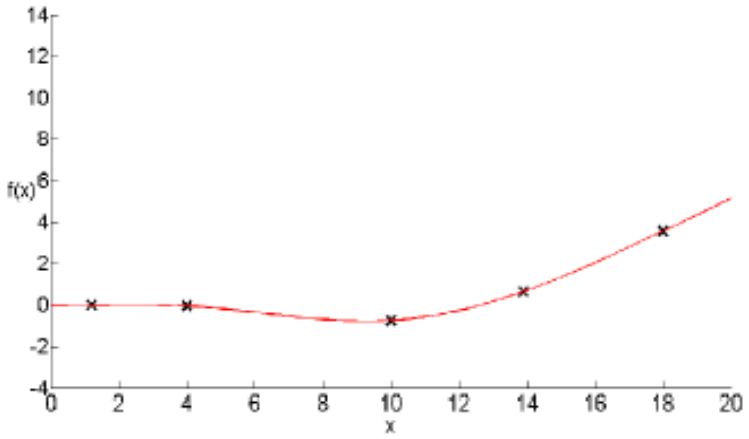
Ο αλγόριθμος MATSuMoTo ακολουθεί τα παρακάτω βήματα προκειμένου να επιλύσει ένα πρόβλημα:

1. Δημιουργία ενός αρχικού μοντέλου με χρήση διάφορων μεθόδων όπως για παράδειγμα η μέθοδος δειγματοληψίας Latin hypercube.
2. Υπολογισμός τιμών αρχικής αντικειμενικής συνάρτησης για τα επιλεγμένα σημεία δειγματοληψίας.
3. Κατασκευή υποκατάστατου μοντέλου (surrogate model) δηλ. μιας απλούστερης συνάρτησης που προσαρμόζεται και ικανοποιεί τα παραπάνω σημεία, με χρήση μεθόδων όπως η Radial Basis Function (RBF) και η Multivariate Adaptive Regression Splines (MARS).
4. Επιλογή νέων σημείων από το πεδίο ορισμού, βάσει διάφορων κριτηρίων και με χρήση πληροφορίας πλέον από το νέο (απλούστερο) υποκατάστατο μοντέλο.
5. Υπολογισμός τιμών αρχικής αντικειμενικής συνάρτησης για τα νέα επιλεγμένα σημεία.
6. Έλεγχος κριτηρίων τερματισμού της διαδικασίας (stopping criterion), κριτήρια όπως μέγιστος αριθμός υπολογισμού τιμών αντικειμενικής συνάρτησης, ή μέγιστος αριθμός επαναλήψεων, κ.α.
7. Εφόσον τα κριτήρια τερματισμού δεν έχουν επιτευχθεί, ο αλγόριθμος κάνει μια επαναληπτική διαδικασία στα βήματα 3 έως 6, οπότε γίνεται εκ νέου προσαρμογή του υποκατάστατου μοντέλου (surrogate model) στα νέα επιλεγμένα σημεία κ.ο.κ.

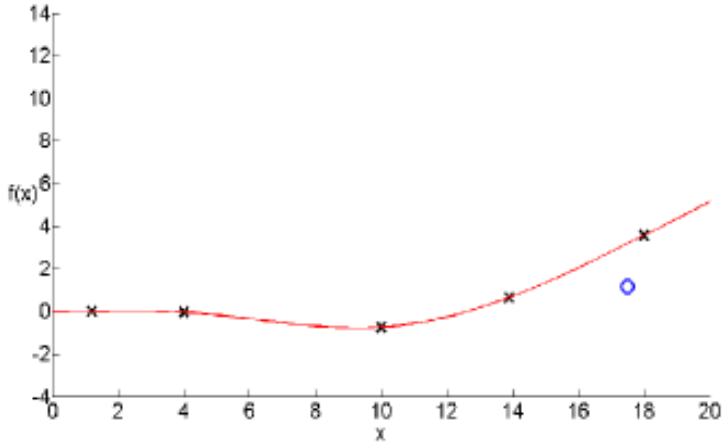
Στα επόμενα διαγράμματα φαίνεται σχηματικά η διαδικασία που ακολουθεί ο αλγόριθμος για μια τυχαία αντικειμενική συνάρτηση f(x) μίας μεταβλητής σχεδιασμού x (Σχ.1 ως 4).



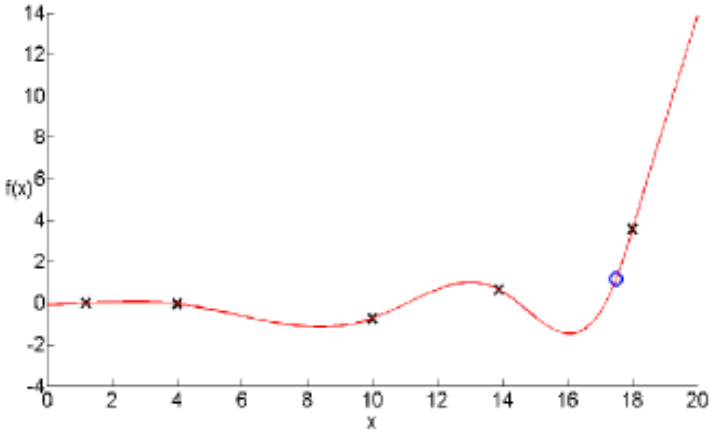
Σχ.1 Δημιουργία ενός αρχικού μοντέλου και υπολογισμός τιμών αρχικής αντικειμενικής συνάρτησης f(x)



Σχ.2 Προσαρμογή υποκατάστατου μοντέλου (surrogate model)(κόκκινη γραμμή)



Σχ.3 Επιλογή νέου σημείου από το πεδίο ορισμού(μπλε κύκλος) και υπολογισμός τιμής αρχικής αντικειμενικής συνάρτησης για τα νέο επιλεγμένο σημείο



Σχ.4 Προσαρμογή υποκατάστατου μοντέλου στο νέο σημείο (νέα κόκκινη γραμμή)

Η επιλογή των νέων σημείων από το πεδίο ορισμού, σημεία που θα πρέπει σε κάθε περίπτωση να ικανοποιούν τους σχεδιαστικούς περιορισμούς των μεταβλητών, είναι σημαντική διαδικασία καθόσον θα καθορίσει τελικά το τοπικό ή καθολικό βέλτιστο της αρχικής αντικειμενικής συνάρτησης και ως εκ τούτου τη βέλτιστη λύση του προβλήματος. Για το λόγο αυτό, αν ο αλγόριθμος MATSuMoTo αν ύστερα από κάποιες προκαθορισμένες επαναλήψεις, δεν καταφέρει να βρει νέο σημείο το οποίο θα δώσει τιμή αρχικής αντικειμενικής καλύτερη από πριν, τότε ξεκινάει πάλι από την αρχή (1^{ο} βήμα). Δηλαδή, δημιουργεί ένα νέο αρχικό μοντέλο μέσω νέας

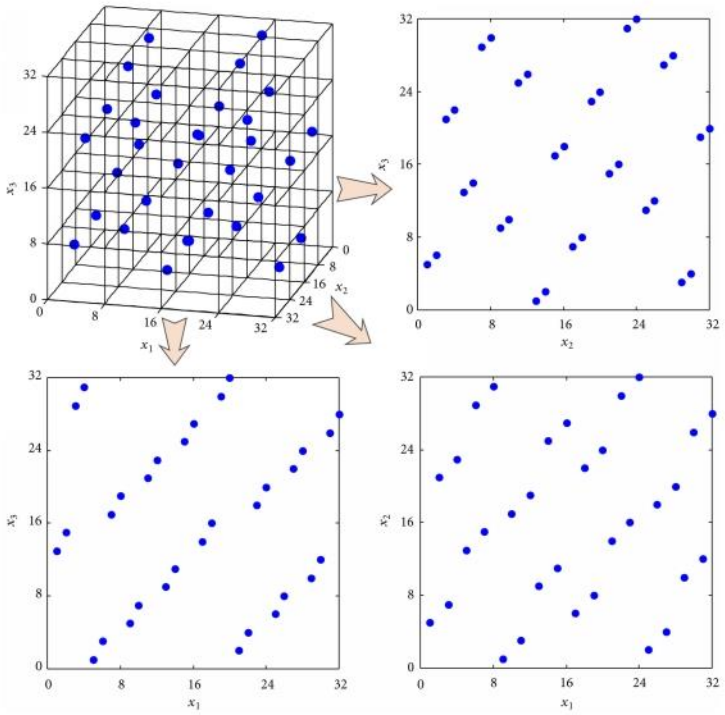
δειγματοληψίας και χωρίς μάλιστα να λαμβάνει υπόψη τα σημεία που ελήφθησαν στο προηγούμενο μοντέλο. Με αυτό τον τρόπο αποφεύγει να εγκλωβιστεί σε τοπικά βέλτιστα και να αναζητήσει καλύτερες λύσεις προκειμένου να βρει τελικά το καθολικό βέλτιστο.

1.5.3 Αλγόριθμος Latin Hypercube

Ο συγκεκριμένος αλγόριθμος είναι στην ουσία μία μέθοδος δειγματοληψίας, μια στατιστική μέθοδος για τη δημιουργία ενός (σχεδόν) τυχαίου δείγματος τιμών παραμέτρων από μια πολυδιάστατη κατανομή. Το Latin Hypercube Sampling (LHS) περιγράφηκε από τον Michael McKay του Εθνικού Εργαστηρίου του Los Alamos το 1979.

Στο πλαίσιο της στατιστικής δειγματοληψίας, ένα τετράγωνο πλέγμα που περιέχει θέσεις δείγματος είναι ένα Latin square, εάν (και μόνο εάν) υπάρχει μόνο ένα δείγμα σε κάθε σειρά και κάθε στήλη. Ένας Latin Hypercube είναι η γενίκευση αυτής της έννοιας σε έναν αυθαίρετο αριθμό διαστάσεων, όπου κάθε δείγμα είναι το μοναδικό σε κάθε άξονα – υπερ-επίπεδο (hyperplane) που το εμπεριέχει.

Κατά τη δειγματοληψία μιας συνάρτησης των N μεταβλητών, το εύρος κάθε μεταβλητής διαιρείται σε M ίδια διαστήματα. Στη συνέχεια τοποθετούνται M δείγματα έτσι ώστε να ικανοποιούνται οι απαιτήσεις του Latin Hypercube. Αυτό υποχρεώνει τον αριθμό των διαιρέσεων M να είναι ίδιος για κάθε μεταβλητή (διάσταση). Αυτή η μορφή δειγματοληψίας δεν απαιτεί περισσότερα δείγματα για περισσότερες διαστάσεις (μεταβλητές). Ένα πλεονέκτημα είναι ότι τα τυχαία δείγματα μπορούν να ληφθούν ένα κάθε φορά και παράλληλα να θυμόμαστε ποια δείγματα ελήφθησαν μέχρι τώρα. Το μόνο που απαιτείται είναι να αποφασίσουμε πόσα σημεία δείγματος να χρησιμοποιήσουμε και για κάθε σημείο δειγματοληψίας να θυμόμαστε σε ποια σειρά και στήλη το σημείο δειγματοληψίας λήφθηκε.

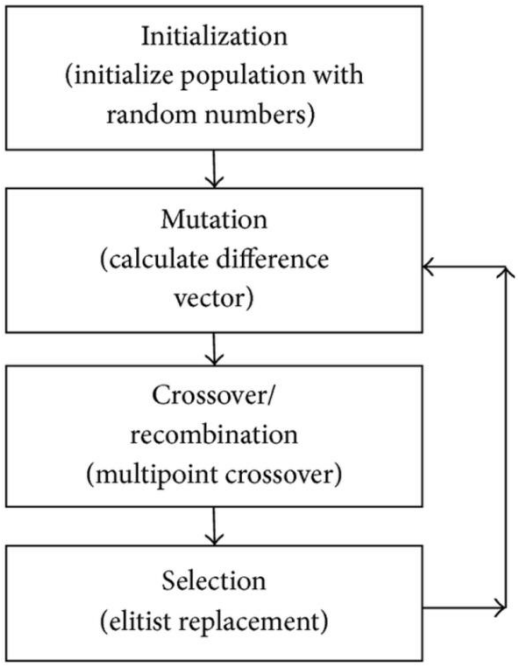


Παράδειγμα δειγματοληψίας 32 δειγμάτων σε συνάρτηση 3 μεταβλητών x1, x2 και x3

1.5.4 Αλγόριθμος Differential Evolution

Ο αλγόριθμος της Διαφορικής Εξέλιξης (Differential Evolution Algorithm) είναι ένας Εξελικτικός Αλγόριθμος που προτάθηκε από τους R.Storn και K.V.Price το 1995. Οι Εξελικτικοί Αλγόριθμοι (Evolutionary Algorithms – EA) είναι μετα-ευρεστικές μέθοδοι βελτιστοποίησης με βάση τον πληθυσμό (population), που έχουν προέλευση και έμπνευση από τον κόσμο της βιολογίας. Οι Εξελικτικοί Αλγόριθμοι ενσωματώνουν τις φυσικές διαδικασίες της επιλογής (selection), της μετάλλαξης (mutation) και της διασταύρωσης (crossover) ή αλλιώς του

ανασυνδυασμού (recombination) και τις χρησιμοποιούν ως μηχανισμούς ή τελεστές αναζήτησης, ώστε να βρουν καλύτερες λύσεις γρηγορότερα σε προβλήματα βελτιστοποίησης.



Τα βασικά στάδια του αλγορίθμου Διαφορικής Εξέλιξης

Η Διαφορική Εξέλιξη είναι μια αποτελεσματική στοχαστική μέθοδος που βασίζεται στον «πληθυσμό» για καθολική βελτιστοποίηση σε συνεχείς χώρους αναζήτησης. Η Διαφορική Εξέλιξη δεν απαιτεί το πρόβλημα βελτιστοποίησης να είναι διαφορίσιμο, όπως απαιτείται από τις κλασικές μεθόδους βελτιστοποίησης. Ως Εξελικτικός Αλγόριθμος που βασίζεται στον πληθυσμό, η Διαφορική Εξέλιξη επιλύει προβλήματα βελτιστοποίησης εξελίσσοντας τον πληθυσμό των υποψήφιων λύσεων χρησιμοποιώντας τους τελεστές της μετάλλαξης, της διασταύρωσης και της επιλογής.

Η πρωταρχική ιδέα της Διαφορικής Εξέλιξης είναι να προσαρμόσει την αναζήτηση κατά τη διάρκεια της εξελικτικής διαδικασίας. Κατά την αρχική φάση της εξέλιξης, οι διακυμάνσεις είναι μεγάλες, καθόσον οι υποψήφιες λύσεις είναι πολύ μακριά η μία από την άλλη. Όσο η εξελικτική διαδικασία προχωράει, ο πληθυσμός συγκλίνει σε μια μικρή περιοχή και οι διακυμάνσεις γίνονται μικρότερες. Πρακτικά δημιουργεί έναν αρχικό πληθυσμό υποψήφιων λύσεων και στη συνέχεια με χρήση απλών μαθηματικών τύπων συνδυάζονται οι θέσεις των υπαρχόντων λύσεων του πληθυσμού. Δημιουργούνται, έτσι, συνεχώς νέες υποψήφιες λύσεις, κρατώντας τελικά εκείνες που έχουν την καλύτερη «επίδοση». Η διαδικασία επαναλαμβάνεται μέχρι να βρεθεί τελικά, αλλά όχι εγγυημένα, η βέλτιστη λύση.

1.6 Ανάλυση αποτελεσμάτων βελτιστοποίησης μοντέλων

Στη συνέχεια αναλύονται διεξοδικά τα αποτελέσματα της βελτιστοποίησης των έξι μοντέλων της παραγράφου 4.2 που χρησιμοποιήθηκαν στην παρούσα εργασία, μετά την εφαρμογή των τεσσάρων αλγορίθμων βελτιστοποίησης που αναφέρθηκαν νωρίτερα. Πιο συγκεκριμένα, για κάθε μοντέλο και αλγόριθμο βελτιστοποίησης που εφαρμόζεται, παρουσιάζεται η πορεία του αλγορίθμου όσον αφορά:

a. Στο ποσοστό % βελτίωσης της τιμής της αντικειμενικής συνάρτησης σε σχέση με την τιμή σχεδιασμού.

b. Στην τιμή της παραβίασης (violation) κάθε επανάληψης (ανάλυσης του μοντέλου).

c. Στην τιμή της αντικειμενικής συνάρτησης.

d. Στον αριθμό των «εφικτών» λύσεων, δηλ. των επαναλήψεων εκείνων όπου δεν υπάρχει παραβίαση και δίνουν αποδεκτή λύση.

e. Στην ταχύτητα εμφάνισης εφικτών λύσεων και ιδιαίτερα της πρώτης εφικτής λύσης και της βέλτιστης λύσης.

Για την καλύτερη κατανόηση των παραπάνω, παρουσιάζονται διαγράμματα στα οποία απεικονίζεται η πορεία του κάθε αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του κάθε μοντέλου. Στον οριζόντιο άξονα εμφανίζεται πάντα ο αριθμός της επανάληψης ενώ στον κατακόρυφο άξονα εμφανίζονται:

a. **Το ποσοστό βελτίωσης (%)** της τιμής της αντικειμενικής συνάρτησης της κάθε επανάληψης σε σχέση με την τιμή σχεδιασμού, δηλαδή την τιμή της αντικειμενικής που λαμβάνει το μοντέλο με βάση τις αρχικές διατομές σχεδιασμού.

b. **Η τιμή της παραβίασης** της κάθε επανάληψης. Με τον όρο «παραβίαση» εννοείται η υπέρβαση των ορίων σχεδιασμού – περιορισμών που έχουν τεθεί (design violation) και η οποία πρακτικά μας δείχνει την «απόσταση» που έχουν οι επιλεγμένες τιμές των μεταβλητών σχεδιασμού (δηλαδή οι διαστάσεις των διατομών) από τους σχεδιαστικούς περιορισμούς.

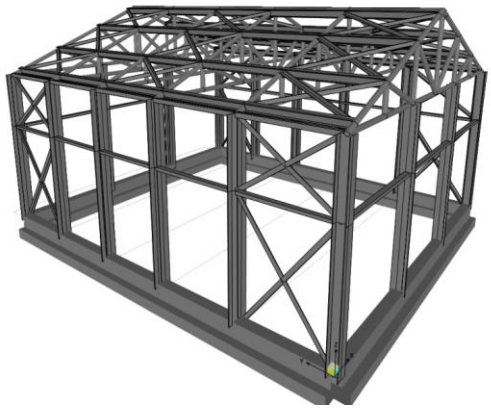
Μία μεγάλη τιμή παραβίασης σημαίνει ότι, οι διατομές που έχουν επιλέγει είναι ενδεχομένως πολύ μικρές για να ικανοποιήσουν τα σχεδιαστικά κριτήρια, αν και δίνουν μικρή τιμή αντικειμενικής δηλ. μικρό βάρος κατασκευής. Σε αντίθεση, πολύ μικρή παραβίαση σημαίνει ότι έχουμε ενδεχομένως μία άκαμπτη κατασκευή με μεγάλες διατομές, η οποία όμως επιφέρει μεγάλο βάρος και άρα μεγάλη τιμή αντικειμενικής, κάτι που είναι αντίθετο με το ζητούμενο. Για το λόγο αυτό, η βέλτιστη λύση θα κινείται πάντα στο όριο της τιμής της παραβίασης δηλ. κοντά στο 1,0.

c. **Η τιμή της αντικειμενικής συνάρτησης** αφενός κάθε επανάληψης και αφετέρου των επαναλήψεων εκείνων που δίνουν εφικτές λύσεις δηλ. που έχουν παραβίαση μικρότερη ή ίση με το 1,0.

Υπενθυμίζεται ότι, τα μοντέλα έχουν σχεδιαστεί και αναλύονται σε κάθε επανάληψη στο πρόγραμμα SAP2000 v.20 της CSI, ενώ για τη βελτιστοποίηση χρησιμοποιείται η πλατφόρμα Optimization Computing Platform OCP. Επιπλέον, σε όλες τις περιπτώσεις που μελετήθηκαν, η αντικειμενική συνάρτηση (objective function) είναι η ελαχιστοποίηση του βάρους της κατασκευής και οι περιορισμοί που έχουν τεθεί είναι οι σχεδιαστικοί (Design Violations) δηλ. οι κατασκευαστικοί και σεισμικοί κώδικες (EC-2, EC-3, EC-8, κτλ.) οι οποίοι εμπεριέχονται στο SAP2000.

1.7 Ανάλυση 1ου Μοντέλου

Το 1ο μοντέλο, όπως προαναφέρθηκε, αφορά σε έναν αχυρώνα κατασκευασμένο από χαλύβδινα στοιχεία διαφόρων διατομών και θεμελίωση από οπλισμένο σκυρόδεμα. Αποτελείται συνολικά από 337 στοιχεία και 154 κόμβους, το ύψος μέχρι το στέγαστρο είναι 4,85m και το συνολικό ύψος 6,85m. Τέλος έχει διαστάσεις σε κάτοψη 11,25 x 9,80 m. και εμβαδό 110,25 m^2. Οι μεταλλικές διατομές είναι από χάλυβα S275 και η θεμελίωση από σκυρόδεμα C30/37.



Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι οι ακόλουθες:

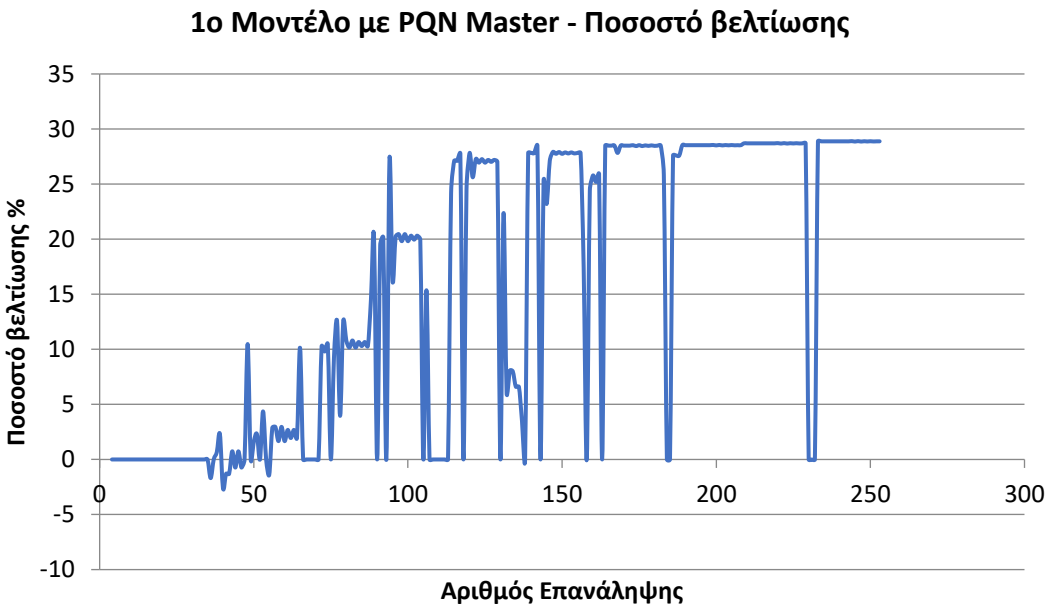
- Υποστυλώματα HEA280
- Θεμελίωση διατομής T h=700mm, w=700mm, t_f=300mm, t_w=300mm
- Κύριες Δοκοί HEA100
- Μηκίδες 125x75x6RHS
- Χιαστί σύνδεσμοι οροφής L90x12
- Χιαστί πλευρικοί σύνδεσμοι L100x10
- Τεγίδες IPE200
- Στοιχεία δικτυωμάτων οροφής 114.3x6CHS

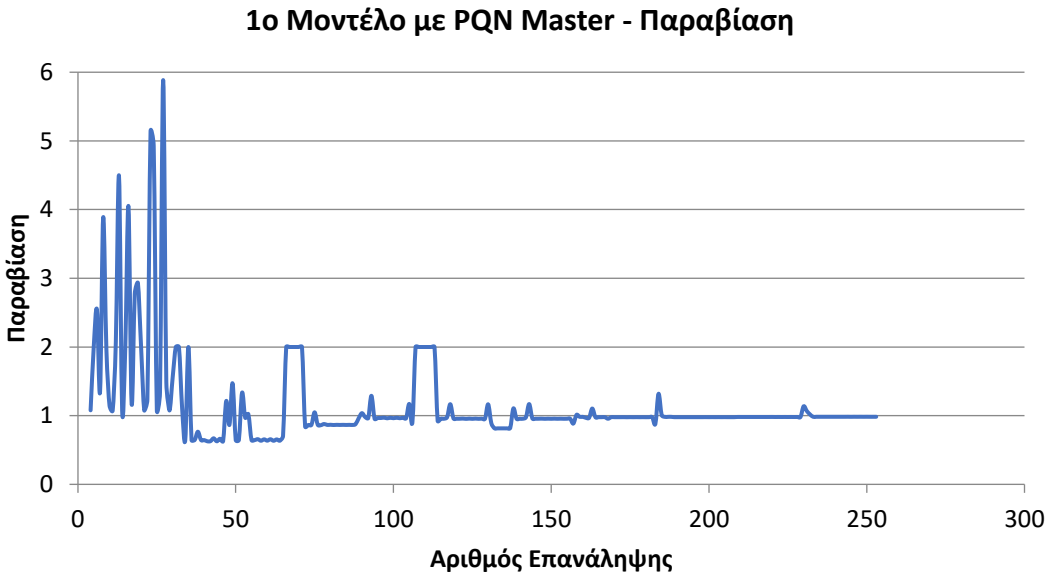
1.7.1 Εφαρμογή του αλγορίθμου PQN Master

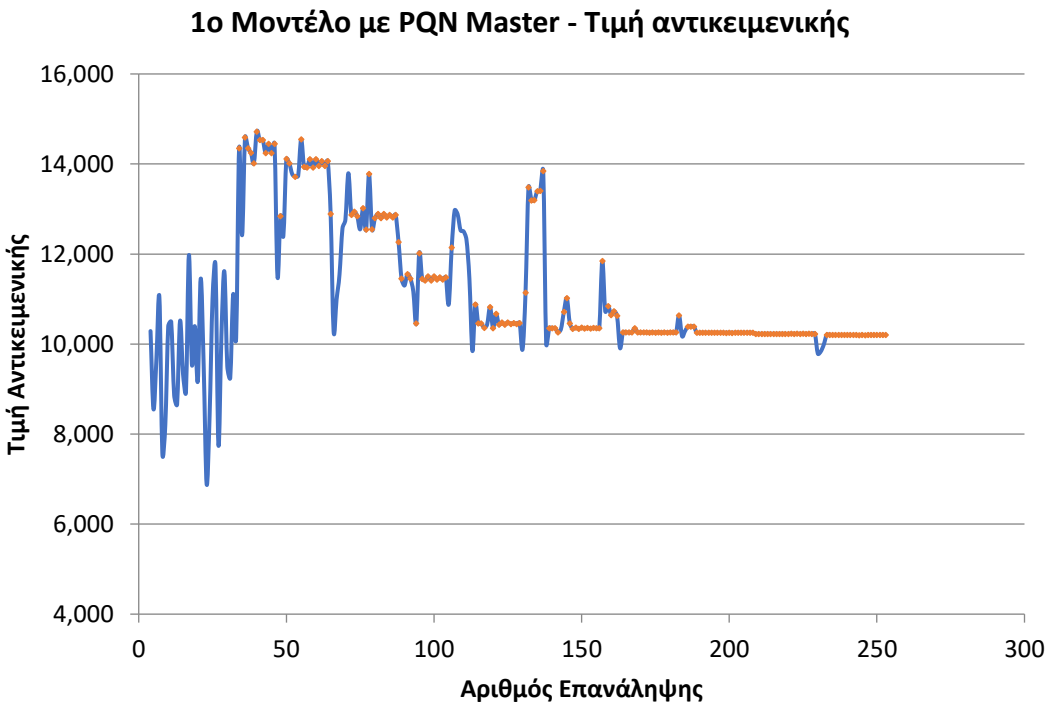
Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού, δηλ. η τιμή της αντικειμενικής συνάρτησης του αρχικού μοντέλου πριν γίνει βελτιστοποίηση, είναι 14.347. Συνολικά γίνονται 253 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 244η επανάληψη με τιμή 10.201 (βελτίωση σε ποσοστό 28,90% επί της τιμής σχεδιασμού).

Οι αντικειμενικές τιμές στις πρώτες επαναλήψεις εμφανίζουν μεγάλη διακύμανση γύρω από την τιμή σχεδιασμού, μέχρι την επανάληψη εκείνη που δίνει την πρώτη εφικτή λύση. Από εκεί και μετά, ο αλγόριθμος αναζητάει «καλύτερες» εφικτές λύσεις σε σημεία κοντά στο υπάρχον τοπικό βέλτιστο, μειώνοντας έτσι συνεχώς την τιμή της αντικειμενικής συνάρτησης μέχρι να βρει την βέλτιστη τιμή και φυσικά ωσότου σταματήσει να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Για το λόγο αυτό, αντίστοιχα και οι παραβιάσεις αρχικά είναι μεγάλες αλλά μόλις εντοπίσει την πρώτη εφικτή λύση μειώνονται σημαντικά και κινούνται γύρω από την τιμή 1,0.

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.


1ο Μοντέλο με PQN Master - Ποσοστό βελτίωσης
Ποσοστό βελτίωσης %
35
30
25
20
15
10
5
0
-5
-10
0
50
100
150
200
250
300
Αριθμός Επανάληψης


1ο Μοντέλο με PQN Master - Παραβίαση
Παραβίαση
6
5
4
3
2
1
0
0
50
100
150
200
250
300
Αριθμός Επανάληψης



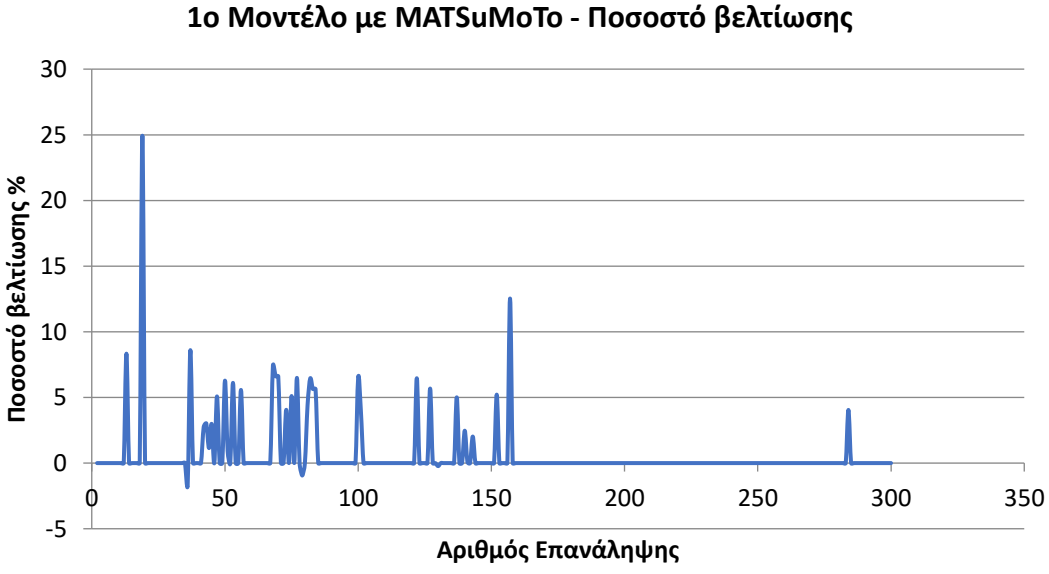
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

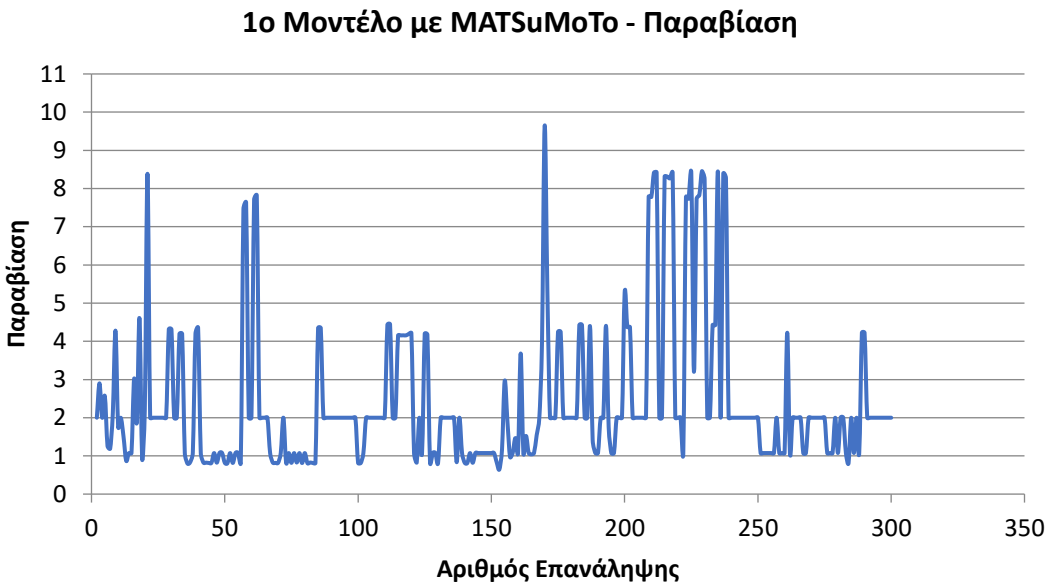
1.7.2 Εφαρμογή του αλγορίθμου MATSuMoTo

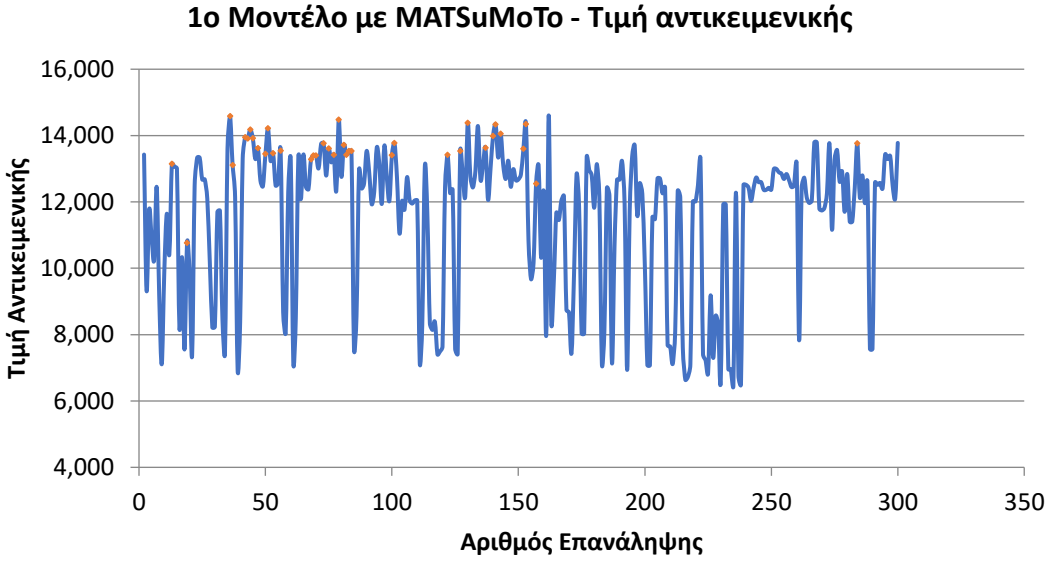
Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 14.347. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 19η επανάληψη με τιμή 10.771 (βελτίωση σε ποσοστό 24,93% επί της τιμής σχεδιασμού).

Σε όλες τις επαναλήψεις υπάρχει μεγάλη διακύμανση τιμών, τόσο της αντικειμενικής συνάρτησης όσο και των παραβιάσεων. Ο αλγόριθμος αναζητάει συνεχώς καλύτερες λύσεις σε όλο το πεδίο σχεδιασμού και έτσι δεν εγκλωβίζεται σε κάποιο ενδεχομένως τοπικό βέλτιστο, ενώ εντοπίζει πολύ γρήγορα τη βέλτιστη τιμή.

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.







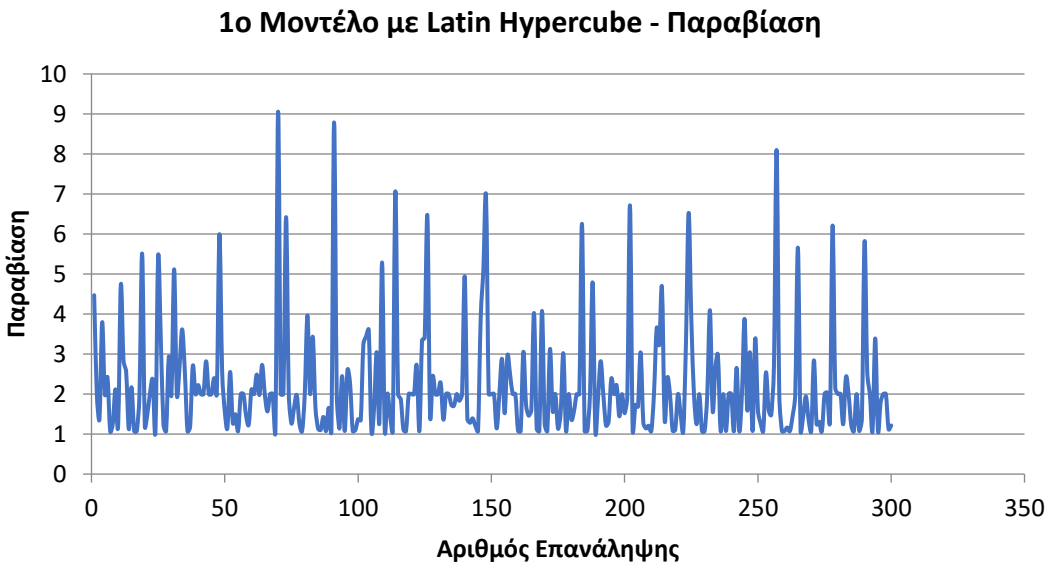
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

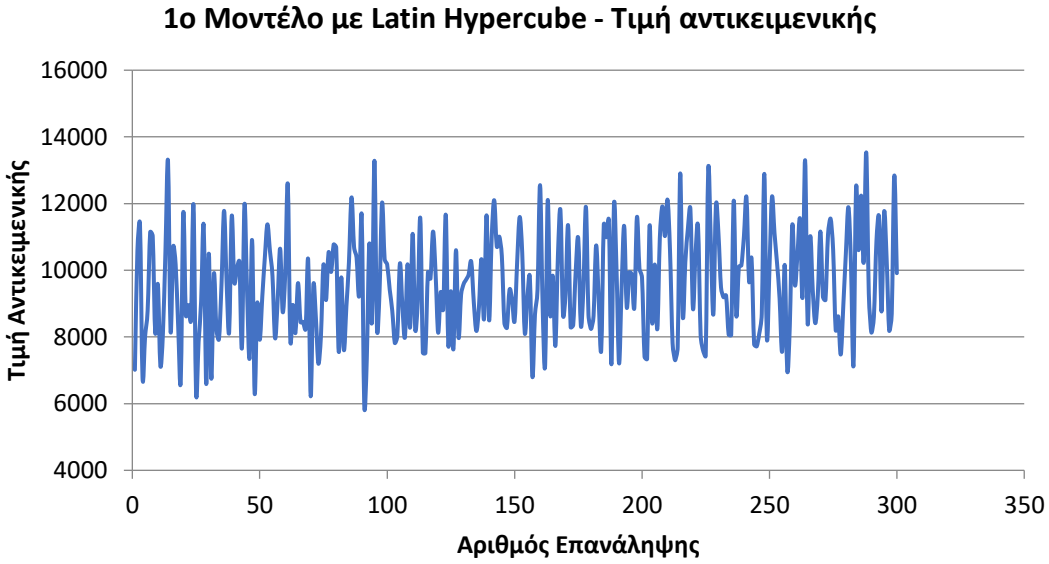
1.7.3 Εφαρμογή του αλγορίθμου Latin Hypercube

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 7.015 εμφανίζοντας αρκετά μεγάλη παραβίαση 4,471. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί.

Γενικά ο αλγόριθμος καθ' όλη τη διάρκεια της εφαρμογής του δεν καταφέρνει να βρει καμία εφικτή λύση. Οι τιμές της αντικειμενικής έχουν μεγάλη διακύμανση σε όλο το εύρος των επαναλήψεων με μέγιστη τιμή στην 288η επανάληψη με τιμή 13.536 (παραβίαση 1,075) και ελάχιστη στην 91η επανάληψη με τιμή 5.960 (παραβίαση 8,789).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

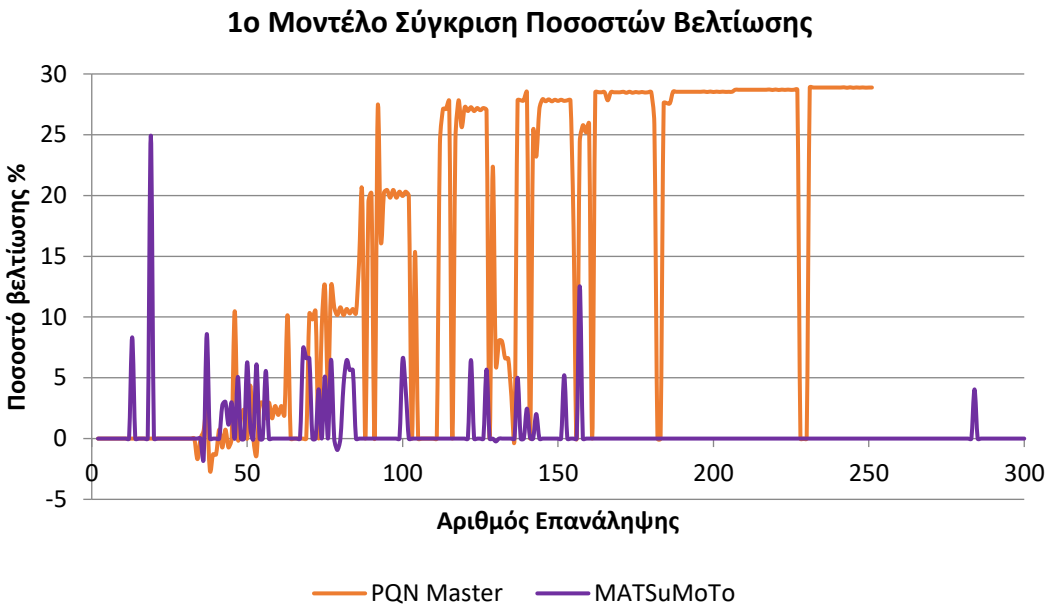
1.7.4 Συγκριτική ανάλυση αποτελεσμάτων

Κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου PQN Master, οι αντικειμενικές τιμές στις πρώτες επαναλήψεις εμφανίζουν μεγάλη διακύμανση γύρω από την τιμή σχεδιασμού, μέχρι την επανάληψη εκείνη που δίνει την πρώτη εφικτή λύση. Από εκεί και μετά, ο αλγόριθμος αναζητάει «καλύτερες» εφικτές λύσεις σε σημεία κοντά στο υπάρχον τοπικό βέλτιστο.

Κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου MATSuMoTo σε όλες τις επαναλήψεις υπάρχει μεγάλη διακύμανση τιμών, τόσο της αντικειμενικής συνάρτησης όσο και των παραβιάσεων. Εντοπίζει πολύ πιο γρήγορα τη βέλτιστη τιμή η οποία όμως αποδεικνύεται τελικά «χειρότερη» από την βέλτιστη τιμή που δίνει ο PQN Master.

Κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου Latin Hypercube, δεν καταφέρνει να βρει κάποια αποδεκτή λύση.

Προς επίρρωση των παραπάνω, στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου. Στο διάγραμμα που απεικονίζεται το ποσοστό βελτίωσης της αντικειμενικής συνάρτησης, δεν έχει σκόπιμα συμπεριληφθεί ο αλγόριθμος Latin Hypercube, καθόσον ο συγκεκριμένος αλγόριθμος δεν καταφέρνει να βρει κάποια αποδεκτή λύση.

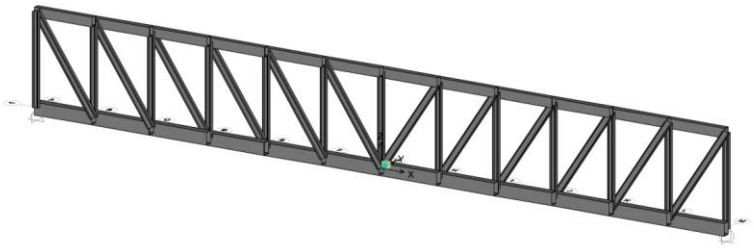


Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	14.769	14.608	13.536
Ελάχιστη τιμή	6.442	6.460	5.960
Τιμή σχεδιασμού	14.347 (εφικτή)	14.347 (εφικτή)	-
Αριθμός Επαναλήψεων	253	300	300
Πρώτη εφικτή λύση	(34η) 14.347 0%	(13η) 13.152 8,33%	-
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	(38η) 14.252 0,66%	(13η) 13.152 8,33%	-
Βέλτιστη τιμή	(244η) 10.201	(19η) 10.771	-
Ποσοστό βελτίωσης %	**28,90%**	**24,93%**	-
Πορεία αλγορίθμου	(34η) 14.347 0% (38η) 14.252 0,66% (48η) 12.844 10,48% (77η) 12.542 12,58% (89η) 11.458 20,14% (94η) 10.458 27,11% (244η) 10.201 28,90%	(13η) 13.152 8,33% (19η) 10.771 24,93%	Δεν βρίσκει εφικτές λύσεις

1.8 Ανάλυση 2ου Μοντέλου

Το 2ο μοντέλο, όπως προαναφέρθηκε, αφορά σε μία 2D δικτυωτή γέφυρα με μεταλλικά στοιχεία διαφόρων διατομών. Αποτελείται συνολικά από 49 στοιχεία και 26 κόμβους, το ύψος είναι 6m και το συνολικό μήκος 48m. Όλες οι μεταλλικές διατομές είναι από χάλυβα S355.



Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι οι ακόλουθες:

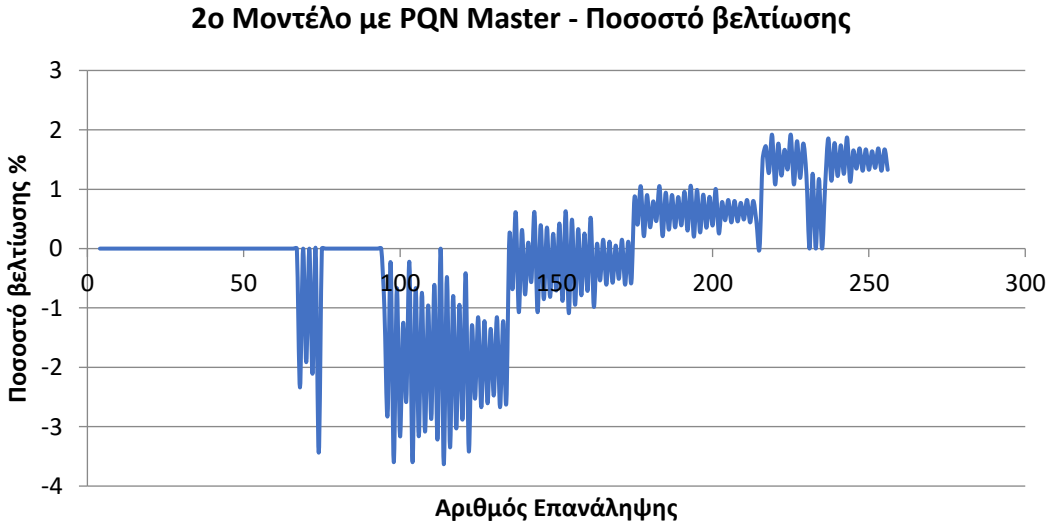
- Άνω πέλμα κοίλης ορθογωνικής διατομής με h=600mm, w=400mm, t_f=18mm, t_w=18mm
- Κάτω πέλμα κοίλης ορθογωνικής διατομής με h=800mm, w=500mm, t_f=12mm, t_w=12mm
- Ορθοστάτες HEM300
- Διαγώνιοι HEA400

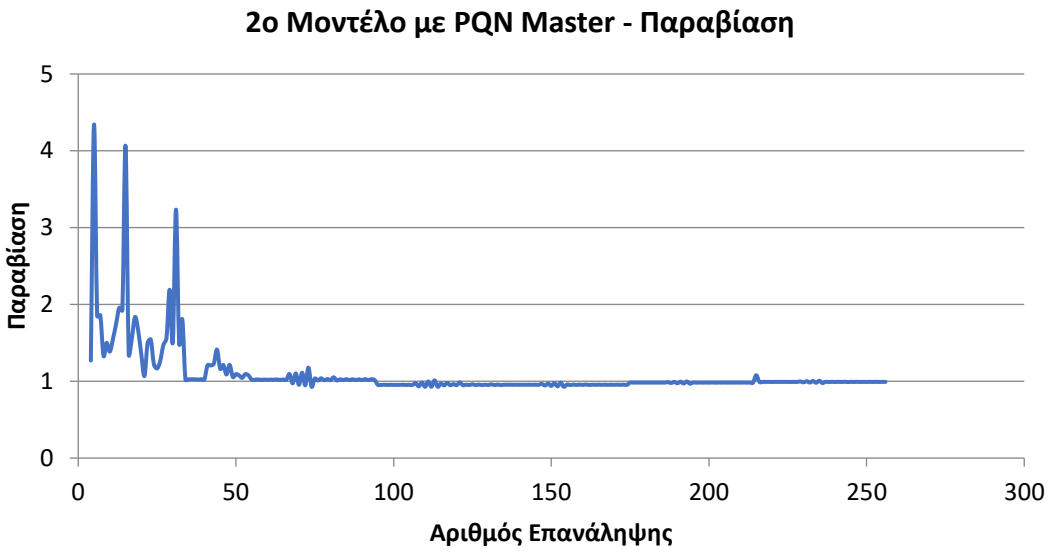
1.8.1 Εφαρμογή του αλγορίθμου PQN Master

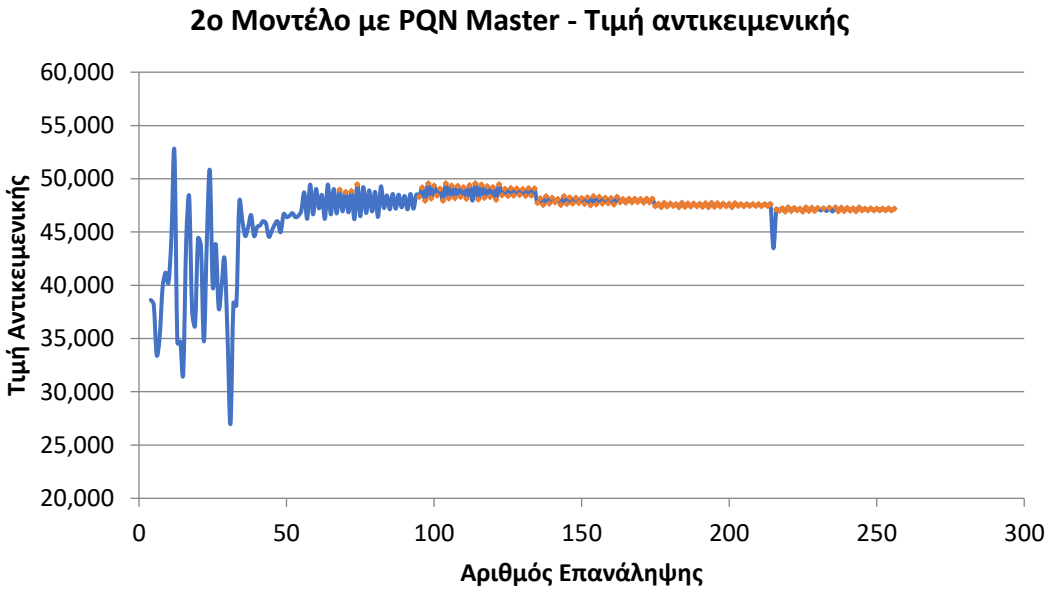
Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού, δηλ. η τιμή της αντικειμενικής συνάρτησης του αρχικού μοντέλου πριν γίνει βελτιστοποίηση, είναι 47.847. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν μικρή παραβίαση 1,02 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 256 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 225η επανάληψη με τιμή 46.929 (μικρή βελτίωση σε ποσοστό 1,92% επί της τιμής σχεδιασμού).

Όπως και στο προηγούμενο μοντέλο, κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου PQN Master, οι αντικειμενικές τιμές στις πρώτες επαναλήψεις εμφανίζουν μεγάλη διακύμανση γύρω από την τιμή σχεδιασμού, μέχρι την επανάληψη εκείνη που δίνει την πρώτη εφικτή λύση. Από εκεί και μετά, ο αλγόριθμος αναζητάει «καλύτερες» εφικτές λύσεις σε σημεία κοντά στο υπάρχον τοπικό βέλτιστο, μειώνοντας έτσι συνεχώς την τιμή της αντικειμενικής συνάρτησης μέχρι να βρει την βέλτιστη τιμή.

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.







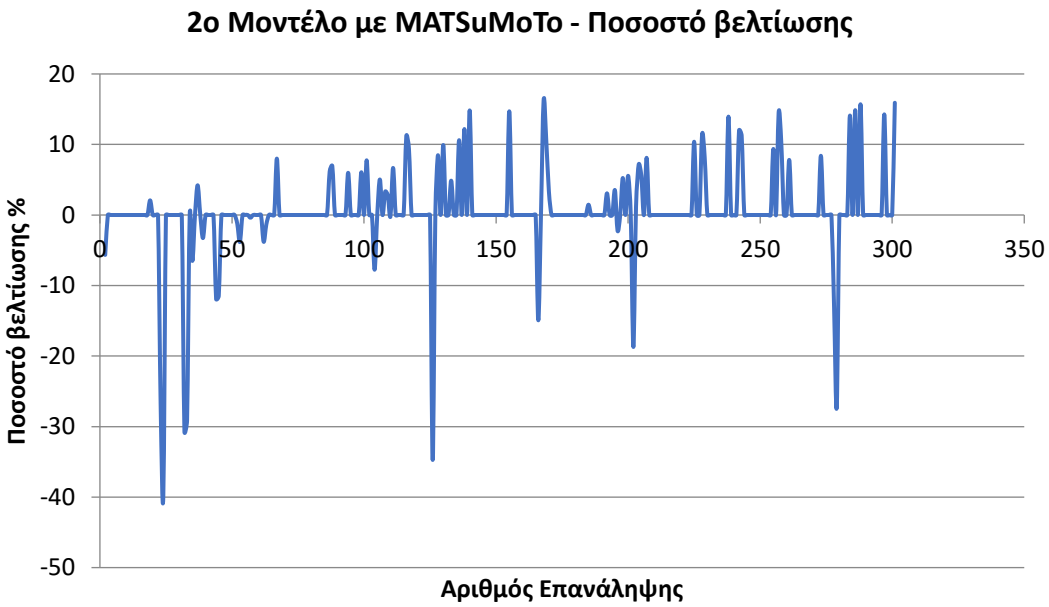
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

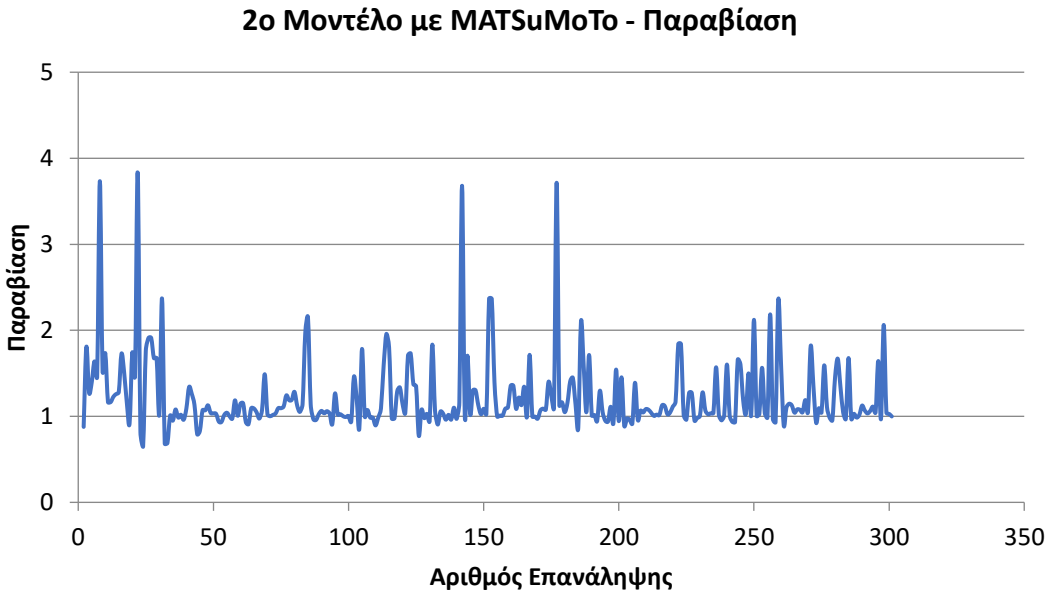
1.8.2 Εφαρμογή του αλγορίθμου MATSuMoTo

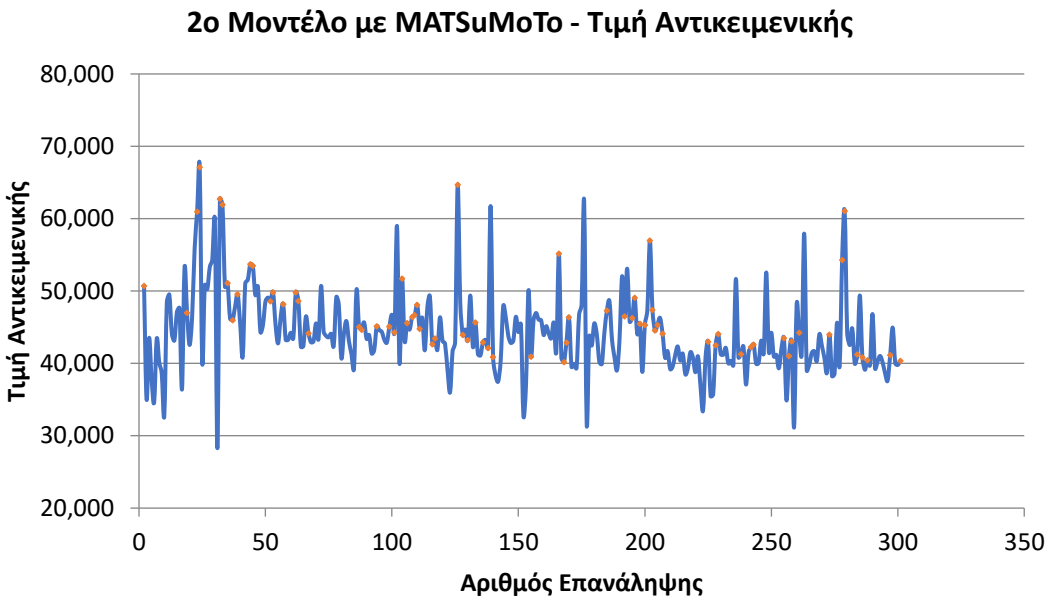
Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 48.001. Συνολικά γίνονται 301 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 168^{η} επανάληψη με τιμή 40.202 (βελτίωση σε ποσοστό 16,25% επί της τιμής σχεδιασμού).

Αντίστοιχα με το προηγούμενο μοντέλο και σε αυτό, κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου MATSuMoTo, διαπιστώνεται ότι σε όλες τις επαναλήψεις υπάρχει μεγάλη διακύμανση τιμών, τόσο της αντικειμενικής συνάρτησης όσο και των παραβιάσεων.

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.


2ο Μοντέλο με MATSuMoTo - Ποσοστό βελτίωσης
Ποσοστό βελτίωσης %
20
10
0
-10
-20
-30
-40
-50
0
50
100
150
200
250
300
350
Αριθμός Επανάληψης


2ο Μοντέλο με MATSuMoTo - Παραβίαση
Παραβίαση
5
4
3
2
1
0
0
50
100
150
200
250
300
350
Αριθμός Επανάληψης



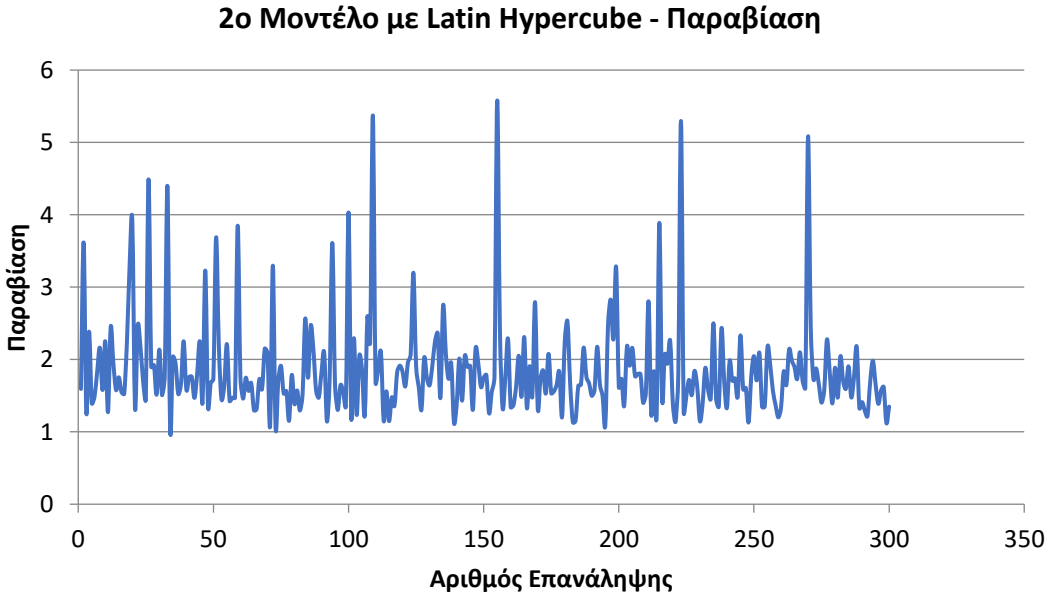
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

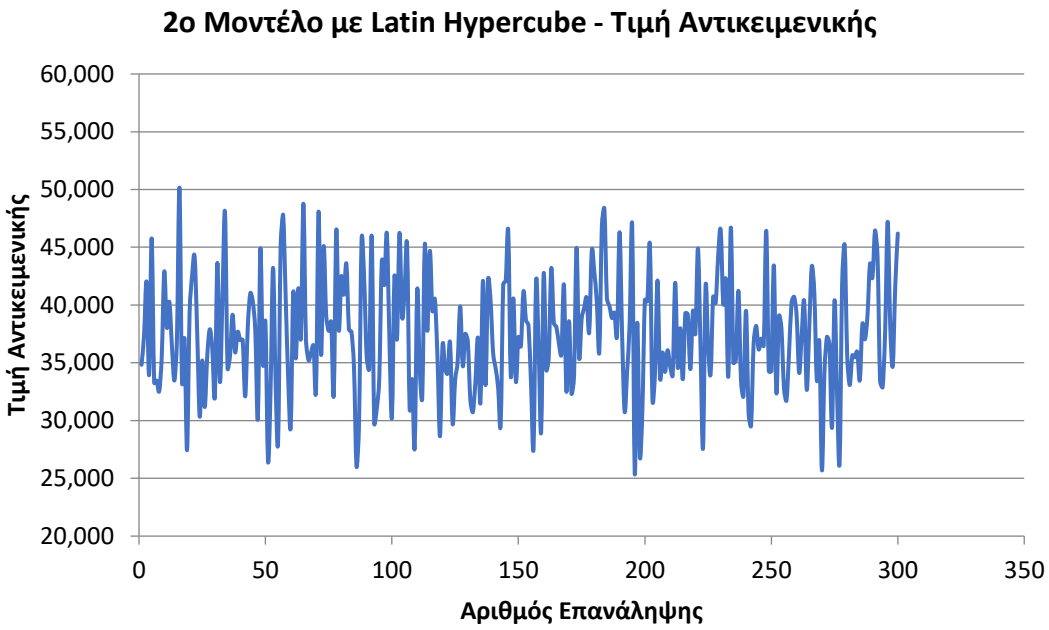
1.8.3 Εφαρμογή του αλγορίθμου Latin Hypercube

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 34.826 εμφανίζοντας μικρή παραβίαση 1,593. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί.

Γενικά ο αλγόριθμος καθ' όλη τη διάρκεια της εφαρμογής του δεν καταφέρνει να βρει καμία εφικτή λύση. Οι τιμές της αντικειμενικής έχουν μεγάλη διακύμανση σε όλο το εύρος των επαναλήψεων με μέγιστη τιμή στην 16η επανάληψη με τιμή 50.142 (παραβίαση 1,549) και ελάχιστη στην 196η επανάληψη με τιμή 25.428 (παραβίαση 2,468).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





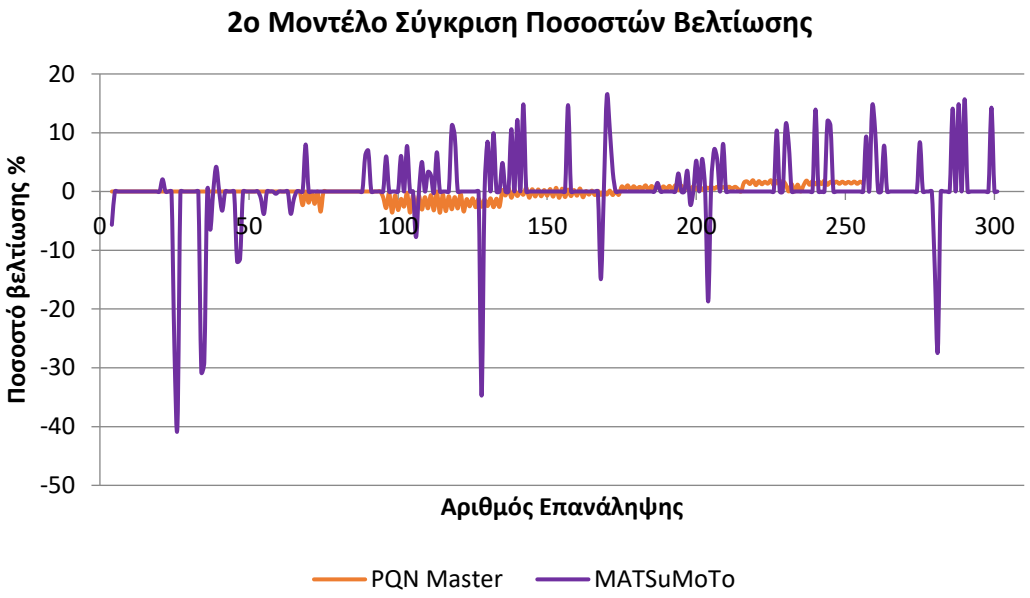
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.8.4 Συγκριτική ανάλυση αποτελεσμάτων

Όπως και στο προηγούμενο μοντέλο, κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου PQN Master, οι αντικειμενικές τιμές στις πρώτες επαναλήψεις εμφανίζουν μεγάλη διακύμανση γύρω από την τιμή σχεδιασμού, μέχρι την επανάληψη εκείνη που δίνει την πρώτη εφικτή λύση. Αντίστοιχα, κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου MATSuMoTo, διαπιστώνεται ότι σε όλες τις επαναλήψεις υπάρχει μεγάλη διακύμανση τιμών, τόσο της αντικειμενικής συνάρτησης όσο και των παραβιάσεων. Σε αυτήν την περίπτωση όμως η βέλτιστη τιμή αποδεικνύεται πολύ «καλύτερη» από τη βέλτιστη τιμή που δίνει ο PQN Master, ο οποίος εδώ εγκλωβίστηκε σε κάποιο τοπικό βέλτιστο, σε αντίθεση με τον αλγόριθμο MATSuMoTo ο οποίος αναζητώντας συνεχώς καλύτερες λύσεις σε όλο το πεδίο σχεδιασμού τελικά εντόπισε πολύ καλύτερη λύση.

Κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου Latin Hypercube, διαπιστώνεται και πάλι ότι δεν καταφέρνει να βρει κάποια αποδεκτή λύση.

Προς επίρρωση των παραπάνω, στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου.

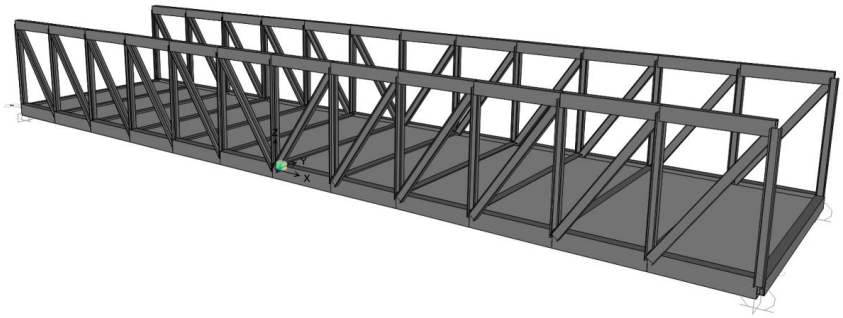


Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	90.477	67.149	50.142
Ελάχιστη τιμή	12.786	28.273	25.428
Τιμή σχεδιασμού	47.847 (μη εφικτή)	48.001 (εφικτή)	-
Αριθμός Επαναλήψεων	256	301	300
Πρώτη εφικτή λύση	(68η) 48.966 -2,34%	(19η) 47.001 2,08%	-
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	(135η) 47.740 0,22%	(19η) 47.001 2,08%	-
Βέλτιστη τιμή	(225η) 46.929	(168η) 40.202	-
Ποσοστό βελτίωσης %	**1,92%**	**16,25%**	**-**
Πορεία αλγορίθμου	(135η) 47.740 0,22% (153η) 47.546 0,63% (175η) 47.437 0,86% (193η) 47.340 1,06% (216η) 47.130 1,50% (225η) 46.929 1,92%	(19η) 47.001 2,08% (37η) 45.976 4,22% (67η) 44.160 8,00% (116η) 42.651 11,15% (168η) 40.202 16,25%	Δεν βρίσκει εφικτές λύσεις

1.9 Ανάλυση 3ου Μοντέλου

Το 3ο μοντέλο, όπως προαναφέρθηκε, αφορά σε μία 3D γέφυρα κάτω διάβασης, με χαλύβδινο κατάστρωμα και χαλύβδινες δικτυωτές κύριες δοκούς. Αποτελείται συνολικά από 111 στοιχεία, 12 κελύφη και 52 κόμβους, το ύψος είναι 6m, το συνολικό μήκος 48m και το άνοιγμα του καταστρώματος 12m. Οι μεταλλικές διατομές των στοιχείων είναι από χάλυβα S355.



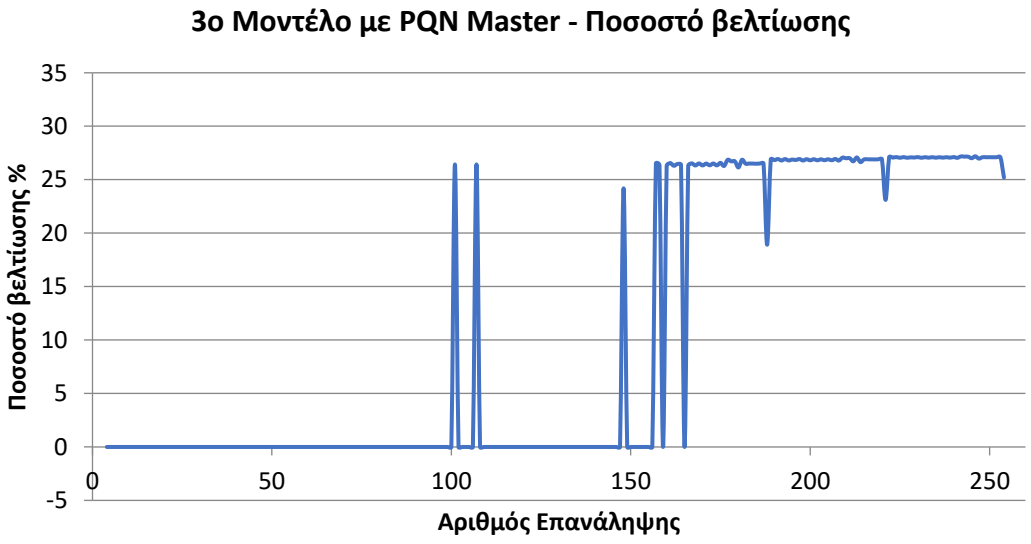
Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι οι ακόλουθες:

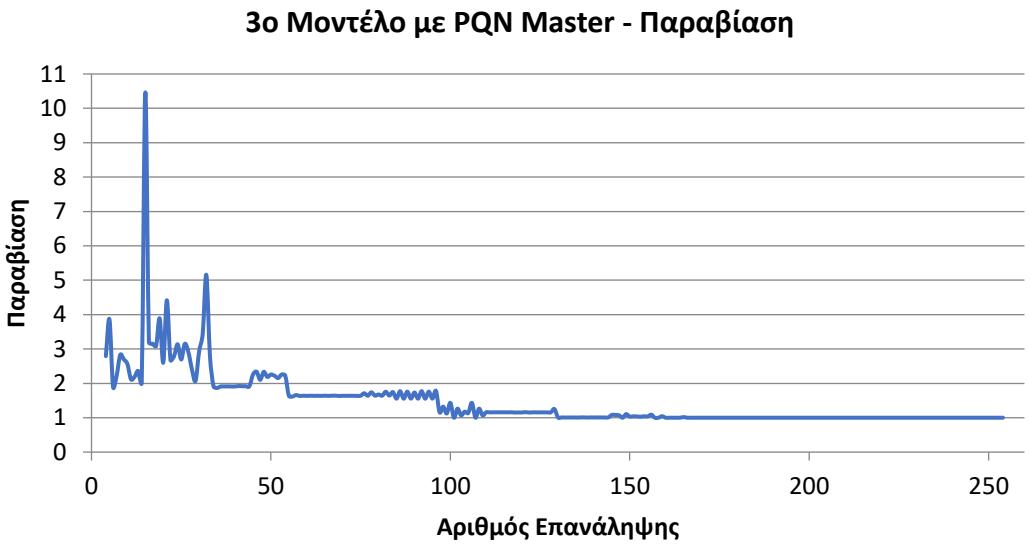
- Άνω πέλμα κύριων δοκών κοίλης ορθογωνικής διατομής με h=600mm, w=400mm, t_f=18mm, t_w=18mm
- Κάτω πέλμα κύριων δοκών κοίλης ορθογωνικής διατομής με h=800mm, w=500mm, t_f=12mm, t_w=12mm
- Ορθοστάτες κύριων δοκών HEM300
- Διαγώνιοι κύριων δοκών HEA400
- Διαδοκίδες κοίλης ορθογωνικής διατομής με h=600mm, w=400mm, t_f=18mm, t_w=18mm
- Κατάστρωμα συμπαγούς διατομής πάχους 250mm.

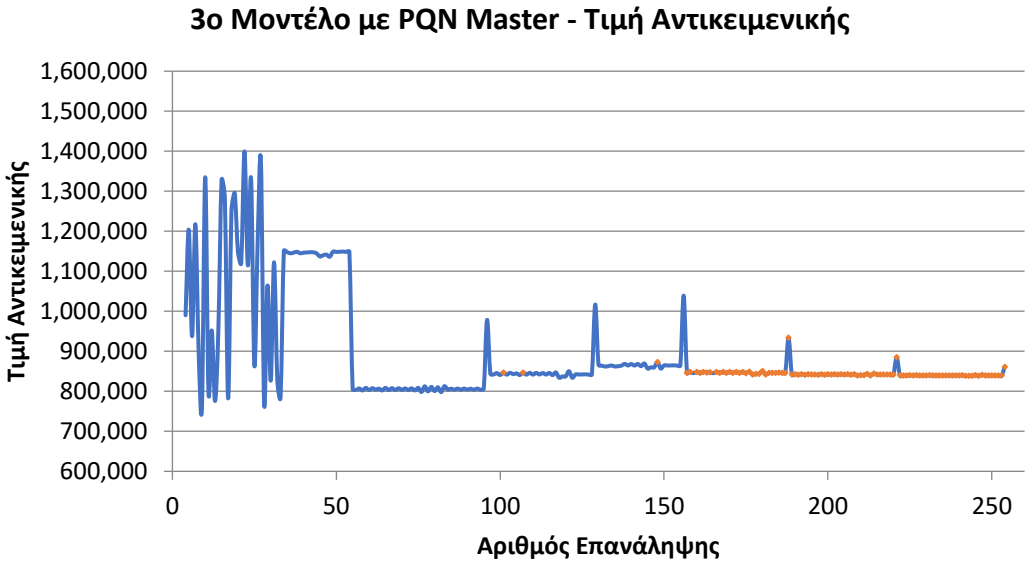
1.9.1 Εφαρμογή του αλγορίθμου PQN Master

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού, δηλ. η τιμή της αντικειμενικής συνάρτησης του αρχικού μοντέλου πριν γίνει βελτιστοποίηση, είναι 1.151.303. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν παραβίαση 1,914 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 254 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 246η επανάληψη με τιμή 838.242 (βελτίωση σε ποσοστό 27,19% επί της τιμής σχεδιασμού).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





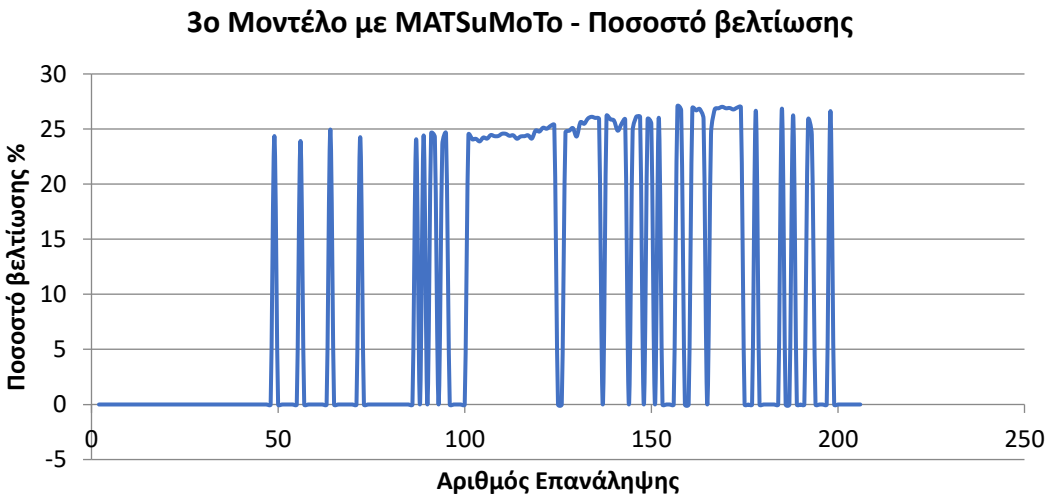


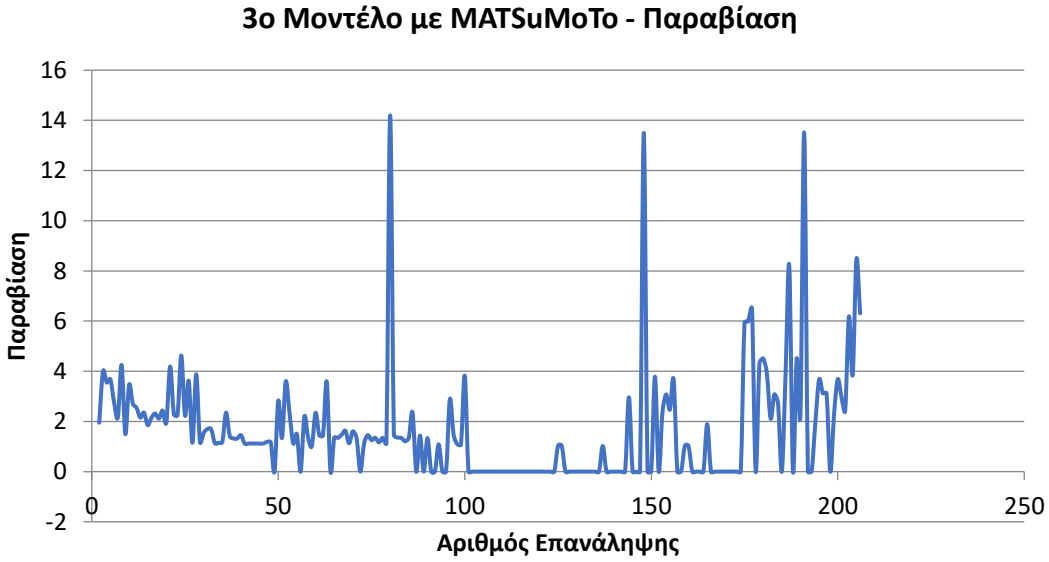
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

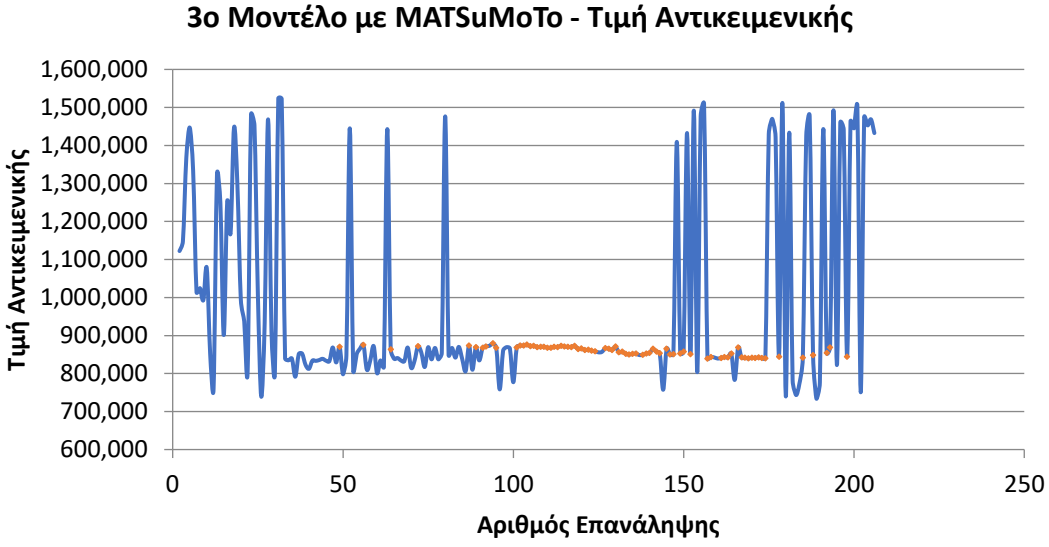
1.9.2 Εφαρμογή του αλγορίθμου MATSuMoTo

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 1.150.588. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν παραβίαση 1,85 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 206 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 157η επανάληψη με τιμή 839.367 (βελτίωση σε ποσοστό 27,05% επί της τιμής σχεδιασμού).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





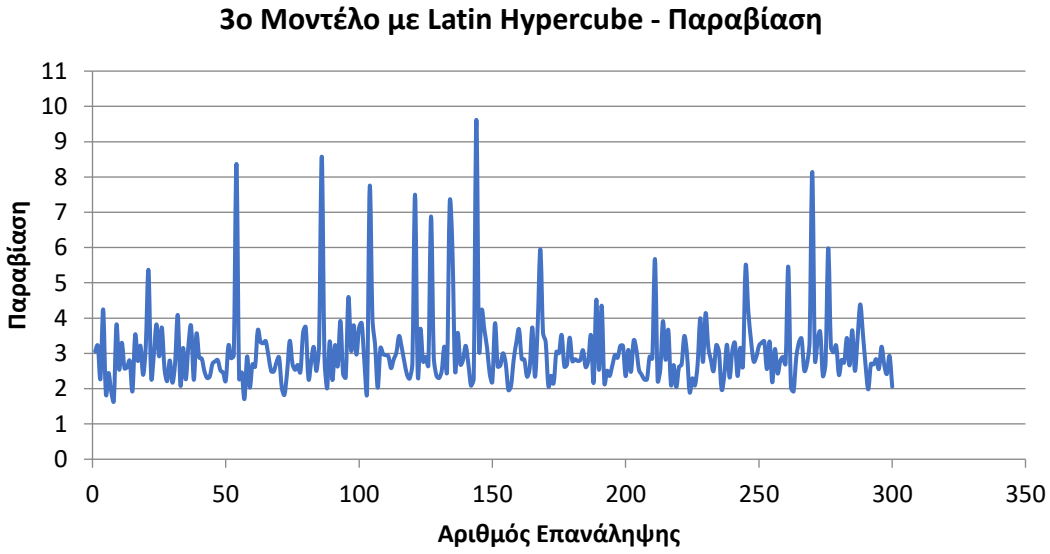


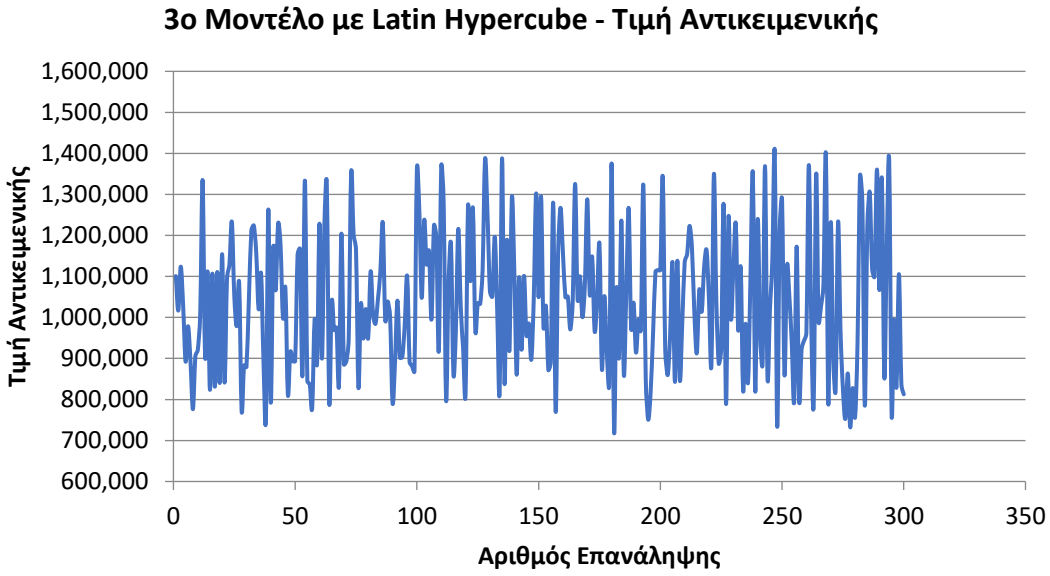
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.9.3 Εφαρμογή του αλγορίθμου Latin Hypercube

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 1.100.831 εμφανίζοντας παραβίαση 3,055. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Γενικά ο αλγόριθμος καθ' όλη τη διάρκεια της εφαρμογής του δεν καταφέρνει να βρει καμία εφικτή λύση. Οι τιμές της αντικειμενικής έχουν μεγάλη διακύμανση σε όλο το εύρος των επαναλήψεων με μέγιστη τιμή στην 247η επανάληψη με τιμή 1.401.455 (παραβίαση 3,395) και ελάχιστη στην 181η επανάληψη με τιμή 721.368 (παραβίαση 2,835).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





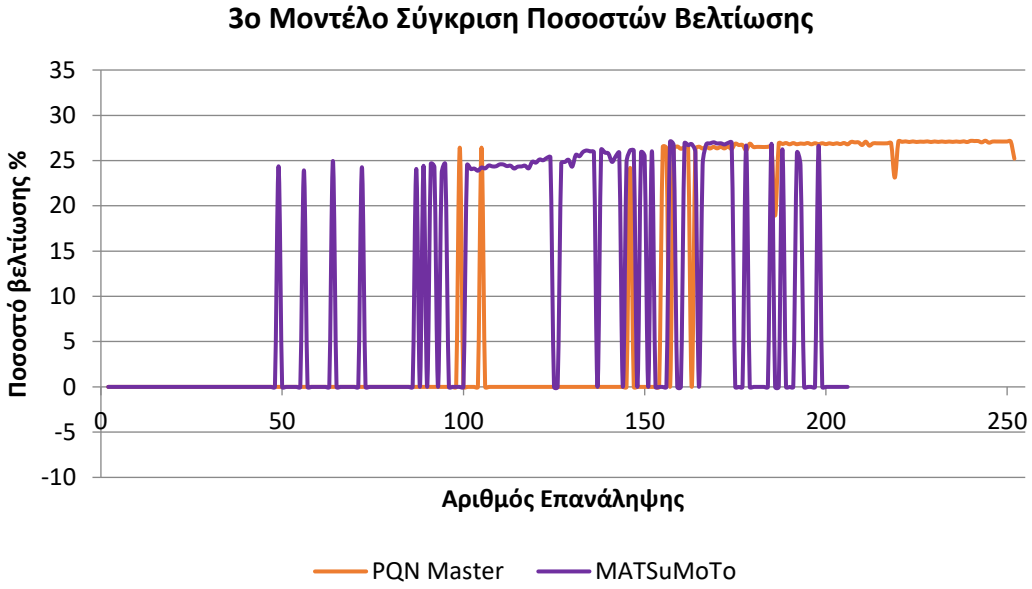
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.9.4 Συγκριτική ανάλυση αποτελεσμάτων

Στο συγκεκριμένο μοντέλο, τόσο κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου PQN Master, όσο και με εφαρμογή του αλγορίθμου MATSuMoTo, οι βέλτιστες λύσεις έχουν παραπλήσιες τιμές. Η κύρια διαφορά έγκειται στο ότι ο MATSuMoTo έφτασε πολύ γρηγορότερα

σε αυτήν. Και πάλι, ο αλγόριθμος Latin Hypercube, δεν καταφέρνει να βρει κάποια αποδεκτή λύση.

Στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου.

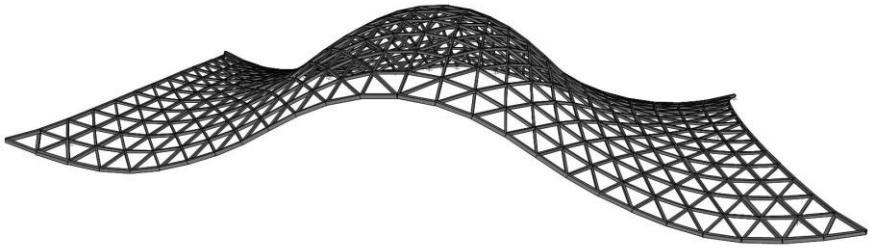


Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	1.595.508	1.523.646	1.401.455
Ελάχιστη τιμή	688.366	733.501	721.368
Τιμή σχεδιασμού	1.151.303 (μη εφικτή)	1.150.588 (μη εφικτή)	-
Αριθμός Επαναλήψεων	254	206	300
Πρώτη εφικτή λύση	(101η) 847.031 26,43%	(49η) 870.251 24,36%	-
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	(101η) 847.031 26,43%	(49η) 870.251 24,36%	-
Βέλτιστη τιμή	(246η) 838.242	(157η) 839.367	-
Ποσοστό βελτίωσης %	**27,19%**	**27,05%**	-
Πορεία αλγορίθμου	(101η) 847.031 26,43% (157η) 846.174 26,50% (246η) 838.242 27,19%	(49η) 870.251 24,36% (157η) 839.367 27,05%	Δεν βρίσκει εφικτές λύσεις

1.10 Ανάλυση 4ου Μοντέλου

Το 4ο μοντέλο, όπως προαναφέρθηκε, αφορά σε ένα στέγαστρο σχεδιασμένο ως χωροδικτύωμα με χαλύβδινα στοιχεία διαφόρων διατομών. Αποτελείται συνολικά από 1148 στοιχεία και 419 κόμβους και το μέγιστο ύψος είναι 16,6m. Τέλος έχει διαστάσεις σε κάτοψη 62 x 38,80 m.



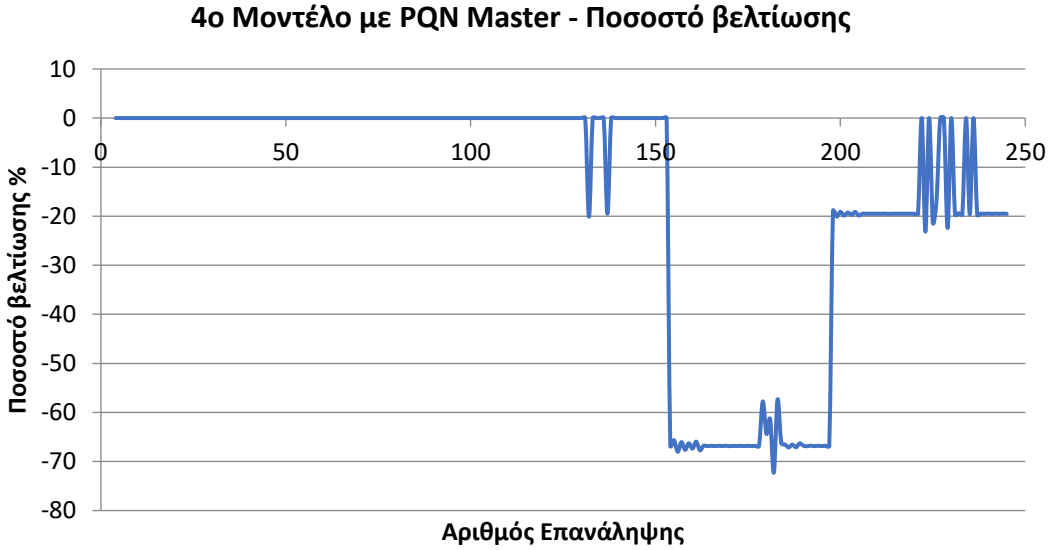
Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι όλες τυποποιημένες κοίλες ορθογωνικές από χάλυβα S355 ή S450 με τις κάτωθι διαστάσεις:

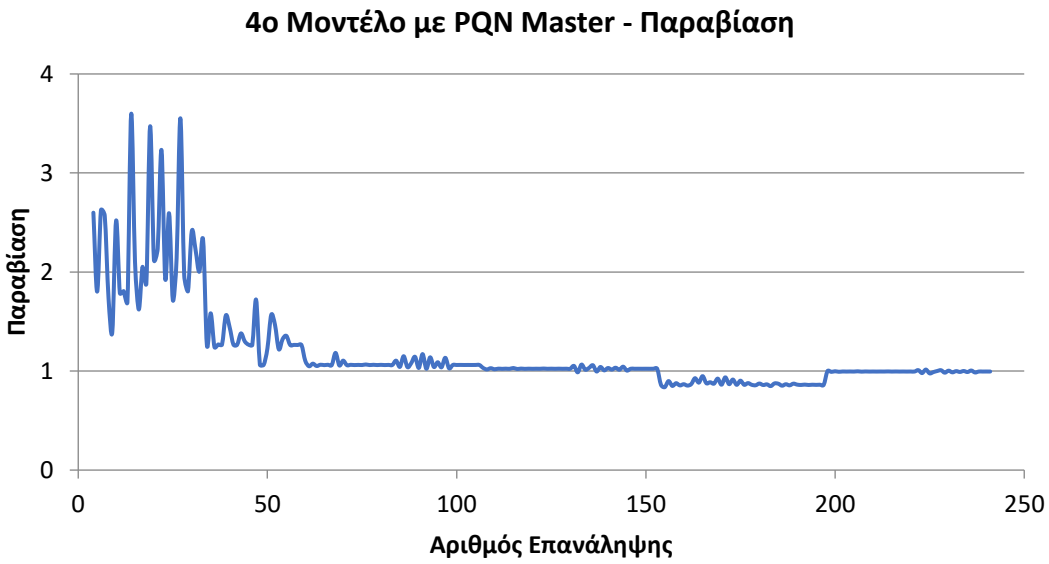
- TUBO200x140x14.2
- TUBO240x168x14.2
- TUBO280x196x22.2
- TUBO280x280x16
- TUBO300x210x25
- TUBO300x300x20

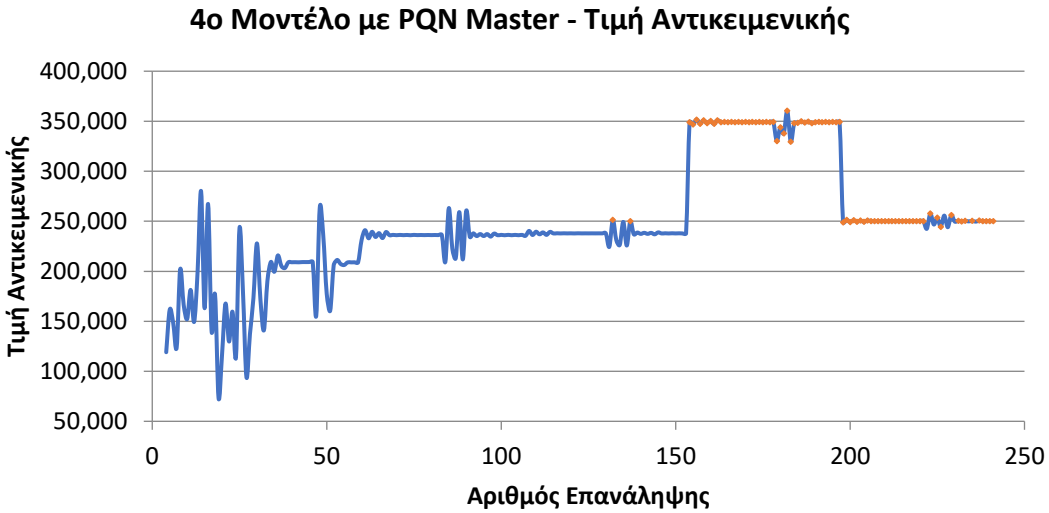
1.10.1 Εφαρμογή του αλγορίθμου PQN Master

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού είναι 209.290. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν μικρή παραβίαση 1,263 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 245 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης είναι μεγαλύτερη από την τιμή σχεδιασμού και εμφανίζεται στην 226η επανάληψη με τιμή 244.490 (ποσοστό -16,82% επί της τιμής σχεδιασμού).

Ο αλγόριθμος βρίσκει αρκετές εφικτές λύσεις, όλες όμως με τιμή μεγαλύτερη της τιμής σχεδιασμού. Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





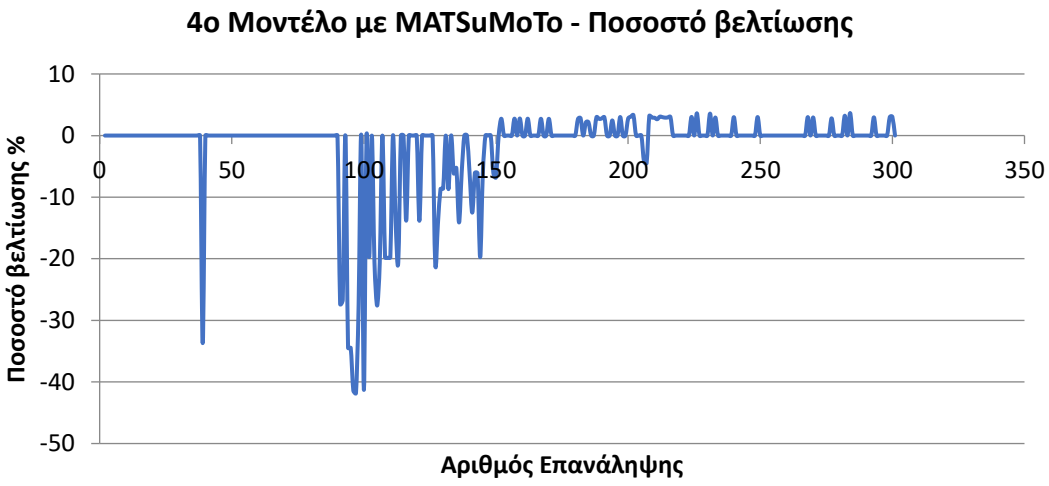


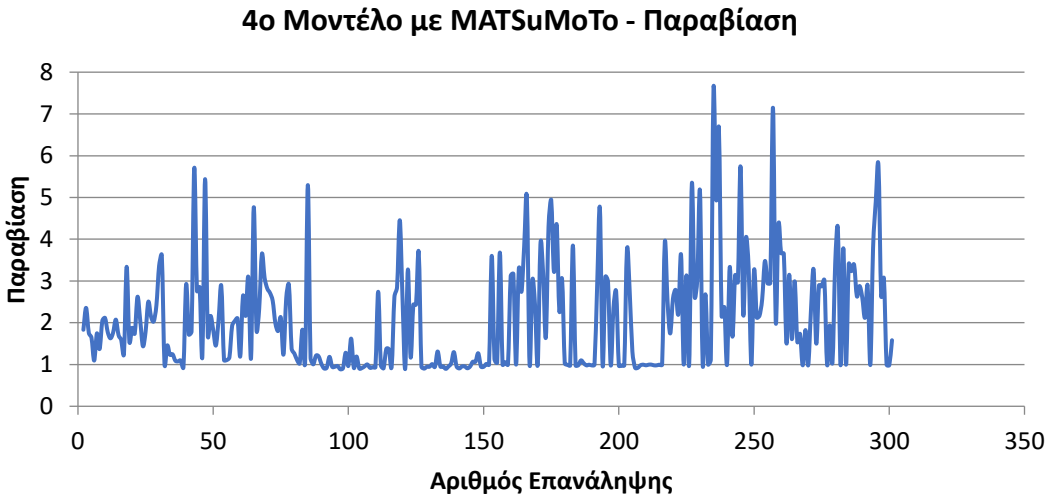
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

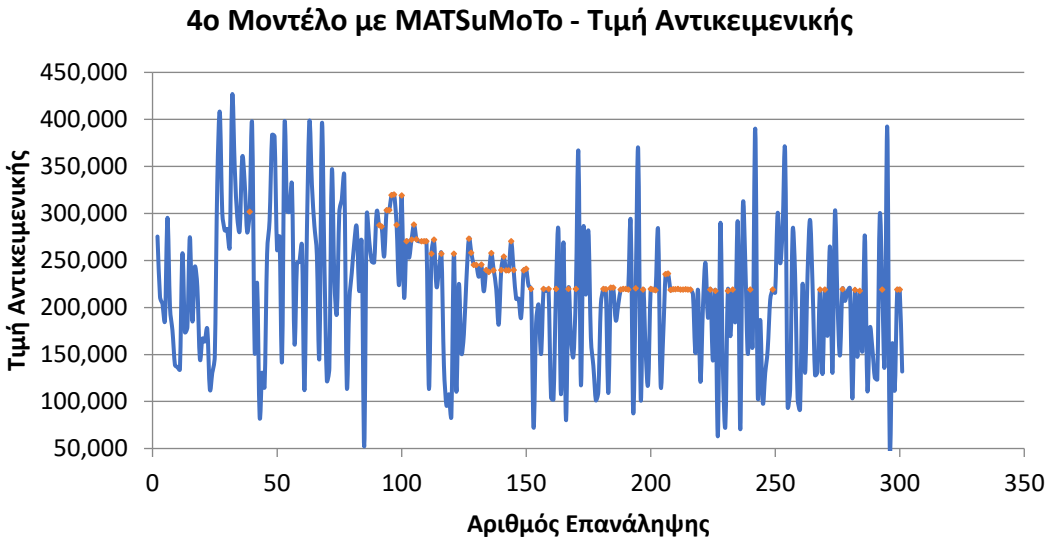
1.10.2 Εφαρμογή του αλγορίθμου MATSuMoTo

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 225.743. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν μικρή παραβίαση 1,186 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 301 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 284η επανάληψη με τιμή 217.476 (βελτίωση σε ποσοστό 3,66% επί της τιμής σχεδιασμού).

Γενικά ο αλγόριθμος παρουσιάζει αρκετές διακυμάνσεις σε όλο το εύρος των επαναλήψεων, δίνοντας αρκετές εφικτές λύσεις με τιμές όμως ελαφρά μικρότερες της τιμής σχεδιασμού. Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.







** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

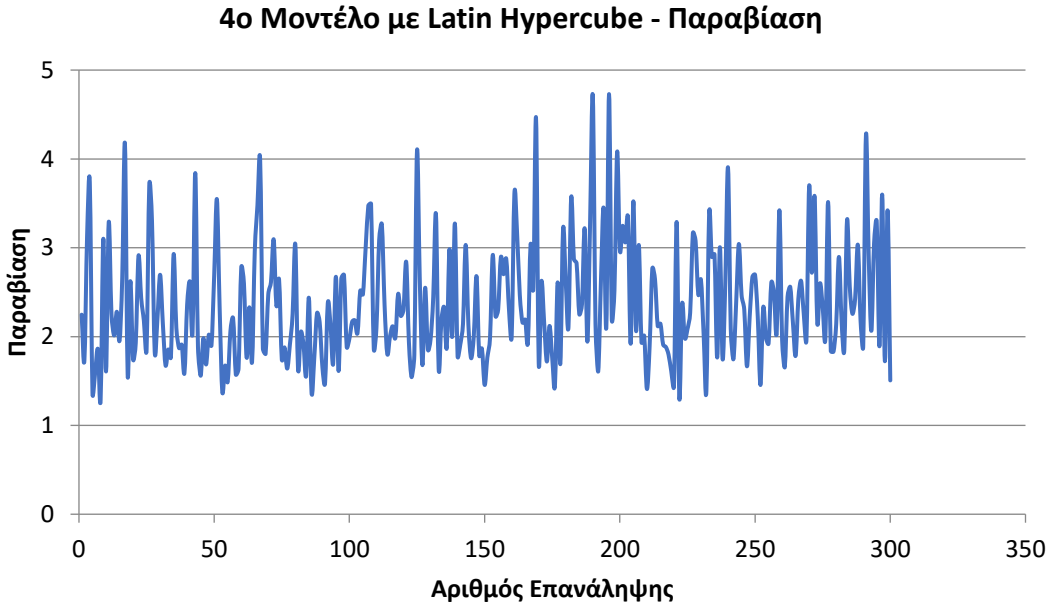
1.10.3 Εφαρμογή του αλγορίθμου Latin Hypercube

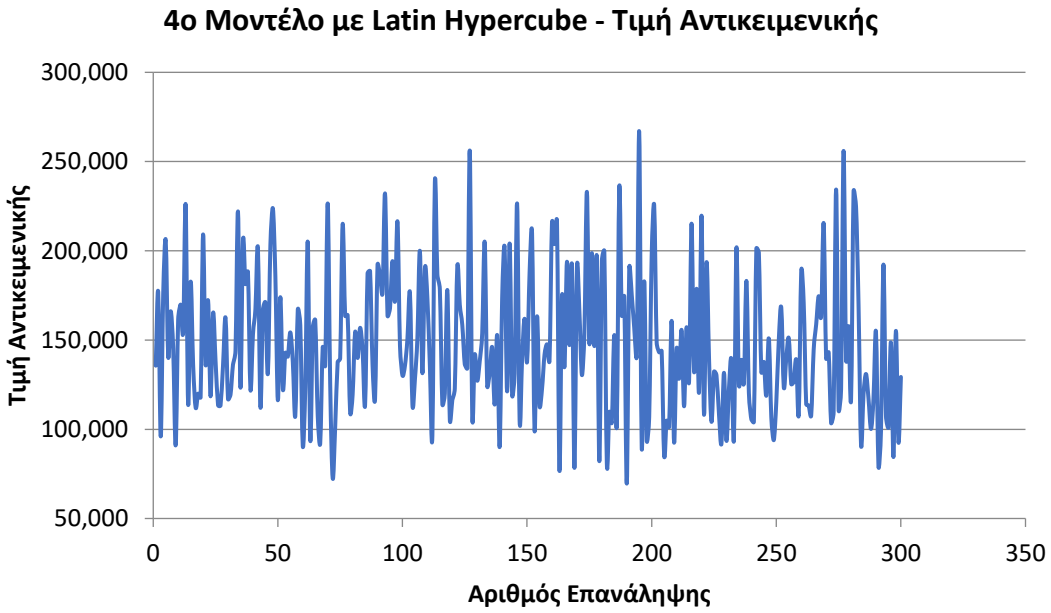
Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 135.623 εμφανίζοντας παραβίαση 2,247. Συνολικά γίνονται

300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί.

Γενικά ο αλγόριθμος καθ' όλη τη διάρκεια της εφαρμογής του δεν καταφέρνει να βρει καμία εφικτή λύση. Οι τιμές της αντικειμενικής έχουν μεγάλη διακύμανση σε όλο το εύρος των επαναλήψεων με μέγιστη τιμή στην 195η επανάληψη με τιμή 266.464 (παραβίαση 2,107) και ελάχιστη στην 190η επανάληψη με τιμή 69.556 (παραβίαση 4,71).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.



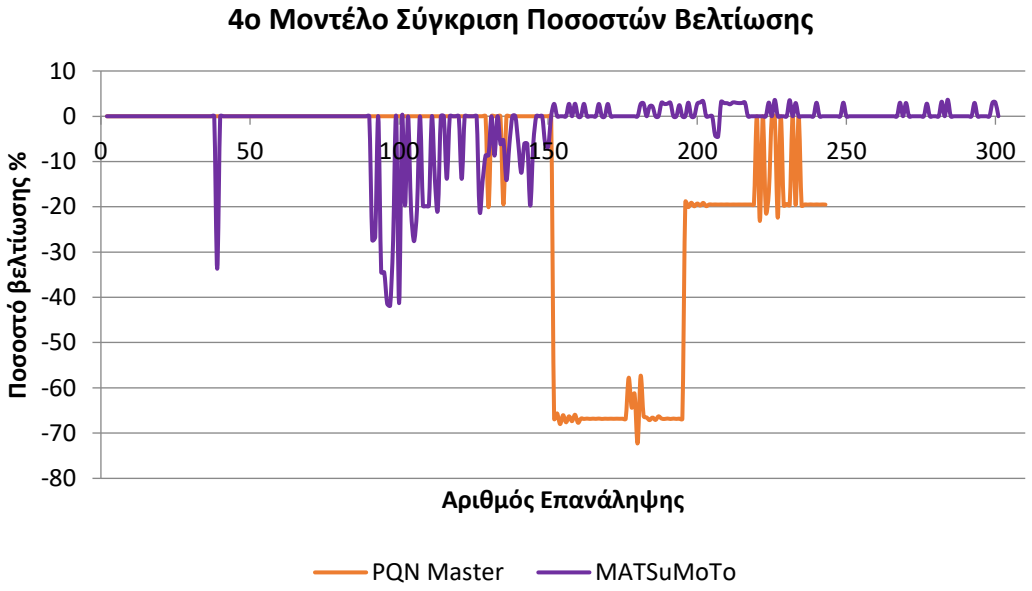


** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.10.4 Συγκριτική ανάλυση αποτελεσμάτων

Στο συγκεκριμένο μοντέλο ο αλγόριθμος PQN Master, αν και βρίσκει αρκετές εφικτές λύσεις, ωστόσο δεν μπορεί να εντοπίσει λύση με τιμή μικρότερη της τιμής σχεδιασμού, σε αντίθεση με τον MATSuMoTo που βρίσκει λίγο καλύτερη λύση από αυτή του σχεδιασμού. Και πάλι, ο αλγόριθμος Latin Hypercube, δεν καταφέρνει να βρει κάποια αποδεκτή λύση.

Στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου.

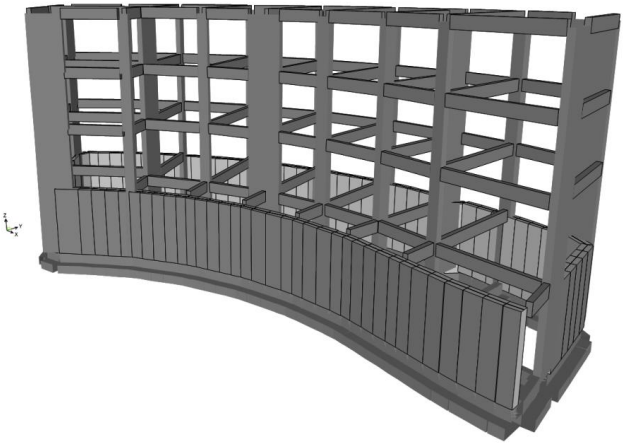


Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	432.623	425.003	266.464
Ελάχιστη τιμή	44.905	51.391	69.556
Τιμή σχεδιασμού	209.290 (μη εφικτή)	225.743 (μη εφικτή)	-
Αριθμός Επαναλήψεων	245	301	300
Πρώτη εφικτή λύση	(132η) 251.361 -20,10%	(39η) 301.807 -33,69%	-
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	-	(152η) 219.562 2,74%	-
Βέλτιστη τιμή	(226η) 244.490	(284η) 217.476	-
Ποσοστό βελτίωσης %	**-16,82%**	**3,66%**	**-**
Πορεία αλγορίθμου	Όλες οι λύσεις έχουν τιμές αντικειμενικής μεγαλύτερες από την τιμή σχεδιασμού	(152η) 219.562 2,74% (284η) 217.476 3,66%	Δεν βρίσκει εφικτές λύσεις

1.11 Ανάλυση 5ου Μοντέλου

Το 5ο μοντέλο, όπως προαναφέρθηκε, αφορά σε ένα σχολείο κατασκευασμένο από οπλισμένο σκυρόδεμα, με υπόγειο και τρεις ορόφους. Αποτελείται συνολικά από 378 ραβδωτά στοιχεία, 92 κελύφη και 255 κόμβους. Το ύψος του κάθε ορόφου είναι αντίστοιχα 3,50m (1ος), 3,25m (2ος) και 3,25m (3ος). Το συνολικό ύψος από τη στάθμη του υπογείου μέχρι το δώμα είναι 14,00m. Τέλος έχει μέγιστες διαστάσεις σε κάτοψη 32,60 x 8,75 m. και συνολικό εμβαδό 275 m^2. Οι διατομές είναι από οπλισμένο σκυρόδεμα κατηγοριών C20/25, C25/30 και C30/37.



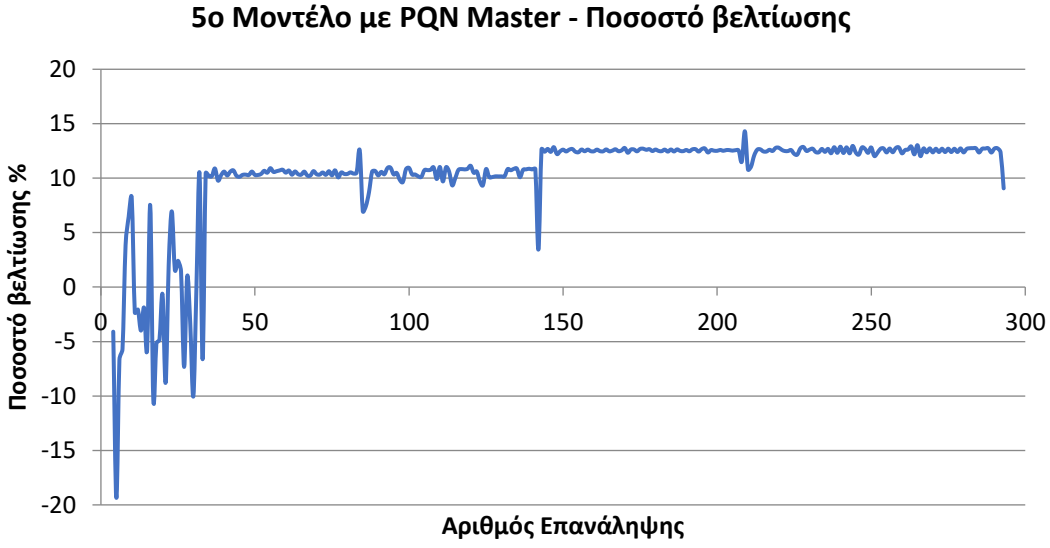
Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι οι ακόλουθες:

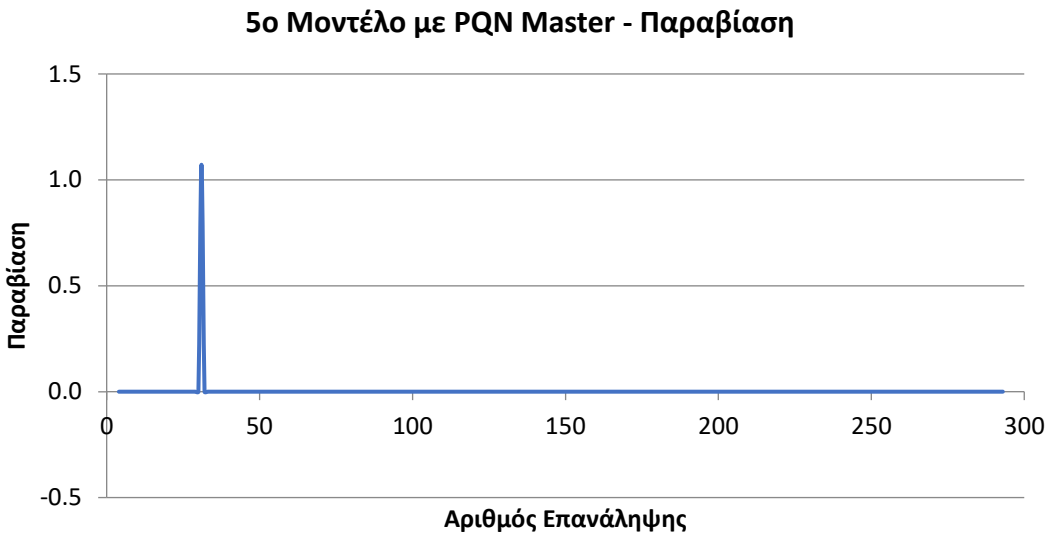
- Δοκοί ορθογωνικής διατομής 600x250
- Υποστυλώματα και τοιχία ορθογωνικής διατομής διαφόρων διαστάσεων 2000x300, 600x300, 700x300, 700x400
- Περιμετρικό τοιχίο υπογείου πάχους 500mm
- Θεμελίωση διατομής Τ με h=1100mm, w=1350mm, t_f=500mm, t_w=350mm

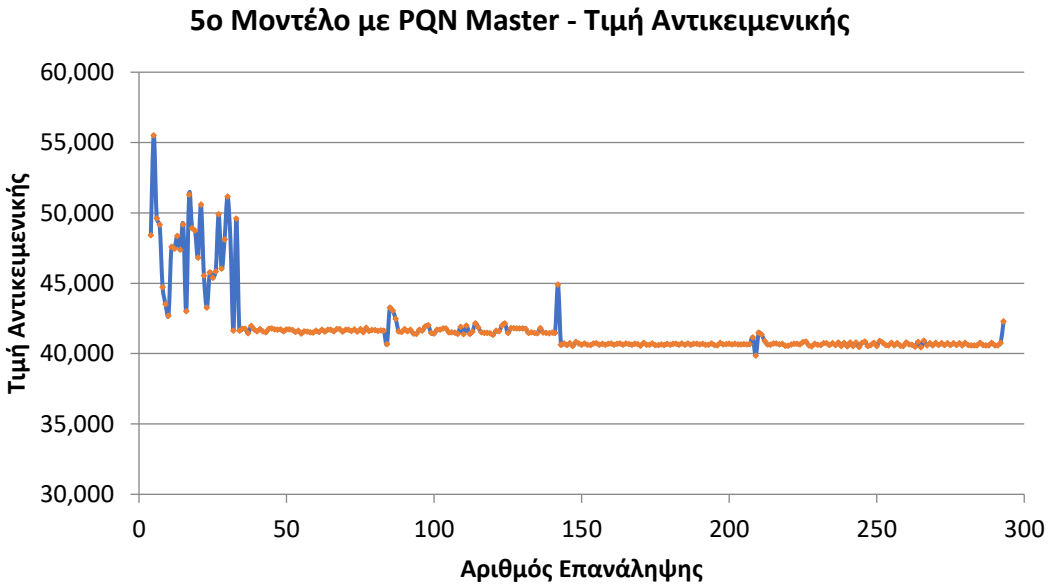
1.11.1 Εφαρμογή του αλγορίθμου PQN Master

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού είναι 46.503. Συνολικά γίνονται 293 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 209η επανάληψη με τιμή 39.849 (βελτίωση σε ποσοστό 14,31% επί της τιμής σχεδιασμού). Όλες οι λύσεις πλην της 31ης επανάληψης είναι εφικτές.

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





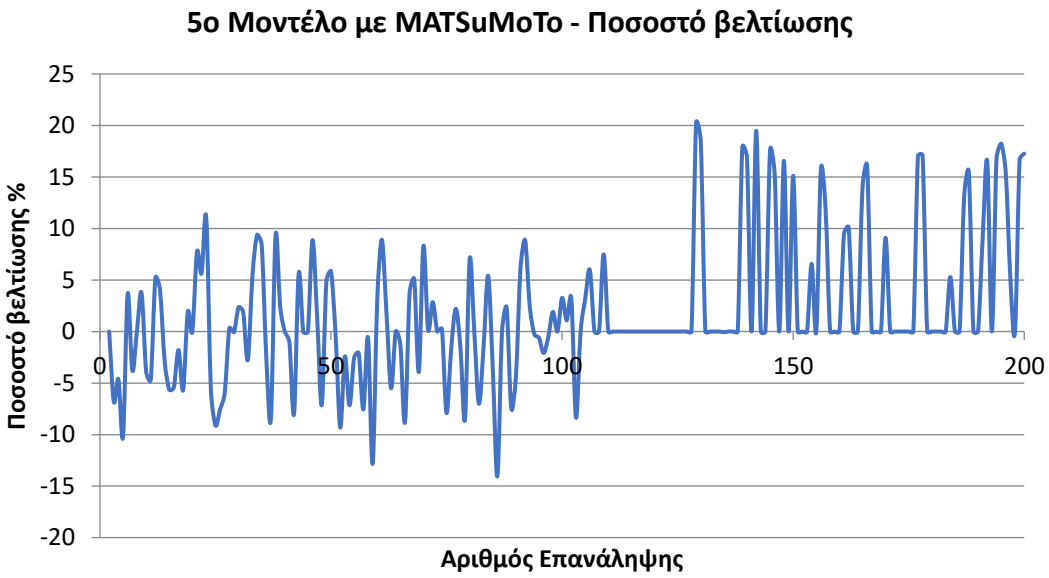


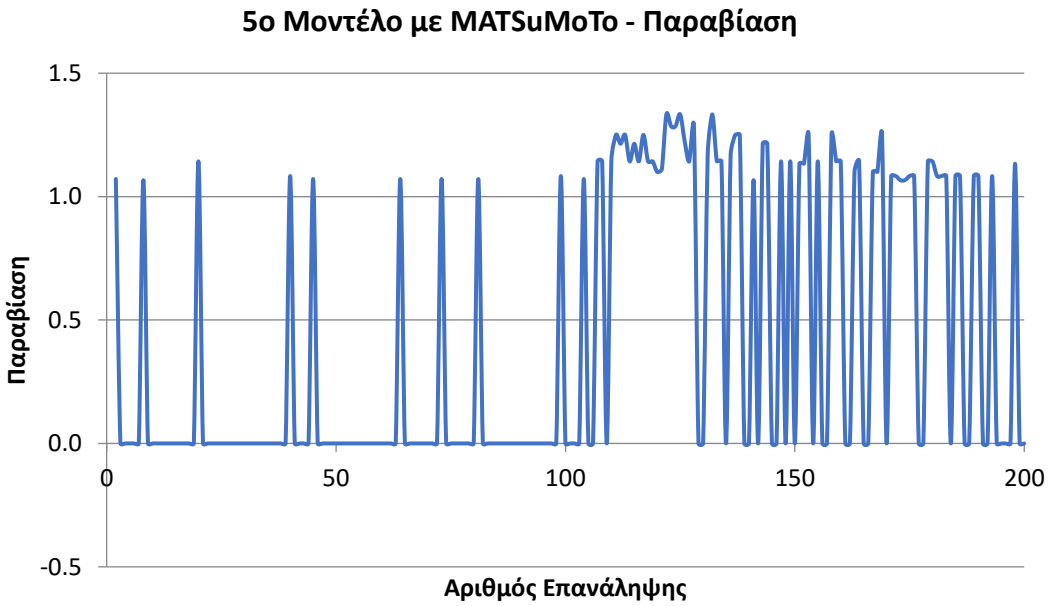
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

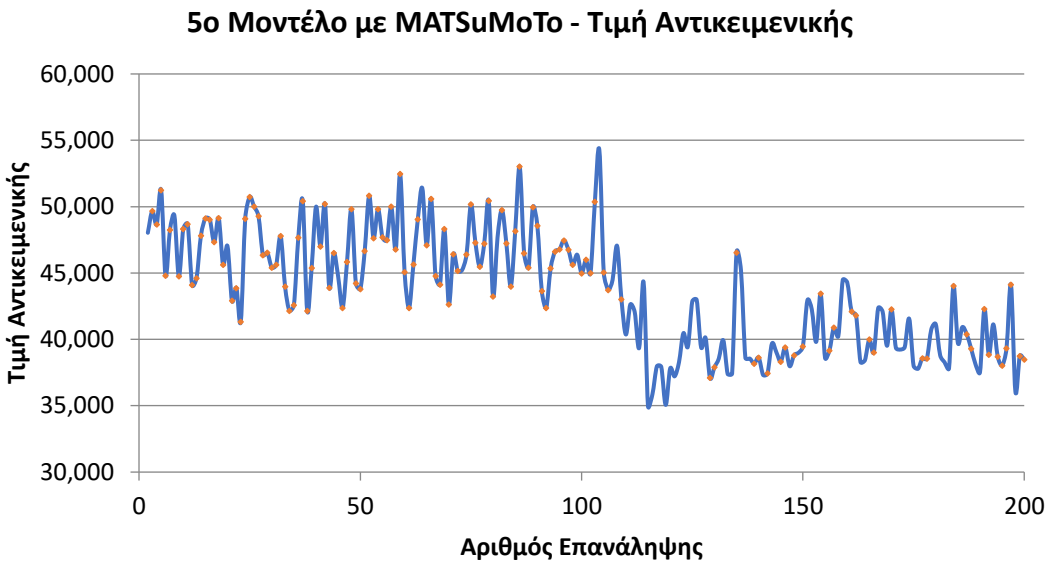
1.11.2 Εφαρμογή του αλγορίθμου MATSuMoTo

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 46.503. Συνολικά γίνονται 200 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην $129^{η}$ επανάληψη με τιμή 37.109 (βελτίωση σε ποσοστό 20,20% επί της τιμής σχεδιασμού).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.


5ο Μοντέλο με MATSuMoTo - Ποσοστό βελτίωσης
Ποσοστό βελτίωσης %
25
20
15
10
5
0
-5
-10
-15
-20
0
50
100
150
200
Αριθμός Επανάληψης


5ο Μοντέλο με MATSuMoTo - Παραβίαση
Παραβίαση
1.5
1.0
0.5
0.0
-0.5
0
50
100
150
200
Αριθμός Επανάληψης

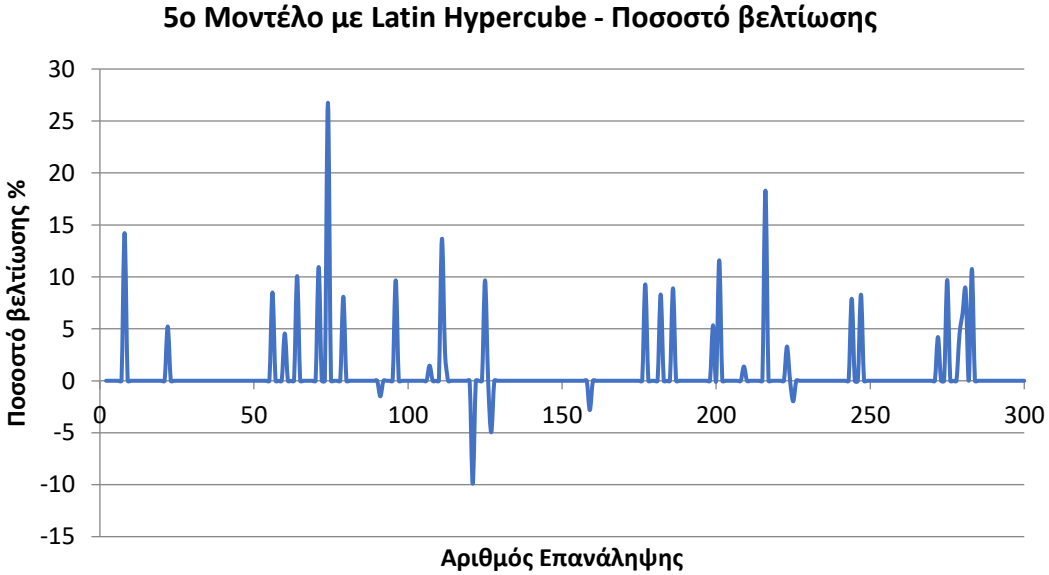


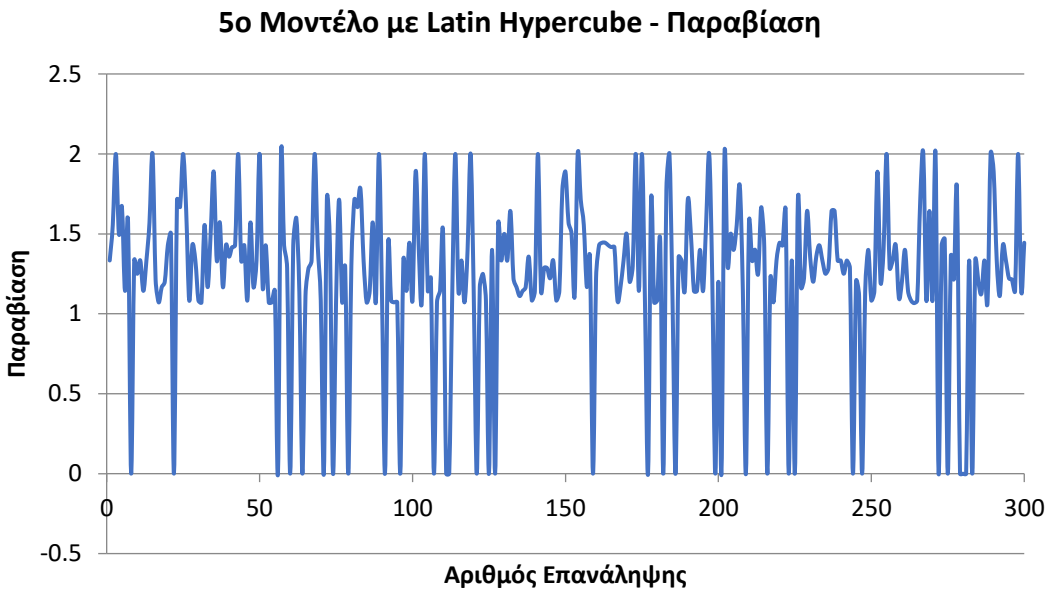
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

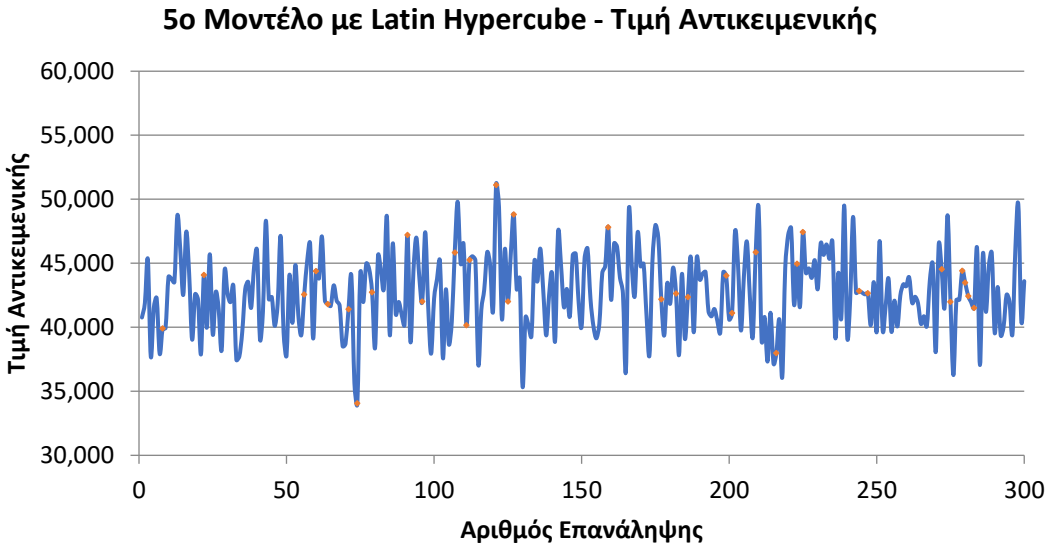
1.11.3 Εφαρμογή του αλγορίθμου Latin Hypercube

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 40.765 εμφανίζοντας παραβίαση 1,333. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 74η επανάληψη με τιμή 34.062 (βελτίωση σε ποσοστό 16,44% με βάση την τιμή σχεδιασμού του αλγόριθμου PQN Master).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





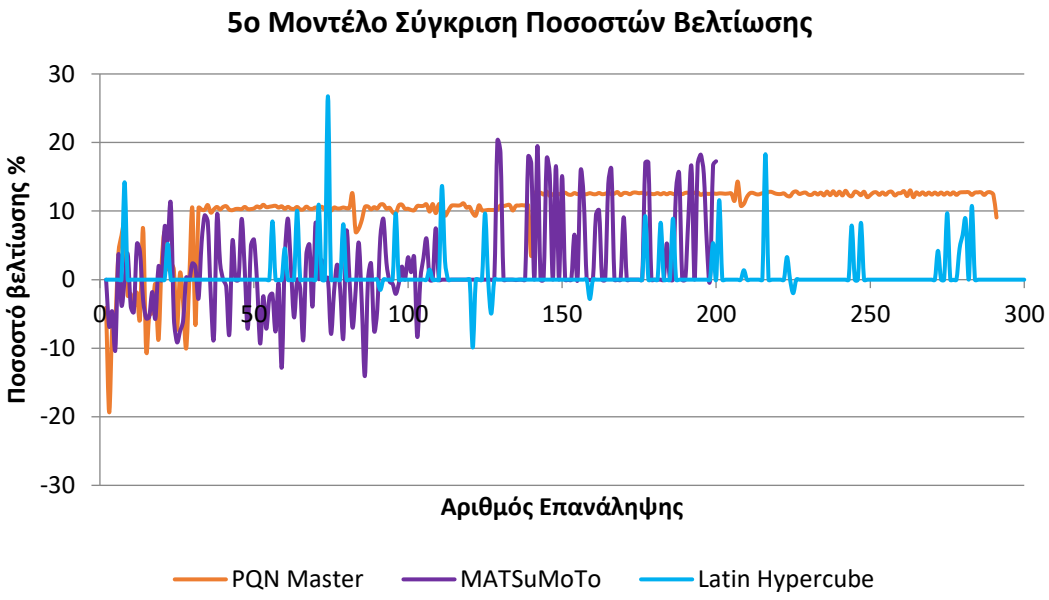


** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.11.4 Συγκριτική ανάλυση αποτελεσμάτων

Στο συγκεκριμένο μοντέλο ο αλγόριθμος MATSuMoTo βρίσκει αρκετά καλύτερη λύση από τον PQN Master και πολύ πιο γρήγορα. Όμως στην περίπτωση αυτή ο αλγόριθμος Latin Hypercube εντοπίζει τελικά την πιο καλή εφικτή λύση με την μικρότερη τιμή αντικειμενικής και από τις άλλες δύο μεθόδους.

Στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου.

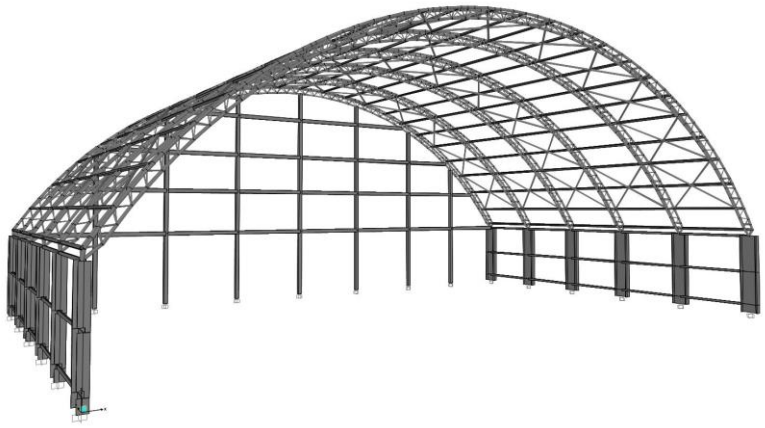


Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	73.155	54.278	51.107
Ελάχιστη τιμή	27.995	34.967	34.062
Τιμή σχεδιασμού	46.503 (εφικτή)	46.503 (εφικτή)	46.503 (PQN Master)
Αριθμός Επαναλήψεων	293	200	300
Πρώτη εφικτή λύση	(4η) 48.407 -4,10%	(4η) 48.663 -4,64%	(8η) 39.895 14,21%
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	(8η) 44.717 3,84%	(6η) 44.804 3,65%	(8η) 39.895 14,21%
Βέλτιστη τιμή	(209η) 39.849	(129η) 37.109	(74η) 34.062
Ποσοστό βελτίωσης %	**14,31%**	**20,20%**	**26,75%**
Πορεία αλγορίθμου	(8η) 44.717 3,84% (9η) 43.505 6,45% (10η) 42.706 8,16% (32η) 41.634 10,47% (84η) 40.665 12,55% (209η) 39.849 14,31%	(6η) 44.804 3,65% (12η) 44.102 5,16% (23η) 41.343 11,09% (129η) 37.109 20,20%	(8η) 39.895 14,21% (74η) 34.062 26,75%

1.12 Ανάλυση 6ου Μοντέλου

Το 6ο μοντέλο, όπως προαναφέρθηκε, αφορά σε ένα υπόστεγο από χαλύβδινα στοιχεία διαφόρων διατομών. Αποτελείται συνολικά από 3294 ραβδωτά στοιχεία και 1379 κόμβους, το ύψος μέχρι το καμπύλο στέγαστρο είναι 2,45m και το συνολικό ύψος 7,87m. Τέλος έχει διαστάσεις σε κάτοψη 12,35 x 19,94 m. και εμβαδό 246,26 m^2. Όλες οι μεταλλικές διατομές είναι από χάλυβα S355.



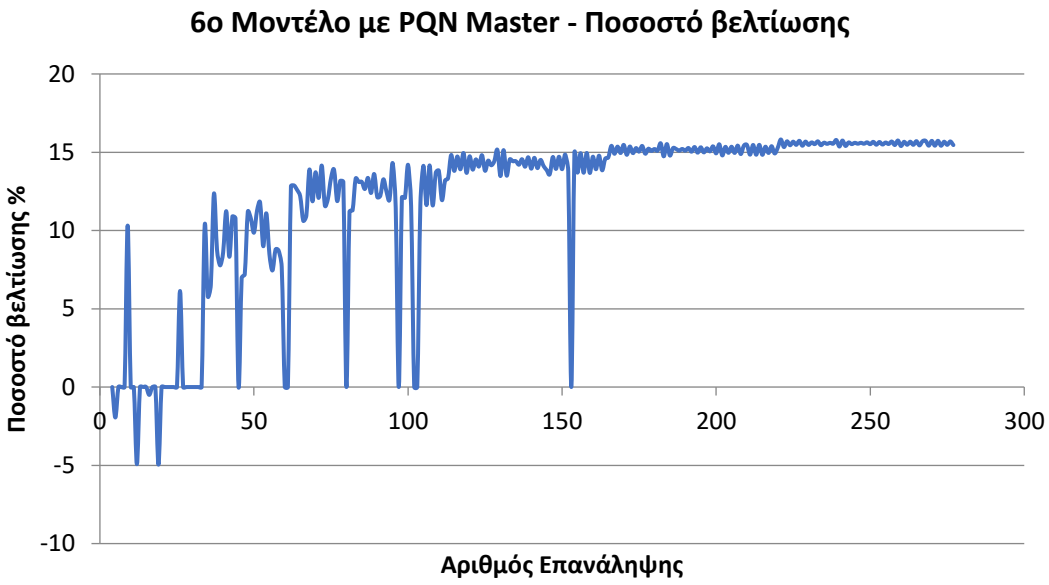
Οι διατομές που έχουν επιλεχθεί κατά το σχεδιασμό του αρχικού μοντέλου είναι οι ακόλουθες:

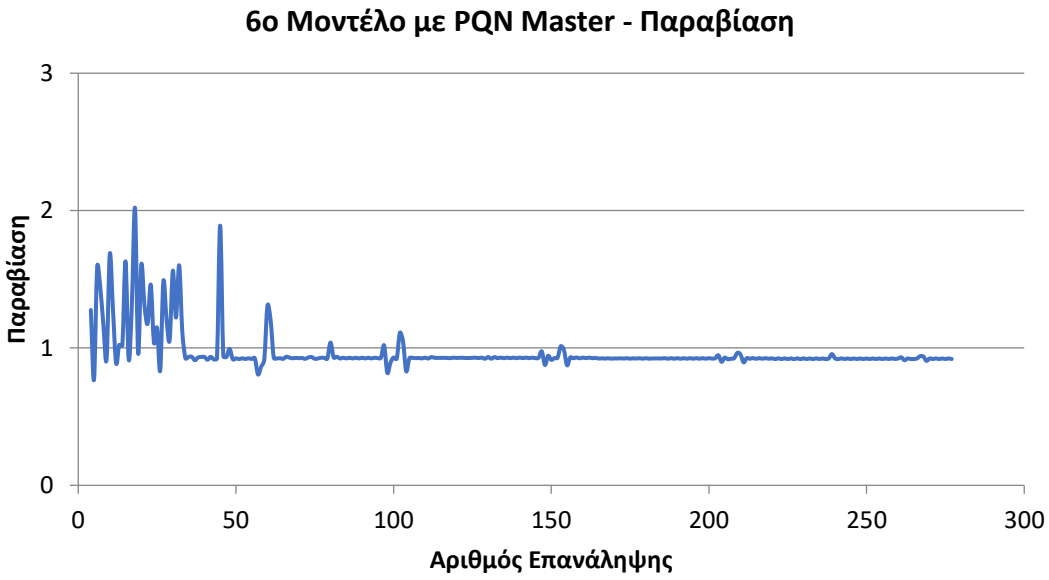
- Δοκοί οπίσθιας πλευράς κοίλης ορθογωνικής διατομής h=125mm, w=75mm, t_f=6mm, t_w=6mm
- Πλευρικοί δοκοί κοίλης ορθογωνικής διατομής h=65mm, w=35mm, t_f=4mm, t_w=4mm
- Υποστυλώματα οπίσθιας πλευράς κοίλης ορθογωνικής διατομής h=125mm, w=125mm, t_f=9mm, t_w=9mm
- Πλευρικά υποστυλώματα διατομής διπλού T h=500mm, w_f=200mm, t_f=16mm, t_w=10.2mm
- Δικτυωτές καμπύλες δοκοί κοίλης κυκλικής διατομής 48.3x5.4, 21.3x3.2 και 60.3x3.6
- Χιαστί σύνδεσμοι οροφής 48.3x5.4
- - Τεγίδες κοίλης ορθογωνικής διατομής h=65mm, w=35mm, t_f=3mm, t_w=3mm

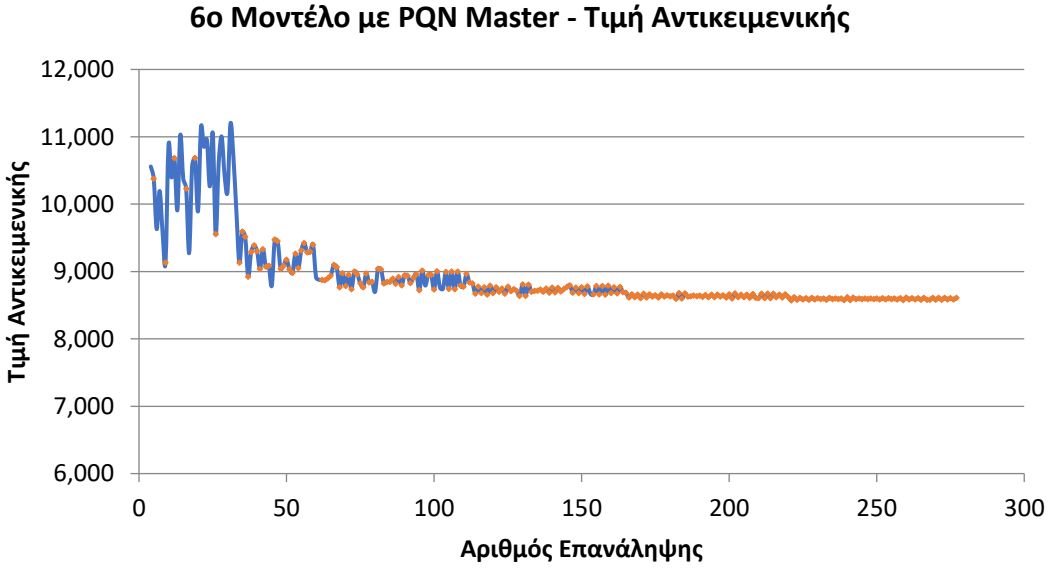
1.12.1 Εφαρμογή του αλγορίθμου PQN Master

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης PQN Master, η τιμή σχεδιασμού είναι 10.180. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν μικρή παραβίαση 1,072 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 277 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην $221^η$ επανάληψη με τιμή 8.570 (βελτίωση σε ποσοστό 15,82% επί της τιμής σχεδιασμού).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.


6ο Μοντέλο με PQN Master - Ποσοστό βελτίωσης
Ποσοστό βελτίωσης %
Αριθμός Επανάληψης
20
15
10
5
0
-5
-10
0
50
100
150
200
250
300


6ο Μοντέλο με PQN Master - Παραβίαση
Παραβίαση
Αριθμός Επανάληψης
3
2
1
0
0
50
100
150
200
250
300

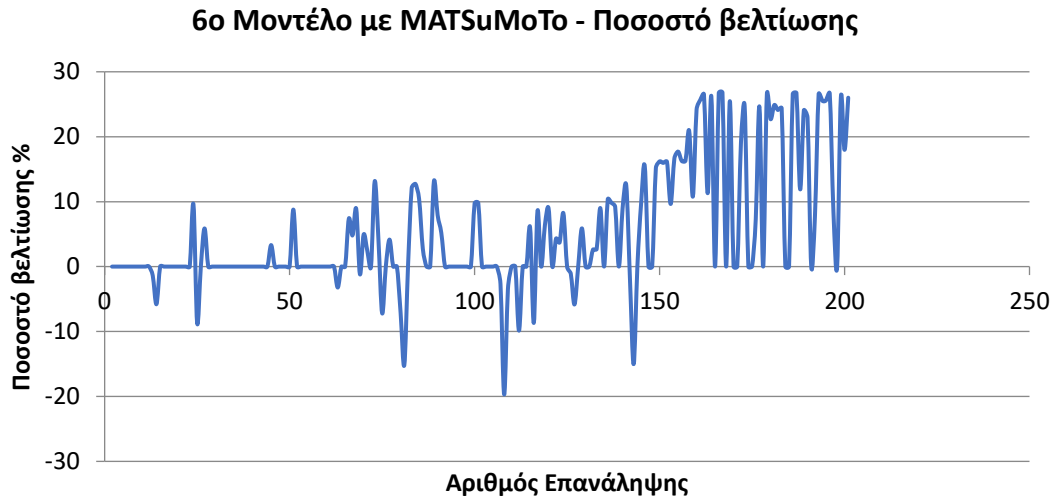

6ο Μοντέλο με PQN Master - Τιμή Αντικειμενικής
Τιμή Αντικειμενικής
Αριθμός Επανάληψης
12,000
11,000
10,000
9,000
8,000
7,000
6,000
0
50
100
150
200
250
300

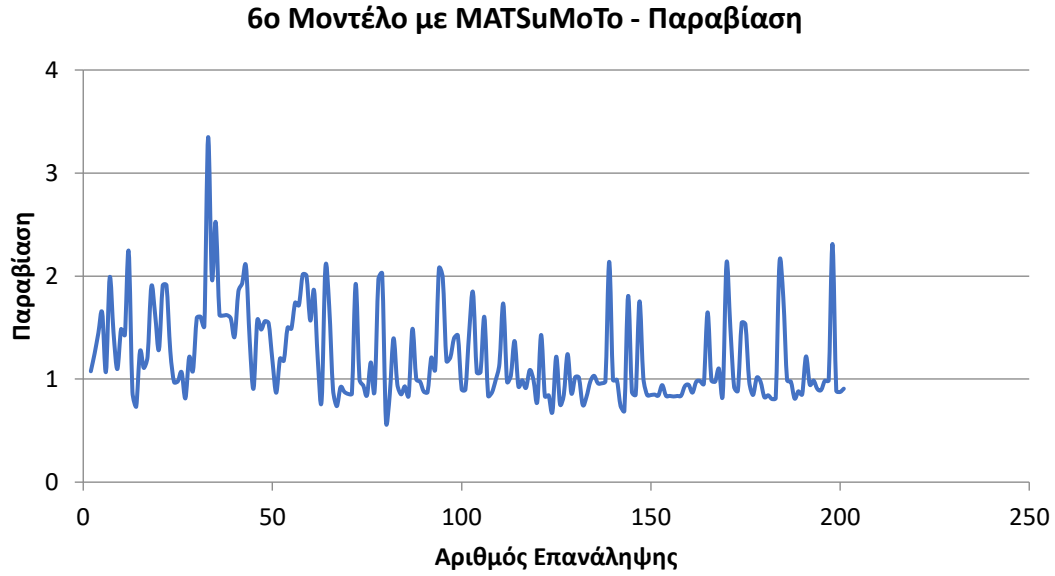
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

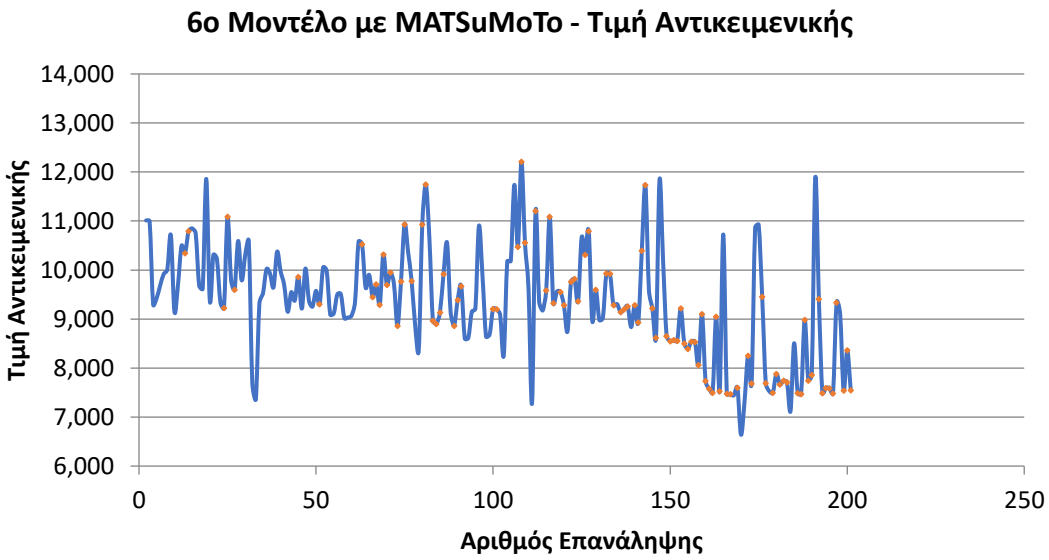
1.12.2 Εφαρμογή του αλγορίθμου MATSuMoTo

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης MATSuMoTo η τιμή σχεδιασμού είναι 10.200. Η επιλογή όμως των συγκεκριμένων (αρχικών) διατομών σχεδιασμού εμφανίζουν μικρή παραβίαση 1,072 γεγονός που καθιστά τη συγκεκριμένη λύση μη εφικτή. Συνολικά γίνονται 201 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 167η επανάληψη με τιμή 7.467 (βελτίωση σε ποσοστό 26,79% επί της τιμής σχεδιασμού).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.





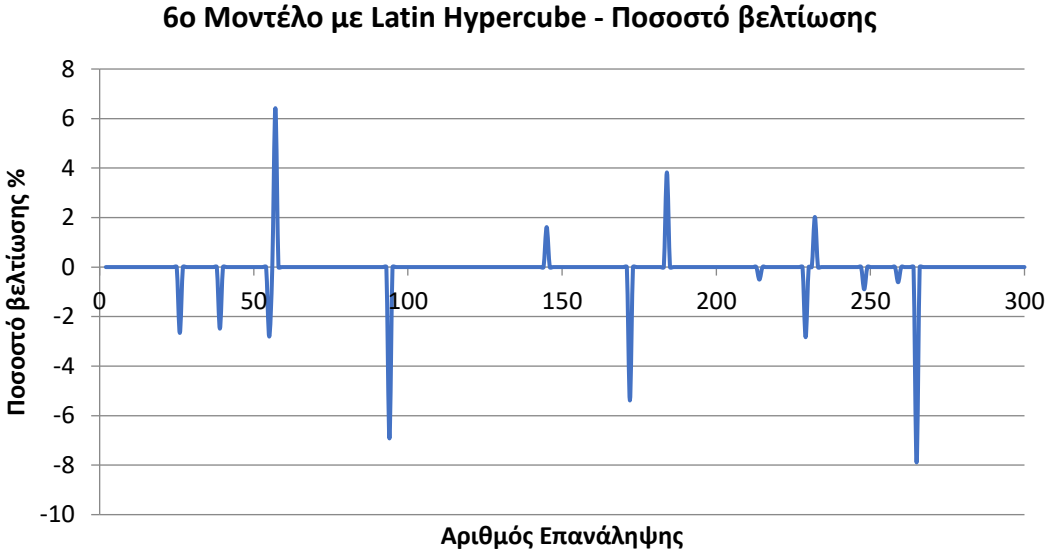


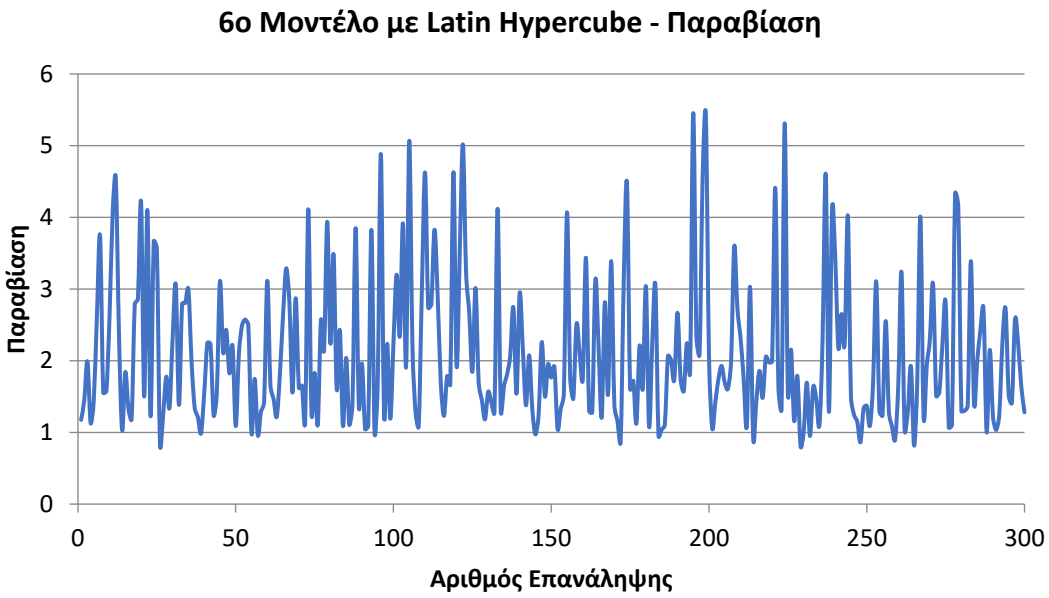
** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

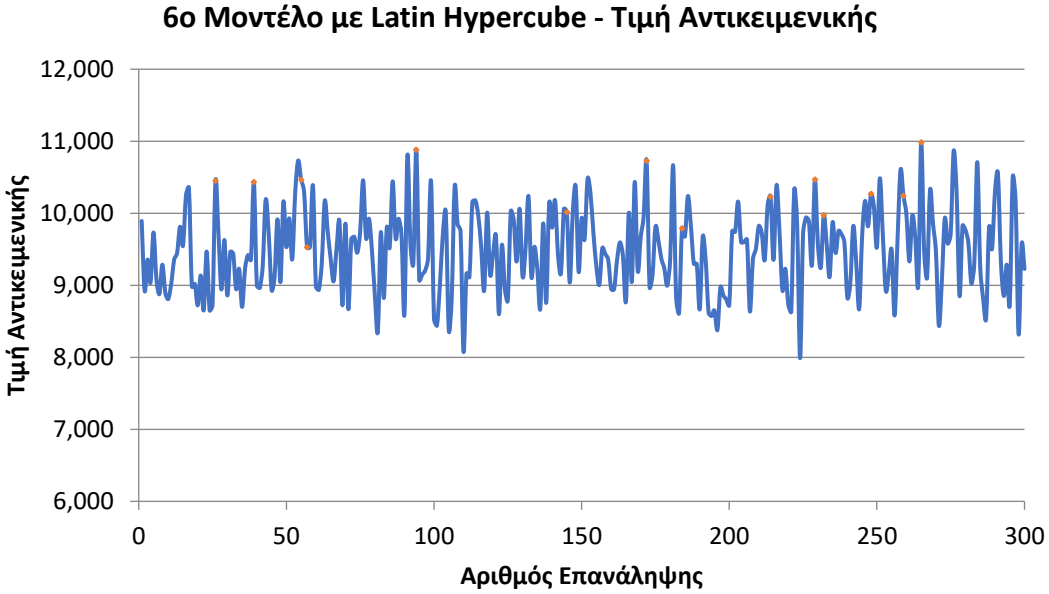
1.12.3 Εφαρμογή του αλγορίθμου Latin Hypercube

Κατά την ανάλυση με εφαρμογή του αλγορίθμου βελτιστοποίησης Latin Hypercube η πρώτη τιμή της αντικειμενικής συνάρτησης είναι 9.892 εμφανίζοντας παραβίαση 1,174. Συνολικά γίνονται 300 επαναλήψεις πριν σταματήσει ο αλγόριθμος να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Η βέλτιστη τιμή της αντικειμενικής συνάρτησης εμφανίζεται στην 57η επανάληψη με τιμή 9.527 (βελτίωση σε ποσοστό 3,69% με βάση την τιμή σχεδιασμού του αλγόριθμου PQN Master).

Στα επόμενα διαγράμματα απεικονίζεται η πορεία του συγκεκριμένου αλγορίθμου κατά τη διαδικασία βελτιστοποίησης του μοντέλου.



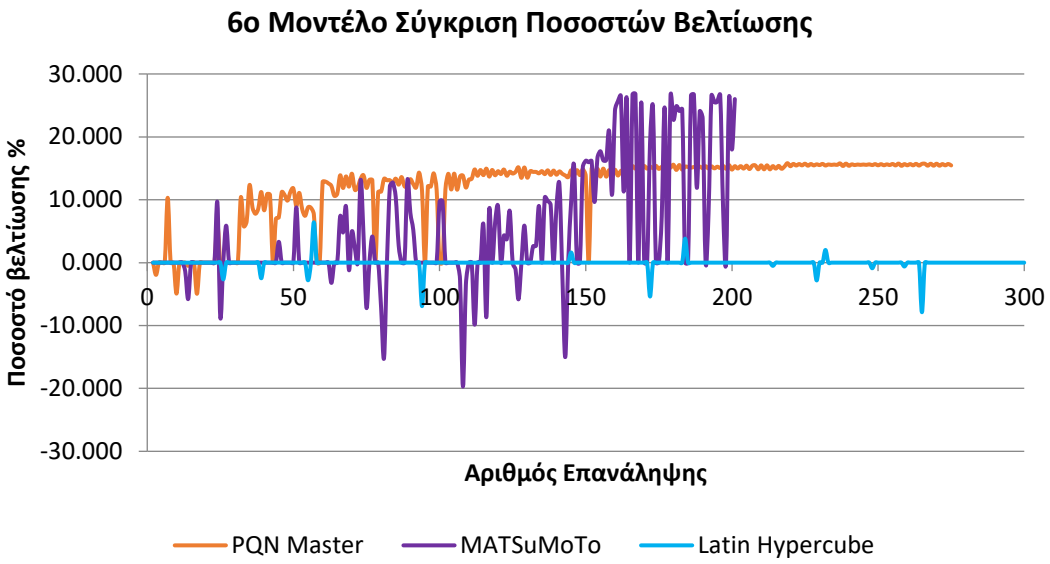




** Με κόκκινο χρώμα υποδεικνύονται οι εφικτές λύσεις (Παραβίαση μικρότερη ή ίση με 1,0)*

1.12.4 Συγκριτική ανάλυση αποτελεσμάτων

Στο συγκεκριμένο μοντέλο ο αλγόριθμος MATSuMoTo βρίσκει αρκετά καλύτερη λύση τόσο από τον PQN Master όσο και από τον Latin Hypercube. Στο επόμενο διάγραμμα και πίνακα απεικονίζεται ταυτόχρονα η πορεία όλων των αλγορίθμων κατά τη διαδικασία βελτιστοποίησης του μοντέλου.



Συνοπτικός πίνακας σύγκρισης πορείας αλγορίθμων

	PQN Master	MATSuMoTo	Latin Hypercube
Μέγιστη τιμή	17.310	12.210	10.984
Ελάχιστη τιμή	4.854	6.640	7.988
Τιμή σχεδιασμού	10.180 (μη εφικτή)	10.200 (μη εφικτή)	10.180 (PQN Master)
Αριθμός Επαναλήψεων	277	201	300
Πρώτη εφικτή λύση	(5η) 10.378 -1,95%	(13η) 10.341 -1,39%	(26η) 10.451 -2,66%
Πρώτη εφικτή με τιμή μικρότερη του σχεδιασμού	(9η) 9.130 10,32%	(24η) 9.223 9,57%	(57η) 9.527 6,42%
Βέλτιστη τιμή	(221η) 8.570	(167η) 7.467	(57η) 9.527
Ποσοστό βελτίωσης %	**15,82%**	**26,79%**	**6,42%**
Πορεία αλγορίθμου	(9η) 9.130 10,32% (37η) 8.923 12,35% (62η) 8.874 12,83% (68η) 8.766 13,89% (72η) 8.739 14,15% (221η) 8.570 15,82%	(24η) 9.223 9,57% (73η) 8.860 13,14% (146η) 8.621 15,48% (158η) 8.065 20,93% (167η) 7.467 26,79%	(57η) 9.527 6,42%

1.13 Συμπεράσματα

Όπως προαναφέρθηκε διαπιστώθηκε ότι, κάθε αλγόριθμος συμπεριφέρεται τελείως διαφορετικά ανάλογα με τα χαρακτηριστικά της κάθε κατασκευής. Έτσι, κάποιοι αλγόριθμοι έδωσαν εξαιρετικά αποτελέσματα όταν εφαρμόστηκαν σε ένα συγκεκριμένο δομικό σύστημα αλλά οι ίδιοι αλγόριθμοι, σε άλλη κατασκευή, είτε εμφάνισαν «κακή» επίδοση είτε δεν κατάφεραν ποτέ να βρουν εφικτές λύσεις κάτω από τους τιθέμενους περιορισμούς της αντικειμενικής συνάρτησης.

Κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου PQN Master διαπιστώνεται ο πρακτικά ντετερμινιστικός χαρακτήρας αυτού βασισμένος σε gradient-based μεθόδους. Οι τιμές της αντικειμενικής συνάρτησης στις πρώτες επαναλήψεις εμφανίζουν μεγάλη διακύμανση γύρω από την τιμή σχεδιασμού, μέχρι την επανάληψη εκείνη που εντοπίζει την πρώτη εφικτή λύση. Από εκεί και μετά, ο αλγόριθμος αναζητάει «καλύτερες» εφικτές λύσεις σε σημεία κοντά στο υπάρχον τοπικό βέλτιστο, μειώνοντας έτσι συνεχώς την τιμή της αντικειμενικής συνάρτησης μέχρι να βρει την βέλτιστη τιμή και φυσικά ωσότου σταματήσει να «τρέχει» με βάσει τα κριτήρια τερματισμού (stopping criterion) που έχουν τεθεί. Συχνά όμως εγκλωβίζεται σε τοπικά βέλτιστα και λόγω της φύσης του δεν καταφέρνει να εντοπίσει το καθολικό βέλτιστο που αποτελεί και το ζητούμενο.

Σε αντίθεση με τα παραπάνω, κατά την βελτιστοποίηση με εφαρμογή του αλγορίθμου MATSuMoTo διαπιστώνεται ο καθαρά στοχαστικός χαρακτήρας του ως derivative-free μέθοδος. Σε όλες τις επαναλήψεις υπάρχει μεγάλη διακύμανση τιμών, τόσο της αντικειμενικής συνάρτησης όσο και των παραβιάσεων. Εντοπίζει πολύ πιο γρήγορα τη βέλτιστη τιμή η οποία όπως αποδείχθηκε τελικά είναι στις περισσότερες περιπτώσεις «καλύτερη» από την βέλτιστη τιμή που δίνουν οι άλλοι αλγόριθμοι.

Τέλος, κατά την βελτιστοποίηση με απλή εφαρμογή του αλγορίθμου Latin Hypercube, διαπιστώνεται ότι η, πρακτικά, μέθοδος τυχαίας δειγματοληψίας είναι ανεπαρκής από μόνη της στις περισσότερες περιπτώσεις, αφού δεν καταφέρνει να βρει κάποια εφικτή και άρα αποδεκτή λύση. Ο συνδυασμός, όμως, με άλλες αποτελεσματικές μεθόδους, όπως στην περίπτωση του αλγορίθμου MATSuMoTo, δίνει τελείως διαφορετικά αποτελέσματα.

Στον επόμενο πίνακα απεικονίζονται συνοπτικά τα ποσοστά βελτίωσης που πέτυχε κάθε αλγόριθμος ανά μοντέλο.

Πίνακας σύγκρισης ποσοστών βελτίωσης

	PQN Master	**MATSuMoTo**	**Latin Hypercube**
1ο Μοντέλο	**28,90%**	24,93%	-
2ο Μοντέλο	1,92%	**16,25%**	-
3ο Μοντέλο	**27,19%**	27,05%	-
4ο Μοντέλο	-16,82%	**3,66%**	-
5ο Μοντέλο	14,31%	20,20%	**26,75%**
6ο Μοντέλο	15,82%	**26,79%**	6,42%

HP-OCP: Dynamic Link Library

Guidelines

Version 1.0
March 2020

Contents

OCP: Dynamic Link Library - Documentation

Linking with the OCP_SolverDLL.dll. Three are the basic steps for linking with the function of OCP_SolverDLL.dll: (1) Definition of the parameters of the basic function, (2) calling the basic function and (3) per iteration communication with the analysis/design function.

Introduction

The HP-OCP Dynamic Link Library (DLL) is a powerful optimization tool that allows external users to link with third-party software, providing a path for two-way exchange of information with other programs.

Definition of the parameters of the basic function

The parameters of the basic function are distinguished in two types: (i) algorithm and (ii) formulation related ones.

The parameters related to the search algorithm that will be used for solving the problem at hand are provided in the "**opt_inp.dat**" file that is described below.

"**opt_inp.dat**" file:

Description

The algorithmic information related to the problem are denoted into the "**opt_inp.dat**" file, that is structured as follows:

Syntax

read: **iAlg** (integer)
read: **noCycles** (integer)
read: **maxFEA** (integer)
read: **improvementPercentage** (real*8)

File parameters

iAlg

The line #1 of the file denotes id of the algorithm.

noCycles

The line #2 of the file denotes the maximum number of consecutive cycles permitted. Convergence is achieved if improvement less than "**improvementPercentage**" is obtained for "**noCycles**" consecutive cycles.

maxFEA

The line #3 of the file denotes the maximum number of FE analyses permitted. If **maxFEA**=0.0 then the previous convergence criterion will be used, otherwise maximum number of FE analyses will be the criterion.

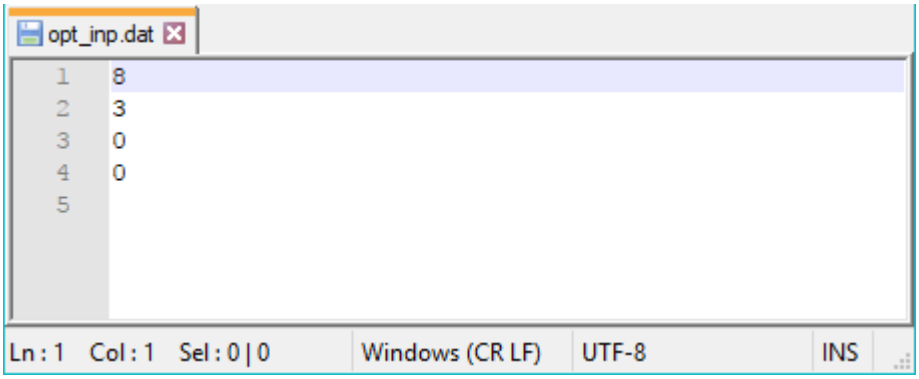
improvementPercentage

The line #4 of the file denotes the lowest percentage value of improvement of the objective function value.

Note

This file needs to be provided by the third-part software to the OCP solver.

Example



The parameters related to the problem formulation are provided in the "**bounds.dat**" file that is described below.

"**bounds.dat**" file:

Description

In order to provide the optimization problem formulation ones, the structural elements (beams, columns, slabs, shear walls etc.) should be divided into groups (in a similar fashion to the frame and area sections used in the CSi products). The information for all these groups is reported in the "**bounds.dat**" file as follows:

Syntax

read: **noElements** (integer)
for i=1:**noElements**
read: **Elements(i).id**, (character*255)
Elements(i).N_dvs, (integer)
(**Elements(i).up(j)**,j=1, Elements(i).N_dvs), (real*8)
(**Elements(i).lb(j)**,j=1,Elements(i).N_dvs), (real*8)
(**Elements(i).step(j)**,j=1,Elements(i).N_dvs), (real*8)
(**Elements(i).ifixed(j)**,j=1, Elements(i).N_dvs), (integer)
Elements(i).sectionType, (integer)
(**Elements(i).init(j)**,j=1,Elements(i).N_dvs) (real*8)
end

File parameters

noElements

The line #1 of the file denotes the number of section elements used for grouping the structural elements of the structure.

for each section element all the following parameters needs to be provided in one line (one next to the other) per section element:

Elements(i).id

This is the name of the ith section element.

Elements(i).N_dvs

This is the number of dimensions (design parameters) for the ith section element

(maximum number of design parameters is equal to 6).

Elements(i).up

This is an array containing the upper bounds of the design parameters for the ith section element.

Elements(i).lb

This is an array containing the lower bounds of the design parameters for the ith section element.

Elements(i).step

This is an array containing the changing step of the design parameters for the ith section element.

Elements(i).ifixed

This is an array denoting if any of the design parameters for the ith section element will remain fixed (**ifixed(j)**=1) (equal to the dimension of the original model) or not (**ifixed(j)**=0).

Elements(i).sectionType

This is an id denoting the section type of the ith section element, **sectionType** is equal to:

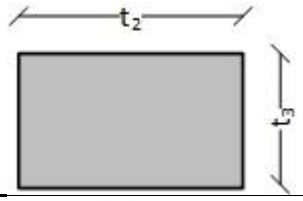
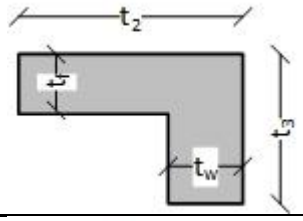
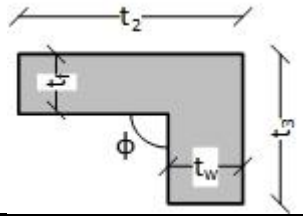
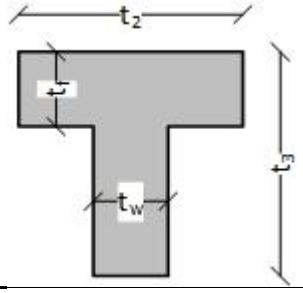
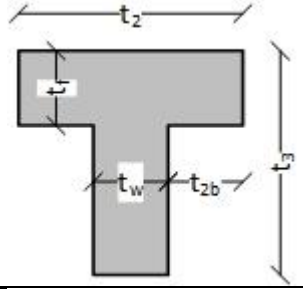
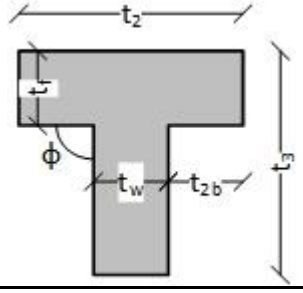
Name	Type	Name	Type	Name	Type	Name	Type	Name	Type
None	0	I-shape	1, 1001	Channel	2, 2001, 2002	SteelTee	3	SteelAngle	4
DoubleAngle	5	DoubleChannel	6	SteelPipe	7	SteelTube	8	AutoSelect	9
SteelJoist	10	HybridI	11	PlatedI	12	Rectangle	13	Circle	14
PrecastI	15	PrecastU	16	ColdC	17	ColdZ	18	ColdHat	19
General	20	NonPrismatic	21	SectionDesigner	22	TeeFoot	23	RectangleFoot	24
Shell	25	Plane	26	Asolid	27	WallAutoSelect	28	Wall	29
Slab	30	Deck	31	FilledSteelTube	50	FilledSteelPipe	51	BUICoverPlate	52
SteelPlate	53	SteelRod	54	EncasedRectangle	55	EncasedCircle	56	ConcreteCross	57
ConcreteTee	103, 1031, 1033	ConcreteL	104, 1041	ConcretePipe	107	ConcreteBox	108, 801	Cable	201
Tendon	301	Zhta	60, 601	Triangle	70				

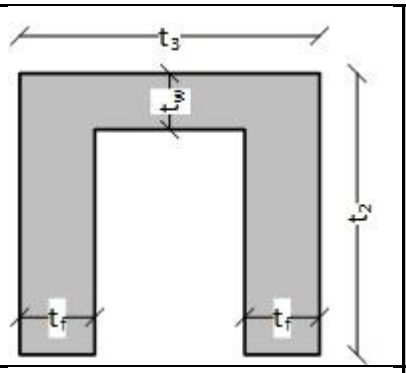
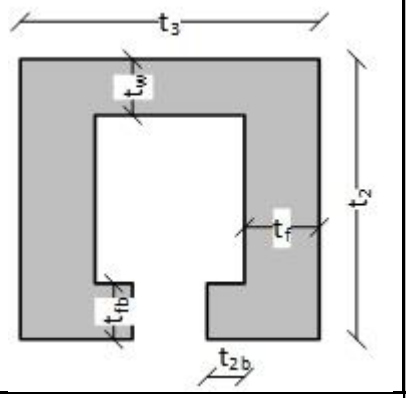
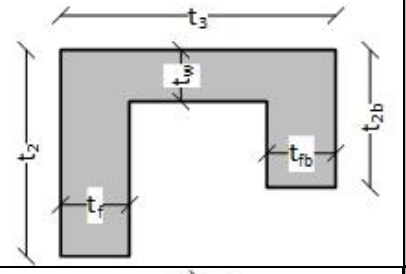
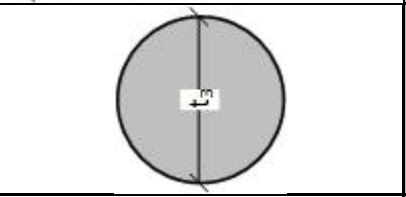
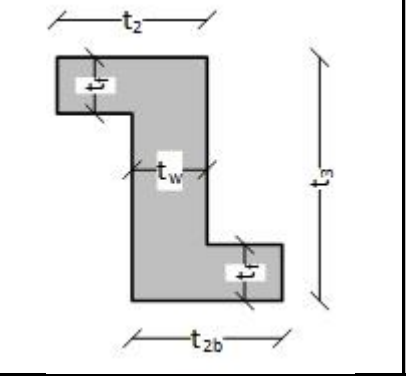
Note1

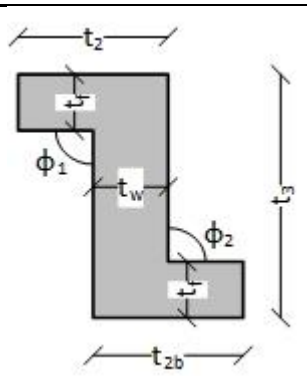
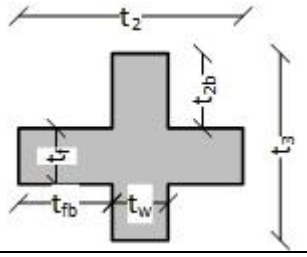
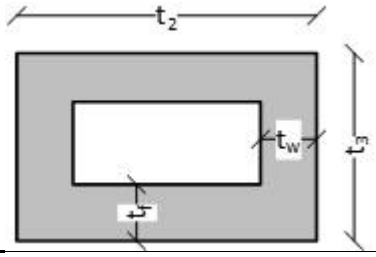
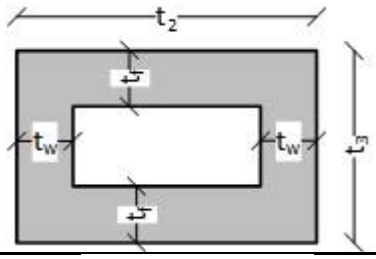
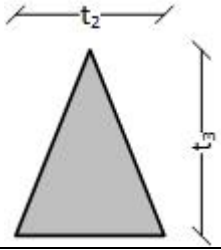
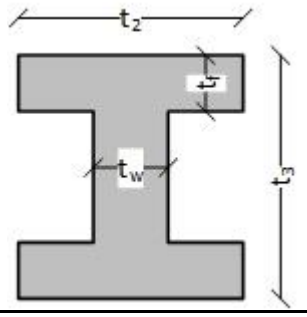
The order of the cross-sectional dimensions are specific:

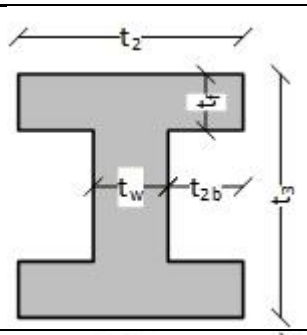
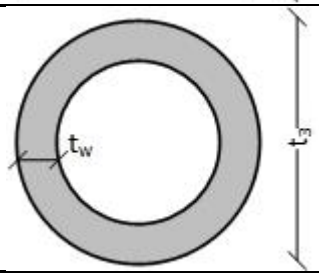
Order:	1st	2nd	3rd	4th	5th	6th
Dimension:	t_3	t_2	t_f	t_w	t_{2b}	t_{fb}

The graphical representation of the cross-section denoted in the following table depict the location of the above mentioned dimensions.

Name	Type	Figure	Note
Rectangular	13		
Angle (SteelAngle, ConcreteL)	4, 104		
Angle2 (ConcreteL)	1041		φ is fixed
Tee (SteelTee, ConcreteTee)	3, 103		Equal legs
Tee2 (ConcreteTee)	1031		Unequal legs
Tee3 (ConcreteTee)	1033		φ is fixed

Channel	2		
Channel2	2001		
Channel3	2002		φ_1 and φ_2 are fixed
Circle	14		
Zhta	60		

Zhta2	601		φ1 and φ2 are fixed
Cross (ConcreteCross)	57		
Tube (SteelTube, ConcreteBox)	8, 108		
Tube2 (ConcreteBox)	108, 801		
Triangle	70		
I-shape	1		Equal legs

I-shape2	1001		Unequal legs
Pile (SteelPipe, ConcretePipe)	7, 107		
Select out of a list (AutoSelect, WallAutoSelect)	9, 28		Elements(i).lb(j) = 1, Elements(i).up(j) = Number of elements in the list

Elements(i).init

This is an array containing the initial values of the design parameters for the ith section element.

Note2

This file needs to be provided by the third-part software to the OCP solver.

Example

bounds.dat

```
10
'Angle_a' 2 65 45 45 25 5 5 0 0 13 60 25
'Angle_b' 2 65 45 45 25 5 5 0 0 13 60 25
'Angle_c' 2 65 45 45 25 5 5 0 0 13 60 25
'Tee_a' 2 65 45 45 25 5 5 0 0 13 60 25
'Tee_b' 2 65 45 45 25 5 5 0 0 13 60 25
'TeeFoot_a' 4 110 135 50 35 80 100 35 25 5 5 0.5 0.5 0 0 0 0 3 110 135 50 35
'TeeFoot_b' 4 110 135 50 35 80 100 35 25 5 5 0.5 0.5 0 0 0 0 3 110 132.5 50 32.5
'Rectangle_a' 2 220 60 80 30 5 5 0 0 13 200 30
'Rectangle_b' 2 80 60 35 20 5 5 0 0 13 60 30
'Base' 1 50 20 1 0 25 50
```

length : 492 lines : 13 | Ln : 1 Col : 3 Sel : 0 | 0 | Windows (CR LF) | UTF-8 | INS

"**obj_funcs.dat**" file:

Description

Part of the formulation of the optimization problem is to select the objective functions and the corresponding weight coefficients along with the corresponding flag denoting minimization or maximization. This information needs to be reported in the "**obj_funcs.out**" file as follows:

Syntax

for i=1:**noObjFunctions** (integer)
read: **objFunct(i).weight**, (real*8)
objFunct(i).flag, (logical)
end

File parameters

noObjFunctions

This variable denotes the number of different objective functions that the solver currently handles (**noObjFunctions**=9).

For each objective function the following information needs to be provided:

objFunct(i).weight

Is the weight coefficient for the ith objective function. NOTE: the sum of the weight coefficients over all the objective functions needs to be equal to 1.0. Where indicatively if i is equal to:
1: Material Cost
2: Construction Cost
3: Life Cycle CO2 Emission
4: Life Cycle Energy
5: Stiffness Eccentricity
6: Strength Eccentricity
7: Compliance
8: Drift Coefficient of Variation
9: First Eigen Frequancy
The above mentioned objective functions are indicative, additional ones can also be used.

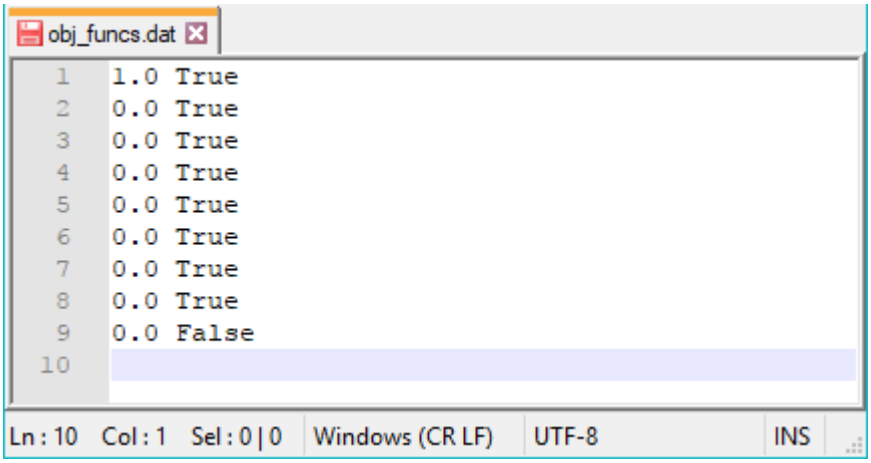
objFunct(i).flag

Is a logical flag denoting if the ith objective function is to be minimized or maximized (True-> min, False-> max).

Note

This file needs to be provided by the third-part software to the OCP solver.

Example


obj_funcs.dat
1 1.0 True
2 0.0 True
3 0.0 True
4 0.0 True
5 0.0 True
6 0.0 True
7 0.0 True
8 0.0 True
9 0.0 False
10
Ln : 10 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

Calling the basic function

When calling the basic function (**OCP_SOLVERDLL**) five input arguments needs to be provided:

(i) software id (an integer variable denoting the id that it will be agreed that corresponds to your software),

(ii) the path denoting where the original model is located (a string variable of maximum length 1024 characters, [character*1024]),

(iii) the name of your analysis and design function,

the rest two input arguments are used for the CSi products (both are string variables of maximum length 1024 characters, [character*1024]).

Description

This is the main function of the library.

Syntax

subroutine OCP_SOLVERDLL(**idSoftwareInp**, (integer)
string1Inp, (character*1024)
funcInp, (pointer)
string2Inp, (character*1024)
string3Inp) (character*1024)

File parameters

idSoftwareInp

Third-party software id.

string1Inp

The path denoting where the original model is located.

funcInp

Third-party software function.

string2Inp

Input argument are used for the CSi products only.

string3Inp

Input argument are used for the CSi products only.

Example

for the case of a C++ implementation:

```
HINSTANCE HandleFortran = ::LoadLibrary(L"OCP_SolverDLL.dll");
if (HandleFortran){
   typedef void (* TRun_OCP_Auto_Anlysis_Design)(int *, char *, void
   (*Run_OCP_Auto_Anlysis_Design)(), char *, char *);
   TRun_OCP_Auto_Anlysis_Design Run_OCP_Auto_Anlysis_Design;
   Run_OCP_Auto_Anlysis_Design =
   (TRun_OCP_Auto_Anlysis_Design)GetProcAddress(HandleFortran,
```

```
    "OCP_SOLVERDLL");
    if (Run_OCP_Auto_Anlysis_Design){
      Run_OCP_Auto_Anlysis_Design(&idSoftware,namefile,
      Function_Run_OCP_Auto_Anlysis_Design,namefile,namefile);
    }
    FreeLibrary(HandleFortran);
    HandleFortran=NULL;
}
```

Per iteration communication with the analysis/design function

The following parts are required in order to perform the iterative runs:

1) a subroutine needs to be prepared that when called will perform the following:
- Create a new design, based on the information provided in the "**diastaseis.dat**" file, more information will be provided below.
- Run analysis, in order to calculate element forces and deformation over the loading combinations that were assigned to the original model.
- Perform all the design checks (according to the design codes that were initially assigned to the original model).
- Calculate the criteria (cost and/or performance ones) and
- Calculate the maximum strength ration: (Demand over capacity) for all checks required by the design code referred previously.

2) Communicate with the subroutine referred to part (1):
- Definition of the new design, the new design that is generated by the OCP_solver is provided in the "**diastaseis.dat**" file, in the following form:

"**diastaseis.txt**" file:

Description

In order to provide the optimization problem formulation ones, the structural elements (beams, columns, slabs, shear walls etc.) should be divided into groups (in a similar fashion to the frame and area sections used in the CSi products). The information for all these groups is reported in the "**diastaseis.txt**" file as follows:

Syntax

```
write: noElements (integer)
for i=1:noElements
        write: Elements(i).id, (character*255)
        for i=1:Elements(i).N_dvs
                write: Elements(i).value(j) (real*8)
        end
end
```

File parameters

noElements

The line #1 of the file denotes the number of section elements used for grouping the structural elements of the structure.

for each section element the following parameters needs to be provided in one line per section element:

Elements(i).id

This is the name of the ith section element.

for the ith section element the values of the design parameters needs to be provided in one

line per dimension:

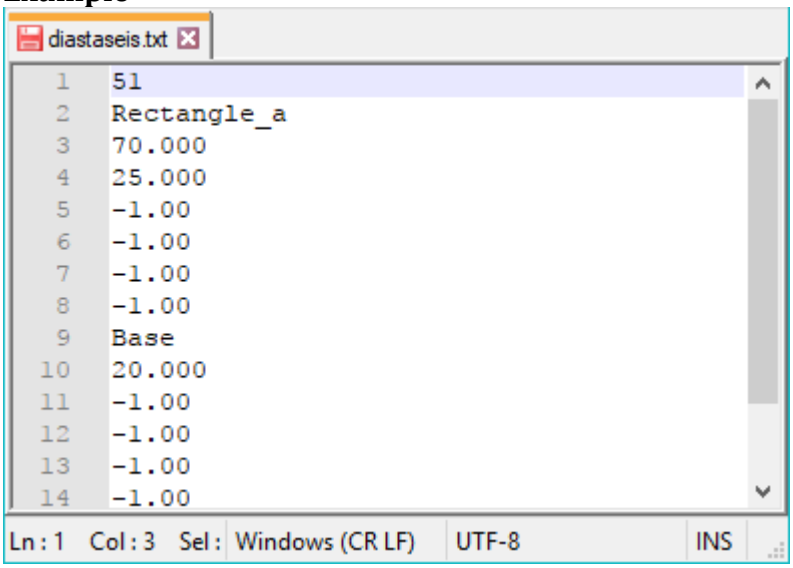
Elements(i).value

This is an array containing the current values of the design parameters for the ith section element. If **N_dvs** less than 6, the remaining components of the array **value** are set equal to -1.

Note

This file is generated by the OCP solver every time a new design is generated and needs to be assessed.

Example



- Assessment of the new design, in order to assess the new designs that are generated through the iterative procedure the function of your software that performs the analysis and design steps needs to generate the "**assess.dat**" file, in the following form.

"**assess.txt**" file:

Description

The information related to the assessment of the current design needs to be written in the "**assess.txt**" file as follows:

Syntax

for i=1:**noObjFunctions** (integer)
read: **objFunct(i)** (real*8)
end
read: **percentViolation** (real*8)

File parameters

noObjFunctions

This variable denotes the number of different objective functions that the solver currently handles (**noObjFunctions**=9).

objFunct

This is an array containing the values of the various objective functions that the solver currently handles for the current design, where:

1: Material Cost
2: Construction Cost
3: Life Cycle CO2 Emission
4: Life Cycle Energy
5: Stiffness Eccentricity
6: Strength Eccentricity
7: Compliance
8: Drift Coefficient of Variation
9: First Eigen Frequancy

The above mentioned objective functions are indicative, additional ones can also be used.

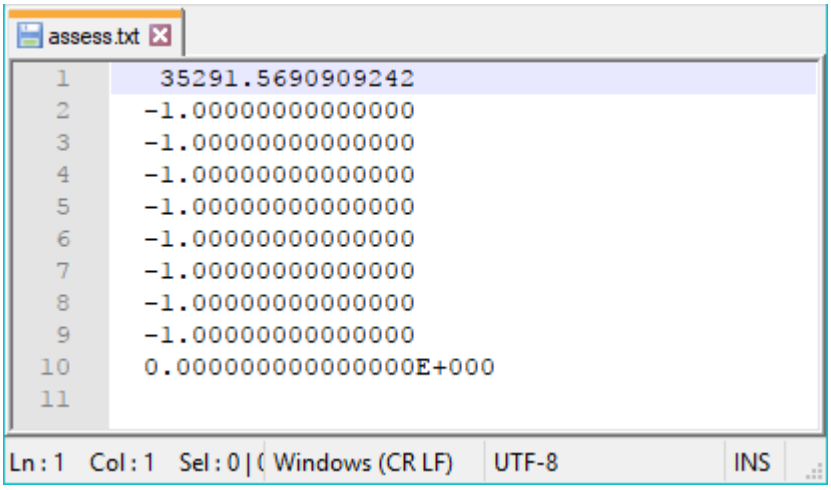
percentViolation

This is the value of maximum percentage of violation for the current design. If **percentViolation**=0.0 then the current design satisfies all the design requirements, otherwise it should be greater than 1.0.

Note

Needs to be provided by the function of the third-party software to the OCP solver for every new design that is assessed.

Example



"**ahistory.out**" file:

Description

The optimization history with respect the objective function value and the maximum

percentage of constraints violation is written in the "**ahistory.out**" file as follows:

Syntax

write: **objFunction** (real*8),
percentViolation (real*8)

File parameters

for each FE analysis performed during the optimization procedure the following information is written in this file:

objFunction

This is the value of the objective function for the current design.

percentViolation

This is the value of maximum percentage of violation for the current design. If **percentViolation**=0.0 then the current design satisfies all the design requirements, otherwise it should be greater than 1.0.

Note

This file is generated by the OCP solver.

Example

ahistory.out

1	76160.4378442561	0.000000000000000E+000
2	29642.8895919963	2.00000000000000
3	46404.8058600176	0.000000000000000E+000
4	45858.1204455459	0.000000000000000E+000
5	43398.9642131185	0.000000000000000E+000
6	42247.9736623608	0.000000000000000E+000
7	35942.8390793223	0.000000000000000E+000
8	36006.1495582108	0.000000000000000E+000
9	36083.3025167056	0.000000000000000E+000
10	35456.8667042658	0.000000000000000E+000
11	35928.7978742712	0.000000000000000E+000
12	35291.5690909242	0.000000000000000E+000
13	35664.2380707653	0.000000000000000E+000
14	35693.3662988720	0.000000000000000E+000
15	36079.3826778642	0.000000000000000E+000
16	35926.4492721545	0.000000000000000E+000
17	36027.1553028158	0.000000000000000E+000
18	39334.7450876357	0.000000000000000E+000
19	35291.5690909242	0.000000000000000E+000
20		

Ln : 12 Col : 2 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

"equal.dat" file:

Description

In order to provide the optimization problem formulation ones, the structural elements (beams, columns, slabs, shear walls etc.) should be divided into groups (in a similar fashion to the frame and area sections used in the CSi products). The information for all these groups is reported in the "**equal.dat**" file as follows:

Syntax

read: **iAlg** (integer)
read: **noCycles** (integer)
read: **maxFEA** (integer)
read: **improvementPercentage** (real*8)

File parameters

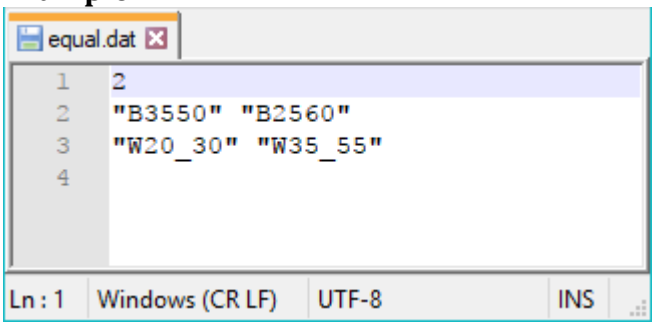
iAlg

The line #1 of the file denotes id of the algorithm.

Note

Needs to be provided to the OCP solver.

Example



"errorlog_for.out" file:

Description

In order to provide the optimization problem formulation ones, the structural elements (beams, columns, slabs, shear walls etc.) should be divided into groups (in a similar fashion to the frame and area sections used in the CSi products). The information for all these groups is reported in the "**errorlog_for.out**" file as follows:

Syntax

read: **iAlg** (integer)
read: **noCycles** (integer)
read: **maxFEA** (integer)
read: **improvementPercentage** (real*8)

File parameters

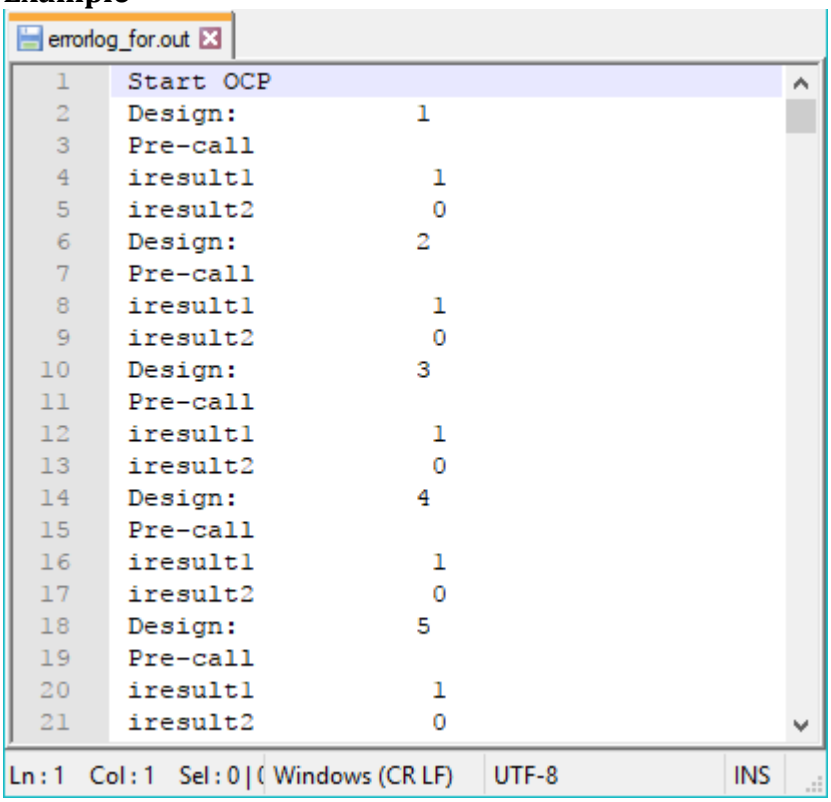
iAlg

The line #1 of the file denotes id of the algorithm.

Note

This file is generated by the OCP solver.

Example



"out_cost.dat" file:

Description

In order to provide the optimization problem formulation ones, the structural elements (beams, columns, slabs, shear walls etc.) should be divided into groups (in a similar fashion to the frame and area sections used in the CSi products). The information for all these groups is reported in the "**out_cost.dat**" file as follows:

Syntax

read: **iAlg** (integer)
read: **noCycles** (integer)
read: **maxFEA** (integer)
read: **improvementPercentage** (real*8)

File parameters

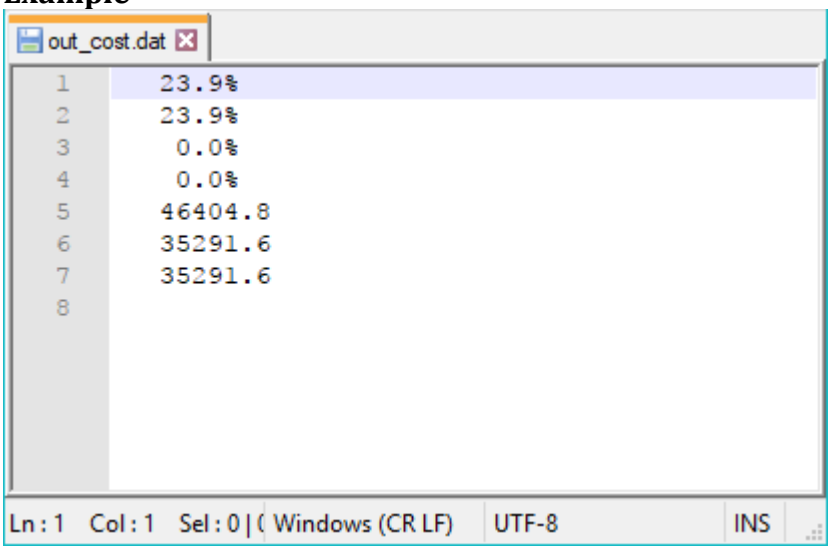
iAlg

The line #1 of the file denotes id of the algorithm.

Note

This file is generated by the OCP solver.

Example



"**stop.dat**" file:

Description

This file is used for stopping the search procedure before convergence. This information is reported in the "**stop.dat**" file as follows:

Syntax

read: **stopFlag** (integer)

File parameters

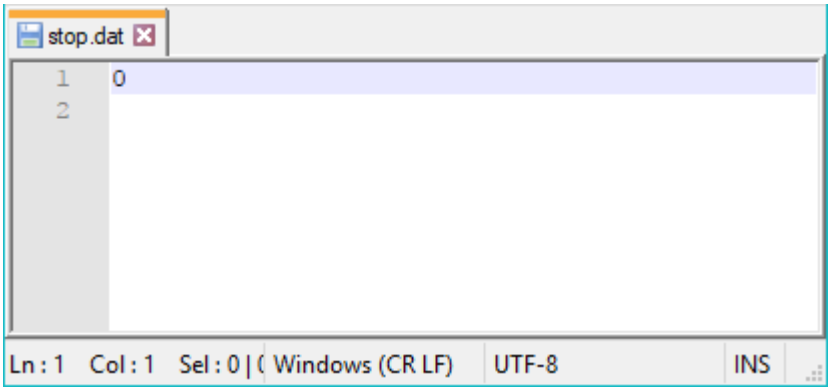
stopFlag

The default value of this flag variable is 0, if **percentViolation**=-1 then the search procedure will be stopped.

Note

Needs to be provided to the OCP solver.

Example


stop.dat
1
0
2
Ln : 1 Col : 1 Sel : 0 | (Windows (CR LF) UTF-8 INS

Security

NOTE: Part of the search procedure is performed online, therefore a licensed version of the OCP_SolverDLL.dll needs to be used in order to work properly.

Σχόλια - Προβλήματα - Παρατηρήσεις

Δεν υπήρχαν παρατηρήσεις

	Επιστημονικός Υπεύθυνος Έργου	Συντονιστής Έργου
Υπογραφή:		
Ονοματεπώνυμο :	Ν. ΛΑΓΑΡΟΣ	Χ. ΚΩΣΤΟΠΑΝΑΓΙΩΤΗΣ
Ημ/νία :	10/09/2020	10/09/2020