

Πολυτεχνείο Κρήτης  
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

---

## Διπλωματική Εργασία

---

«Εφαρμογές Προσομοιωμένης Ανόπτησης στην Εκτίμηση Στατιστικών  
Μοντέλων»



Ευάγγελος Εμμ. Καστρινάκης  
Α.Μ.: 2017030052

Εξεταστική Επιτροπή:

Καθ. Διονύσιος Χριστόπουλος (Επιβλέπων)

Καθ. Αθανάσιος Λιάβας

Καθ. Μιχαήλ Λαγουδάκης

Χανιά, Οκτώβριος 2024



Technical University of Crete  
School of Electrical and Computer Engineering

---

**Diploma Thesis**

---

**«Applications of Simulated Annealing in Statistical Model Estimation»**



**Evangelos Emm. Kastrinakis**  
**Student ID: 2017030052**

**Examination Committee:**

Prof. Dionissios Christopoulos (Supervisor)

Prof. Athanasios Liavas

Prof. Michail Lagoudakis

**Chania, October 2024**



# Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται τη μελέτη πολύπλοκων προβλημάτων καθολικής βελτιστοποίησης. Η καθολική βελτιστοποίηση είναι θέμα μείζονος σημασίας σε διάφορους τομείς, όπως σε πολλούς κλάδους της μηχανικής και των επιστημών. Η επίλυση πολύπλοκων προβλημάτων με πολλαπλά τοπικά ακρότατα καθίσταται ιδιαίτερα δύσκολη, όταν χρησιμοποιούνται συμβατικές μέθοδοι τοπικής βελτιστοποίησης. Αντίθετα, οι μέθοδοι καθολικής βελτιστοποίησης μπορούν να αντιμετωπίσουν προβλήματα σύγκλισης και οδηγούν σε βελτίωση απόδοσης. Ειδικότερα, η διπλωματική εργασία εξετάζει τις μεθόδους της προσομοιωμένης ανόπτησης και της καθολικής αναζήτησης (Matlab GlobalSearch) σε δύο διακριτά προβλήματα. Το πρώτο πρόβλημα αναφέρεται στην εύρεση καθολικού βέλτιστου σε συναρτήσεις ελέγχου υψηλών διαστάσεων (σε χώρους 10 και 30 διαστάσεων) με πολλαπλά τοπικά ακρότατα. Το δεύτερο πρόβλημα περιλαμβάνει τη μεγιστοποίηση της πιθανοφάνειας ενός συνόλου χωρικών δεδομένων (δείγμα χωρικής στοχαστικής διαδικασίας) ως προς τις παραμέτρους του χωρικού στοχαστικού μοντέλου τοπικών αλληλεπιδράσεων. Εξετάζονται τόσο συνθετικά όσο και πραγματικά χωρικά δεδομένα (πάχος στρωμάτων άνθρακα). Σε αυτήν την περίπτωση χρησιμοποιείται ένας παραμετρικός χώρος τεσσάρων διαστάσεων.

Ο σκοπός της παρούσας μελέτης είναι η εξέταση της απόδοσης των παραπάνω μεθόδων βελτιστοποίησης ως προς την ακρίβεια εντοπισμού του καθολικού βέλτιστου (όπου μπορεί να υπολογιστεί) και ως προς τον υπολογιστικό χρόνο. Παρουσιάζονται τα αποτελέσματα της εφαρμογής των δύο τεχνικών βελτιστοποίησης στα δύο προβλήματα, δηλαδή στις συναρτήσεις ελέγχου και στην πιθανοφάνεια του στοχαστικού μοντέλου τοπικών αλληλεπιδράσεων. Πιο συγκεκριμένα, η σύγκριση αφορά στον υπολογιστικό χρόνο βελτιστοποίησης, την ακρίβεια της επιτυγχανόμενης λύσης, και την ευαισθησία της κάθε μεθόδου σε διάφορες παραμετροποιήσεις. Από τη βελτιστοποίηση των συναρτήσεων ελέγχου διακρίνεται ότι η μέθοδος καθολικής αναζήτησης είναι αποδοτικότερη ως προς τον υπολογιστικό χρόνο και εμφανίζει μεγαλύτερη συνέπεια στην εύρεση του καθολικού βέλτιστου ανεξάρτητα από την παραμετροποίηση. Όσον αφορά στη βελτιστοποίηση της πιθανοφάνειας των χωρικών δεδομένων,

παρατηρείται ότι η μέθοδος προσομοιωμένης ανόπτησης είναι αποτελεσματικότερη ως προς τον υπολογιστικό χρόνο. Για τα χωρικά συνθετικά δεδομένα, και οι δύο μέθοδοι επιτυγχάνουν παρόμοιες λύσεις για τη μεγιστοποίηση της πιθανοφάνειας. Για τα δεδομένα πάχους στρωμάτων άνθρακα, η καθολική αναζήτηση δεν συγκλίνει σε περίοδο μεγαλύτερη των δύο ωρών. Στο πρόβλημα της πιθανοφάνειας, καμία από τις δύο μεθόδους δεν επιτυγχάνει σημαντική βελτίωση συγκριτικά με τα αποτελέσματα της τοπικής βελτιστοποίησης.

#### **Λέξεις κλειδιά**

Καθολική Βελτιστοποίηση, Προσομοιωμένη Ανόπτηση, Καθολική Αναζήτηση, Συναρτήσεις ελέγχου, Μεγιστοποίηση Πιθανοφάνειας, Σύνολα χωρικών δεδομένων (πραγματικά και συνθετικά), Στοχαστικές τοπικές αλληλεπιδράσεις

# ABSTRACT

This diploma thesis deals with the study of complex global optimization problems. Global optimization is a subject of major importance in various fields in science and engineering. Solving complex optimization problems with multiple local extrema is particularly difficult, when using conventional methods of local optimization. On the other hand, global optimization methods can address convergence problems and lead to improved performance. Specifically, the thesis examines the methods of simulated annealing and global search (Matlab GlobalSearch) in the solution of two distinct problems. The first problem refers to finding the global optimum of high-dimensional control functions (in 10 and 30 dimensional spaces) with multiple local extrema. The second problem involves maximizing the likelihood of a set of spatial data (a sample of a spatial stochastic process) in terms of the parameters of the so-called stochastic local interaction model. Both synthetic and real (coal thickness) spatial data are examined. A four-dimensional parametric space is used in this latter case.

The present study aims to examine the performance of the above optimization methods in terms of global optimum localization accuracy (where applicable) and computational time. The results of applying the two optimization techniques to the two problems (i.e., the control functions and the likelihood of the stochastic local interactions model) are presented. More specifically, the comparison concerns the computational optimization time, the accuracy of the achieved solution, and the sensitivity of each method to various parameterizations. Regarding the optimization of the control functions, it is evident that the global search method is more efficient in terms of computational time and finds the global optimum more consistently regardless of the parameterization. Regarding the optimization of the likelihood, it is observed that the simulated annealing is more efficient in terms of computational time. For the synthetic spatial data, both methods achieve similar solutions for maximum likelihood. For the coal thickness data, global search does not converge over a period longer than two hours. However, for the spatial datasets neither method significantly improves the likelihood compared to local optimization.

## Keywords

Global Optimization, Simulated Annealing, Global Search, Control functions, Maximization of likelihood, spatial datasets (real and synthetic), Stochastic Local Interaction



# Ευχαριστίες

Κατ' αρχήν, θα ήθελα να ευχαριστήσω και να εκφράσω την βαθύτατη ευγνωμοσύνη μου στον επιβλέποντα καθηγητή μου κύριο Διονύσιο Χριστόπουλο για την καθοδήγηση και υποστήριξή του καθ' όλη τη διάρκεια της ερευνητικής εργασίας. Η ολοκλήρωση της εργασίας δεν θα ήταν δυνατή, χωρίς τις γνώσεις και την εμπειρία του. Επιπλέον, θα ήθελα να ευχαριστήσω τα υπόλοιπα μέλη της εξεταστικής επιτροπής, τον καθηγητή Αθανάσιο Λιάβα και καθηγητή Μιχαήλ Λαγουδάκη, για τη συμμετοχή τους και το χρόνο που αφιέρωσαν για την ανάγνωση της εργασίας. Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια, τους φίλους και συμφοιτητές μου για την αδιάλειπτη υποστήριξη και ενθάρρυνσή τους καθ' όλη τη διάρκεια των σπουδών μου.



# Περιεχόμενα

Περίληψη	i
ABSTRACT	iii
1 Εισαγωγή	1
2 Μέθοδος Προσομοιωμένης Ανόπτησης (ΠΑ)	5
2.1 Εισαγωγή	5
2.2 Η περιγραφή της μεθόδου ΠΑ	7
2.3 Παράμετροι ελέγχου	10
3 Εφαρμογή Μεθόδου ΠΑ σε Συναρτήσεις Ελέγχου	15
3.1 Περιγραφή της υβριδικής προσέγγισης	15
3.2 Συνάρτηση Ackley	17
3.2.1 Αποτελέσματα σε χώρο 10 διαστάσεων	18
3.2.2 Αποτελέσματα σε χώρο 30 διαστάσεων	28
3.3 Συνάρτηση Cross-in-Tray	34
3.3.1 Αποτελέσματα σε χώρο 10 διαστάσεων	35
3.3.2 Αποτελέσματα σε χώρο 30 διαστάσεων	42
4 Μεγιστοποίηση Συνάρτησης Πιθανοφάνειας με Εφαρμογή ΠΑ	49
4.1 Περιγραφή του προβλήματος	49
4.1.1 Αποτελέσματα για το μοντέλο SLI και χωρικά συνθετικά δεδομένα	51

4.1.2	Αποτελέσματα για το μοντέλο SLI και χωρικά πραγματικά δεδομένα	59
<b>5</b>	<b>Μέθοδος Καθολικής Αναζήτησης (Matlab GlobalSearch)</b>	<b>69</b>
5.1	Εισαγωγή	69
5.2	Η περιγραφή της μεθόδου GS	71
5.2.1	Η περιγραφή της μεθόδου SS	75
5.3	Παράμετροι ελέγχου της μεθόδου GS	78
<b>6</b>	<b>Εφαρμογή Καθολικής Αναζήτησης σε Συναρτήσεις Ελέγχου</b>	<b>83</b>
6.1	Παράμετροι ελέγχου της μεθόδου τοπικής βελτιστοποίησης fmincon	83
6.2	Συνάρτηση Ackley σε χώρο 10 διαστάσεων	83
6.3	Συνάρτηση Ackley σε χώρο 30 διαστάσεων	88
6.4	Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων	89
6.5	Συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων	93
<b>7</b>	<b>Μεγιστοποίηση Συνάρτησης Πιθανοφάνειας με Εφαρμογή Καθολικής Αναζήτησης</b>	<b>97</b>
7.1	Παράμετροι ελέγχου της μεθόδου τοπικής βελτιστοποίησης fmincon	97
7.2	Περιγραφή του προβλήματος	97
7.2.1	Αποτελέσματα για το μοντέλο SLI και χωρικά συνθετικά δεδομένα	97
7.2.2	Αποτελέσματα για το μοντέλο SLI και χωρικά πραγματικά δεδομένα	104
<b>8</b>	<b>Συμπεράσματα</b>	<b>105</b>
	<b>Βιβλιογραφία</b>	<b>107</b>
	<b>Παράρτημα Α' Κώδικας Matlab</b>	<b>113</b>
A'.1	Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων	114
A'.2	Καθολική αναζήτηση για τους δύο τύπους προβλημάτων	144
A'.3	Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας	159

# Κατάλογος Σχημάτων

2.1	Διάγραμμα ροής του αλγόριθμου ΠΑ . . . . .	9
3.1	Η αναπαράσταση της συνάρτησης Ackley σε χώρο δύο διαστάσεων ( $d = 2$ ). . . . .	17
3.2	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	18
3.3	Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	20
3.4	Η εξέλιξη της τιμής συνάρτησης κόστους σε λογαριθμική κλίμακα (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο $\log$ για καλύτερα οπτικά αποτελέσματα. . . . .	21
3.5	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. . . . .	22
3.6	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. . . . .	23
3.7	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . Η <code>fmincon</code> εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	24

3.8	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . . . . .	25
3.9	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . . . . .	26
3.10	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. Η κόκκινη οριζόντια γραμμή αντιστοιχεί στην τιμή συνάρτησης κόστους του σημείου που βρίσκεται το καθολικό ελάχιστο. . . . .	28
3.11	Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	30
3.12	Η εξέλιξη της τιμής συνάρτησης κόστους σε λογαριθμική κλίμακα (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο $\log$ για καλύτερα οπτικά αποτελέσματα. . . . .	31
3.13	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . Η <code>fmincon</code> εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	32
3.14	Η αναπαράσταση της συνάρτησης Cross-in-Tray σε χώρο δύο διαστάσεων ( $n = 2$ ). . . . .	34
3.15	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	35
3.16	Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	37

3.17 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. . . . .	38
3.18 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. . . . .	38
3.19 Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. Η patternsearch εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	39
3.20 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. .	40
3.21 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. .	41
3.22 Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	43
3.23 Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. . . . .	45
3.24 Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. Η patternsearch εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	46
4.1 Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με $3.52864 \cdot 10^3$ και λαμβάνεται στην 1,700η επανάληψη. . . . .	52

4.2	Η εξέλιξη της τιμής NLL (σε λογαριθμική κλίμακα) στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο $\log$ για καλύτερα οπτικά αποτελέσματα. . . . .	53
4.3	Η εξέλιξη της τιμής παραμέτρου $m$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	54
4.4	Η εξέλιξη της τιμής παραμέτρου $\lambda$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	55
4.5	Η εξέλιξη της τιμής παραμέτρου $c1$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	56
4.6	Η εξέλιξη της τιμής παραμέτρου $\mu$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	57
4.7	Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . Η <code>fmincon</code> εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	58
4.8	Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με $2.29545 \cdot 10^4$ και λαμβάνεται στην 631η επανάληψη. . . . .	59
4.9	Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Η εικόνα είναι μεγενθυμένη για καλύτερη οπτική παρατήρηση των συμβάντων επανόπτησης. . . . .	60
4.10	Η εξέλιξη της τιμής NLL (σε λογαριθμική κλίμακα) στα χωρικά πραγματικά δεδομένα συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο $\log$ για καλύτερα οπτικά αποτελέσματα. . . . .	61



4.11 Η εξέλιξη της τιμής παραμέτρου $m$ του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	62
4.12 Η εξέλιξη της τιμής παραμέτρου $\lambda$ του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	63
4.13 Η εξέλιξη της τιμής παραμέτρου $c1$ του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	64
4.14 Η εξέλιξη της τιμής παραμέτρου $\mu$ του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . . . .	65
4.15 Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου <code>fmincon</code> . Η <code>fmincon</code> εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ. . . . .	66
5.1 Διάγραμμα ροής του αλγόριθμου SS . . . . .	77
5.2 Ο κύριος βρόγχος της διαδικασίας του αλγόριθμου SS και πως κάθε βήμα του αλγόριθμου διαχειρίζεται το σύνολο των σημείων. . . . .	78
6.1 Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης ( <code>fmincon</code> ). . . . .	84
6.2 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. . . . .	86
6.3 Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. . . . .	86

6.4	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon). . . . .	88
6.5	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon). . . . .	90
6.6	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. . . . .	92
6.7	Η εξέλιξη της μεταβλητής $x$ σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. . . . .	92
6.8	Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon). . . .	94
7.1	Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon). Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με $3.5285127 \cdot 10^3$ και λαμβάνεται στην 735η επανάληψη. . .	98
7.2	Η εξέλιξη της τιμής παραμέτρου $m$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . .	100
7.3	Η εξέλιξη της τιμής παραμέτρου $\lambda$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . .	101
7.4	Η εξέλιξη της τιμής παραμέτρου $c1$ του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . .	102

- 7.5 Η εξέλιξη της τιμής παραμέτρου  $\mu$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτῆσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL. . 103



# Κατάλογος Πινάκων

3.1	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων . . . . .	27
3.2	Παράμετροι ελέγχου της τοπικής μεθόδου fmincon για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων . . . . .	27
3.3	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων . . . . .	33
3.4	Παράμετροι ελέγχου της τοπικής μεθόδου fmincon για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων . . . . .	33
3.5	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Cross-In-Tray στις 10 διαστάσεις . . . . .	42
3.6	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για την συνάρτηση Cross-In-Tray στις 30 διαστάσεις . . . . .	47
4.1	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τα χωρικά συνθετικά δεδομένα . . . . .	59
4.2	Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τα χωρικά πραγματικά δεδομένα . . . . .	67
6.1	Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων . . . . .	87
6.2	Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης fmincon για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων . . . . .	87
6.3	Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων . . . . .	89
6.4	Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης fmincon για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων . . . . .	89

6.5	Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων . . . . .	93
6.6	Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων . . . . .	95
7.1	Παράμετροι ελέγχου του αλγόριθμου GS για τα χωρικά συνθετικά δεδομένα . . . . .	104
7.2	Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης fmincon για τα χωρικά συνθετικά δεδομένα . . . . .	104

# Κεφάλαιο 1

## Εισαγωγή

Τα προβλήματα βελτιστοποίησης αποτελούν ερευνητικό αντικείμενο για μεγάλο χρονικό διάστημα. Σε διάφορους τομείς, όπως της επιστήμης και μηχανικής, παρουσιάζονται σύνθετα προβλήματα που περιλαμβάνουν πολλαπλά τοπικά ακρότατα και διακρίνονται από παραμετρικούς χώρους υψηλών διαστάσεων. Οι μέθοδοι τοπικής βελτιστοποίησης στα συγκεκριμένα προβλήματα δεν είναι αποτελεσματικοί, διότι καταλήγουν σε κάποιο τοπικό ακρότατο, μακριά από τη γειτονία του καθολικού βέλτιστου [1] [2]. Για την αντιμετώπιση των συγκεκριμένων προβλημάτων υπάρχουν πολλές μέθοδοι καθολικής βελτιστοποίησης. Στην παρούσα εργασία, χρησιμοποιούνται οι μέθοδοι της προσομοιωμένης ανόπτωσης (ΠΑ) και καθολικής αναζήτησης-Matlab GlobalSearch (GS) σε δύο διακριτά προβλήματα. Το πρώτο πρόβλημα αναφέρεται στην εύρεση καθολικού βέλτιστου σε συναρτήσεις ελέγχου υψηλών διαστάσεων (χώροι 10 και 30 διαστάσεων) με πολλαπλά τοπικά ακρότατα. Αναλυτικότερα, οι συναρτήσεις ελέγχου που ελαχιστοποιούνται είναι η Ackley και Cross-In-Tray. Ο σκοπός της μελέτης είναι να εξεταστεί η αποτελεσματικότητα των δύο μεθόδων σε αυτά τα πολύπλοκα τοπία ως προς την ακρίβεια εντοπισμού του καθολικού βέλτιστου (όπου μπορεί να υπολογιστεί) και του υπολογιστικού χρόνου.

Η ΠΑ είναι πιθανολογική, μετά-ευρετική μέθοδος τυχαίας αναζήτησης που είναι ιδιαίτερα αποτελεσματική σε προβλήματα με μεγάλους χώρους αναζήτησης [3] [4]. Το κύριο χαρακτηριστικό της είναι ότι παρέχει τη δυνατότητα διαφυγής από τα τοπικά ακρότατα επιτρέποντας κινήσεις που επιδεινώνουν την τιμή συνάρτησης κόστους. Η GS είναι ευρετική μέθοδος πολλαπλών εκκινήσεων που έχει σχεδιαστεί για την εύρεση του καθολικού βέλτιστου σε μη γραμμικά προβλήματα [5]. Η GS εκκινεί μία μέθοδο τοπικής βελτιστοποίησης από πολλαπλά σημεία εκκίνησης του συνόλου. Τα σημεία εκκίνησης δημιουργούνται από τη μέθοδο αναζήτησης διασποράς (scatter-search).

Η παρούσα μελέτη αφορά προβλήματα υψηλών διαστάσεων και η εύρεση του

---

καθολικού βέλτιστου αποτελεί δύσκολη διαδικασία. Οι μέθοδοι ΠΑ και GS δε διασφαλίζουν πάντα εγγυημένη σύγκλιση στο καθολικό βέλτιστο. Έτσι, απαιτείται προσεκτικός συντονισμός των παραμέτρων ελέγχου για την επίτευξη ισορροπίας μεταξύ ακρίβειας και υπολογιστικού χρόνου. Συγκεκριμένα, παρουσιάζονται διάφορες παραμετροποιήσεις και στρατηγικές. Η μέθοδος ΠΑ σε πολύπλοκους χώρους αναζήτησης απαιτεί υψηλό αριθμό επαναλήψεων και επιτυγχάνει αργή σύγκλιση κοντά στη γειτονία του καθολικού βέλτιστου. Επομένως, χρησιμοποιείται το μοντέλο υβριδικής προσέγγισης που συνδυάζει την ΠΑ με μία τοπική μέθοδο με στόχο τη σύγκλιση και βελτίωση της απόδοσης. Τέλος, για κάθε μέθοδο προτείνονται κριτήρια τερματισμού που δεν επιτρέπουν την υψηλή αύξηση του υπολογιστικού χρόνου.

Στην παρούσα μελέτη, το δεύτερο πρόβλημα περιλαμβάνει τη μεγιστοποίηση της πιθανοφάνειας ενός συνόλου χωρικών δεδομένων (δείγμα χωρικής στοχαστικής διαδικασίας) ως προς τις παραμέτρους του χωρικού στοχαστικού μοντέλου τοπικών αλληλεπιδράσεων (SLI). Συγκεκριμένα, πραγματοποιείται η επιλογή ανάμεσα σε δύο σύνολα δεδομένων, στα χωρικά πραγματικά ή συνθετικά, και ο παραμετρικός χώρος είναι τεσσάρων διαστάσεων ( $m, \lambda, c1, \mu$ ). Το μοντέλο SLI είναι ιδιαίτερα αποτελεσματικό στη διαχείριση μεγάλων συνόλων δεδομένων. Η συγκεκριμένη έρευνα πραγματοποιείται για την παρατήρηση συμπεριφοράς της κάθε μεθόδου στα μεγάλα χωρικά δεδομένα και στην εξέταση της αποτελεσματικότητάς της ως προς την ακρίβεια μεγιστοποίησης της πιθανοφάνειας και του υπολογιστικού χρόνου.

Το τοπίο της συνάρτησης κόστους είναι περίπλοκο και χρήζει περαιτέρω διερεύνησης για την κατανόηση του πρότυπου τοπικών ελαχίστων. Αναλυτικότερα, η μεγιστοποίηση της πιθανοφάνειας σε μεγάλα σύνολα δεδομένων αποτελεί δύσκολη διαδικασία. Άρα, είναι ιδιαίτερης σημασίας η κατάλληλη επιλογή των παραμέτρων ελέγχου για την επίτευξη ισορροπίας μεταξύ ακρίβειας και υπολογιστικού χρόνου. Έτσι, χρησιμοποιείται το μοντέλο υβριδικής προσέγγισης που συνδυάζει την ΠΑ με μία τοπική μέθοδο με στόχο τη σύγκλιση στο βέλτιστο αποτέλεσμα.

Η παρούσα εργασία αποσκοπεί στη βελτιστοποίηση δύο τύπων προβλημάτων, δηλαδή των συναρτήσεων ελέγχου και της πιθανοφάνειας του μοντέλου SLI. Επιπλέον, γίνεται σύγκριση των αποτελεσμάτων των μεθόδων στα θέματα του υπολογιστικού χρόνου, της ακρίβειας λύσης και της ευαισθησίας των μεθόδων στις διάφορες παραμετροποιήσεις τους.

Η δομή των επόμενων κεφαλαίων της εργασίας οργανώνεται ως εξής: στα Κεφάλαια 2 και 5 περιγράφονται οι μέθοδοι ΠΑ και GS. Στα Κεφάλαια 3 και 6 παρουσιάζονται τα αποτελέσματα των μεθόδων ΠΑ και GS στα προβλήματα βελτιστοποίησης των συναρτήσεων Ackley, Cross-In-Tray. Στα Κεφάλαια 4 και 7 παρουσιάζονται τα αποτελέσματα των μεθόδων ΠΑ και GS στο πρόβλημα της μεγιστοποίησης της



---

συνάρτησης πιθανοφάνειας του μοντέλου (SLI). Στην παρούσα εργασία, οι αριθμητικές τιμές που περιλαμβάνουν τον τελεστή («.») αντιπροσωπεύουν το τμήμα των δεκαδικών ψηφίων. Επιπλέον, οι αριθμητικές τιμές που περιλαμβάνουν τον τελεστή («,») αντιπροσωπεύουν το διαχωρισμό των χιλιάδων.

---

## Κεφάλαιο 2

# Μέθοδος Προσομοιωμένης Ανόπτωσης (ΠΑ)

### 2.1 Εισαγωγή

Η ΠΑ είναι μία πιθανολογική μέθοδος βελτιστοποίησης που προτάθηκε από τους Kirkpatrick, Gelett, Vecchi το 1983 και Chenry το 1985 [3]. Ειδικά, χρησιμοποιήθηκε για τον προσδιορισμό του καθολικού βέλτιστου δεδομένης συνάρτησης κόστους που περιέχει πολλαπλά τοπικά ακρότατα. Η μέθοδος μιμείται τη φυσική διαδικασία ανόπτωσης στη μεταλλουργία, δηλαδή την ελεγχόμενη ψύξη για τη βελτίωση των αποτελεσμάτων.

Με τον όρο **ανόπτωση** νοείται η διαδικασία θέρμανσης και ελεγχόμενης ψύξης ενός στερεού μέχρι να επιτευχθεί η καταλληλότερη δυνατή διαμόρφωση κρυσταλλικού πλέγματος (η ελάχιστη κατάσταση του ενεργειακού πλέγματος). Ειδικότερα, στις υψηλές θερμοκρασίες, τα άτομα στο υλικό έχουν υψηλή ενέργεια με αποτέλεσμα να έχουν μεγαλύτερη ελευθερία διάταξης και τυχαίας επιταχυνόμενης κίνησης. Έτσι, με την αργή ψύξη, το σύστημα έχει περισσότερες πιθανότητες να φτάσει στην ελάχιστη κατάσταση του ενεργειακού πλέγματος. Σε αυτή την κατάσταση, τα άτομα είναι ελεύθερα από τα κρυσταλλικά ελαττώματα. Ως εκ τούτου, η ΠΑ θεσπίζει τη σύνδεση μεταξύ του παραπάνω είδους της θερμοδυναμικής συμπεριφοράς και της αναζήτησης για το καθολικό ελάχιστο σε ένα πρόβλημα βελτιστοποίησης [6].

Η ανάπτυξη της μεθόδου ΠΑ το 1983 αποσκοπούσε στην αντιμετώπιση των έντονα μη γραμμικών προβλημάτων [4]. Αναλυτικότερα, η ΠΑ είναι στοχαστικός, μετά-ευρετικός αλγόριθμος τυχαίας αναζήτησης [4], που αξιοποιείται σε διακριτά προβλήματα βελτιστοποίησης σε μεγάλους χώρους αναζήτησης [7] [8]. Επιπρόσθετα, η εύκολη εφαρμογή της, οι ιδιότητες σύγκλισης και το γεγονός ότι δεν προαπαιτεί

## 2.1. Εισαγωγή

γνώση του χώρου αναζήτησης, την καθιστούν δημοφιλή τεχνική κατά τις τελευταίες δεκαετίες. Το κύριο χαρακτηριστικό αυτής της μεθόδου είναι, ότι παρέχει τη δυνατότητα διαφυγής από τα τοπικά ακρότατα επιτρέποντας κινήσεις που επιδεινώνουν την τιμή της συνάρτησης κόστους [6]. Ακόμη, ο αλγόριθμος ΠΑ εφαρμόζεται ευρέως σε πολλούς τομείς της βιομηχανίας και είναι ικανός στη διαχείριση πολύπλοκων προβλημάτων όπως το πρόβλημα του «Ταξιδιώτη Πωλητή» [9].

Ο αλγόριθμος ΠΑ ξεκινά με τον ορισμό της συνάρτησης κόστους με σκοπό την ελαχιστοποίηση ή την μεγιστοποίησή της ανάλογα τις απαιτήσεις κάθε προβλήματος [9]. Έπειτα, γίνεται η επιλογή αρχικής λύσης και ορίζονται οι παράμετροι ελέγχου του αλγόριθμου. Σε κάθε βήμα του, δημιουργείται μια νέα λύση. Στη συνέχεια, υπολογίζεται η διαφορά κόστους μεταξύ της τρέχουσας και της νέας λύσης. Εάν η διαφορά είναι μικρότερη, τότε γίνεται πάντα αποδεκτή ως νέα τρέχουσα λύση, αλλιώς η νέα λύση άλλοτε απορρίπτεται και άλλοτε γίνεται αποδεκτή βάσει συγκεκριμένης πιθανότητας. Με αυτόν τον τρόπο, η μέθοδος ΠΑ δίνει τη δυνατότητα αποδοχής λύσεων υψηλότερου κόστους έτσι ώστε να ξεφύγει από τα τοπικά ακρότατα και να κινηθεί σε περιοχές που πιθανόν βρίσκεται το καθολικό βέλτιστο. Επιπλέον, σε κάθε βήμα σημειώνεται μείωση της θερμοκρασίας μέσω του προγράμματος ενημέρωσης της θερμοκρασίας έτσι ώστε να μειωθεί η πιθανότητα αποδοχής λύσεων υψηλότερου κόστους. Τέλος, επαναλαμβάνεται η παραπάνω διαδικασία μέχρι να ικανοποιηθεί ένα κριτήριο τερματισμού [6].

Η **προσαρμοστική προσομοιωμένη ανόπτηση** (ΠΠΑ) αποτελεί βελτιωμένη έκδοση της ΠΑ που αναπτύχθηκε για να βρίσκει στατιστικά την καλύτερη καθολική κατά προσέγγιση λύση μιας μη γραμμικής, περιορισμένης, μη κυρτής συνάρτησης κόστους σε  $D$ -διάστατο χώρο [10]. Πιο συγκεκριμένα, εισήχθη ο όρος της «πολύ γρήγορης προσομοιωμένης ανόπτησης» για να τονιστεί το γεγονός της γρήγορης σύγκλισης της μεθόδου σε σύγκριση με την τυπική προσέγγιση ΠΑ [11]. Τελικώς, προτιμώμενος όρος είναι η ΠΠΑ. Η μέθοδος αυτή εισάγει βελτιώσεις έναντι της τυπικής ΠΑ σε συγκεκριμένους τομείς. Αναλυτικότερα, η μέθοδος ΠΠΑ συντονίζει τις θερμοκρασίες ανάλογα με τις ευαισθησίες της συνάρτησης κόστους για κάθε παράμετρο, δηλαδή επιτρέπει διαφορετικές θερμοκρασίες για διαφορετικές παραμέτρους. Αυτό έχει ως αποτέλεσμα, ο αλγόριθμος να διερευνά το χώρο των παραμέτρων βελτιώνοντας την απόδοση και την ταχύτητα σύγκλισης [12].

Εν συνεχεία, σημαντική διαφορά είναι η επανανόπτηση. Σε εκτενέστερη ανάλυση, ο ρυθμός αλλαγής του προγράμματος ανόπτησης μεταβάλλεται ανεξαρτήτως για κάθε παράμετρο καθ' όλη τη διαδικασία της βελτιστοποίησης. Αυτό έχει ως αποτέλεσμα, ο αλγόριθμος να κινείται σε περιοχές πολύ κοντά στο καθολικό βέλτιστο, καθώς προσαρμόζει την αναζήτησή του στο χώρο των παραμέτρων μέσω σημείων που εμφανίζουν διαφορετική τοπολογία. Επιπλέον, η επανανόπτηση επαναφέρει τη

## 2.2. Η περιγραφή της μεθόδου ΠΑ

θερμοκρασία σε υψηλότερη τιμή (συγκριτικά με την τρέχουσα) μετά από ορισμένο αριθμό αποδεκτών καταστάσεων. Έτσι, ο αλγόριθμος αποφεύγει την παγίδευση σε κάποιο τοπικό ακρότατο και αναζητά ευρύτερα το καθολικό. Η μέθοδος χρησιμοποιεί ένα διαφορετικό πρόγραμμα ανόπτησης που η μείωση της θερμοκρασίας πραγματοποιείται από μία εκθετική συνάρτηση με αποτέλεσμα ο αλγόριθμος να συγκλίνει ταχύτερα στο καθολικό βέλτιστο [12].

Η ΠΠΑ συνιστά στοχαστική μέθοδο καθολικής βελτιστοποίησης που είναι αποτελεσματική σε προβλήματα που περιέχουν πολύπλοκες, μη γραμμικές συναρτήσεις κόστους με πολλά τοπικά ελάχιστα [13]. Τα πλεονεκτήματα της μεθόδου ΠΠΑ έναντι του κλασικού αλγόριθμου ΠΑ είναι ότι προσαρμόζει ξεχωριστά την κάθε παράμετρο στην τοπική τοπολογία της συνάρτησης κόστους και περιλαμβάνει ταχύτερο πρόγραμμα ενημέρωσης της θερμοκρασίας. Έτσι, διαθέτει ταχύτερες ιδιότητες σύγκλισης. Ωστόσο, λόγω αυτών των πλεονεκτημάτων, η υλοποίηση του αλγόριθμου ΠΠΑ καθίσταται πιο περίπλοκη από τον κλασικό αλγόριθμο ΠΑ [12]. Τέλος, η μέθοδος ΠΠΑ είναι αποτελεσματική στην επίλυση του προβλήματος ρύπανσης των υπόγειων υδάτων [14] αλλά και του προσδιορισμού των ακτινικών ταχυτήτων των δυαδικών αστεριών [12].

Στην συγκεκριμένη εργασία, τα προβλήματα βελτιστοποίησης που επιλύονται από την εφαρμογή της μεθόδου ΠΑ, υλοποιούνται στο προγραμματιστικό περιβάλλον της Matlab. Η δομή της μεθόδου παρουσιάζει κοινά χαρακτηριστικά με το μοντέλο του αλγόριθμου ΠΠΑ. Επιπροσθέτως, οι τελικές τιμές που επιστρέφονται από τον αλγόριθμο δεν είναι οι τιμές που λαμβάνονται από την τελευταία εκτέλεσή του. Αντιθέτως, είναι οι επιτυγχανόμενες ελάχιστες τιμές που προκύπτουν κατά τη διάρκεια της διαδικασίας βελτιστοποίησης. Επιπλέον, η τρέχουσα θερμοκρασία  $T$ , περιγράφεται ως διάνυσμα μήκους ίσο με τον αριθμό των στοιχείων της μεταβλητής  $x$ . Τα συγκεκριμένα προβλήματα βελτιστοποίησης χαρακτηρίζονται από παραμετρικούς χώρους υψηλών διαστάσεων. Έτσι, χάριν απλότητας πραγματοποιείται ενδεικτικά η επιλογή της πρώτης διάστασης της θερμοκρασίας για την οπτική απεικόνισή της κατά τη διάρκεια της διαδικασίας βελτιστοποίησης.

## 2.2 Η περιγραφή της μεθόδου ΠΑ

Παρακάτω παρουσιάζεται αναλυτικά η διαδικασία βελτιστοποίησης της μεθόδου ΠΑ [9] [8]:

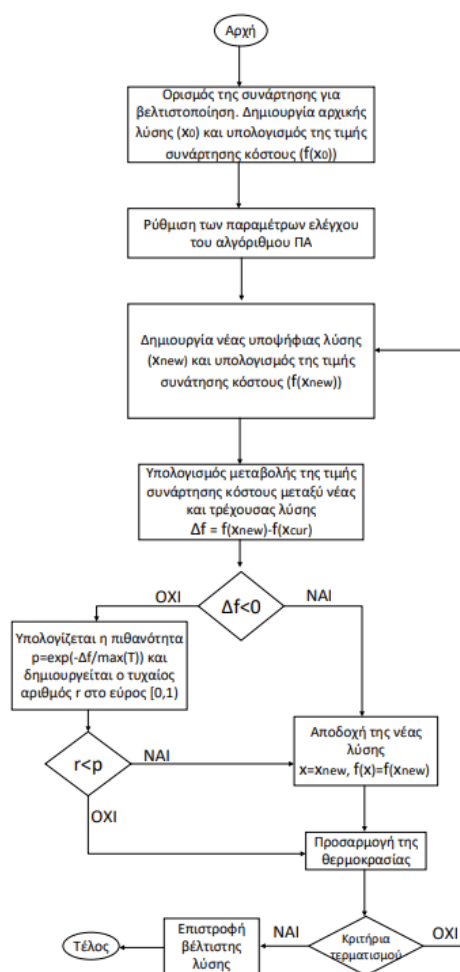
Βήμα 1: Ορισμός του προβλήματος για βελτιστοποίηση. Αυτό περιλαμβάνει τον ορισμό της συνάρτησης κόστους, δηλαδή της συνάρτησης που πρόκειται να ελαχιστοποιηθεί.

## 2.2. Η περιγραφή της μεθόδου ΠΑ

---

- Βήμα 2: Δημιουργία αρχικής λύσης  $x_0$  και υπολογισμός της τιμής συνάρτησης κόστους  $f(x_0)$ . Η αρχική λύση δημιουργείται τυχαία. Ακόμη, ορίζονται οι παράμετροι ελέγχου.
- Βήμα 3: Δημιουργία νέας υποψήφιας λύσης  $x_{\text{new}}$  και υπολογισμός της τιμής συνάρτησης κόστους  $f(x_{\text{new}})$ . Η συνάρτηση γειτονικής λύσης καθορίζει τον τρόπο με τον οποίο δημιουργούνται οι νέες λύσεις σε κάθε επανάληψη.
- Βήμα 4: Υπολογισμός της μεταβολής της τιμής συνάρτησης κόστους μεταξύ της νέας και τρέχουσας λύσης.  $\Delta f = f(x_{\text{new}}) - f(x_{\text{cur}})$
- Βήμα 5: Αποδοχή ή απόρριψη της υποψήφιας λύσης σύμφωνα με το κριτήριο Μητρόπολης. Εάν  $\Delta f < 0$ , γίνεται αποδεκτή ως νέα τρέχουσα λύση. Σε διαφορετική περίπτωση, η νέα λύση άλλοτε απορρίπτεται και άλλοτε γίνεται αποδεκτή με βάση την πιθανότητα  $P = \frac{1}{1 + \exp\left(\frac{\Delta f}{\max(T)}\right)} \approx \exp\left(\frac{-\Delta f}{\max(T)}\right)$ . Η πιθανότητα αυτή μειώνεται αναλογικά με τη τρέχουσα θερμοκρασία  $T$ . Στη συνέχεια, στην περίπτωση που  $\Delta f > 0$  δημιουργείται ένας τυχαίος αριθμός  $r$  στο διάστημα  $[0, 1)$ . Έπειτα, γίνεται η σύγκριση  $r < p$ . Αν ισχύει, τότε συμβαίνει η αποδοχή της λύσης υψηλότερου κόστους ως νέα τρέχουσα, αλλιώς δεν αλλάζει η τρέχουσα λύση και μεταβαίνει η εκτέλεση στο βήμα 6.
- Βήμα 6: Προσαρμογή της θερμοκρασίας. Η θερμοκρασία μειώνεται σύμφωνα με το πρόγραμμα ενημέρωσής της που έχει επιλεχθεί εξ' αρχής ή η θερμοκρασία αυξάνεται όταν συμβεί η επανανόπτηση.
- Βήμα 7: Έλεγχος κριτηρίων τερματισμού. Εάν συμβαίνει αυτό, τότε εξάγεται η βέλτιστη τιμή συνάρτησης κόστους και η διαδικασία βελτιστοποίησης τερματίζεται. Σε διαφορετική περίπτωση, επιστρέφει η εκτέλεση στο βήμα 3.

## 2.2. Η περιγραφή της μεθόδου ΠΑ



Σχήμα 2.1: Διάγραμμα ροής του αλγόριθμου ΠΑ

## 2.3 Παράμετροι ελέγχου

Οι παράμετροι ελέγχου που επηρεάζουν την απόδοση του αλγόριθμου ΠΑ είναι η αρχική θερμοκρασία, το πρόγραμμα ψύξης, τα κριτήρια τερματισμού, το κριτήριο αποδοχής, η επανανόπτηση και η συνάρτηση γειτονικής λύσης. Σε οποιοδήποτε πρόβλημα βελτιστοποίησης είναι απαραίτητο να συμβούν πολλαπλές εκτελέσεις με διαφορετικές παραμετροποιήσεις, έτσι ώστε να προσδιοριστεί στατιστικά ο πιο αποτελεσματικός συνδυασμός.

Η εύρεση της κατάλληλης τιμής της **αρχικής θερμοκρασίας** (InitialTemperature) σε οποιοδήποτε πρόβλημα βελτιστοποίησης είναι μείζονος σημασίας για την αποτελεσματικότητα του αλγόριθμου ΠΑ. Ειδικότερα, η υψηλή τιμή της επιτρέπει με μεγαλύτερη πιθανότητα την αποδοχή λύσεων υψηλότερου κόστους και δημιουργεί τοπίο που κρατάει τον αλγόριθμο μακριά από το καθολικό ελάχιστο για πολλές επαναλήψεις. Αυτό έχει ως αποτέλεσμα, τη σημαντική αύξηση του υπολογιστικού χρόνου βελτιστοποίησης και ίσως ο αλγόριθμος να μη φτάσει σε επιτυχημένη αναζήτηση, αν οι υπόλοιποι παράμετροι ελέγχου δεν είναι ορισμένοι κατάλληλα. Η αρχική θερμοκρασία και το πρόγραμμα ενημέρωσής της καθορίζουν τον ρυθμό σύγκλισης του αλγόριθμου. Από την άλλη, η χαμηλή τιμή της, ενδεχομένως να περιορίσει την αναζήτηση γύρω από το σημείο εκκίνησης. Αυτό έχει ως συνέπεια, ο αλγόριθμος να τερματίσει πρόωρα την αναζήτησή του σε περιοχές μακριά από το καθολικό ελάχιστο [15] [6].

Το **πρόγραμμα ψύξης** ή το **πρόγραμμα ενημέρωσης της θερμοκρασίας** καθορίζει τον ρυθμό με τον οποίο μειώνεται η θερμοκρασία σε κάθε επανάληψη του αλγόριθμου. Η επιλογή αργού προγράμματος ψύξης απαιτεί περισσότερες επαναλήψεις για την εύρεση λύσης κοντά στο καθολικό ελάχιστο. Εάν, ο αλγόριθμος διαθέτει περιορισμένο αριθμό επαναλήψεων θα οδηγηθεί σε μη βέλτιστο αποτέλεσμα. Αντιθέτως, η επιλογή πολύ γρήγορου προγράμματος ψύξης, ενδέχεται να παγιδεύσει τον αλγόριθμο σε κάποιο τοπικό ακρότατο [15].

Στα συγκεκριμένα προβλήματα βελτιστοποίησης, η συνάρτηση TemperatureFcn είναι υπεύθυνη για την επιλογή του προγράμματος ψύξης. Πιο συγκεκριμένα, διαθέτει τρεις επιλογές (εκθετική, γραμμική, λογαριθμική). Η παράμετρος  $k$  αντιστοιχεί στον τρέχοντα αριθμό επανάληψης μέχρι την επανανόπτηση. Επίσης,  $T_0$  είναι η αρχική θερμοκρασία του συστήματος και η θερμοκρασία  $T$ , περιγράφεται ως διάνυσμα μήκους ίσο με τον αριθμό των στοιχείων του τρέχοντος σημείου  $x$ . Επί της ουσίας, για την παράμετρο  $T$  εφαρμόζεται διαφορετική θερμοκρασία σε κάθε διάσταση [16]. Η λειτουργία εκθετικής ενημέρωσης πραγματοποιείται από τη συνάρτηση temperatureexp, κατά την οποία η θερμοκρασία μειώνεται εκθετικά καθώς αυξάνε-



### 2.3. Παράμετροι ελέγχου

---

ται ο αριθμός των επαναλήψεων και ορίζεται ως:

$$T = T_0 \cdot 0.95^k$$

Η λειτουργία γραμμικής ενημέρωσης υλοποιείται από τη συνάρτηση `temperaturefast`, κατά την οποία μειώνεται η θερμοκρασία γραμμικά καθώς αυξάνεται ο αριθμός των επαναλήψεων και ορίζεται ως:

$$T = T_0/k$$

Η λειτουργία λογαριθμικής ενημέρωσης διεκπεραιώνεται από τη συνάρτηση `temperatureboltz`, κατά την οποία η θερμοκρασία μειώνεται λογαριθμικά καθώς αυξάνεται ο αριθμός των επαναλήψεων και ορίζεται ως:

$$T = T_0/\ln(k)$$

Τα **κριτήρια τερματισμού** καθορίζουν πότε πρέπει να τερματιστεί η διαδικασία βελτιστοποίησης [16]. Η κατάλληλη επιλογή των κριτηρίων τερματισμού είναι απαραίτητη για να διασφαλιστεί η ισορροπία μεταξύ του υπολογιστικού χρόνου και της ακρίβειας λύσης. Τα κριτήρια τερματισμού είναι τα εξής: `MaxIterations`, `MaxFunctionEvaluations`, `MaxTime`, `FunctionTolerance`, `ObjectiveLimit`.

Αναφορικά με το κριτήριο τερματισμού `MaxIterations`, ο αλγόριθμος τερματίζει την εκτέλεσή του μετά από τον μέγιστο ορισμένο αριθμό επαναλήψεων. Αυτό είναι σημαντικό, διότι ο αλγόριθμος δεν εκτελεί απεριόριστες επαναλήψεις, αλλά τερματίζει σε πεπερασμένο χρονικό διάστημα.

Το κριτήριο τερματισμού `MaxFunctionEvaluations` καθορίζει τον τερματισμό του αλγόριθμου μετά από τον μέγιστο ορισμένο αριθμό αξιολογήσεων της συνάρτησης κόστους. Πιο συγκεκριμένα, με τη δημιουργία κάθε νέου σημείου, η συνάρτηση αξιολογείται σε αυτό το σημείο για να καθοριστεί η αποδοχή ή απόρριψη του. Επιπλέον, η συνάρτηση αξιολογείται περισσότερο από μία φορά κατά τη διαδικασία που συμβαίνει η επανανόπτηση. Αναλυτικότερα, είναι απαραίτητος ο υπολογισμός των κλίσεων της συνάρτησης κόστους σε κάθε διάσταση. Έτσι, αξιολογείται η συνάρτηση κόστους χωριστά σε κάθε διάσταση.

Το κριτήριο τερματισμού `MaxTime` καθορίζει το μέγιστο χρονικό διάστημα σε δευτερόλεπτα που επιτρέπεται να εκτελεστεί ο αλγόριθμος πριν τον τερματισμό του.

Το κριτήριο τερματισμού `FunctionTolerance` ορίζει την ανοχή για τον έλεγχο μεταβολής της βέλτιστης τιμής συνάρτησης κόστους για ένα μέγιστο ορισμένο αριθμό επαναλήψεων. Επεξηγηματικά, ο μέγιστος αριθμός των επαναλήψεων καθορίζεται

### 2.3. Παράμετροι ελέγχου

από την παράμετρο `MaxStallIterations`. Στην περίπτωση που η μεταβολή της βέλτιστης τιμής συνάρτησης κόστους για τις επαναλήψεις `MaxStallIterations`, είναι μικρότερη από την ανοχή `FunctionTolerance`, τότε ο αλγόριθμος τερματίζει την εκτέλεσή του. Επί της ουσίας, το κριτήριο υποδεικνύει ότι περαιτέρω επαναλήψεις δεν πραγματοποιούν σημαντικές αλλαγές στην τιμή συνάρτησης κόστους σύμφωνα με την καθορισμένη ανοχή.

Σε κάθε επανάληψη, η τιμή συνάρτησης κόστους συγκρίνεται με την παράμετρο `ObjectiveLimit`. Ο αλγόριθμος τερματίζει την εκτέλεσή του όταν η τιμή συνάρτησης κόστους είναι ίση ή μικρότερη από το συγκεκριμένο όριο.

Το **κριτήριο αποδοχής** καθορίζει την έγκριση ή απόρριψη του νέου σημείου στο χώρο αναζήτησης. Στα συγκεκριμένα προβλήματα, το κριτήριο καθορίζεται από τη συνάρτηση `AcceptanceFcn` η οποία συνιστά παραλλαγή του κλασικού κριτηρίου Μη-τρόπολης. Αναλυτικότερα, στην περίπτωση που το κόστος της νέας λύσης είναι μικρότερο από αυτό της τρέχουσας, τότε το νέο σημείο γίνεται αυτόματα αποδεκτό. Αντιθέτως, εάν το κόστος της νέας λύσης είναι μεγαλύτερο, καθορίζεται με ορισμένη πιθανότητα η αποδοχή ή απόρριψή του σημείου. Αυτή η πιθανότητα ορίζεται ως:  $P = \frac{1}{1 + \exp\left(\frac{\Delta E}{\max(T)}\right)}$ , όπου  $\Delta E$  είναι η διαφορά κόστους μεταξύ της τρέχουσας και νέας λύσης και  $T$  είναι η τρέχουσα θερμοκρασία [16]. Η διαφορά που παρατηρείται συγκριτικά με το κλασικό κριτήριο είναι ότι χρησιμοποιείται η μέγιστη τρέχουσα θερμοκρασία. Αυτό συμβαίνει διότι χρησιμοποιούνται διαφορετικές θερμοκρασίες σε κάθε διάσταση.

Η **επανανόπτηση** είναι υπεύθυνη για την αύξηση της θερμοκρασίας αφού ο αλγόριθμος ΠΑ δεχτεί έναν ορισμένο αριθμό νέων σημείων [16]. Έτσι, πραγματοποιείται η επανεκκίνηση της αναζήτησης σε υψηλότερη θερμοκρασία. Αυτό είναι και το βασικό χαρακτηριστικό του αλγόριθμου ΠΑ που τον κάνει να διαφέρει από τους υπόλοιπους στα προβλήματα βελτιστοποίησης. Πιο συγκεκριμένα, αποδέχεται λύσεις υψηλότερου κόστους για τη διαφυγή από τα τοπικά βέλτιστα με σκοπό την ευρεία αναζήτηση για την εύρεση του καθολικού.

Στα συγκεκριμένα προβλήματα, το διάστημα επανανόπτησης καθορίζεται από την παράμετρο `ReannealInterval`. Σε εκτενέστερη ανάλυση, η θερμοκρασία αυξάνεται σε κάθε διάσταση όταν οι παράμετροι ανόπτησης ρυθμίζονται σε χαμηλότερη τιμή από τον τρέχοντα αριθμό επανάληψης [11]. Οι παράμετροι ανόπτησης εξαρτώνται από τις τιμές των εκτιμώμενων κλίσεων της συνάρτησης κόστους σε κάθε διάσταση και προσδιορίζονται από τον τύπο:

$$k_i = \log \left( \frac{T_0}{T_i} \cdot \frac{\max(s_j)}{s_i} \right),$$

που  $k_i$  είναι η παράμετρος ανόπτησης για το στοιχείο  $i$ ,  $T_0$  είναι η αρχική θερμο-

### 2.3. Παράμετροι ελέγχου

κρασία του στοιχείου  $i$ ,  $T_i$  είναι η τρέχουσα θερμοκρασία του στοιχείου  $i$ ,  $s_i$  είναι η κλίση της συνάρτησης κόστους στην κατεύθυνση  $i$ , πολλαπλασιασμένη με τη διαφορά των ορίων στην κατεύθυνση  $i$  [47]. Ωστόσο, η ακριβής αύξηση της θερμοκρασίας εξαρτάται από τις κλίσεις της συνάρτησης κόστους σε κάθε κατεύθυνση συγκριτικά με τη μέγιστη κλίση μεταξύ των διάφορων στοιχείων  $i$ .

Η **συνάρτηση γειτονικής λύσης** είναι υπεύθυνη για τη δημιουργία νέων σημείων σε κάθε επανάληψη στη γειτονία του τρέχοντος. Διευκρινιστικά, για τον τυπικό αλγόριθμο ΠΑ, ορισμένες κοινές στρατηγικές για τη δημιουργία νέων σημείων είναι η εναλλαγή δύο στοιχείων και η τυχαία αλλαγή της τιμής ενός ή περισσότερων στοιχείων [48]. Στον αλγόριθμο ΠΠΑ τα νέα δοκιμαστικά σημεία που βρίσκονται εντός του χώρου αναζήτησης και οριοθετούνται από τα διανύσματα  $A_i$  (άνω όρια) και  $B_i$  (κάτω όρια) προσδιορίζονται ως  $x_{new} = x_{old} + q_i(B_i - A_i)$  με  $i = 1, \dots, D$ . Ειδικά, κάθε παράμετρος  $x_i$  ενημερώνεται ως  $x_{i,new}$  χρησιμοποιώντας τη συνάρτηση διανομής  $q_i$ . Επίσης, το  $q_i$  προστίθεται στην γειτονία του τρέχοντος σημείου για τη δημιουργία νέου. Επομένως, για κάθε διάσταση  $i$  η συνάρτηση διανομής προσδιορίζεται ως:

$$q_i = \text{sgn} \left( u_i - \frac{1}{2} \right) T_i(k_i) \left( \left( 1 + \frac{1}{T_i(k_i)} \right)^{2u_i-1} - 1 \right)$$

Η συνάρτηση διανομής  $q_i$  είναι το μέγεθος του τυχαίου βήματος που χρησιμοποιείται στο τρέχον σημείο για τη δημιουργία του νέου και ορίζεται στο διάστημα  $[-1, 1]$ . Από την εξίσωση, η παράμετρος  $u_i$  είναι μια ομοιόμορφα κατανεμημένη τυχαία μεταβλητή στο εύρος  $[0, 1]$ . Αυτή η παράμετρος εισάγει την τυχειότητα στη δημιουργία νέων σημείων. Ακόμη, η μεταβλητή  $T_i$  είναι η θερμοκρασία για την παράμετρο  $x_i$  και ελέγχει την κλίμακα της μετατόπισης. Συμπληρωματικά, η μεταβλητή  $k_i$  αναφέρεται ως χρόνος ανόπτησης και αποτελεί δείκτη που ελέγχει τον αριθμό ενημερώσεων της θερμοκρασίας για κάθε παράμετρο. Συνεπώς, η συνάρτηση  $\text{sgn}()$  καθορίζει το πρόσημο έκφρασης μέσα στις παρενθέσεις, δηλαδή την κατεύθυνση της απόστασης που θα έχει το νέο σημείο από το τρέχον [42] [41].

Στα συγκεκριμένα προβλήματα, η συνάρτηση `AnnealingFcn` είναι υπεύθυνη για τη δημιουργία νέων σημείων σε κάθε επανάληψη [46]. Ύπ'αυτή την έννοια, χρησιμοποιούνται δύο τρόποι για τη δημιουργία νέων σημείων. Η διαφορά μεταξύ αυτών των δύο, αφορά στην κατανομή πιθανότητας με κλίμακα που εξαρτάται από την τρέχουσα θερμοκρασία. Αναντίρρητα, η διαφορά των δύο επιλογών σημειώνεται στην απόσταση που υπάρχει μεταξύ νέου και τρέχοντος σημείου (βλέπε ενότητα (A)). Πιο συγκεκριμένα, και οι δύο παράμετροι `annealingfast`, `annealingboltz` δημιουργούν νέα σημεία με κατεύθυνση ομοιόμορφα τυχαία με όλες τις κατευθύνσεις να είναι εξίσου πιθανές. Στην περίπτωση του `annealingfast`, επιλέγεται η κλίμακα μετατόπισης του νέου σημείου από το τρέχον να είναι ανάλογη της τρέχουσας θερμοκρασίας. Αντιθέτως, στην περίπτωση του `annealingboltz` επιλέγεται η κλίμακα μετατόπισης

### 2.3. Παράμετροι ελέγχου

---

του νέου σημείου από το τρέχον να είναι ανάλογη με την τετραγωνική ρίζα της τρέχουσας θερμοκρασίας.

## Κεφάλαιο 3

# Εφαρμογή Μεθόδου ΠΑ σε Συναρτήσεις Ελέγχου

### 3.1 Περιγραφή της υβριδικής προσέγγισης

Στα συγκεκριμένα προβλήματα βελτιστοποίησης, η τυπική εκτέλεση του αλγόριθμου ΠΑ απαιτεί υψηλό υπολογιστικό χρόνο για την εύρεση λύσης υψηλής ακρίβειας κοντά στο καθολικό βέλτιστο [19]. Πιο συγκεκριμένα, σε ορισμένες περιπτώσεις ενδέχεται η σύγκλιση του αλγόριθμου να είναι μακριά από το καθολικό βέλτιστο. Για την αποφυγή αυτών των προβλημάτων ορίζεται μία υβριδική προσέγγιση ως ο συνδυασμός των μεθόδων καθολικής και τοπικής βελτιστοποίησης. Στη συγκεκριμένη εργασία προτείνεται η υβριδική προσέγγιση που συγχωνεύει τον αλγόριθμο ΠΑ με τοπικές μεθόδους βελτιστοποίησης. Η τοπική μέθοδος εκτελείται κατά τη διάρκεια ή στο τέλος των επαναλήψεων του αλγόριθμου ΠΑ [16]. Στην περίπτωση που εκτελείται αμέσως μετά τον τερματισμό, εκκινεί από το βέλτιστο σημείο στο οποίο τερματίζει ο αλγόριθμος ΠΑ. Η υβριδική προσέγγιση συμβάλλει στη σημαντική μείωση του υπολογιστικού χρόνου και αυξάνει την πιθανότητα σύγκλισης στο καθολικό βέλτιστο. Ο αλγόριθμος ΠΑ διαθέτει την επιλογή τεσσάρων διαφορετικών τοπικών μεθόδων βελτιστοποίησης για τον ορισμό του μοντέλου υβριδικής προσέγγισης.

Στα προβλήματα βελτιστοποίησης της συνάρτησης Ackley σε χώρο 10 και 30 διαστάσεων, αξιοποιείται η υβριδική προσέγγιση με τη τοπική βελτιστοποίηση **fmincon**. Η συνάρτηση λειτουργεί ως τοπική μέθοδος βελτιστοποίησης σε μη γραμμικά προβλήματα [20]. Συγκεκριμένα, είναι σχεδιασμένη για την εύρεση του ελαχίστου συνάρτησης πολλών μεταβλητών που επιδέχεται ορίων, μη γραμμικών και γραμμικών περιορισμών [21]. Η μέθοδος εκκινεί από ένα αρχικό σημείο και αξιοποιεί τις τιμές και κλίσεις της συνάρτησης του προβλήματος προκειμένου να επιτευχθεί η σύγκλιση σε τοπικό ελάχιστο. Επιπρόσθετα, διαθέτει διαφορετικούς αλγόριθμους και

### 3.1. Περιγραφή της υβριδικής προσέγγισης

επιλέγει τον κατάλληλο με βάση τις συγκεκριμένες απαιτήσεις κάθε προβλήματος [22]. Στα συγκεκριμένα προβλήματα υψηλής διάστασης, χρησιμοποιείται ο αλγόριθμος *interior-point*. Η επιλογή του συγκεκριμένου αλγόριθμου γίνεται διότι είναι ικανός στη διαχείριση προβλημάτων μεγάλης κλίμακας. Ως εκ τούτου, επιτυγχάνει καλύτερη κατανομή μνήμης και υπολογιστικού χρόνου.

Ακολούθως, πραγματοποιείται ο υπολογισμός παραγώγου της συνάρτησης Ackley στο χώρο των 10 και 30 διαστάσεων. Αυτό συμβαίνει διότι η πληροφορία της παραγώγου βελτιώνει σημαντικά την ακρίβεια λύσης και την ταχύτητα σύγκλισης. Πιο συγκεκριμένα, η μέθοδος γνωρίζει την ακριβή κατεύθυνση της τιμής σε ένα σημείο, αντί να βασίζεται στις αριθμητικές παραγώγους που υπολογίζονται με την προσέγγιση πεπερασμένων διαφορών (προκαθορισμένη επιλογή) [23]. Ακόμη, χρησιμοποιείται η ανοχή *StepTolerance* για τον έλεγχο του τερματισμού της διαδικασίας [20]. Η παράμετρος καθορίζει το ελάχιστο αποδεκτό μέγεθος βήματος των μεταβλητών  $x$  μεταξύ των επαναλήψεων. Έτσι, εάν η αλλαγή στο μέγεθος βήματος του  $x$  μεταξύ δύο επαναλήψεων είναι μικρότερη από αυτήν την ανοχή, τότε η διαδικασία βελτιστοποίησης τερματίζει. Για αυτόν τον λόγο, η παράμετρος είναι ορισμένη σε χαμηλή τιμή.

Στα προβλήματα βελτιστοποίησης της συνάρτησης *Cross-In-Tray* σε χώρο 10 και 30 διαστάσεων, αξιοποιείται η υβριδική προσέγγιση με τη τοπική βελτιστοποίηση **pattern search**. Η μέθοδος χρησιμοποιείται για πολυδιάστατες, μη ομαλές ή μη διαφορίσιμες συναρτήσεις [24]. Το κύριο χαρακτηριστικό που την ξεχωρίζει από τις υπόλοιπες μεθόδους είναι ότι δε βασίζεται στον υπολογισμό κλίσης ή των ανώτερων παραγώγων για την εύρεση του βέλτιστου σημείου [25]. Η μέθοδος αξιολογεί διάφορα σημεία που είναι ορισμένα σε ένα πλέγμα. Αν δε συμβεί βελτίωση στην τιμή συνάρτησης κόστους, τότε το πλέγμα συρρικνώνεται και η αναζήτηση περιορίζεται γύρω από το τρέχον σημείο. Στην αντίθετη περίπτωση, το σημείο γίνεται αποδεκτό στο επόμενο βήμα του αλγόριθμου και το πλέγμα επεκτείνεται πραγματοποιώντας μεγαλύτερα βήματα αναζήτησης [26].

Στη συνέχεια, μέχρι την ολοκλήρωση της διαδικασίας η εκτέλεση εναλλάσσεται συνεχώς μεταξύ των βημάτων επιτυχημένης ή μη αναζήτησης. Αναλυτικότερα, το πλέγμα ορίζεται ως το σύνολο των σημείων γύρω από το τρέχον βέλτιστο. Το μέγεθος και το σχήμα του πλέγματος προσαρμόζεται συνεχώς με βάση την πρόοδο αναζήτησης. Στα συγκεκριμένα προβλήματα, η ανοχή στο μέγεθος του πλέγματος (*MeshTolerance*) είναι υπεύθυνη για τον τερματισμό της διαδικασίας. Έτσι, εάν η αλλαγή στο μέγεθος του πλέγματος μεταξύ δύο επαναλήψεων είναι μικρότερη από αυτή την ανοχή, τότε η διαδικασία τερματίζει. Στα συγκεκριμένα προβλήματα έγινε χρήση της μεθόδου βελτιστοποίησης με τις προκαθορισμένες επιλογές παραμέτρων ελέγχου [26]. Η μέθοδος μπορεί να χρησιμοποιηθεί τόσο σε προβλήματα καθολικής

### 3.2. Συνάρτηση Ackley

όσο και σε τοπικής βελτιστοποίησης.

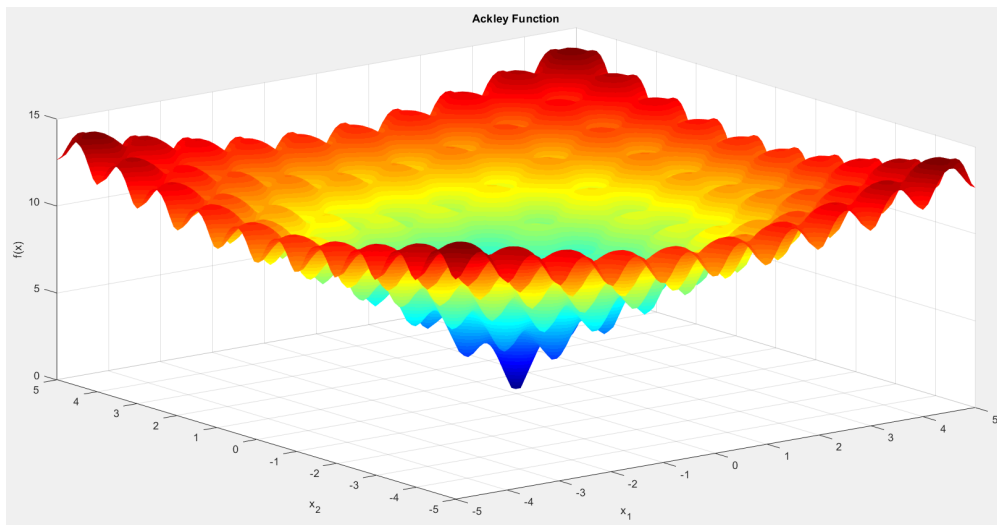
## 3.2 Συνάρτηση Ackley

Η πολυδιάστατη συνάρτηση Ackley ορίζεται από την ακόλουθη εξίσωση:

$$f(\mathbf{x}) = -a \exp \left( -b \sqrt{\frac{1}{d} \sum_{i=1}^d x_i^2} \right) - \exp \left( \frac{1}{d} \sum_{i=1}^d \cos(cx_i) \right) + a + \exp(1)$$

Στην εξίσωση της συνάρτησης η παράμετρος  $x$  είναι διάνυσμα πραγματικών αριθμών, ενώ τα  $a, b, c$  είναι σταθερές. Συνήθως ορίζονται ως  $a = 20$ ,  $b = 0.2$ ,  $c = 2\pi$  και το  $x_i$  είναι το  $i$ -οστό στοιχείο του διανύσματος  $x$ . Στις περισσότερες εφαρμογές η συνάρτηση συνήθως αξιολογείται στον υπερκύβο με  $x_i \in [-32.768, 32.768]$ , για όλα τα  $i = 1, \dots, d$

Στις δύο διαστάσεις ( $d = 2$ ), το γράφημα της συνάρτησης Ackley είναι μία επιφάνεια σε τρεις διαστάσεις. Οι άξονες  $x_1$  και  $x_2$  αντιπροσωπεύουν τις μεταβλητές εισόδου και ο κατακόρυφος άξονας  $z$  αντιπροσωπεύει την τιμή εξόδου της συνάρτησης,  $f(\mathbf{x})$ . Η συνάρτηση Ackley είναι συνεχής, μη κυρτή, πολυτροπική και διαφορίσιμη συνάρτηση που έχει πολλά τοπικά ελάχιστα αλλά και ένα καθολικό [27]. Η συνάρτηση είναι δημοφιλής και χρησιμοποιείται σε προβλήματα δοκιμής απόδοσης των μεθόδων βελτιστοποίησης με στόχο την εύρεση του καθολικού ελάχιστου. Από το διάγραμμα, η συνάρτηση χαρακτηρίζεται από μια σχεδόν επίπεδη εξωτερική περιοχή και περιέχει μια μεγάλη κοιλάδα στο κέντρο [28]. Ειδικά, η μορφή της επιφάνειας ομοιάζει με ωειδές σχήμα και παρουσιάζει ένα καθολικό ελάχιστο στο κέντρο, δηλαδή στο σημείο  $[0, 0]$  με τιμή ίση με μηδέν. Παρακάτω φαίνεται το διάγραμμα της συνάρτησης:



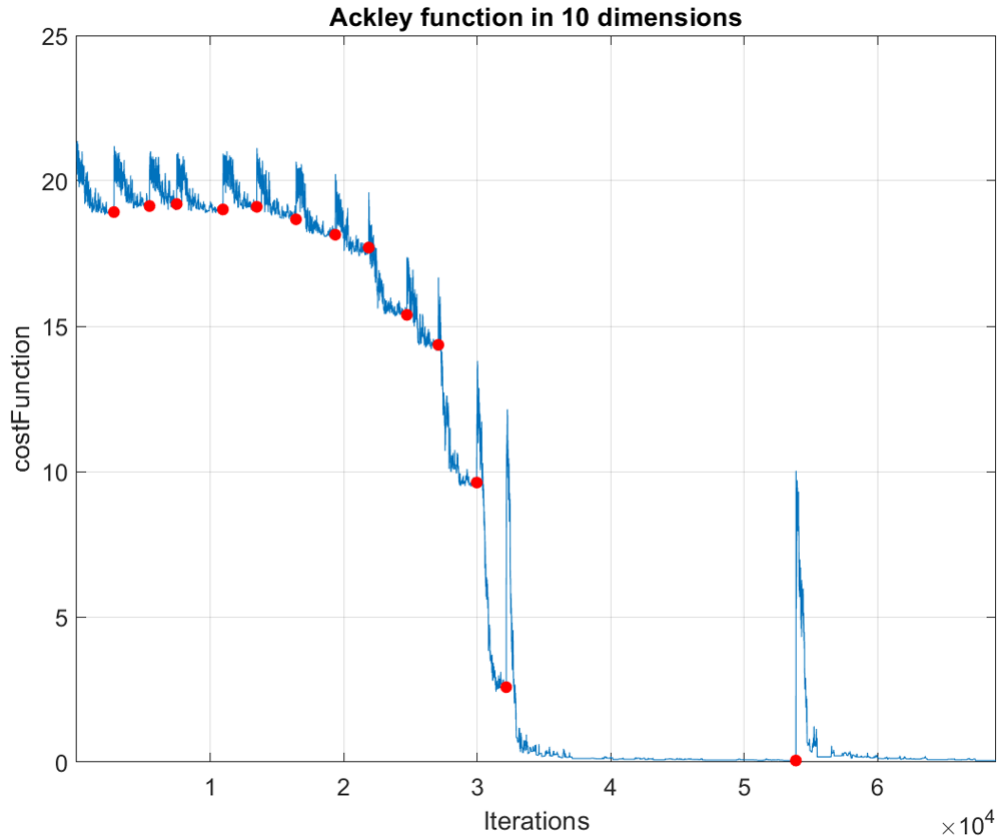
Σχήμα 3.1: Η αναπαράσταση της συνάρτησης Ackley σε χώρο δύο διαστάσεων ( $d = 2$ ).



## 3.2. Συνάρτηση Ackley

### 3.2.1 Αποτελέσματα σε χώρο 10 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Ackley σε χώρο 10 διαστάσεων από την εφαρμογή των μεθόδων ΠΑ και fmincon:



**Σχήμα 3.2:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

Από το Σχήμα 3.2 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος ΠΑ ελαχιστοποιεί τη συνάρτηση Ackley με την πάροδο του χρόνου. Οι κόκκινες ακίδες στην τιμή συνάρτησης κόστους αντιστοιχούν στα συμβάντα επανανόπτησης. Ο αλγόριθμος ΠΑ εκκινεί την αναζήτηση του από τυχαίο αρχικό σημείο με τιμή ίση 21.0923.

Από το Σχήμα 3.2 διακρίνεται ότι στις πρώτες περίπου 22,000 επαναλήψεις δεν πραγματοποιείται σημαντική μείωση στην τιμή. Η τιμή της αρχικής θερμοκρασίας θεωρείται σχετικά υψηλή. Ωστόσο, χρησιμοποιείται το γραμμικό πρόγραμμα ενημέρωσής της και η θερμοκρασία σε σύντομες επαναλήψεις λαμβάνει τιμές κοντά στο μηδέν. Αυτό έχει ως αποτέλεσμα, ο αλγόριθμος σε σύντομες επαναλήψεις να είναι λιγότερο πιθανό να δεχτεί λύσεις που αυξάνουν την τιμή της. Έτσι, εστιάζει την αναζήτηση του στη βελτίωση γύρω από την τρέχουσα τιμή. Στην αρχική φάση βελτιστοποίησης, ο αλγόριθμος ΠΑ βρίσκεται μακριά από το καθολικό ελάχιστο



### 3.2. Συνάρτηση Ackley

---

και κινείται σε περιοχές που περιλαμβάνουν πολλαπλά τοπικά ελάχιστα. Σε αυτή τη φάση, πραγματοποιείται ο συνδυασμός ενός γρήγορου προγράμματος ψύξης με ένα σχετικά μικρό διάστημα επανανόπτησης για την αποφυγή παγίδευσης σε κάποιο τοπικό ελάχιστο. Επιπλέον, η ΠΑ περιλαμβάνει την τυχειότητα για την επιλογή νέων υποψήφιων λύσεων. Αυτό έχει ως συνέπεια, οι τυχαίες κινήσεις στις πρώτες επαναλήψεις να μην οδηγούν σε σημαντικές βελτιώσεις.

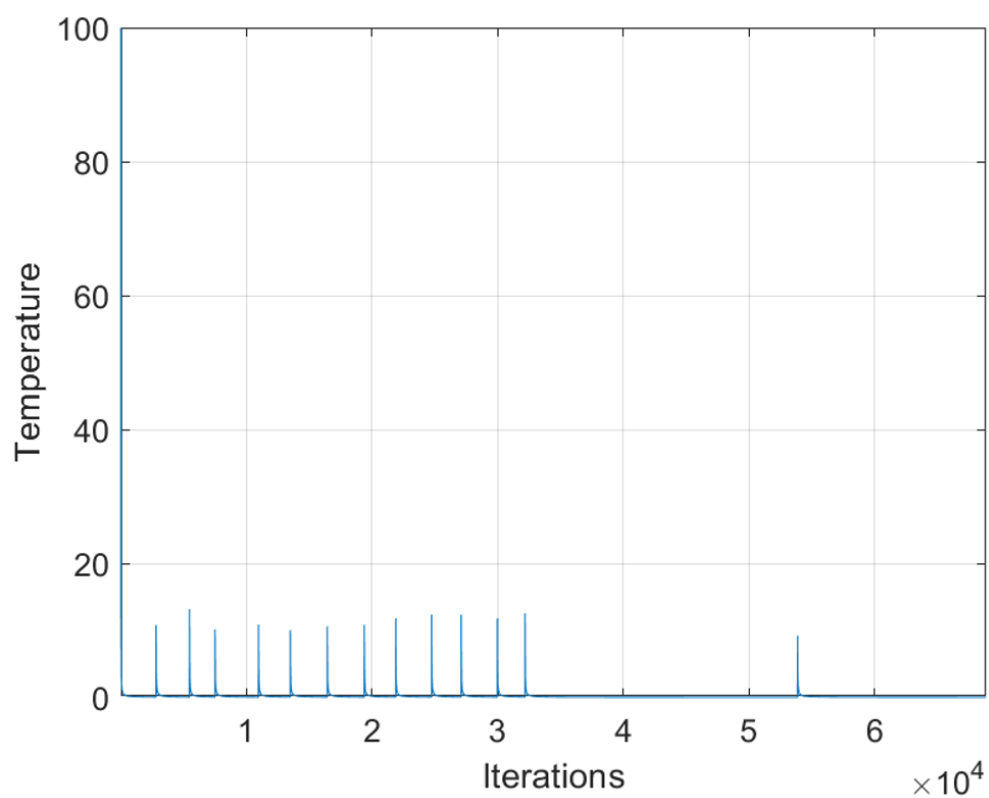
Στο διάστημα επαναλήψεων από 22,000 έως 33,000, διακρίνεται σημαντική μείωση από μια τιμή περίπου 17 σε μια τιμή περίπου 0.8. Πιο συγκεκριμένα, πραγματοποιούνται τέσσερα συμβάντα επανανόπτησης. Μετά την περιοδική αύξηση της θερμοκρασίας, ο αλγόριθμος κινείται σε περιοχές που συνεχώς μειώνουν την τιμή. Έτσι, η πτώση υποδηλώνει ότι ο αλγόριθμος έχει πέσει πάνω σε μια διαδρομή που οδηγεί σε περιοχές κοντά στο καθολικό ελάχιστο. Ως εκ τούτου, οι επιλογές των παραμέτρων για την ενημέρωση θερμοκρασίας, την επανανόπτηση και τη συνάρτηση γειτονικής λύσης στο συγκεκριμένο διάστημα, είναι κατάλληλα διαμορφωμένες.

Από το Σχήμα 3.2 μετά την πάροδο των 32,000 επαναλήψεων μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης παρατηρούνται μόνο δύο γεγονότα επανανόπτησης. Αυτό συμβαίνει διότι, η θερμοκρασία σε αυτό το διάστημα λαμβάνει χαμηλές τιμές και πραγματοποιείται η αποδοχή μόνο των νέων σημείων που βελτιώνουν την τρέχουσα τιμή. Πιο συγκεκριμένα, επειδή ο αλγόριθμος βρίσκεται σε περιοχές κοντά στο καθολικό ελάχιστο, απαιτούνται περαιτέρω επαναλήψεις για να γίνει η αποδοχή των νέων σημείων έτσι ώστε να συμβεί η επανανόπτηση. Συμπερασματικά, ο αλγόριθμος συγκλίνει σε μία λύση πολύ κοντά στο καθολικό ελάχιστο και φτάνει σε κατάσταση ισορροπίας, καθώς οι επαναλήψεις σε αυτά τα διαστήματα δεν παράγουν σημαντικές βελτιώσεις.

Η διαδικασία βελτιστοποίησης, χρησιμοποιώντας τον αλγόριθμο ΠΑ, τερματίζει στις 68,000 επαναλήψεις επαναλήψεις εξαιτίας της ανοχής `FunctionTolerance` που έχει οριστεί σε τιμή ίση με  $1 \cdot 10^{-04}$ . Συγκεκριμένα, η μέση μεταβολή στη βέλτιστη τιμή συνάρτησης κόστους, στις τελευταίες 15,000 επαναλήψεις (`MaxStallIterations`), είναι μικρότερη της ανοχής `FunctionTolerance`. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στις 53,000 επαναλήψεις με τιμή ίση με 0.057.

### 3.2. Συνάρτηση Ackley

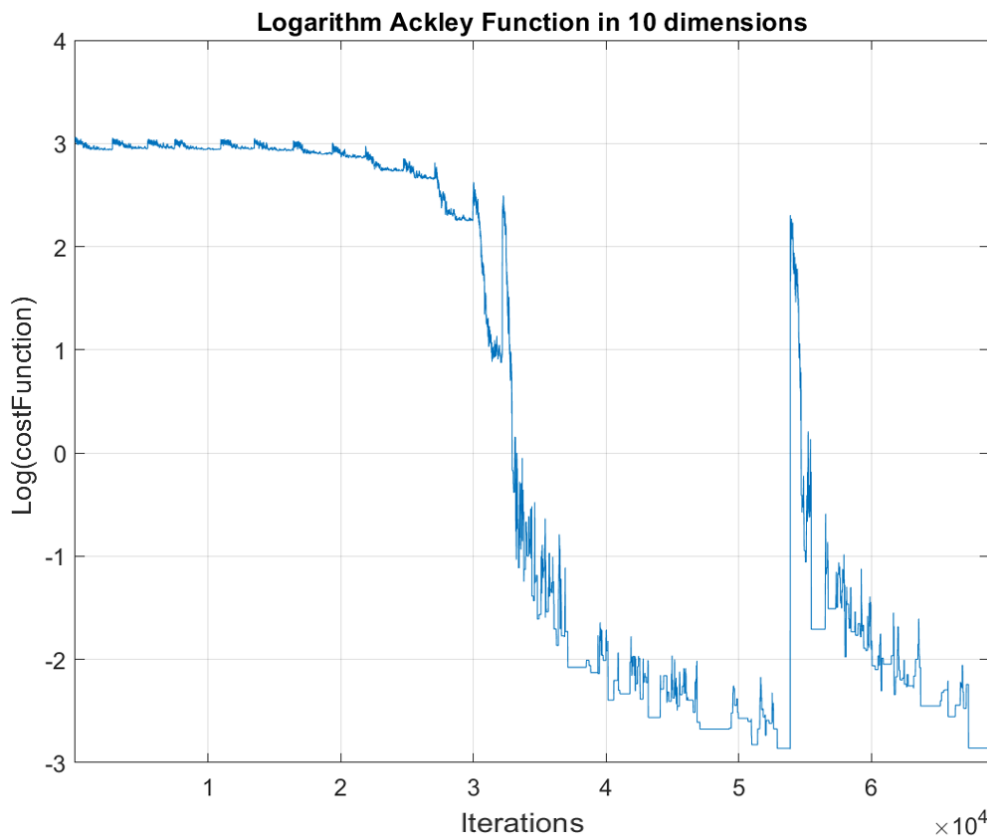
---



**Σχήμα 3.3:** Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

### 3.2. Συνάρτηση Ackley

Το Σχήμα 3.3 περιέχει τη γραφική παράσταση της παραμέτρου θερμοκρασίας για την πρώτη διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Αναλυτικότερα, οι κορυφές οφείλονται στα συμβάντα επανανόπτησης που πραγματοποιούνται σε συγκεκριμένα διαστήματα (ύστερα από την αποδοχή 650 νέων σημείων). Η αρχική θερμοκρασία έχει οριστεί σε τιμή ίση με 100. Ακόμη, μετά από κάθε κορυφή, παρατηρείται ότι η θερμοκρασία μειώνεται γρήγορα αλλά σταθερά, σύμφωνα με το πρόγραμμα ενημέρωσης θερμοκρασίας. Στις πρώτες 32,000 επαναλήψεις διακρίνεται ότι η επανανόπτηση πραγματοποιείται σε παρόμοιο αριθμό επαναλήψεων. Αντιθέτως, στο διάστημα από τις 32,000 επαναλήψεις μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης παρατηρούνται δύο συμβάντα επανανόπτησης. Αυτό το γεγονός οφείλεται στις χαμηλές τιμές θερμοκρασίας και στο ότι ο αλγόριθμος κινείται σε περιοχές κοντά στο καθολικό ελάχιστο. Αυτό έχει ως αποτέλεσμα, να απαιτούνται περαιτέρω επαναλήψεις για να γίνει η αποδοχή του καθορισμένου διαστήματος επανανόπτησης.

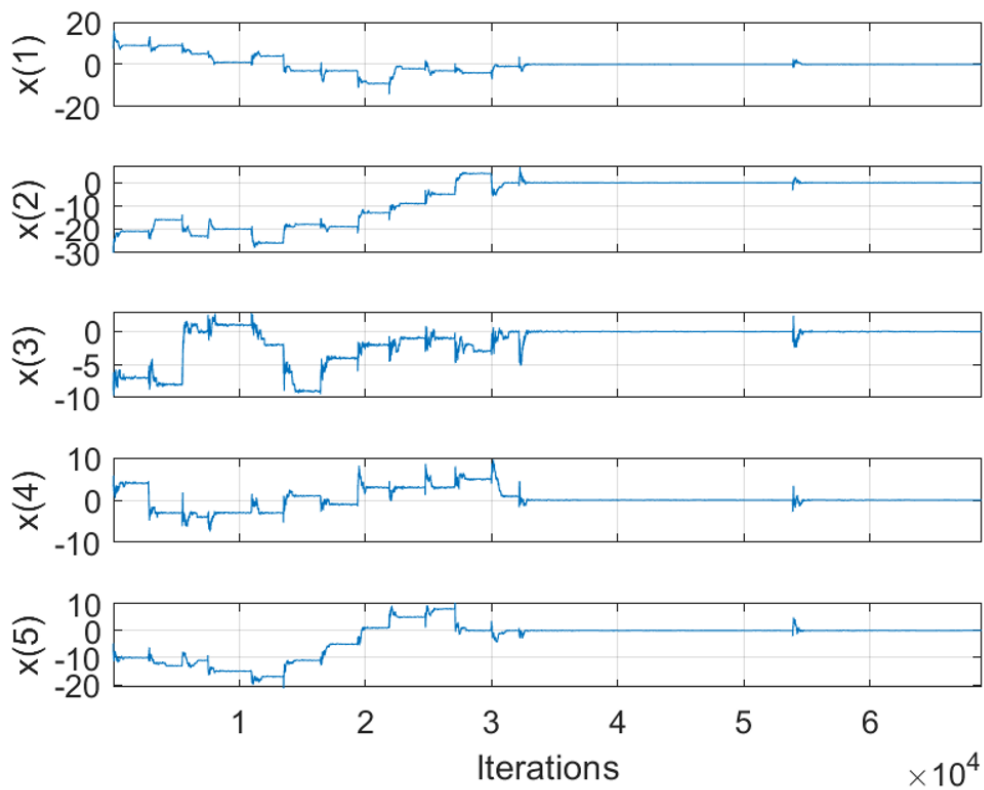


**Σχήμα 3.4:** Η εξέλιξη της τιμής συνάρτησης κόστους σε λογαριθμική κλίμακα (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο log για καλύτερα οπτικά αποτελέσματα.

Το Σχήμα 3.4 περιέχει τη γραφική παράσταση της λογαριθμικής τιμής συνάρτησης κόστους, συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ.

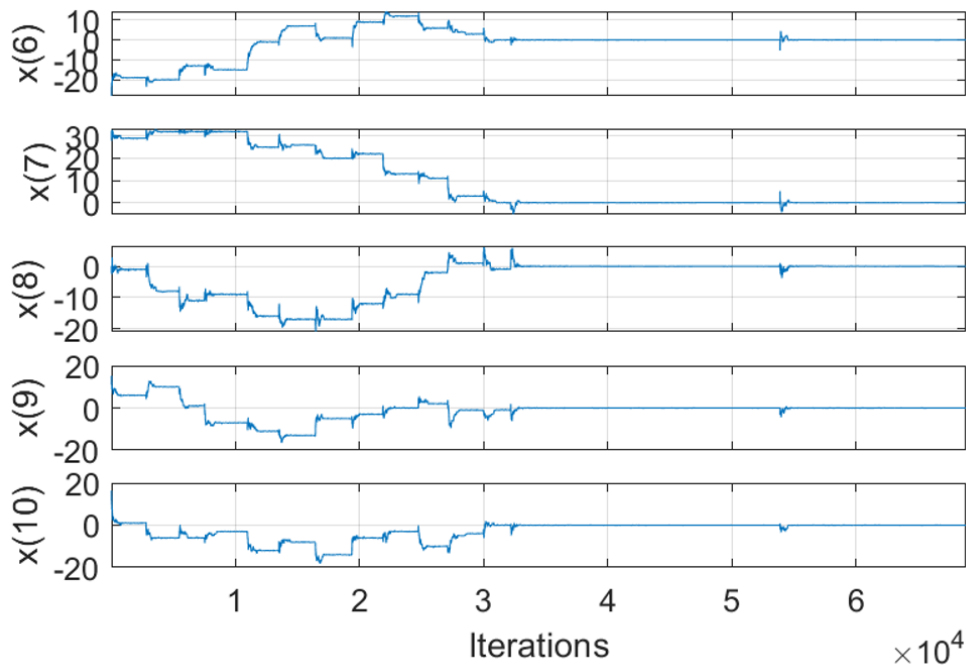
### 3.2. Συνάρτηση Ackley

Η αξιοποίηση της λογαριθμικής κλίμακας πραγματοποιείται για να διασαφηνισθεί η καλύτερη οπτική εικόνα στις αλλαγές των τιμών. Η λογαριθμική κλίμακα συμπιέζει στον κατακόρυφο άξονα μεγάλες αριθμητικές περιοχές σε ένα μικρότερο οπτικό χώρο. Έτσι, δίνεται η δυνατότητα για πιο ευκρινή παρατήρηση στο αν υπάρχει ουσιαστική μεταβολή. Στις πρώτες 27,000 επαναλήψεις διακρίνεται επίπεδη σχεδόν περιοχή με τις τιμές να μην εμφανίζουν σημαντικές βελτιώσεις. Στο διάστημα από τις 27,000 επαναλήψεις μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης, διακρίνονται βελτιώσεις με την εύρεση συνεχών λύσεων χαμηλότερου κόστους. Αυτές οι συνεχείς βελτιώσεις οδηγούν τον αλγόριθμο σε λύση που βρίσκεται κοντά στο καθολικό ελάχιστο.



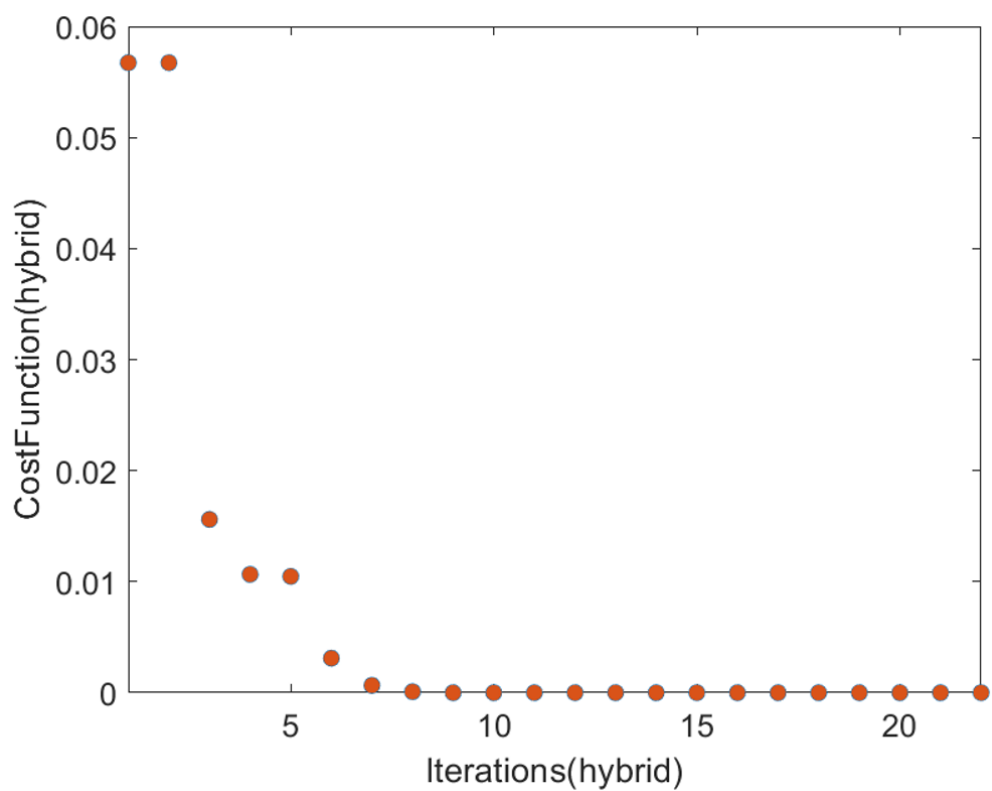
**Σχήμα 3.5:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ.

### 3.2. Συνάρτηση Ackley



Σχήμα 3.6: Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ.

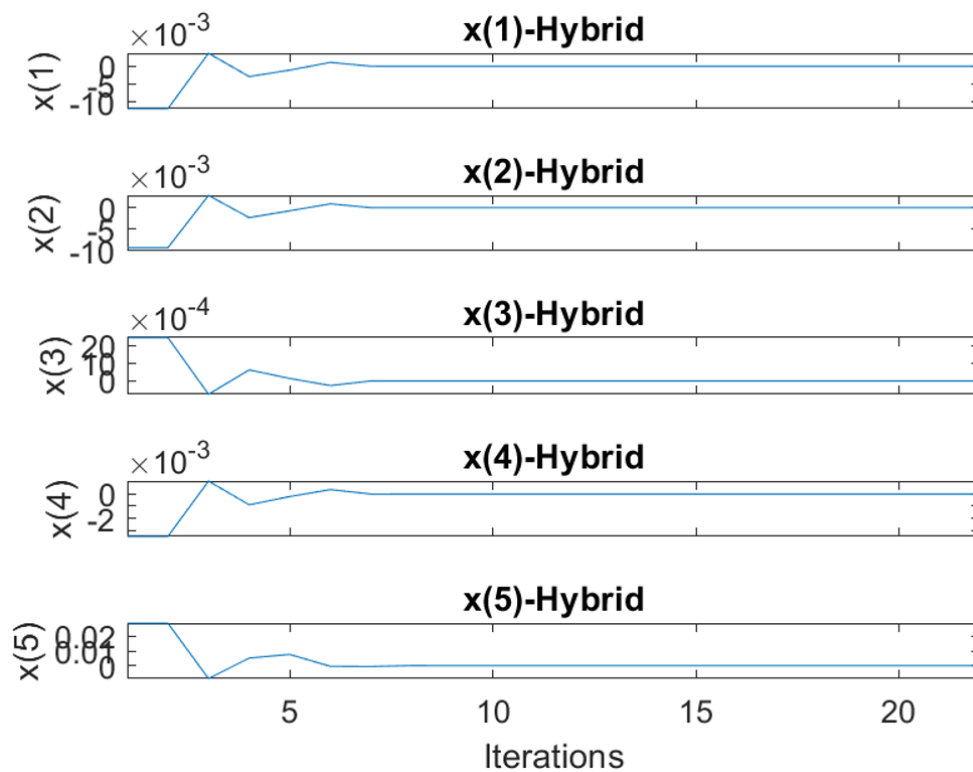
Τα σχήματα 3.5 και 3.6 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής εισόδου  $x$  σε κάθε διάσταση, συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Σε κάθε διάσταση διακρίνεται ότι στις πρώτες 32,000 επαναλήψεις, υπάρχουν μεγάλες διακυμάνσεις και συμβαίνουν συνεχώς αλλαγές στις τιμές. Αυτό συμβαίνει εξαιτίας του συνδυασμού του γρήγορου προγράμματος ενημέρωσης θερμοκρασίας και του διαστήματος επανανόπτησης. Έτσι, ο αλγόριθμος μεταβαίνει σε διαφορετικά σημεία του χώρου αναζήτησης καθώς λαμβάνει σε σύντομες επαναλήψεις και λύσεις υψηλότερου κόστους. Στο διάστημα από τις 32,000 επαναλήψεις μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης παρατηρούνται μικρές αλλαγές. Αυτό συμβαίνει διότι η θερμοκρασία λαμβάνει χαμηλές τιμές σε αυτές τις επαναλήψεις και ο αλγόριθμος αποδέχεται νέα σημεία σε κοντινή απόσταση από τα τρέχοντα. Τα άνω και κάτω όρια της μεταβλητής  $x$  σε κάθε διάσταση προσδιορίζονται από το εύρος  $[-32.7680, 32.7680]$ .



**Σχήμα 3.7:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου `fmincon`. Η `fmincon` εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

### 3.2. Συνάρτηση Ackley

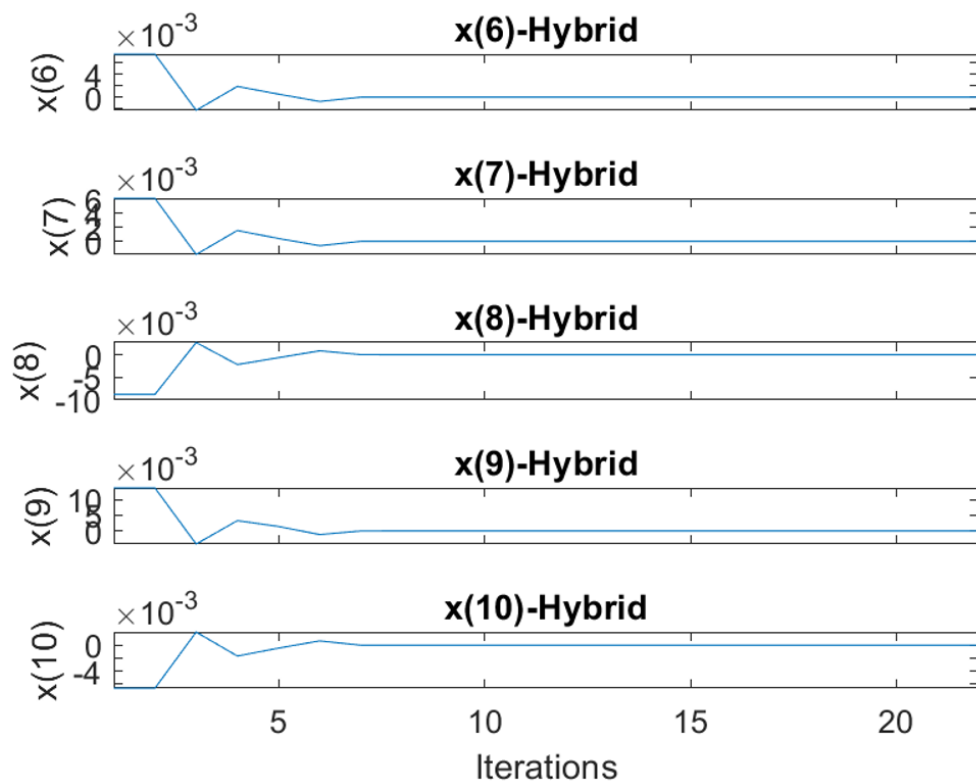
Το Σχήμα 3.7 περιέχει τη γραφική παράσταση της τιμής συνάρτησης κόστους συναρτήσει των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου `fmincon`. Στις αρχικές επαναλήψεις παρατηρείται ταχεία πτώση. Στη συνέχεια, πραγματοποιούνται ελάχιστες αλλαγές καθώς αυξάνεται ο αριθμός των επαναλήψεων. Η τιμή που λαμβάνεται είναι της τάξης  $10^{-16}$  που τυπικά θεωρείται ισοδύναμη με μηδέν, λόγω υπολογιστικών σφαλμάτων. Ο αλγόριθμος ΠΑ δίνει τη δυνατότητα στην τοπική μέθοδο `fmincon` να κινηθεί σε περιοχές που είναι κοντά στο καθολικό ελάχιστο. Έτσι, η εκτέλεσή της επιτυγχάνει λύση υψηλής ακρίβειας σε σύντομο χρονικό διάστημα. Η επιτυγχανόμενη ελάχιστη τιμή που λαμβάνεται μέσω της υβριδικής προσέγγισης είναι ίση με  $8.88 \cdot 10^{-16}$ .



**Σχήμα 3.8:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου `fmincon`.

### 3.2. Συνάρτηση Ackley

---



**Σχήμα 3.9:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου `fmincon`.



### 3.2. Συνάρτηση Ackley

Τα Σχήματα 3.8 και 3.9 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής εισόδου  $x$  σε κάθε διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου `fmincon`. Στις πρώτες επαναλήψεις, σε κάθε διάσταση διακρίνονται σημαντικές αλλαγές στις τιμές. Έπειτα, συμβαίνουν ελάχιστες αλλαγές καθώς αυξάνεται ο αριθμός των επαναλήψεων. Οι επιτυγχανόμενες τιμές των  $x$  είναι εξαιρετικά κοντά στο μηδέν και είναι των τάξεων  $10^{-17}$ ,  $10^{-18}$ .

Οι παραμετροποιήσεις που αξιοποιούνται για τα αποτελέσματα των παραπάνω σχημάτων από τις μεθόδους ΠΑ και `fmincon` φαίνονται παρακάτω:

**Πίνακας 3.1:** Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων

Παράμετρος	Επιλογή
MaxFunctionEvaluations	75,000
MaxIterations	70,000
ObjectiveLimit	$1.00 \cdot 10^{-02}$
FunctionTolerance	$1.00 \cdot 10^{-04}$
InitialTemperature	100
MaxStallIterations	15,000
TemperatureFcn	temperaturefast
ReannealInterval	650
AnnealingFcn	annealingboltz
Display	iter
DisplayInterval	1,000

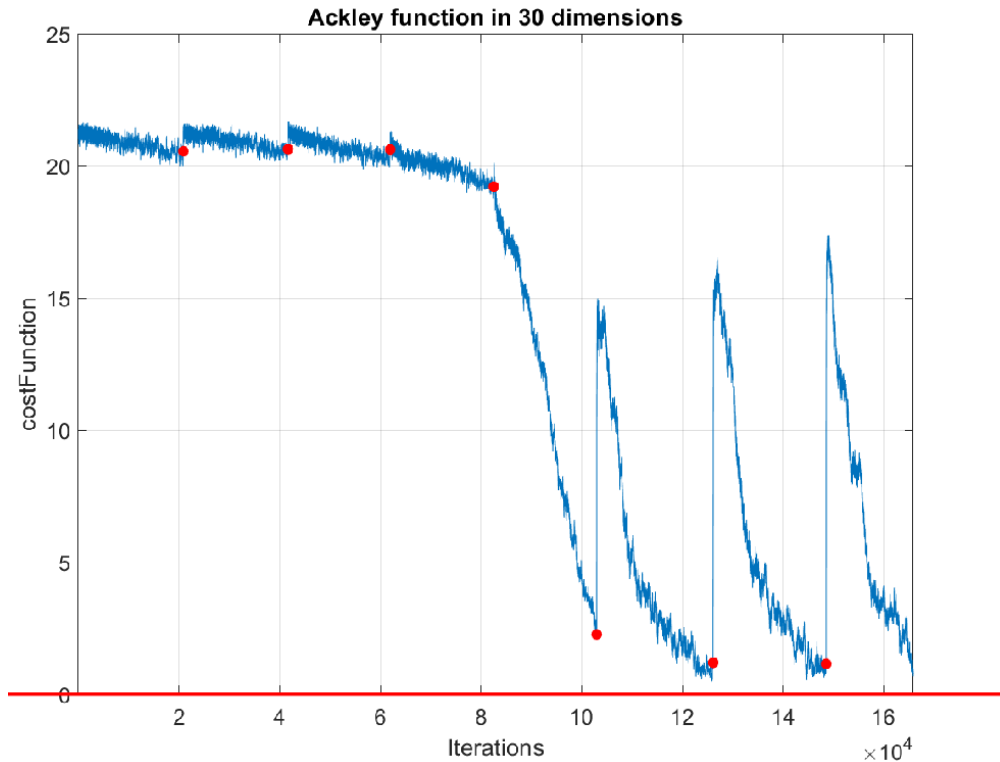
**Πίνακας 3.2:** Παράμετροι ελέγχου της τοπικής μεθόδου `fmincon` για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων

Παράμετρος	Επιλογή
Algorithm	interior-point
StepTolerance	$1.00 \cdot 10^{-15}$
SpecifyObjectiveGradient	true

### 3.2. Συνάρτηση Ackley

#### 3.2.2 Αποτελέσματα σε χώρο 30 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Ackley σε χώρο 30 διαστάσεων από την εφαρμογή των μεθόδων ΠΑ και fmincon:



**Σχήμα 3.10:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση. Η κόκκινη οριζόντια γραμμή αντιστοιχεί στην τιμή συνάρτησης κόστους του σημείου που βρίσκεται το καθολικό ελάχιστο.

Από το Σχήμα 3.10 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, επισημαίνοντας ότι ο αλγόριθμος ΠΑ ελαχιστοποιεί τη συνάρτηση Ackley με την πάροδο του χρόνου. Οι κόκκινες ακίδες στην τιμή συνάρτησης κόστους αντιστοιχούν στα συμβάντα επανανόπτησης. Ο αλγόριθμος ΠΑ εκκινεί την αναζήτηση του από τυχαίο αρχικό σημείο με τιμή ίση 21.0747.

Από το Σχήμα 3.10 παρατηρείται ότι στα τρία πρώτα συμβάντα επανανόπτησης δεν πραγματοποιείται σημαντική μείωση. Η τιμή της αρχικής θερμοκρασίας θεωρείται υψηλή. Παρόλα αυτά, χρησιμοποιείται το γραμμικό πρόγραμμα ενημέρωσής της και η θερμοκρασία σε σύντομες επαναλήψεις λαμβάνει τιμές κοντά στο μηδέν. Έτσι, ο αλγόριθμος ΠΑ δέχεται με υψηλότερη πιθανότητα λύσεις που μειώνουν την τιμή. Η πολυπλοκότητα του τοπίου της συνάρτησης στο χώρο των 30 διαστάσεων είναι υψηλή και χαρακτηρίζεται από πολλαπλά τοπικά ελάχιστα καθώς ο αλγόριθμος βρίσκεται μακριά από το καθολικό. Επιπλέον, χρησιμοποιείται υψηλό διάστημα

### 3.2. Συνάρτηση Ackley

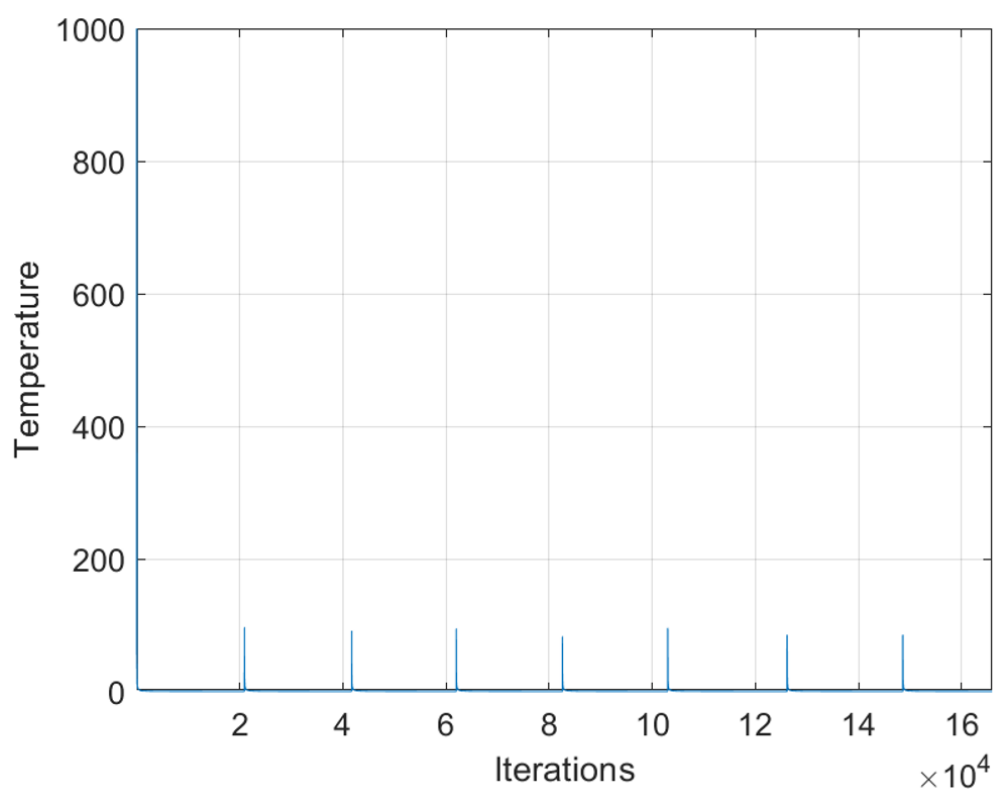
---

επανανόπτησης. Άρα, βάσει των παραπάνω, ο αλγόριθμος εστιάζει την αναζήτησή του στην διερεύνηση των περιοχών γύρω από τα τοπικά ελάχιστα και αξιοποιεί την επανανόπτηση έτσι ώστε να αποφύγει να παγιδευτεί σε αυτά.

Στο διάστημα επαναλήψεων από 82,000 έως 103,000, σημειώνεται απότομη μείωση, από τιμή περίπου 19 σε τιμή περίπου 2. Πιο συγκεκριμένα, η απότομη πτώση υποδηλώνει ότι ο αλγόριθμος έχει πέσει πάνω σε μια διαδρομή που οδηγεί σε περιοχές κοντά στο καθολικό ελάχιστο. Επομένως, οι επιλογές των παραμέτρων για την ενημέρωση θερμοκρασίας, την επανανόπτηση και τη συνάρτηση γειτονικής λύσης στο συγκεκριμένο διάστημα, είναι κατάλληλα διαμορφωμένες.

Στο Σχήμα 3.10 από τις 103,000 επαναλήψεις μέχρι τον τερματισμό της διαδικασίας, συμβαίνουν τρία συμβάντα επανανόπτησης που ακολουθούνται από σημαντικές μειώσεις. Σε κάθε εκ νέου επανανόπτηση, ο αλγόριθμος δέχεται με υψηλότερη πιθανότητα, λύσεις υψηλότερου κόστους. Έπειτα, λόγω του γραμμικού προγράμματος ενημέρωσης θερμοκρασίας, η θερμοκρασία σε σύντομες επαναλήψεις λαμβάνει χαμηλές τιμές. Έτσι, η τιμή συνάρτησης κόστους ελαχιστοποιείται συνεχώς με την πάροδο των επαναλήψεων μέχρι να συμβεί η επανανόπτηση. Συνοπτικά, ο αλγόριθμος συγκλίνει σε λύση κοντά στο καθολικό ελάχιστο και φτάνει σε κατάσταση ισορροπίας καθώς οι περαιτέρω επαναλήψεις δεν παράγουν σημαντικές βελτιώσεις.

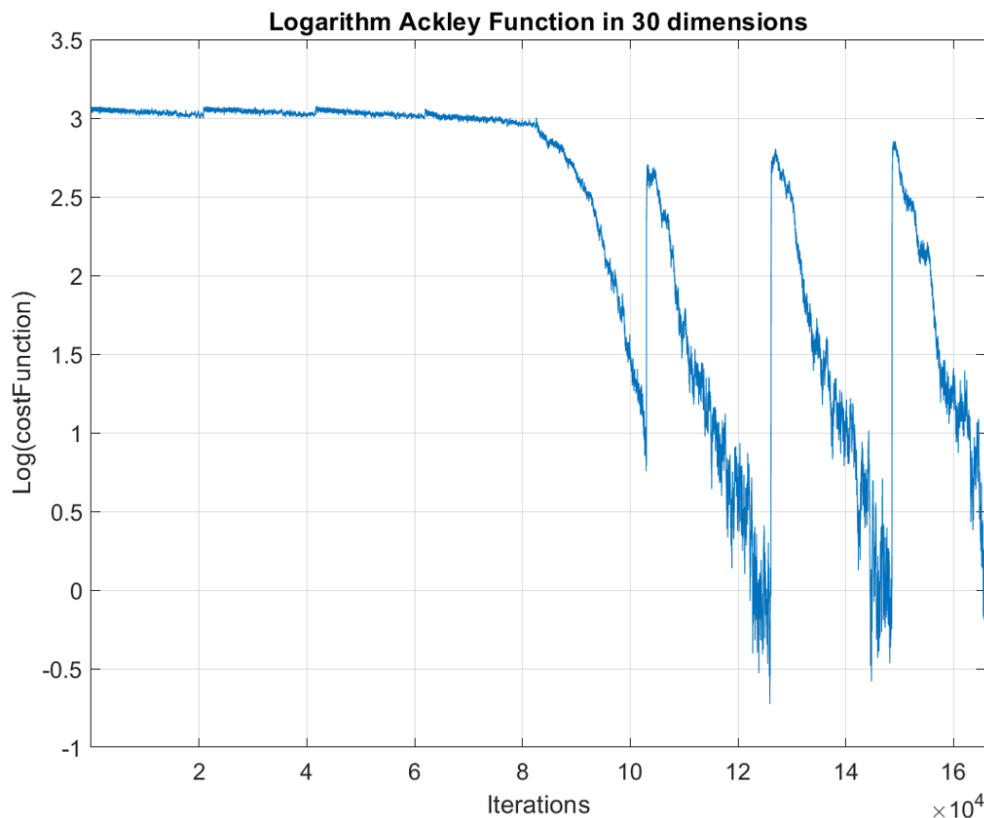
Η διαδικασία βελτιστοποίησης, χρησιμοποιώντας τον αλγόριθμο ΠΑ, τερματίζει στις 165,000 επαναλήψεις εξαιτίας της ανοχής `FunctionTolerance` που έχει οριστεί σε τιμή ίση με  $1 \cdot 10^{-04}$ . Συγκεκριμένα, η μέση μεταβολή στην βέλτιστη τιμή συνάρτησης κόστους, στις τελευταίες 40,000 επαναλήψεις (`MaxStallIterations`), είναι μικρότερη της ανοχής `FunctionTolerance`. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στις 125,000 επαναλήψεις με τιμή ίση με 0.484.



**Σχήμα 3.11:** Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

### 3.2. Συνάρτηση Ackley

Το Σχήμα 3.11 περιέχει τη γραφική παράσταση της παραμέτρου θερμοκρασίας για την πρώτη διάσταση συναρτήσεως των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Συγκεκριμένα, οι κορυφές οφείλονται στα συμβάντα επανανόπτησης που πραγματοποιούνται σε συγκεκριμένα διαστήματα (ύστερα από την αποδοχή 11,500 νέων σημείων). Η αρχική τιμή της θερμοκρασίας έχει οριστεί σε τιμή ίση με 1,000. Τα συμβάντα επανανόπτησης πραγματοποιούνται σε παρόμοιο αριθμό επαναλήψεων. Επιπλέον, μετά τις κορυφές υπάρχουν απότομες πτώσεις επειδή χρησιμοποιείται το γραμμικό πρόγραμμα ψύξης. Έτσι, η θερμοκρασία λαμβάνει τιμές κοντά στο μηδέν για εκτεταμένες επαναλήψεις. Αυτό αποσκοπεί στην αποφυγή λύσεων υψηλότερου κόστους και τη διερεύνηση συγκεκριμένων περιοχών του χώρου.

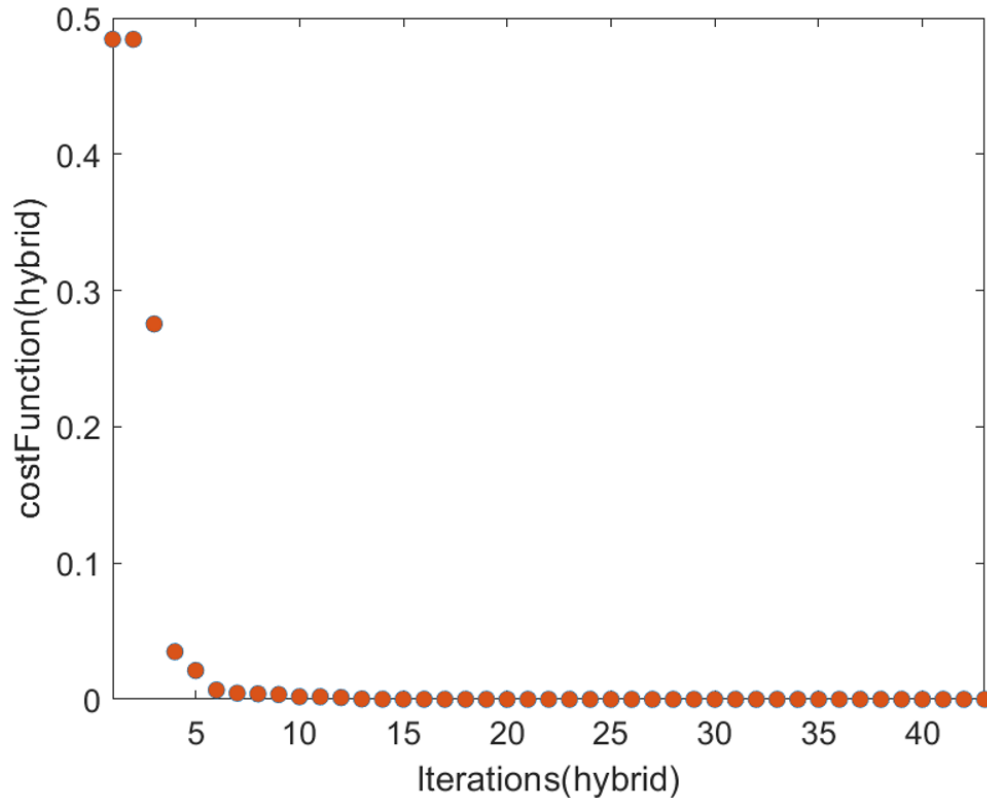


**Σχήμα 3.12:** Η εξέλιξη της τιμής συνάρτησης κόστους σε λογαριθμική κλίμακα (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσεως των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο  $\log$  για καλύτερα οπτικά αποτελέσματα.

Το Σχήμα 3.12 περιέχει τη γραφική παράσταση της λογαριθμικής τιμής συνάρτησης κόστους, συναρτήσεως των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αξιοποίηση της λογαριθμικής κλίμακας πραγματοποιείται για να διασαφηνισθεί η καλύτερη οπτική εικόνα στις αλλαγές των τιμών. Η λογαριθμική κλίμακα συμπίεζει στον κατακόρυφο άξονα μεγάλες αριθμητικές περιοχές σε ένα μικρότερο οπτικό χώρο. Επομένως, δίνεται η δυνατότητα για πιο ευκρινή παρατήρηση στο

### 3.2. Συνάρτηση Ackley

αν υπάρχει ουσιαστική μεταβολή. Στις πρώτες 82,000 επαναλήψεις παρατηρείται επίπεδη σχεδόν περιοχή με τις τιμές να μεταβάλλονται ελάχιστα. Στο διάστημα από τις 82,000 επαναλήψεις μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης, παρατηρούνται βελτιώσεις με την εύρεση συνεχών λύσεων χαμηλότερου κόστους.



**Σχήμα 3.13:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου `fmincon`. Η `fmincon` εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

Το Σχήμα 3.13 περιέχει τη γραφική παράσταση της τιμής συνάρτησης κόστους συναρτήσει των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου `fmincon`. Στις αρχικές επαναλήψεις παρατηρείται ταχεία πτώση. Στη συνέχεια, πραγματοποιούνται ελάχιστες αλλαγές καθώς αυξάνεται ο αριθμός των επαναλήψεων. Η τιμή που λαμβάνεται είναι της τάξης  $10^{-16}$  που τυπικά θεωρείται ισοδύναμη με μηδέν, λόγω υπολογιστικών σφαλμάτων. Ο αλγόριθμος ΠΑ δίνει τη δυνατότητα στην τοπική μέθοδο `fmincon` να κινηθεί σε περιοχές που είναι κοντά στο καθολικό ελάχιστο. Επομένως, η εκτέλεσή της επιτυγχάνει λύση υψηλής ακρίβειας σε σύντομο χρονικό διάστημα. Η επιτυγχανόμενη ελάχιστη τιμή που λαμβάνεται μέσω της υβριδικής προσέγγισης είναι ίση με  $8.88 \cdot 10^{-16}$ .

Οι παραμετροποιήσεις που αξιοποιούνται για τα αποτελέσματα των παραπάνω σχημάτων από την εφαρμογή των μεθόδων ΠΑ και `fmincon` φαίνονται παρακάτω:

### 3.2. Συνάρτηση Ackley

---

**Πίνακας 3.3:** Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων

Παράμετρος	Επιλογή
MaxFunctionEvaluations	220,000
MaxIterations	200,000
ObjectiveLimit	$1.00 \cdot 10^{-01}$
FunctionTolerance	$1.00 \cdot 10^{-04}$
InitialTemperature	1,000
MaxStallIterations	40,000
TemperatureFcn	temperaturefast
ReannealInterval	11,500
AnnealingFcn	annealingboltz
Display	iter
DisplayInterval	1,000

**Πίνακας 3.4:** Παράμετροι ελέγχου της τοπικής μεθόδου fmincon για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων

Παράμετρος	Επιλογή
Algorithm	interior-point
StepTolerance	$1.00 \cdot 10^{-15}$
SpecifyObjectiveGradient	true

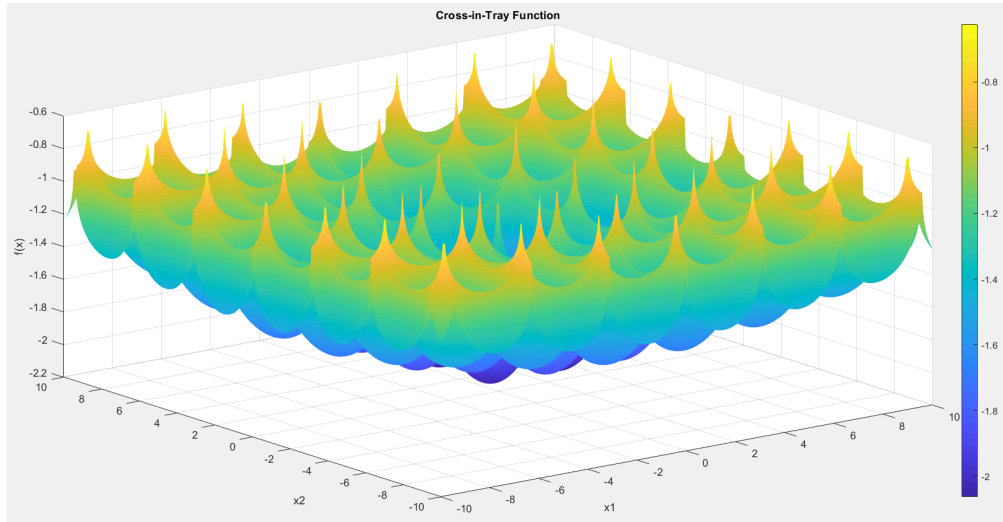
### 3.3 Συνάρτηση Cross-in-Tray

Η συνάρτηση Cross-in-Tray ορίζεται από την ακόλουθη εξίσωση:

$$f(x) = -0.0001 \left( \left| \sin(x_1) \sin(x_2) \exp \left( \left| 100 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi} \right| \right) \right| + 1 \right)^{0.1}$$

Στην εξίσωση της συνάρτησης, η παράμετρος  $x$  είναι διάνυσμα πραγματικών αριθμών και τα  $x_1, x_2$  είναι τα στοιχεία του διανύσματος  $x$ . Στις περισσότερες εφαρμογές η συνάρτηση συνήθως αξιολογείται στο κύβο με  $x_i \in [-10, 10]$ , για τα  $i = 1, 2$ .

Στις δύο διαστάσεις ( $n = 2$ ), το γράφημα της συνάρτησης Cross-in-Tray είναι μία επιφάνεια σε τρεις διαστάσεις. Οι άξονες  $x_1$  και  $x_2$  αντιπροσωπεύουν τις μεταβλητές εισόδου και ο κατακόρυφος άξονας  $z$  αντιπροσωπεύει την τιμή εξόδου της συνάρτησης,  $f(x)$ . Η συνάρτηση Cross-in-Tray είναι συνεχής, μη κυρτή, πολυτροπική και μη διαφορίσιμη συνάρτηση που έχει πολλαπλά τοπικά ελάχιστα [27]. Ακόμη, η συνάρτηση χρησιμοποιείται σε προβλήματα δοκιμής απόδοσης των μεθόδων βελτιστοποίησης με στόχο την εύρεση του καθολικού ελάχιστου. Πιο συγκεκριμένα, σε μεγενθυμένο οπτικό πεδίο, παρατηρείται ότι η μορφή της επιφάνειας εμφανίζει περιγράμματα σε σχήμα σταυρού λόγω των κορυφών και των βαθιών κοιλάδων. Η συνάρτηση παρουσιάζει τέσσερα καθολικά ελάχιστα στα σημεία  $(1.3491, 1.3491)$ ,  $(-1.3491, 1.3491)$ ,  $(1.3491, -1.3491)$ ,  $(-1.3491, -1.3491)$  με τιμή ίση με  $-2.06261$  [29]. Παρακάτω φαίνεται το διάγραμμα της συνάρτησης:



**Σχήμα 3.14:** Η αναπαράσταση της συνάρτησης Cross-in-Tray σε χώρο δύο διαστάσεων ( $n = 2$ ).

Για τη συνάρτηση είναι εφικτή η τροποποίησή της, έτσι ώστε να χρησιμοποιηθεί σε προβλήματα βελτιστοποίησης για διάσταση μεγαλύτερη από δύο. Η εξίσωση της



### 3.3. Συνάρτηση Cross-in-Tray

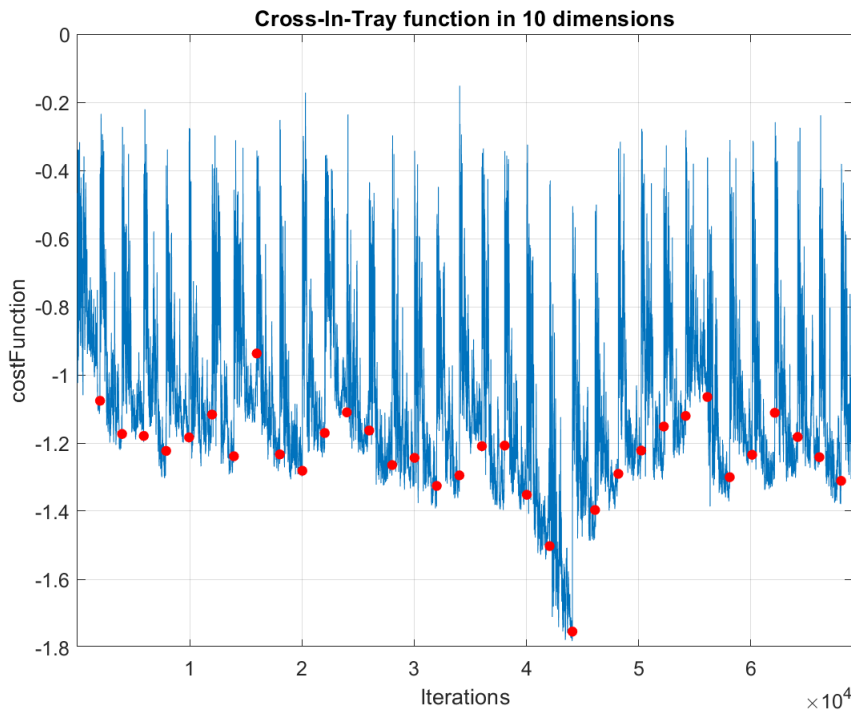
συνάρτησης Cross-in-Tray για διάσταση μεγαλύτερη από δύο ορίζεται ως:

$$f(\mathbf{x}) = -0.0001 \cdot \left( \left| \prod_{i=1}^n \sin(x_i) \exp \left( \left| 100 - \frac{\sqrt{\sum_{i=1}^n x_i^2}}{\pi} \right| \right) \right| + 1 \right)^{0.1}$$

Στην εξίσωση της συνάρτησης η παράμετρος  $x$  είναι διάνυσμα πραγματικών αριθμών και το  $x_i$  είναι το  $i$ -οστό στοιχείο του διανύσματος  $x$ . Για την συνάρτηση Cross-In-Tray σε διάσταση μεγαλύτερη από δύο δεν είναι γνωστή η θέση ή οι θέσεις και η τιμή του καθολικού ελαχίστου.

#### 3.3.1 Αποτελέσματα σε χώρο 10 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Cross-In-Tray σε χώρο 10 διαστάσεων από την εφαρμογή των μεθόδων ΠΑ και patternsearch:



**Σχήμα 3.15:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

Από το Σχήμα 3.15 παρατηρείται ότι η γραφική παράσταση εμφανίζει πολλές διακυμάνσεις μεταξύ λύσεων υψηλότερου και χαμηλότερου κόστους καθ'όλη τη διάρκεια της διαδικασίας βελτιστοποίησης. Βέβαια, ο αλγόριθμος ΠΑ ελαχιστοποιεί τη συνάρτηση Cross-In-Tray με την πάροδο του χρόνου. Οι κόκκινες ακίδες στην τιμή συνάρτησης κόστους αντιστοιχούν στα συμβάντα επανανόπτησης. Ο αλγόριθμος ΠΑ εκκινεί την αναζήτησή του από τυχαίο αρχικό σημείο με τιμή ίση  $-0.4578$ .

### 3.3. Συνάρτηση Cross-in-Tray

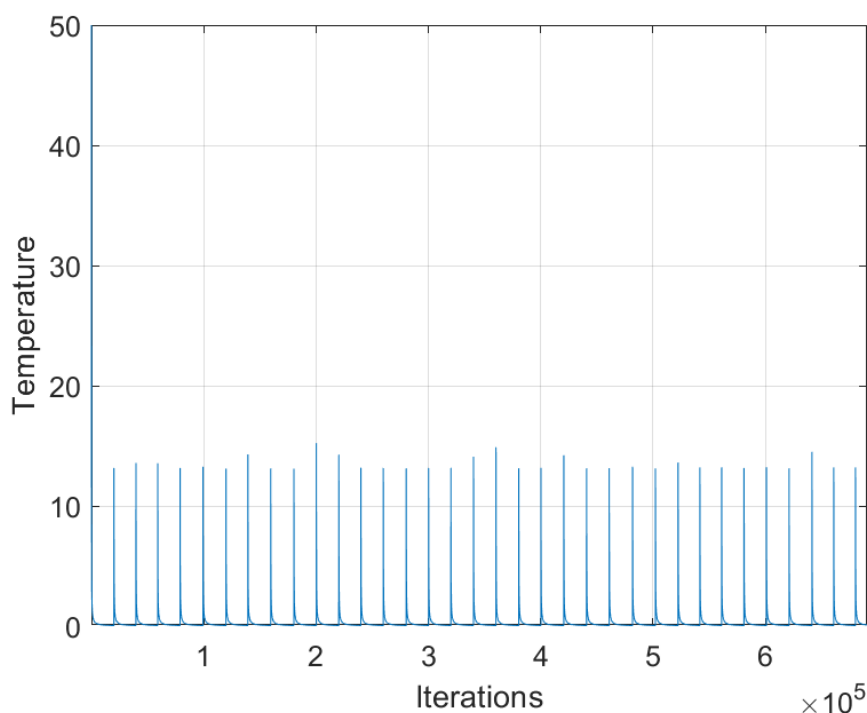
Έχει αναφερθεί ότι για τη συνάρτηση Cross-In-Tray, για διάσταση μεγαλύτερη από δύο, δεν υπάρχει γνώση για τη θέση ή τις θέσεις και την τιμή του καθολικού ελαχίστου. Ακόμη, και στις δύο διαστάσεις η συνάρτηση διακρίνεται από πολύπλοκο τοπίο με πολλά τοπικά ελάχιστα και περιορίζεται σε μικρό χώρο αναζήτησης (βλέπε ενότητα (3.3)). Έτσι, με την αύξηση των διαστάσεων σε δέκα, αυξάνεται εκθετικά η πολυπλοκότητα του χώρου αναζήτησης. Επομένως, για την αποφυγή παγίδευσης σε τοπικό ελάχιστο, χρησιμοποιείται ο συνδυασμός γρήγορου προγράμματος φύξης με ένα σχετικά μικρό διάστημα επανανόπτησης.

Από το Σχήμα 3.15 διακρίνονται κορυφές σε σύντομες επαναλήψεις που αυξάνουν την τιμή συνάρτησης κόστους. Αναλυτικότερα, ύστερα από κάθε συμβάν επανανόπτησης ο αλγόριθμος δέχεται με υψηλότερη πιθανότητα λύσεις υψηλότερου κόστους. Η τιμή της αρχικής θερμοκρασίας θεωρείται σχετικά υψηλή. Ωστόσο, χρησιμοποιείται το γραμμικό πρόγραμμα ενημέρωσής της και η θερμοκρασία σε σύντομες επαναλήψεις λαμβάνει τιμές κοντά στο μηδέν. Έτσι, η τιμή συνάρτησης κόστους ελαχιστοποιείται συνεχώς με την πάροδο των επαναλήψεων μέχρι να συμβεί η επανανόπτηση.

Από το Σχήμα 3.15 παρατηρείται ότι στις πρώτες περίπου 44,000 επαναλήψεις η τιμή συνάρτησης κόστους εμφανίζει πτωτική τάση. Ειδικότερα, παρά το γεγονός των αυξήσεων λόγω συμβάντων επανανόπτησης, διακρίνεται συνεχής βελτίωση σε λύσεις χαμηλότερου κόστους. Αντίθετα, από την πάροδο των 44,000 επαναλήψεων μέχρι τον τερματισμό της διαδικασίας, ο αλγόριθμος κινείται σε περιοχές λύσεων υψηλότερου κόστους που δε βελτιώνουν την τρέχουσα ελάχιστη τιμή.

Στο διάστημα επαναλήψεων από 38,000 έως 44,000, σημειώνεται σημαντική μείωση από τιμή περίπου  $-1.2$  σε τιμή περίπου  $-1.78$ . Συγκεκριμένα, η μείωση υποδηλώνει ότι ο αλγόριθμος έχει πέσει πάνω σε μια διαδρομή που οδηγεί σε περιοχές βελτιωμένης απόδοσης. Ως εκ τούτου, οι επιλογές των παραμέτρων για την ενημέρωση θερμοκρασίας, την επανανόπτηση και τη συνάρτηση γειτονικής λύσης, στο συγκεκριμένο διάστημα είναι κατάλληλα διαμορφωμένες.

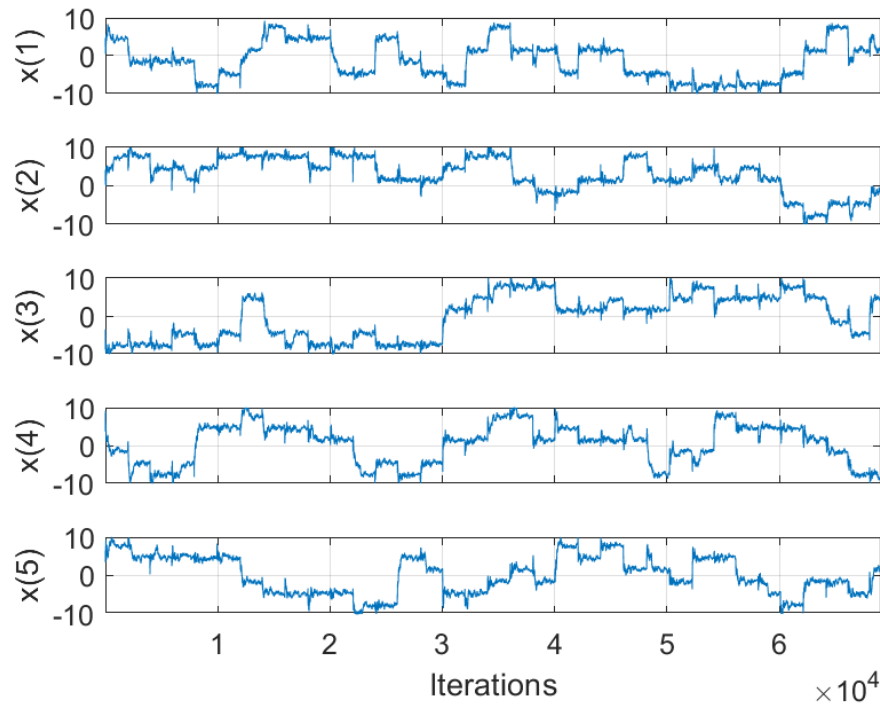
Η διαδικασία βελτιστοποίησης, χρησιμοποιώντας τον αλγόριθμο ΠΑ, τερματίζει στις 69,000 επαναλήψεις εξαιτίας της ανοχής FunctionTolerance που έχει οριστεί σε τιμή ίση με  $1 \cdot 10^{-04}$ . Αυτό σημαίνει ότι η μέση μεταβολή στη βέλτιστη τιμή, στις τελευταίες 25,000 επαναλήψεις (MaxStallIterations), είναι μικρότερη της ανοχής FunctionTolerance. Πιο συγκεκριμένα, η βέλτιστη τιμή παρέμεινε αμετάβλητη σε αυτές τις επαναλήψεις. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στις 44,000 επαναλήψεις με τιμή ίση με  $-1.78$ .



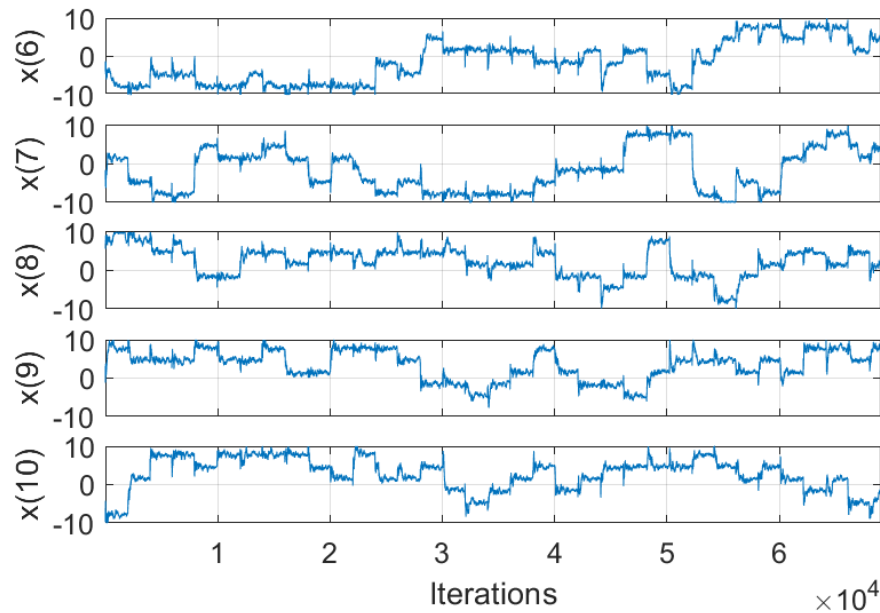
**Σχήμα 3.16:** Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

Το Σχήμα 3.16 περιέχει τη γραφική παράσταση της παραμέτρου θερμοκρασίας για την πρώτη διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Σε εκτενέστερη ανάλυση, οι κορυφές οφείλονται στα συμβάντα επανανόπτησης που πραγματοποιούνται σε συγκεκριμένα διαστήματα (ύστερα από την αποδοχή 1,300 νέων σημείων). Η αρχική θερμοκρασία έχει οριστεί σε τιμή ίση με 100. Έπειτα, μετά τις κορυφές, υπάρχουν απότομες πτώσεις επειδή χρησιμοποιείται το γραμμικό πρόγραμμα ενημέρωσης της θερμοκρασίας. Έτσι, η θερμοκρασία λαμβάνει τιμές κοντά στο μηδέν για εκτεταμένες επαναλήψεις. Τα συμβάντα επανανόπτησης πραγματοποιούνται σε παρόμοιο αριθμό επαναλήψεων. Το γράφημα δεν παρουσιάζει κάποια απότομη και ξαφνική αλλαγή, αλλά δείχνει ένα σταθερό και περιοδικό πρόγραμμα ψύξης.

### 3.3. Συνάρτηση Cross-in-Tray



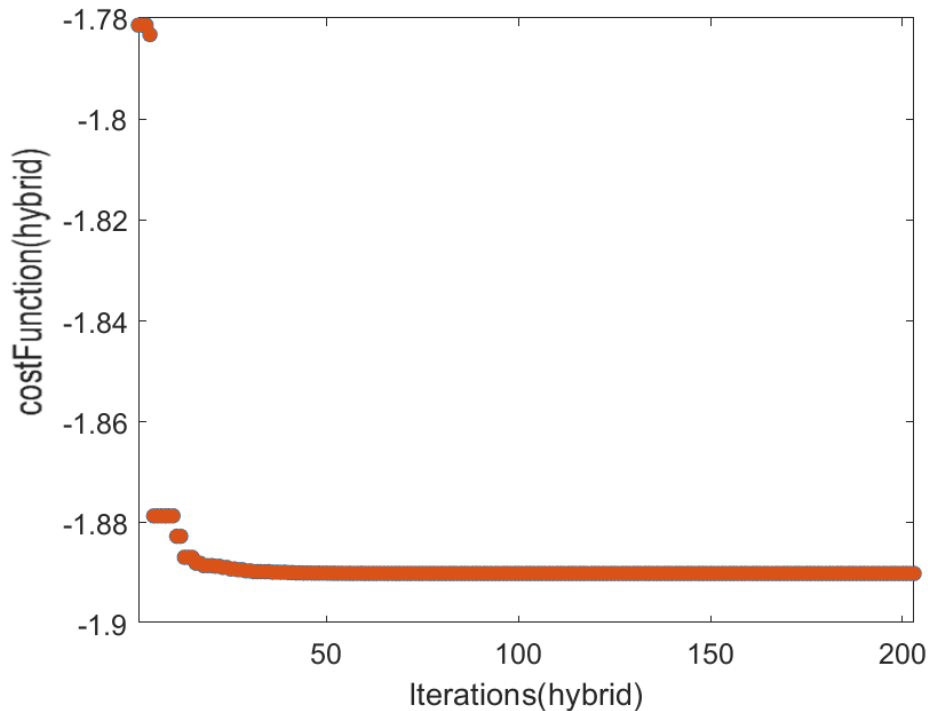
**Σχήμα 3.17:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ.



**Σχήμα 3.18:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ.

### 3.3. Συνάρτηση Cross-in-Tray

Τα σχήματα 3.17 και 3.18 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής εισόδου  $x$  σε κάθε διάσταση συναρτήσεως των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Σε κάθε διάσταση διακρίνεται ότι σε ορισμένες διαδοχικές επαναλήψεις, συμβαίνουν μικρές αλλαγές μεταξύ των τιμών τους. Αυτό συμβαίνει διότι η θερμοκρασία λαμβάνει χαμηλές τιμές σε αυτές τις επαναλήψεις και ο αλγόριθμος αποδέχεται νέα σημεία σε κοντινή απόσταση από τα τρέχοντα. Αντιθέτως, σε κάθε διάσταση μεταξύ ορισμένων διαδοχικών επαναλήψεων, διακρίνονται απότομες αλλαγές. Αυτό οφείλεται στο γεγονός ότι η επανανόπτηση οδηγεί τον αλγόριθμο σε νέες περιοχές του χώρου αναζήτησης. Έτσι, τα σημεία είναι αρκετά διαφορετικά από τα προηγούμενα, εφόσον λαμβάνει και λύσεις υψηλότερου κόστους. Τα άνω και κάτω όρια της μεταβλητής  $x$  σε κάθε διάσταση προσδιορίζονται από το εύρος  $[-10, 10]$ .

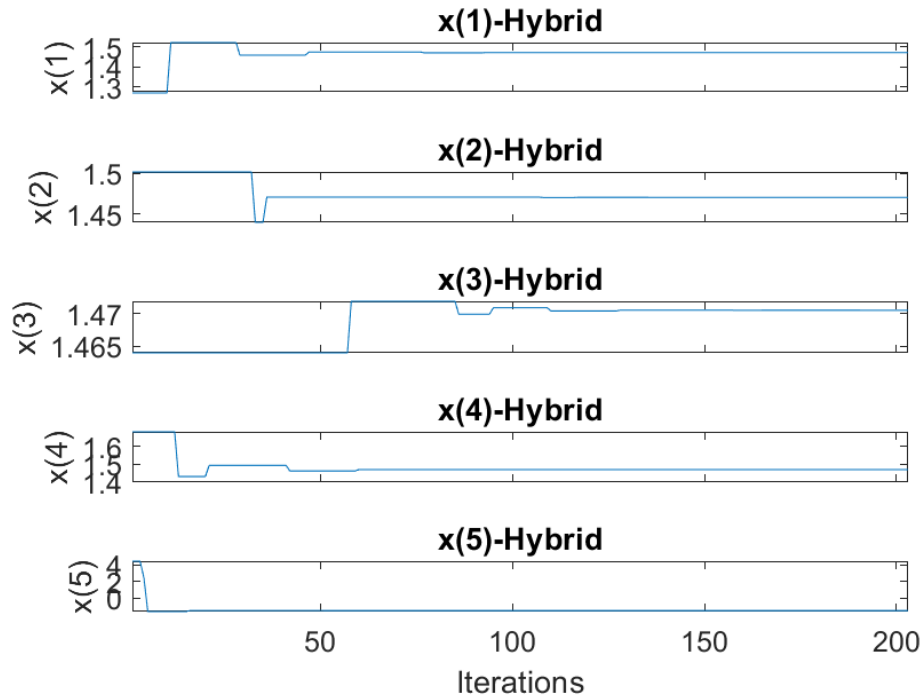


**Σχήμα 3.19:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσεως των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. Η patternsearch εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

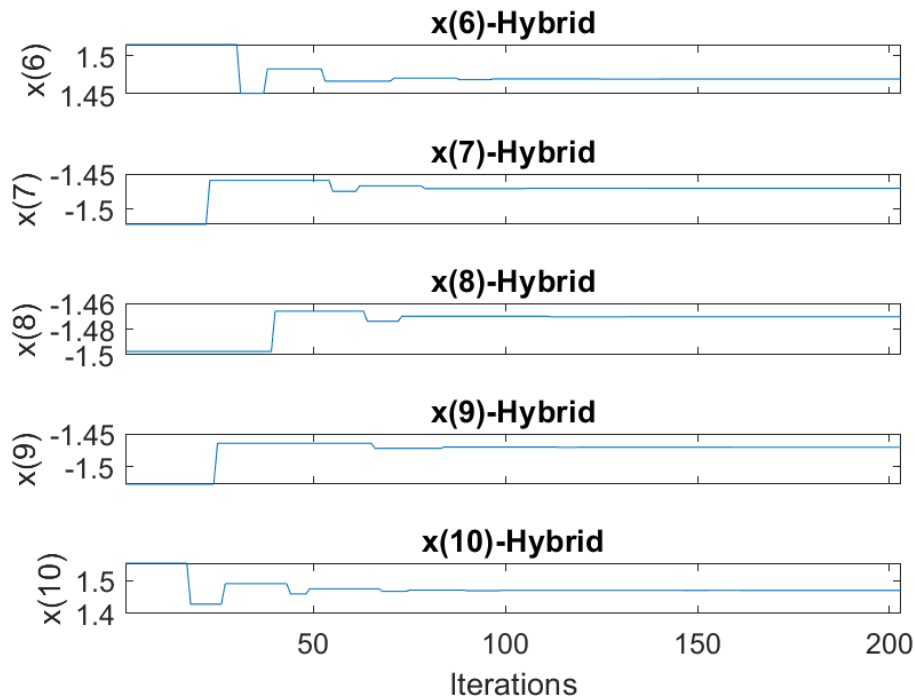
Το Σχήμα 3.19 περιέχει τη γραφική παράσταση της τιμής συνάρτησης κόστους συναρτήσεως των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου patternsearch. Στις αρχικές επαναλήψεις συμβαίνει γρήγορη μείωση, ενώ καθώς αυξάνεται ο αριθμός των επαναλήψεων, η τιμή παραμένει σταθερή. Έτσι, επιτυγχάνεται μείωση σε σύντομο χρονικό διάστημα. Η απόδοση βελτιστοποίησης από την εκτέλεση της τοπικής μεθόδου, εξαρτάται σε μεγάλο βαθμό από το σημείο εκκίνησης που λαμβάνει

### 3.3. Συνάρτηση Cross-in-Tray

από τον αλγόριθμο ΠΑ. Ο αλγόριθμος ΠΑ δίνει τη δυνατότητα στην τοπική μέθοδο να κινηθεί σε περιοχές βελτιωμένης απόδοσης ως προς την τιμή συνάρτησης κόστους. Η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $-1.89$ .



**Σχήμα 3.20:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch.



**Σχήμα 3.21:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch.

Τα σχήματα 3.20 και 3.21 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής εισόδου  $x$  σε κάθε διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου patternsearch. Στις πρώτες επαναλήψεις, σε κάθε διάσταση, διακρίνονται σημαντικές αλλαγές στις τιμές. Ακολούθως, η μέθοδος συγκλίνει σε μία τιμή. Οι επιτυγχανόμενες τιμές των  $x$  είναι ίσες με 1.4705 και  $-1.4705$ .

Για την τοπική μέθοδο patternsearch, αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση της μεθόδου ΠΑ για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:

### 3.3. Συνάρτηση Cross-in-Tray

---

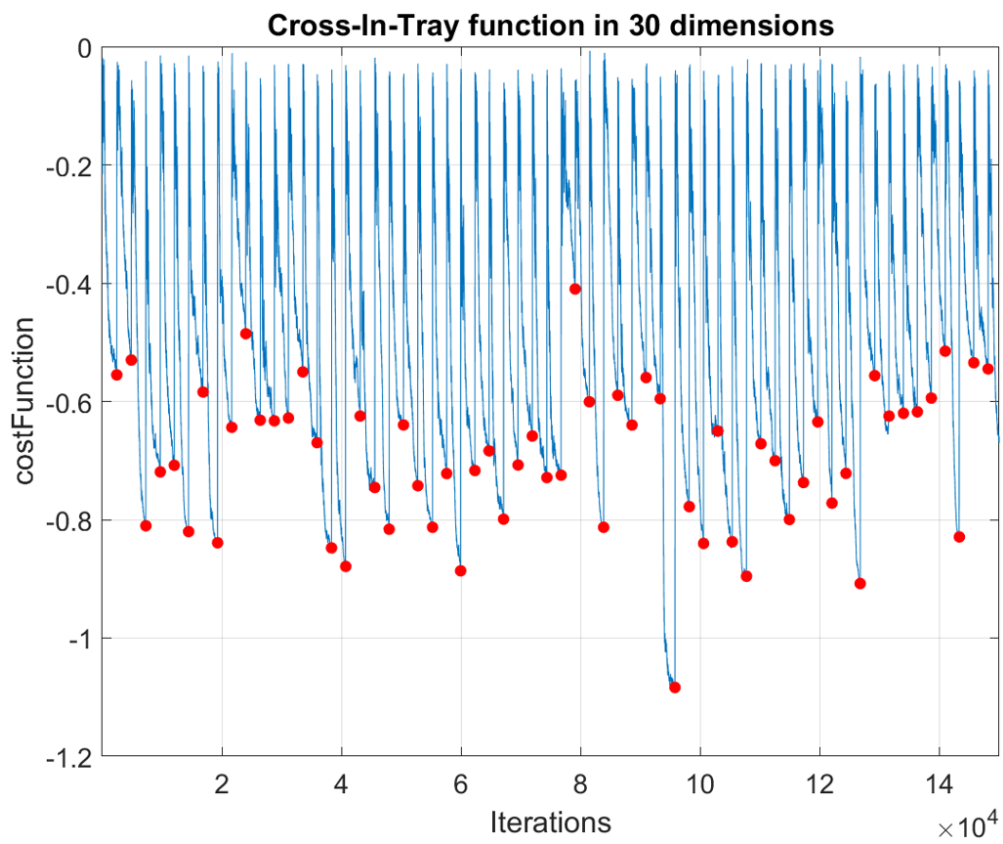
**Πίνακας 3.5:** Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τη συνάρτηση Cross-In-Tray στις 10 διαστάσεις

Παράμετρος	Επιλογή
MaxFunctionEvaluations	85,000
MaxIterations	80,000
ObjectiveLimit	-1.80
FunctionTolerance	$1.00 \cdot 10^{-04}$
InitialTemperature	100
MaxStallIterations	25,000
TemperatureFcn	temperaturefast
ReannealInterval	1,300
AnnealingFcn	annealingboltz
Display	iter
DisplayInterval	1,000

#### 3.3.2 Αποτελέσματα σε χώρο 30 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Cross-In-Tray σε χώρο 30 διαστάσεων από την εφαρμογή των μεθόδων ΠΑ και patternsearch:





**Σχήμα 3.22:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

### 3.3. Συνάρτηση Cross-in-Tray

Στο Σχήμα 3.22 παρατηρείται ότι η γραφική παράσταση εμφανίζει πολλές διακυμάνσεις μεταξύ λύσεων υψηλότερου και χαμηλότερου κόστους καθ' όλη την διάρκεια της διαδικασίας βελτιστοποίησης. Ο αλγόριθμος ΠΑ εκκινεί την αναζήτησή του από τυχαίο αρχικό σημείο με τιμή ίση  $-0.047$ .

Έχει αναφερθεί ότι για τη συνάρτηση Cross-In-Tray, για διάσταση μεγαλύτερη από δύο, δεν υπάρχει γνώση για τη θέση ή τις θέσεις και την τιμή του καθολικού ελαχίστου. Ακόμη, και στις δύο διαστάσεις η συνάρτηση διακρίνεται από πολύπλοκο τοπίο με πολλά τοπικά ελάχιστα και περιορίζεται σε μικρό χώρο αναζήτησης (βλέπε ενότητα (3.3)). Έτσι, με την αύξηση του αριθμού των διαστάσεων σε τριάντα, αυξάνεται εκθετικά η πολυπλοκότητα του χώρου αναζήτησης. Αυτό έχει ως αποτέλεσμα, να καθίσταται δύσκολη η εύρεση της κατάλληλης παραμετροποίησης για τον αλγόριθμο ΠΑ. Στο συγκεκριμένο πρόβλημα ο αλγόριθμος ΠΑ δεν κατάφερε να βρει λύση υψηλής ακρίβειας, όπως ο αλγόριθμος καθολικής αναζήτησης.

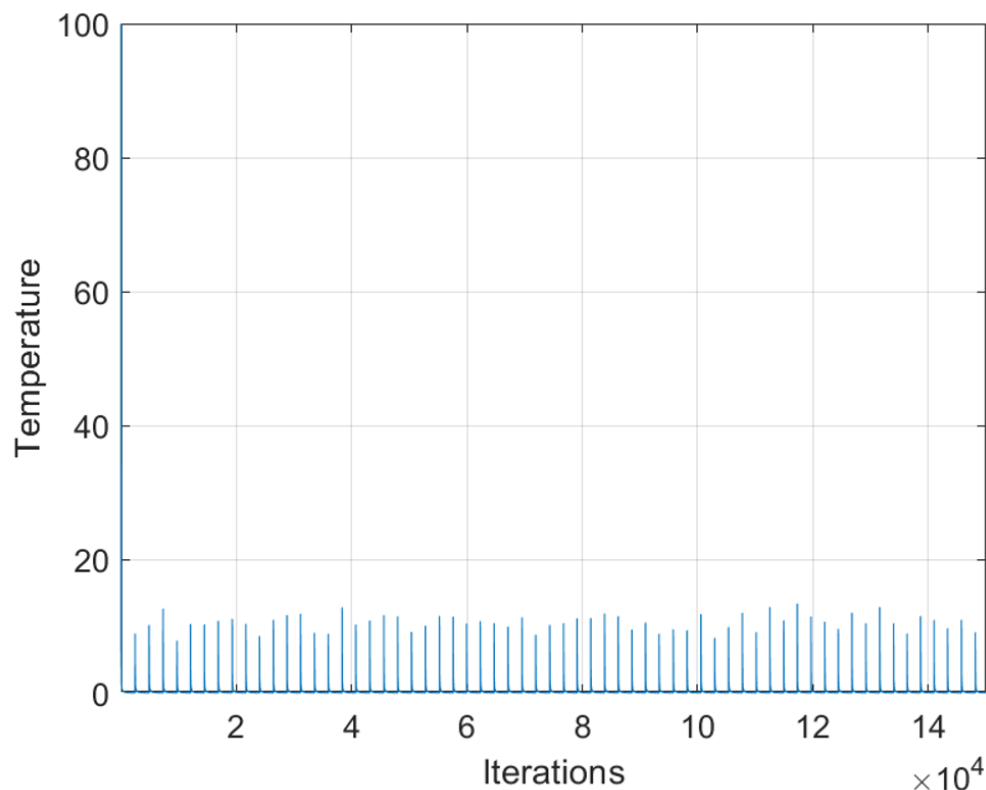
Οι παράμετροι ελέγχου καθορίζουν τη συμπεριφορά και την αποτελεσματική αναζήτηση της ΠΑ στην εύρεση της ελάχιστης τιμής. Η παραμετροποίηση που χρησιμοποιείται στο συγκεκριμένο πρόβλημα δεν είναι αποδοτική. Από το Σχήμα 3.22 παρατηρούνται κορυφές σε σύντομες επαναλήψεις που αυξάνουν την τιμή μετά από κάθε συμβάν επανανόπτησης. Έτσι, με την αύξηση της θερμοκρασίας του συστήματος, αυξάνεται η πιθανότητα αποδοχής λύσεων υψηλότερου κόστους. Το διάστημα επανανόπτησης επιλέγεται σχετικά μικρό για την αποφυγή παγίδευσης σε κάποιο τοπικό ελάχιστο.

Από το Σχήμα 3.22, ύστερα από κάθε συμβάν επανανόπτησης, διακρίνεται πτωτική πορεία. Η τιμή της αρχικής θερμοκρασίας θεωρείται σχετικά υψηλή. Ωστόσο, χρησιμοποιείται το γραμμικό πρόγραμμα ενημέρωσής της και η θερμοκρασία σε σύντομες επαναλήψεις λαμβάνει τιμές κοντά στο μηδέν. Έτσι, η τιμή της συνάρτησης κόστους ελαχιστοποιείται συνεχώς με την πάροδο των επαναλήψεων μέχρι να συμβεί η επανανόπτηση. Για τη δημιουργία των νέων σημείων χρησιμοποιείται η επιλογή annealingfast. Σε αυτήν την περίπτωση, οι αποστάσεις των νέων σημείων από τα τρέχοντα είναι μεγαλύτερες όσο η θερμοκρασία παραμένει σε υψηλά στάδια. Όμως, λόγω του γραμμικού προγράμματος ψύξης, τα νέα σημεία δημιουργούνται σε κοντινή απόσταση από τα τρέχοντα για εκτεταμένες επαναλήψεις. Αυτό έχει ως αποτέλεσμα, ο αλγόριθμος να παγιδεύεται σε τοπικά ελάχιστα χαμηλής ακρίβειας ως προς την τιμή συνάρτησης κόστους.

Η διαδικασία βελτιστοποίησης, χρησιμοποιώντας τον αλγόριθμο ΠΑ, τερματίζει στις 150,000 επαναλήψεις εξαιτίας της εντολής MaxIterations. Έτσι, ο αλγόριθμος τερματίζει την εκτέλεσή του μετά τον προκαθορισμένο αριθμό επαναλήψεων. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στις 96,000 επαναλήψεις με τιμή ίση με

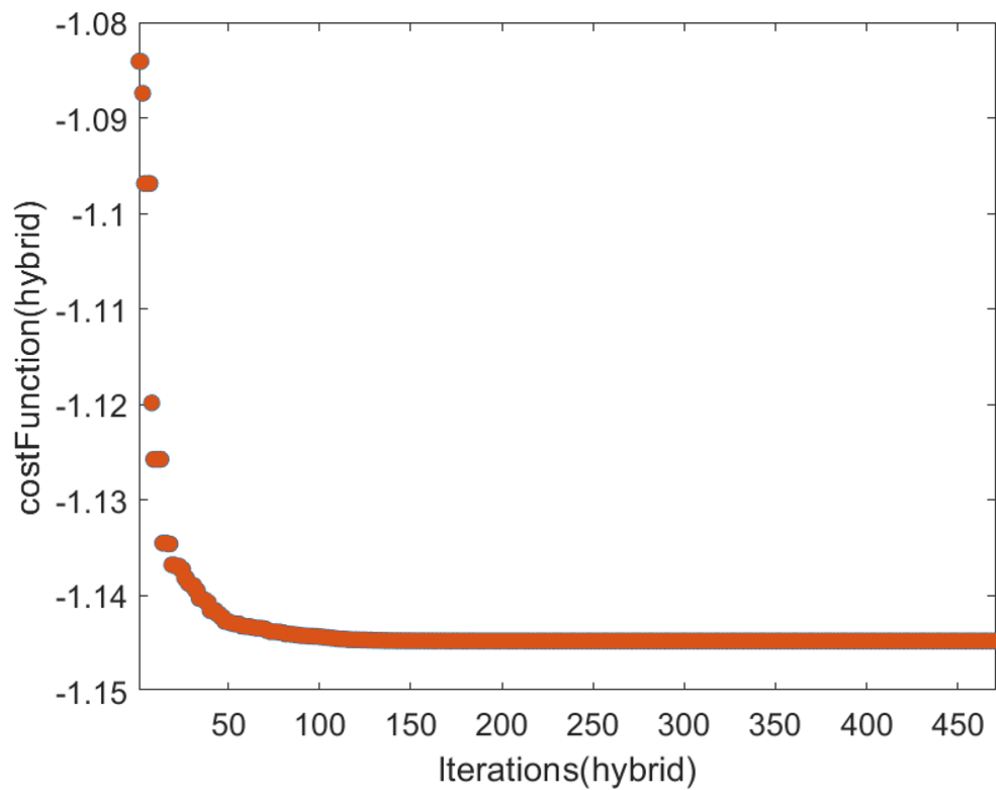
### 3.3. Συνάρτηση Cross-in-Tray

−1.08.



**Σχήμα 3.23:** Η εξέλιξη της τιμής θερμοκρασίας για την πρώτη διάσταση (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Οι κορυφές υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η επανανόπτηση.

Το Σχήμα 3.23 περιέχει τη γραφική παράσταση της παραμέτρου θερμοκρασίας για την πρώτη διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Συγκεκριμένα, οι κορυφές οφείλονται στα συμβάντα επανανόπτησης που πραγματοποιούνται σε συγκεκριμένα διαστήματα (ύστερα από την αποδοχή 1,650 νέων σημείων). Η αρχική θερμοκρασία έχει οριστεί σε τιμή ίση με 100. Τα συμβάντα επανανόπτησης πραγματοποιούνται σε παρόμοιο αριθμό επαναλήψεων. Μετά από κάθε κορυφή, παρατηρείται ότι η θερμοκρασία μειώνεται γρήγορα αλλά σταθερά, σύμφωνα με το πρόγραμμα ψύξης.



**Σχήμα 3.24:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου patternsearch. Η patternsearch εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

### 3.3. Συνάρτηση Cross-in-Tray

Το Σχήμα 3.24 περιέχει τη γραφική παράσταση της τιμής συνάρτησης κόστους συναρτήσει των επαναλήψεων από την εκτέλεση της τοπικής μεθόδου patternsearch. Στις αρχικές επαναλήψεις πραγματοποιείται μικρή σχετικά μείωση, ενώ καθώς αυξάνεται ο αριθμός των επαναλήψεων, η τιμή παραμένει σταθερή. Η απόδοση βελτιστοποίησης από την εκτέλεση της τοπικής μεθόδου εξαρτάται σε μεγάλο βαθμό από το σημείο εκκίνησης που λαμβάνει από τον αλγόριθμο ΠΑ. Έτσι, η τιμή είναι βελτιωμένη αλλά αποτελεί λύση χαμηλής ακρίβειας. Η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $-1.145$ .

Για την τοπική μέθοδο patternsearch, αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση της μεθόδου ΠΑ για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:

**Πίνακας 3.6:** Παράμετροι ελέγχου του αλγόριθμου ΠΑ για την συνάρτηση Cross-In-Tray στις 30 διαστάσεις

Παράμετρος	Επιλογή
MaxFunctionEvaluations	160,000
MaxIterations	150,000
ObjectiveLimit	-1.3
FunctionTolerance	$1.00 \cdot 10^{-06}$
InitialTemperature	100
MaxStallIterations	70,000
TemperatureFcn	temperaturefast
ReannealInterval	1,650
AnnealingFcn	annealingfast
Display	iter
DisplayInterval	1,000



## Κεφάλαιο 4

# Μεγιστοποίηση Συνάρτησης Πιθανοφάνειας με Εφαρμογή ΠΑ

### 4.1 Περιγραφή του προβλήματος

Οι κλασικές γεωστατιστικές και μηχανικής μάθησης μέθοδοι χαρακτηρίζονται από την κακή κλιμάκωση των απαιτούμενων υπολογιστικών πόρων σε εφαρμογές που αφορούν μεγάλα χωρικά σύνολα δεδομένων [30]. Για αυτόν τον λόγο, παρουσιάζεται ένα απλοποιημένο **στοχαστικό μοντέλο τοπικών αλληλεπιδράσεων** (stochastic local interaction (SLI)) που είναι σχεδιασμένο για τη βελτίωση της υπολογιστικής απόδοσης στη χωρική πρόβλεψη μεγάλων δεδομένων [31]. Πιο συγκεκριμένα, το μοντέλο SLI κατασκευάζει ένα πίνακα ισορροπίας που απεικονίζει τις τιμές δεδομένων, τις γεωγραφικές τους τοποθεσίες και τις διακυμάνσεις πυκνότητας δειγματοληψίας, χωρίς να απαιτείται η είσοδος του χρήστη. Το κύριο χαρακτηριστικό και πλεονέκτημα του μοντέλου SLI είναι ότι παρακάμπτει την αντιστροφή πίνακα για την εκτίμηση των παραμέτρων, τη χωρική πρόβλεψη και την εκτίμηση αβεβαιότητας. Αυτό έχει ως αποτέλεσμα, να απαιτεί σημαντικά λιγότερους υπολογιστικούς πόρους από την παραδοσιακή μέθοδο kriging.

Το μοντέλο SLI ορίζεται με όρους από την εκθετική από κοινού πυκνότητα Gibbs-Boltzmann:

$$f(\mathbf{x}_s) = \frac{1}{Z(\boldsymbol{\theta})} e^{-\mathcal{H}(\mathbf{x}_s; \boldsymbol{\theta})},$$

που  $\mathbf{x}_s$  είναι οι τιμές δείγματος στις τοποθεσίες του συνόλου δειγματοληψίας,  $\boldsymbol{\theta}$  είναι το διάνυσμα των παραμέτρων του μοντέλου SLI,  $\mathcal{H}(\mathbf{x}_s; \boldsymbol{\theta})$  είναι η συνάρτηση ενέργειας του μοντέλου SLI και  $Z(\boldsymbol{\theta})$  είναι η συνάρτηση διαμερισμού που αναπαριστά μία σταθερά κανονικοποίησης.

Η συνάρτηση ενέργειας του μοντέλου SLI στα διάσπαρτα δεδομένα δίνεται από

#### 4.1. Περιγραφή του προβλήματος

την εξίσωση:

$$H(\mathbf{x}_s; \boldsymbol{\theta}) = \frac{1}{2\lambda} \left[ \frac{(\mathbf{x}_s - \mathbf{m}_x)^T (\mathbf{x}_s - \mathbf{m}_x)}{N} + c_1 S_1(\mathbf{x}_s; \mathbf{h}) \right],$$

που  $\mathbf{m}_x$  είναι το διάνυσμα των μέσων δειγματοληπτικών τιμών,  $\lambda$  είναι ο συντελεστής κλίμακας που είναι ανάλογος της συνολικής διασποράς των δεδομένων,  $c_1$  είναι θετικός συντελεστής ακαμψίας που καθορίζει το βάρος των όρων κλίσης στα δεδομένα,  $k$  είναι η τάξη του πλησιέστερου γείτονα,  $\mu$  είναι μία θετική παράμετρος ρύθμισης του εύρους ζώνης,  $S_1(\mathbf{x}_s; \mathbf{h})$  είναι η μέση τιμή τετραγωνικής κλίσης του πυρήνα,  $\mathbf{h} = (h_1, \dots, h_N)^T$  είναι το διάνυσμα των τοπικών εύρων ζώνης πυρήνα,  $\boldsymbol{\theta} = (\lambda, m_x, c_1, k, \mu)^T$  είναι το διάνυσμα των παραμέτρων του μοντέλου SLI.

Οι όροι ενέργειας που βασίζονται σε κλίση δίνονται από τον μέσο όρο του πυρήνα:

$$S_1(\mathbf{x}_s; \mathbf{h}) = ((x_n - x_m)^2)_{\mathbf{h}} = \sum_{n=1}^N \sum_{m=1}^N w_{n,m}(\mathbf{h})(x_n - x_m)^2,$$

που  $((x_n - x_m)^2)_{\mathbf{h}}$  αναπαριστά τον μέσο όρο Nadaraya-Watson των τετραγωνικών διαφορών των τιμών των δεδομένων,  $w_{n,m}$  είναι τα βάρη πυρήνα μεταξύ δύο σημείων  $s_n$  και  $s_m$ .

Η εκτίμηση του διανύσματος των παραμέτρων του μοντέλου SLI πραγματοποιείται μέσω της μεγιστοποίησης της πιθανοφάνειας. Η μεγιστοποίηση της πιθανοφάνειας είναι ισοδύναμη με την ελαχιστοποίηση της αρνητικής λογαριθμικής πιθανοφάνειας. Η **αρνητική λογαριθμική πιθανοφάνεια** (negative log-likelihood) δίνεται από την ακόλουθη εξίσωση:

$$\begin{aligned} -\ln \mathcal{L}(\boldsymbol{\theta}; \mathbf{x}_s) &= \frac{1}{2} \left[ \frac{1}{\lambda} (\mathbf{x}_s - \mathbf{m}_s)^\top \tilde{\mathbf{J}}(\boldsymbol{\theta}_{-\lambda}) (\mathbf{x}_s - \mathbf{m}_s) + N \ln 2\pi\lambda - \ln \det \tilde{\mathbf{J}}(\boldsymbol{\theta}_{-\lambda}) \right] \\ &= \frac{1}{\lambda} \tilde{\mathcal{H}}(\mathbf{x}_s; \boldsymbol{\theta}_{-\lambda}) + \frac{N}{2} \ln 2\pi\lambda - \frac{1}{2} \ln \det \tilde{\mathbf{J}}(\boldsymbol{\theta}_{-\lambda}), \end{aligned}$$

που  $\tilde{H}(\mathbf{x}_s; \boldsymbol{\theta}_{-\lambda})$  είναι η συνάρτηση ενέργειας υπολογισμένη για  $\lambda = 1$ , που  $\mathbf{J}(\boldsymbol{\theta}')$  είναι ο πίνακας ισορροπίας, με  $\boldsymbol{\theta}_{-\lambda} = (c_1, k, \mu)$  και  $\tilde{\mathbf{J}}(\boldsymbol{\theta}_{-\lambda}) = \lambda \mathbf{J}(\boldsymbol{\theta}')$ .

Η συνάρτηση `sli_mle_function` (βλέπε ενότητα (A.3)) είναι σχεδιασμένη για τον υπολογισμό της NLL ενός συνόλου χωρικών δεδομένων, ως προς τις παραμέτρους του μοντέλου SLI. Αρχικά, η συνάρτηση πραγματοποιεί έλεγχο στον αριθμό ορισμάτων εισαγωγής και εξάγει τις παραμέτρους του μοντέλου SLI. Επίσης, καθορίζει τη διάσταση χωρικών δεδομένων, ανακτά τον τύπο συνάρτησης πυρήνα και την τάξη του πλησιέστερου γείτονα από τη δομή `ker`. Επιπρόσθετα, περιλαμβάνεται η εκτίμηση του πίνακα αποστάσεων και του πίνακα ισορροπίας. Επιπλέον, πραγματοποιείται ο υπολογισμός της ενέργειας του μοντέλου SLI που ενσωματώνει τις τοπικές αλληλεπιδράσεις. Τέλος, υπολογίζεται ο λογάριθμος της ορίζουσας του πίνακα ισορροπίας και η συνάρτηση επιστρέφει την υπολογισμένη τιμή της NLL.



#### 4.1. Περιγραφή του προβλήματος

---

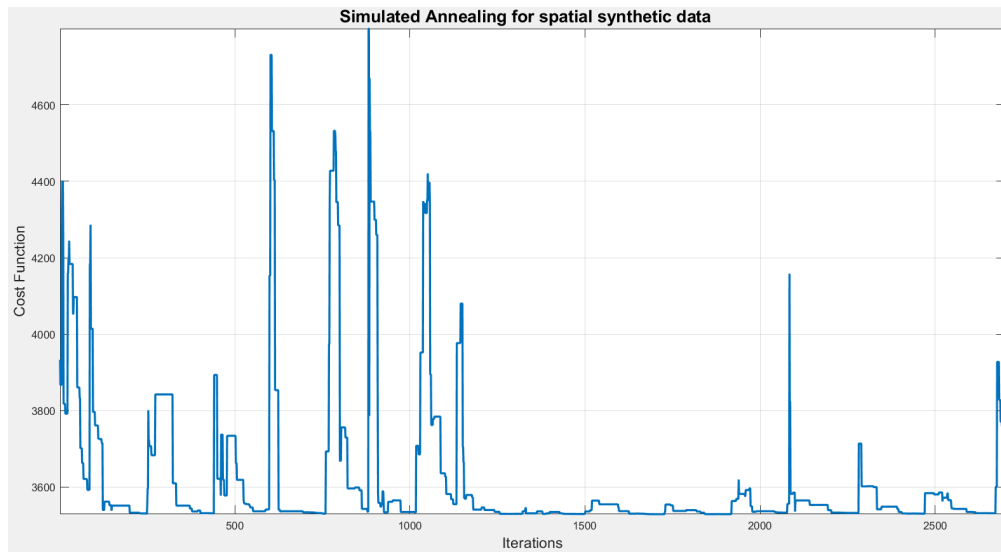
Η συνάρτηση `sli_mle_function` περιλαμβάνει κλήσεις συναρτήσεων για τον υπολογισμό της (NLL). Πιο συγκεκριμένα, η συνάρτηση `kernel_s` χρησιμοποιείται για τον καθορισμό των διαφορετικών τύπων συναρτήσεων πυρήνα. Έπειτα, η συνάρτηση `calc_dist_g` είναι υπεύθυνη για τον υπολογισμό του πίνακα αποστάσεων μεταξύ των χωρικών δεδομένων και του διανύσματος των τοπικών εύρων πυρήνα. Επιπλέον, η συνάρτηση `estim_J2_gra` είναι σχεδιασμένη για να υπολογίζει τον πίνακα ισορροπίας και αντλεί πληροφορίες από τον πίνακα αποστάσεων, το διάνυσμα των τοπικών εύρων πυρήνα και τις παραμέτρους ( $c1, \mu$ ) του μοντέλου SLI. Ακόμη, η συνάρτηση `logdet2` είναι σχεδιασμένη για τον υπολογισμό του φυσικού λογαρίθμου της ορίζουσας ενός θετικά ορισμένου πίνακα διότι απαιτείται η εκτίμηση της συνάρτησης διαμερισμού  $Z(\theta)$ . Σε αυτήν τη συνάρτηση, χρησιμοποιείται η αποσύνθεση Cholesky διότι είναι κατάλληλη για μεγάλους αραιούς πίνακες.

Εν συνεχεία, γίνεται η αρχικοποίηση και η φόρτωση των απαραίτητων δεδομένων. Πιο συγκεκριμένα, ορίζεται ο τύπος συνάρτησης πυρήνα και ο αριθμός των πλησιέστερων γειτόνων. Έπειτα, ορίζεται ο πίνακας που περιέχει τις χωρικές τοποθεσίες των δεδομένων. Τα συνθετικά μη γκαουσιανά δεδομένα περιέχουν 1.741 σημεία σε διδιάστατο χώρο. Τα δεδομένα πάχους στρωμάτων άνθρακα (πραγματικά) περιέχουν 11.416 σημεία (Campbell County, Wyoming, USA)) [31]. Επιπλέον, υπολογίζεται ο πίνακας που περιλαμβάνει τις τιμές των δεδομένων και ορίζονται τυχαία οι τιμές για τις παραμέτρους ( $m, \lambda, c1, \mu$ ) του μοντέλου SLI σε ένα πίνακα *Param0*. Τέλος, πραγματοποιείται η κλήση της συνάρτησης `sli_mle_function` για τη συγκεκριμένη παραμετροποίηση.

##### 4.1.1 Αποτελέσματα για το μοντέλο SLI και χωρικά συνθετικά δεδομένα

Παρακάτω φαίνονται οι γραφικές παραστάσεις από την εφαρμογή της μεθόδου ΠΑ και `fmincon` στα χωρικά συνθετικά δεδομένα:

#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.1:** Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $3.52864 \cdot 10^3$  και λαμβάνεται στην 1,700η επανάληψη.

Από το Σχήμα 4.1 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος βελτιστοποίησης ΠΑ ελαχιστοποιεί την τιμή NLL με την πάροδο του χρόνου. Ο αλγόριθμος συγκλίνει κοντά σε μία τιμή ίση με  $3.528 \cdot 10^3$ . Έτσι, παρά το γεγονός των αυξήσεων, λόγω συμβάντων επανανόπτησης, διακρίνεται συνεχής βελτίωση σε λύσεις χαμηλότερου κόστους. Η τυχαία αρχική τιμή είναι ίση με  $3.9329 \cdot 10^3$  και η τυχαία αρχική τιμή του διανύσματος **Param0** είναι ίση με  $[2.7293 \ 1 \ 1,000 \ 1.5]$ .

Από το Σχήμα 4.1 διακρίνονται κορυφές που αυξάνουν σημαντικά την τιμή συνάρτησης κόστους στις πρώτες περίπου 1,200 επαναλήψεις. Αναλυτικότερα, το διάστημα επανανόπτησης επιλέγεται σχετικά μικρό (ύστερα από την αποδοχή 20 νέων σημείων). Επιπλέον, μέσω της συνάρτησης `annealingfast` οι αποστάσεις των νέων σημείων από τα τρέχοντα είναι μεγαλύτερες για όσο η θερμοκρασία παραμένει σε υψηλά στάδια. Έτσι, είναι αυξημένη η αποδοχή λύσεων υψηλότερου κόστους. Ο σκοπός της συγκεκριμένης παραμετροποίησης είναι η αναζήτηση νέων περιοχών που ενδεχομένως οδηγήσουν τον αλγόριθμο σε πιο ελαχιστοποιημένη λύση, καθώς δεν υπάρχει γνώση για το τοπίο της συνάρτησης κόστους που διερευνάται. Ωστόσο, μετά τις 1,200 επαναλήψεις η τιμή της συνάρτησης κόστους δεν αυξάνεται σημαντικά μετά από κάθε συμβάν επανανόπτησης. Συνεπώς, ο αλγόριθμος κατευθύνει την αναζήτηση του σε περιοχές γύρω από την ελαχιστοποιημένη τιμή.

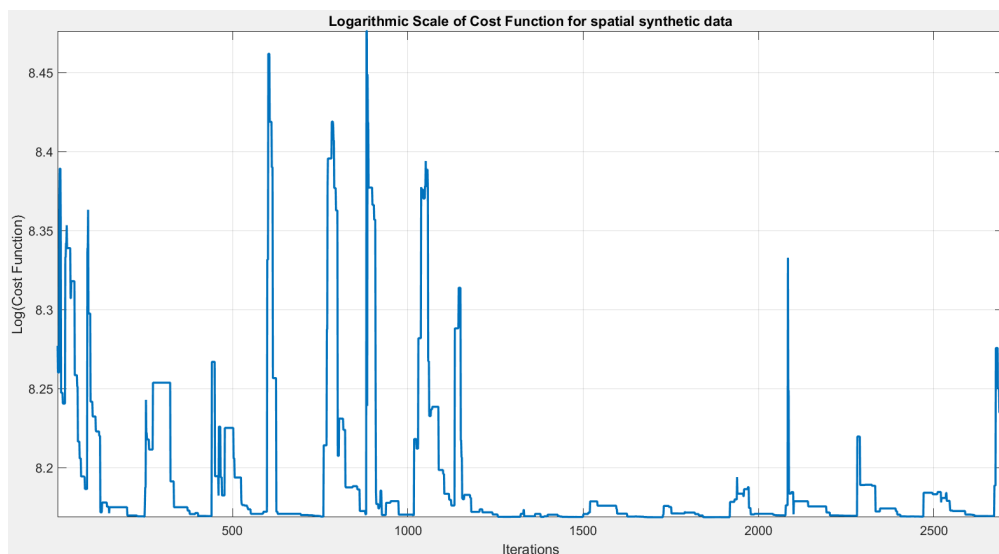
Ακολουθώς, ύστερα από κάθε συμβάν επανανόπτησης διακρίνεται πτωτική πορεία. Αυτό συμβαίνει διότι με τη μείωση θερμοκρασίας, μειώνεται η αποδοχή λύσεων υψηλότερου κόστους. Επιπροσθέτως, πριν την κάθε εκ νέου επανανόπτηση παρατηρούνται τμήματα που οι αλλαγές είναι μικρές. Αυτό συμβαίνει γιατί, η μειωμένη

#### 4.1. Περιγραφή του προβλήματος

θερμοκρασία εστιάζει την αναζήτηση του αλγόριθμου σε βελτιώσεις γύρω από την τρέχουσα τιμή.

Από το Σχήμα 4.1 παρατηρείται ότι πριν από πολλά γεγονότα εκ νέου επανανόπτησης ο αλγόριθμος συγκλίνει σε λύσεις πολύ κοντά στην τιμή που επέστρεψε ως ελάχιστη. Έτσι, ο αλγόριθμος θεωρείται ότι φτάνει σε κατάσταση ισορροπίας σχετικά νωρίς στη διαδικασία βελτιστοποίησης, καθώς όλες οι περαιτέρω εκ νέου αναζητήσεις, επιστρέφουν σε αυτή την κατάσταση.

Η διαδικασία βελτιστοποίησης χρησιμοποιώντας τον αλγόριθμο ΠΑ τερματίζει στις 2,700 επαναλήψεις εξαιτίας της ανοχής `FunctionTolerance` που έχει οριστεί σε τιμή ίση με  $1 \cdot 10^{-04}$ . Συγκεκριμένα, η μέση μεταβολή στη βέλτιστη τιμή, στις τελευταίες 1,000 επαναλήψεις (`MaxStallIterations`), είναι μικρότερη της ανοχής `FunctionTolerance`. Η αρχική τιμή είναι ίση με  $3.9329 \cdot 10^3$ , ενώ η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $3.52864 \cdot 10^3$ . Η συνολική μείωση είναι σημαντική και περίπου ίση με 404. Έτσι, επιδεικνύεται η αποτελεσματικότητα του αλγόριθμου ΠΑ στην ελαχιστοποίηση της NLL για το σύνολο των χωρικών συνθετικών δεδομένων.

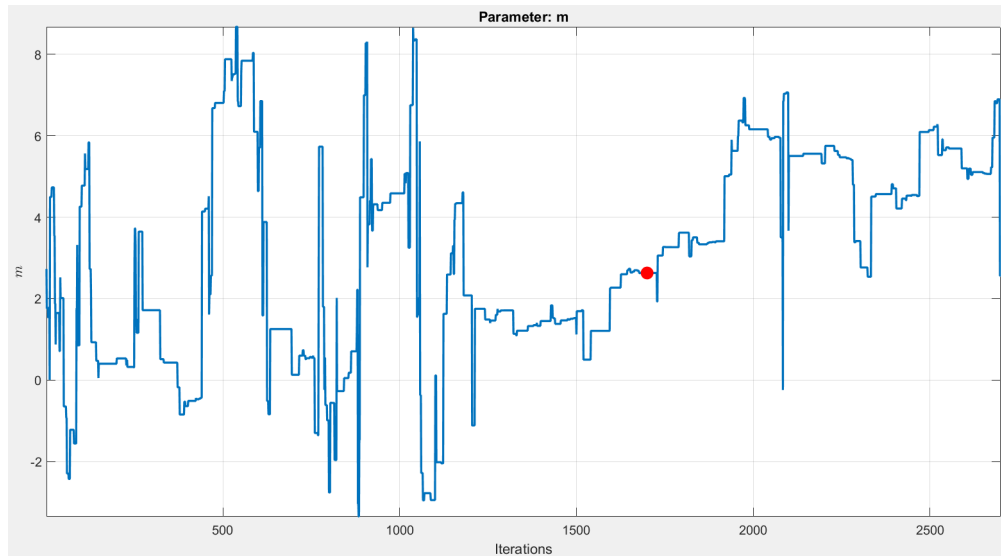


**Σχήμα 4.2:** Η εξέλιξη της τιμής NLL (σε λογαριθμική κλίμακα) στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο  $\log$  για καλύτερα οπτικά αποτελέσματα.

Το Σχήμα 4.2 περιέχει τη γραφική παράσταση της λογαριθμικής τιμής NLL συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Αναλυτικότερα, η αξιοποίηση της λογαριθμικής κλίμακας πραγματοποιείται για να διασαφηνισθεί η καλύτερη οπτική εικόνα στις αλλαγές των τιμών. Η λογαριθμική κλίμακα συμπιέζει στον κατακόρυφο άξονα μεγάλες αριθμητικές περιοχές σε ένα μικρότερο οπτικό χώρο. Έτσι, δίνεται η δυνατότητα για πιο ευκρινή παρατήρηση στο αν υπάρχει ουσιαστική μεταβολή. Συγκρίνοντας τα σχήματα 4.1 και 4.2 διακρίνεται ότι το σχήμα 4.2 δεν εμφανίζει ουσιαστική μεταβολή στην κατακόρυφη κλίμακα. Επομένως, το

#### 4.1. Περιγραφή του προβλήματος

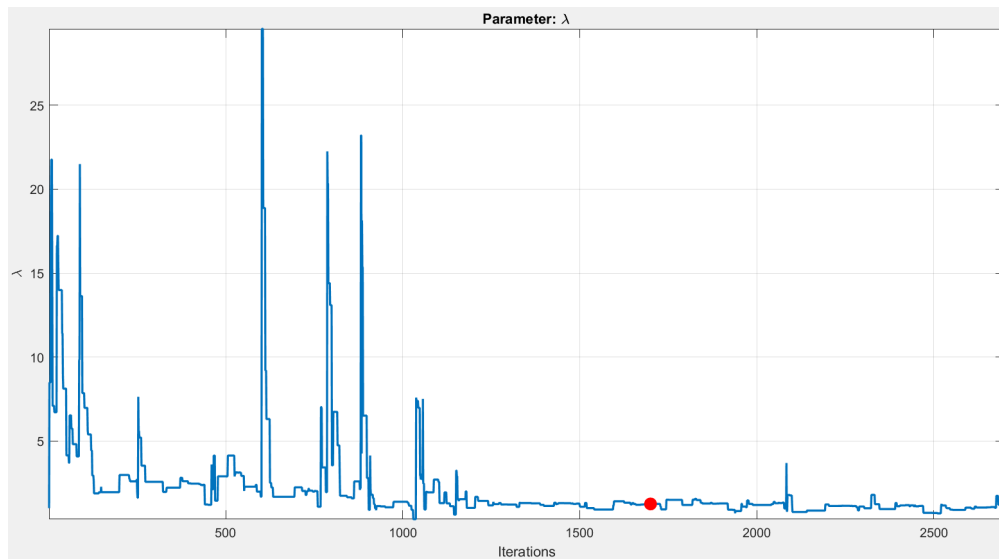
σχήμα 4.2 είναι σχεδόν πανομοιότυπο με το σχήμα 4.1. Αυτό συμβαίνει διότι, οι μεταβολές που γίνονται δεν είναι πολλών τάξεων μεγέθους.



**Σχήμα 4.3:** Η εξέλιξη της τιμής παραμέτρου  $m$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτίζει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.3 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $m$  συναρτίζει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 2.7293, ενώ η βελτιστοποιημένη είναι ίση με 2.6289. Το άνω και κάτω όριο είναι 8.88 και  $-3.4214$  αντίστοιχα. Η βέλτιστη τιμή της βρίσκεται σε κοντινή απόσταση συγκριτικά με την αρχική. Η αρχική τιμή της παραμέτρου  $m$  ορίζεται ως ο μέσος όρος των δειγματοληπτικών τιμών. Αυτό το γεγονός, συνιστά καλή εκτίμηση του πραγματικού μέσου όρου των δεδομένων. Επομένως, η αρχική τιμή της βρίσκεται σε ευνοϊκή περιοχή και είναι ακριβής.

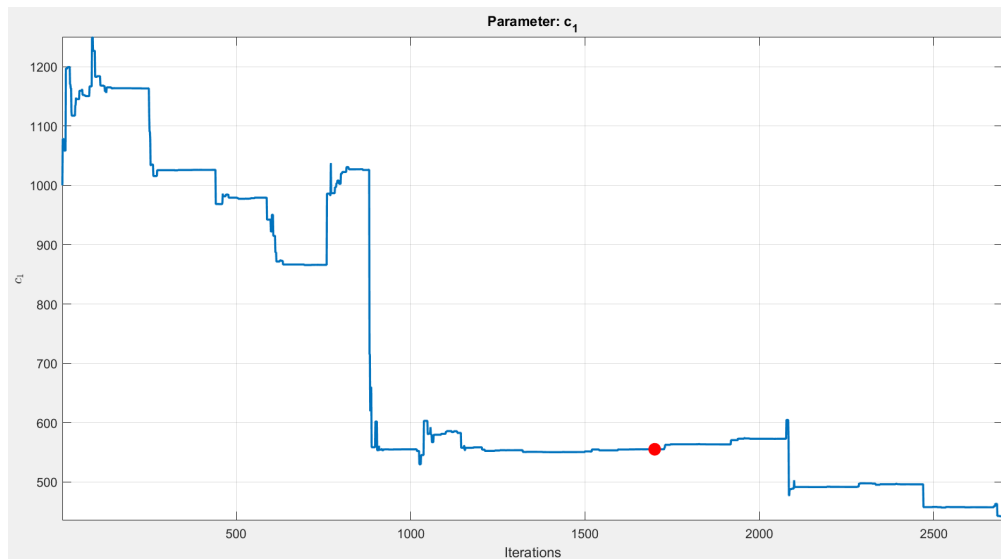
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.4:** Η εξέλιξη της τιμής παραμέτρου  $\lambda$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.4 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\lambda$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1, ενώ η βελτιστοποιημένη είναι ίση με 1.2487. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Ο συντελεστής κλίμακας  $\lambda$  είναι ανάλογος με τη συνολική διασπορά στα δεδομένα. Οι υψηλότερες τιμές της προκαλούν αύξηση στην τιμή NLL, ενώ οι χαμηλότερες προκαλούν μείωση. Η βέλτιστη τιμή της βρίσκεται πιθανόν στο διάστημα  $[1, 2.5]$ . Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής είναι παρατηρήσιμη καθώς υπάρχει μεταβολή στην συνολική διασπορά των δεδομένων.

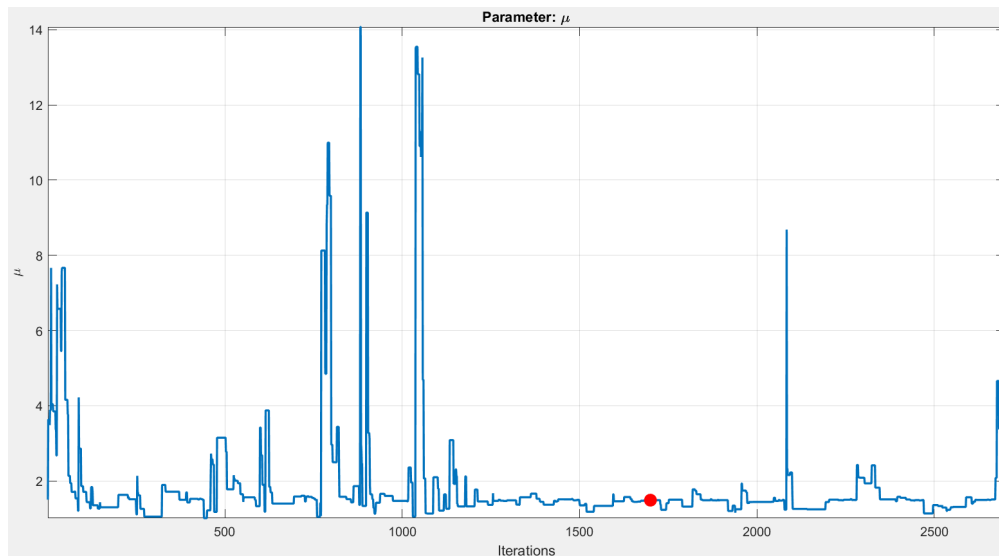
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.5:** Η εξέλιξη της τιμής παραμέτρου  $c_1$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.5 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $c_1$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1,000, ενώ η βελτιστοποιημένη είναι ίση με 554.99. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Στις αρχικές επαναλήψεις διακρίνεται μικρή αύξηση στην τιμή της. Στη συνέχεια, μέχρι το τέλος της διαδικασίας βελτιστοποίησης, παρατηρείται συνεχής πτώση στην τιμή της. Αυτό το γεγονός υποδεικνύει ότι η βέλτιστη τιμή της για την ελαχιστοποίηση της NLL βρίσκεται σε χαμηλότερες τιμές συγκριτικά με την αρχική. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής είναι σημαντική.

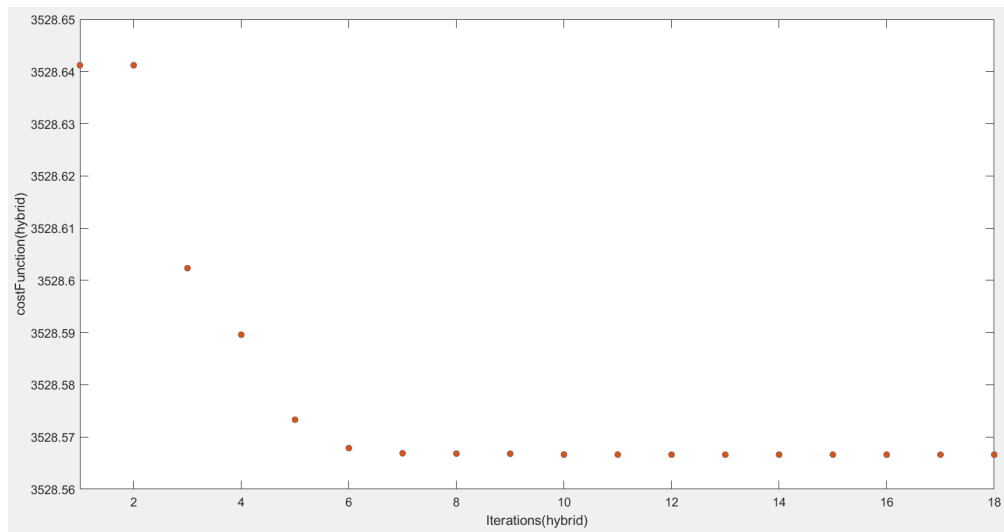
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.6:** Η εξέλιξη της τιμής παραμέτρου  $\mu$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτίζει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.6 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\mu$  συναρτίζει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1.5, ενώ η βελτιστοποιημένη είναι ίση με 1.4866. Το άνω και κάτω όριο είναι 15 και 0.5 αντίστοιχα. Αναλυτικότερα, οι υψηλότερες τιμές της παραμέτρου  $\mu$  προκαλούν αύξηση στην τιμή NLL, ενώ οι χαμηλότερες προκαλούν μείωση. Η βέλτιστη τιμή της βρίσκεται πιθανόν στο διάστημα  $[1.4, 2.5]$ . Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής θεωρείται σχετικά μικρή. Αυτό υποδεικνύει ότι η αρχική εκτίμηση για την παράμετρο  $\mu$  βρίσκεται σε περιοχή κοντά στη βέλτιστη τιμή της.

#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.7:** Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου fmincon. Η fmincon εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

Από το Σχήμα 4.7 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι η τοπική μέθοδος ελαχιστοποιεί την τιμή NLL με την πάροδο του χρόνου. Στις αρχικές επαναλήψεις διακρίνεται απότομη πτώση στην τιμή, που ακολουθείται από μικρές βελτιώσεις καθώς αυξάνεται ο αριθμός των επαναλήψεων. Έτσι, ο αλγόριθμος ΠΑ δίνει την δυνατότητα στην τοπική μέθοδο να κινηθεί σε ευνοϊκές περιοχές. Αυτό έχει ως αποτέλεσμα, η εκτέλεσή της να επιτυγχάνει λύση που ελαχιστοποιεί περαιτέρω την τιμή NLL. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $3.52856 \cdot 10^3$ .

Ακολουθώντας, για την επιτυγχανόμενη ελάχιστη τιμή NLL, οι τιμές των παραμέτρων του μοντέλου SLI είναι ίσες με  $m = 2.7293$ ,  $\lambda = 1.2672$ ,  $c1 = 554.99$ ,  $\mu = 1.4905$ . Άρα, διακρίνεται ότι οι βέλτιστες τιμές των παραμέτρων  $m, \mu$  έχουν μεταβληθεί ελάχιστα συγκριτικά με τις αρχικές τιμές τους. Οι βέλτιστες τιμές των παραμέτρων  $c1, \lambda$  θεωρούνται σημαντικές και παρατηρήσιμες συγκριτικά με τις αρχικές τιμές τους. Αυτό συμβαίνει διότι, η ελαχιστοποίηση της NLL εξαρτάται κυρίως από τον λόγο  $c1/\lambda$ .

Για την τοπική μέθοδο fmincon αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση της μεθόδου ΠΑ για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:



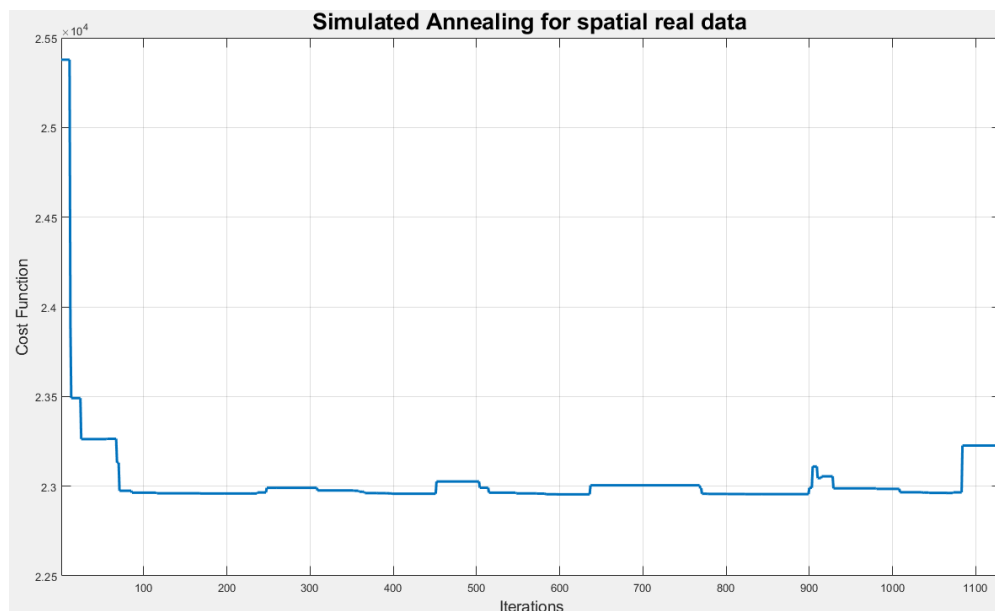
#### 4.1. Περιγραφή του προβλήματος

Πίνακας 4.1: Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τα χωρικά συνθετικά δεδομένα

Παράμετρος	Επιλογή
MaxFunctionEvaluations	3,500
MaxIterations	3,000
ObjectiveLimit	3528.6
FunctionTolerance	$1.00 \cdot 10^{-04}$
InitialTemperature	700
MaxStallIterations	1,000
TemperatureFcn	temperatureexp
ReannealInterval	20
AnnealingFcn	annealingfast
Display	iter
DisplayInterval	10

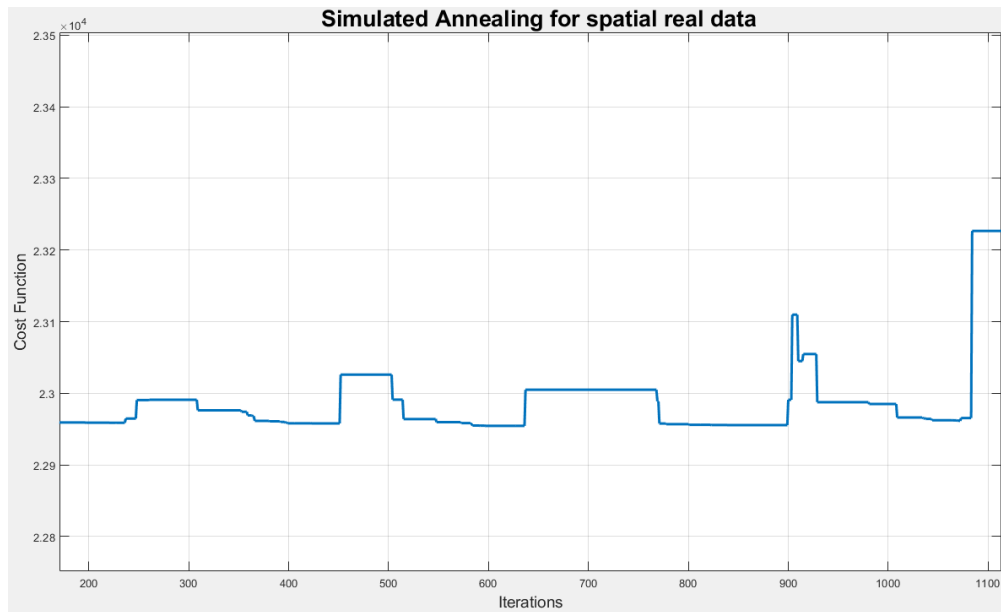
#### 4.1.2 Αποτελέσματα για το μοντέλο SLI και χωρικά πραγματικά δεδομένα

Παρακάτω φαίνονται οι γραφικές παραστάσεις από την εφαρμογή της μεθόδου ΠΑ και `fmincon` στα χωρικά πραγματικά δεδομένα:



Σχήμα 4.8: Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $2.29545 \cdot 10^4$  και λαμβάνεται στην 631η επανάληψη.

#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.9:** Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτῆσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η εικόνα είναι μεγενθυμένη για καλύτερη οπτική παρατήρηση των συμβάντων επανανόπτησης.

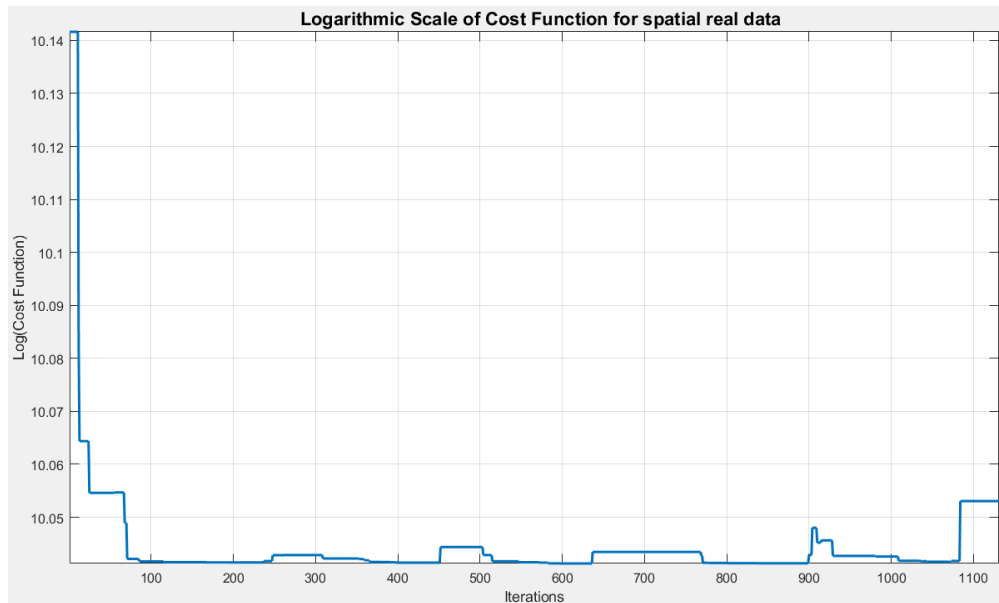
Από το Σχήμα 4.8 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος βελτιστοποίησης ΠΑ ελαχιστοποιεί την τιμή NLL με την πάροδο του χρόνου. Ο αλγόριθμος συγκλίνει κοντά σε μία τιμή ίση  $2.2954 \cdot 10^4$ . Ο συνδυασμός του εκθετικού προγράμματος ενημέρωσης θερμοκρασίας και του μικρού διαστήματος επανανόπτησης, συμβάλλουν στη χαρακτηριστική μορφή της γραφικής παράστασης. Η τυχαία αρχική τιμή είναι ίση με  $2.5378 \cdot 10^4$  και η τυχαία αρχική τιμή του διανύσματος **Param0** είναι ίση με  $[7.8291 \quad 1 \quad 1,000 \quad 1.5]$ .

Από το Σχήμα 4.8 διακρίνεται ότι στις πρώτες 100 επαναλήψεις συμβαίνει ταχεία μείωση στην τιμή NLL. Επιπροσθέτως, παρατηρούνται σε συγκεκριμένες επαναλήψεις μικρές μεταβολές που αυξάνουν ελάχιστα την τιμή NLL μετά από κάθε συμβάν επανανόπτησης. Πιο συγκεκριμένα, ο αλγόριθμος κατευθύνει την αναζήτηση του σε περιοχές γύρω από την τρέχουσα ελαχιστοποιημένη τιμή. Πριν την κάθε εκ νέου επανανόπτηση παρατηρούνται τμήματα που οι αλλαγές είναι ελάχιστες. Σε αυτά τα διαστήματα, η γραφική παράσταση παρουσιάζει πιο επίπεδη περιοχή.

Από το Σχήμα 4.8 παρατηρείται ότι πριν την κάθε εκ νέου επανανόπτηση ο αλγόριθμος συγκλίνει σε λύσεις πολύ κοντά στην τιμή που επέστρεψε ως ελάχιστη και οι περαιτέρω επαναλήψεις οδηγούν σε μικρές αλλαγές γύρω από αυτή την τιμή. Έτσι, ο αλγόριθμος θεωρείται ότι φτάνει σε κατάσταση ισορροπίας σχετικά νωρίς στη διαδικασία βελτιστοποίησης, καθώς όλες οι περαιτέρω εκ νέου αναζητήσεις, επιστρέφουν σε αυτή την κατάσταση.

#### 4.1. Περιγραφή του προβλήματος

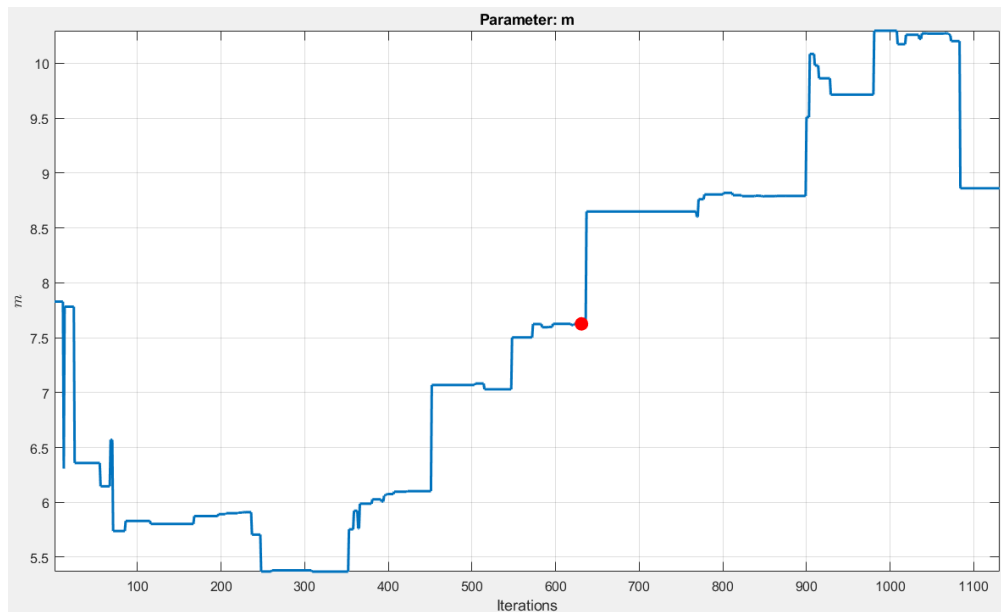
Η διαδικασία βελτιστοποίησης, χρησιμοποιώντας τον αλγόριθμο ΠΑ, τερματίζει στις 1,131 επαναλήψεις εξαιτίας ανοχής FunctionTolerance που έχει οριστεί σε τιμή ίση με  $1 \cdot 10^{-04}$ . Συγκεκριμένα, η μέση μεταβολή στην βέλτιστη τιμή, στις τελευταίες 500 επαναλήψεις (MaxStallIterations), είναι μικρότερη της ανοχής FunctionTolerance. Η αρχική τιμή είναι ίση με  $2.5378 \cdot 10^4$ , ενώ η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $2.29545 \cdot 10^4$ . Η συνολική μείωση είναι σημαντική και περίπου ίση με 2.424. Έτσι, επιδεικνύεται η αποτελεσματικότητα του αλγόριθμου ΠΑ στην ελαχιστοποίηση της NLL για το σύνολο των χωρικών πραγματικών δεδομένων.



**Σχήμα 4.10:** Η εξέλιξη της τιμής NLL (σε λογαριθμική κλίμακα) στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Ο κατακόρυφος άξονας του διαγράμματος αξιοποιεί τον λογάριθμο log για καλύτερα οπτικά αποτελέσματα.

Το Σχήμα 4.10 περιέχει τη γραφική παράσταση της λογαριθμικής τιμής NLL συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Πιο συγκεκριμένα, η αξιοποίηση της λογαριθμικής κλίμακας πραγματοποιείται για να διασαφηνισθεί η καλύτερη οπτική εικόνα στις αλλαγές των τιμών. Η λογαριθμική κλίμακα συμπιέζει στον κατακόρυφο άξονα μεγάλες αριθμητικές περιοχές σε ένα μικρότερο οπτικό χώρο. Άρα, δίνεται η επιλογή για πιο ευκρινή παρατήρηση στο αν υπάρχει ουσιαστική μεταβολή. Συγκρίνοντας τα σχήματα 4.8 και 4.10 διακρίνεται ότι το σχήμα 4.10 δεν εμφανίζει ουσιαστική μεταβολή στην κατακόρυφη κλίμακα. Έτσι, το σχήμα 4.10 είναι σχεδόν πανομοιότυπο με το σχήμα 4.8. Αυτό συμβαίνει διότι, οι μεταβολές των τιμών του διανύσματος Param δεν καλύπτουν πολλές τάξεις μεγέθους.

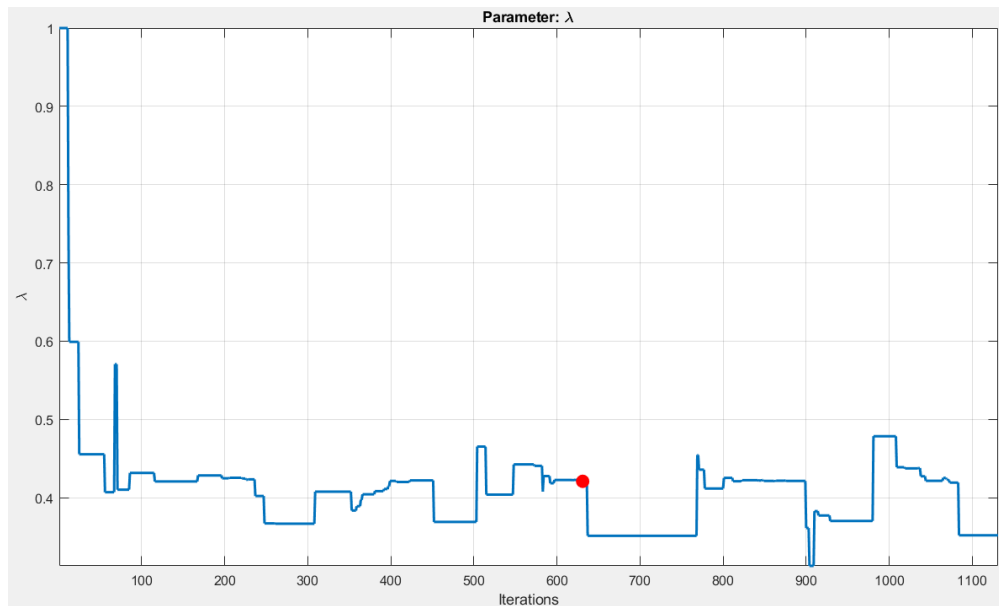
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.11:** Η εξέλιξη της τιμής παραμέτρου  $m$  του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.11 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $m$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 7.8291, ενώ η βελτιστοποιημένη είναι ίση με 7.6261. Το άνω και κάτω όριο είναι 13.8149 και 1.8433 αντίστοιχα. Η βέλτιστη τιμή της βρίσκεται σε κοντινή απόσταση συγκριτικά με την αρχική. Η αρχική τιμή της παραμέτρου  $m$  ορίζεται ως ο μέσος όρος των δειγματοληπτικών τιμών. Αυτό το γεγονός, συνιστά καλή εκτίμηση του πραγματικού μέσου όρου των δεδομένων. Επομένως, η αρχική τιμή της βρίσκεται σε ευνοϊκή περιοχή και είναι ακριβής.

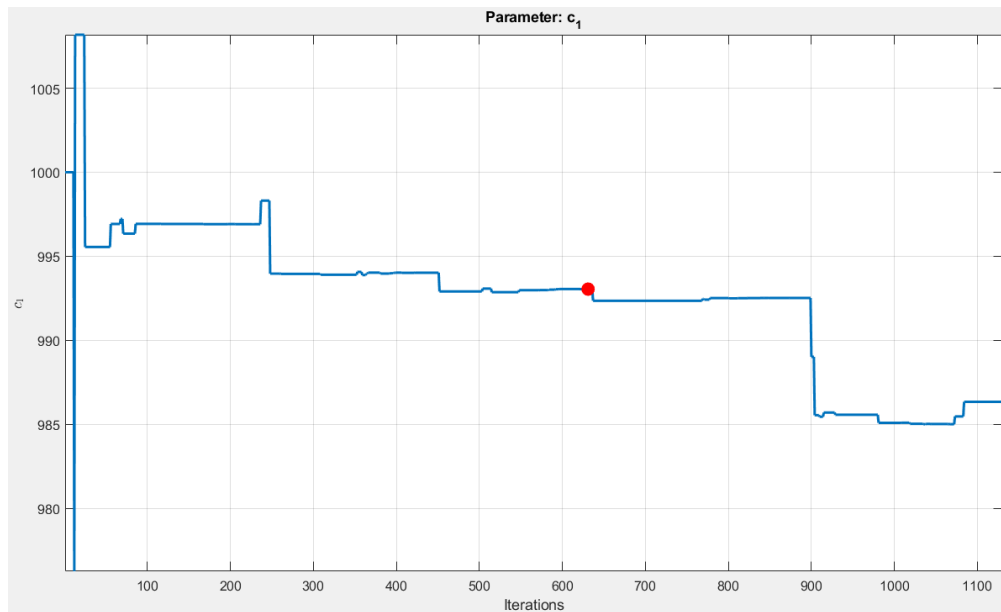
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.12:** Η εξέλιξη της τιμής παραμέτρου  $\lambda$  του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.12 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\lambda$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1, ενώ η βελτιστοποιημένη είναι ίση με 0.4209. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Ο συντελεστής κλίμακας  $\lambda$  είναι ανάλογος με τη συνολική διασπορά στα δεδομένα. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής είναι παρατηρήσιμη καθώς υπάρχει μεταβολή στην συνολική διασπορά των δεδομένων.

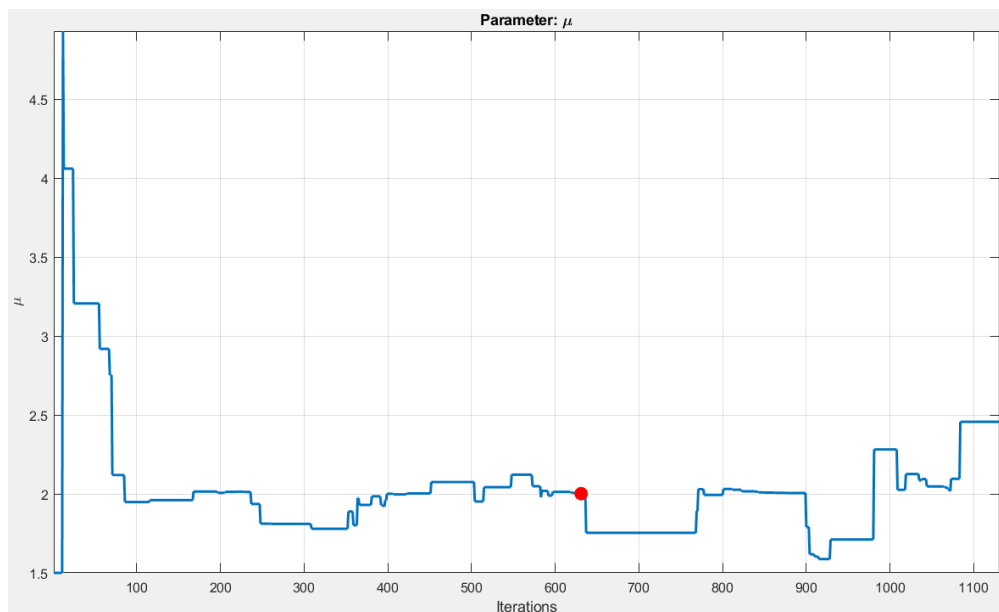
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.13:** Η εξέλιξη της τιμής παραμέτρου  $c_1$  του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.13 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $c_1$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1,000, ενώ η βελτιστοποιημένη είναι ίση με 993.04. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Η παράμετρος λαμβάνει χαμηλότερες τιμές συγκριτικά με την αρχική τιμή της για μεγάλο μέρος της διαδικασίας βελτιστοποίησης. Αυτό το γεγονός υποδεικνύει ότι η βέλτιστη τιμή της για την ελαχιστοποίηση της NLL βρίσκεται σε χαμηλότερες τιμές συγκριτικά με την αρχική. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής θεωρείται παρατηρήσιμη.

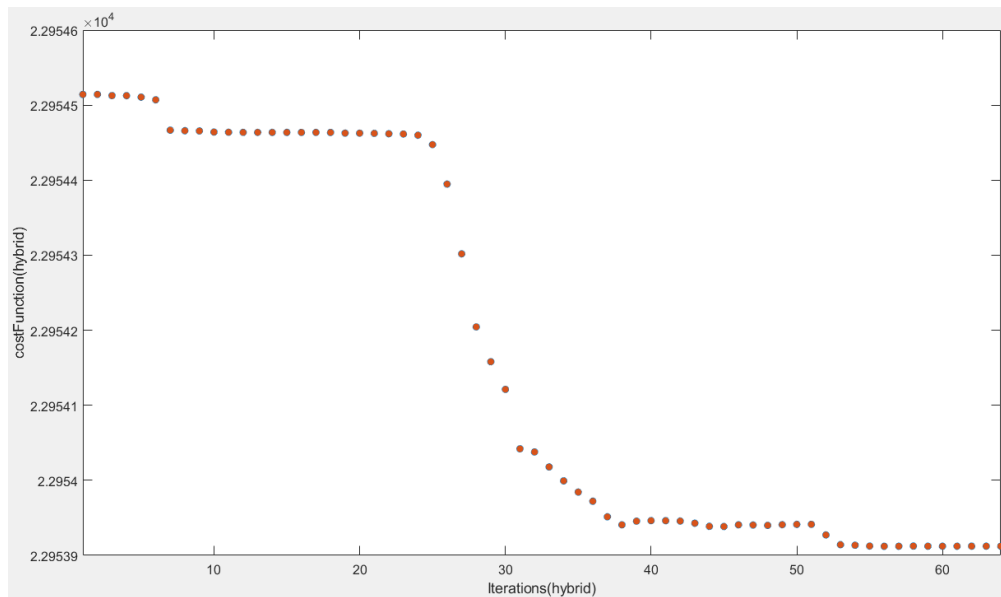
#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.14:** Η εξέλιξη της τιμής παραμέτρου  $\mu$  του μοντέλου SLI στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της ΠΑ. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 4.14 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\mu$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου ΠΑ. Η αρχική τιμή της είναι ίση με 1.5, ενώ η βελτιστοποιημένη είναι ίση με 2. Το άνω και κάτω όριο είναι 0.5 και 15 αντίστοιχα. Οι τιμές της περιορίζονται σε συγκεκριμένη περιοχή του χώρου αναζήτησης  $[1.5, 2.5]$  για μεγάλο μέρος της διαδικασίας βελτιστοποίησης. Αυτό υποδεικνύει ότι η βέλτιστη τιμή της βρίσκεται πιθανόν σε αυτό το διάστημα. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής θεωρείται σχετικά μικρή. Έτσι, η αρχική εκτίμηση για την παράμετρο  $\mu$  βρίσκεται σε περιοχή κοντά στη βέλτιστη τιμή της.

#### 4.1. Περιγραφή του προβλήματος



**Σχήμα 4.15:** Η εξέλιξη της τιμής NLL στα χωρικά πραγματικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή της τοπικής μεθόδου fmincon. Η fmincon εκκινεί από την ελάχιστη τιμή στην οποία τερματίζει ο αλγόριθμος ΠΑ.

Από το Σχήμα 4.15 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός που υποδηλώνει ότι η τοπική μέθοδος ελαχιστοποιεί τη NLL με την πάροδο του χρόνου. Στις αρχικές επαναλήψεις διακρίνονται μικρές αλλαγές στην τιμή. Στη συνέχεια, παρατηρείται πτώση της τιμής που ακολουθείται από μικρές βελτιώσεις καθώς αυξάνεται ο αριθμός των επαναλήψεων. Έτσι, ο αλγόριθμος ΠΑ δίνει τη δυνατότητα στην τοπική μέθοδο να κινηθεί σε ευνοϊκές περιοχές. Αυτό έχει ως αποτέλεσμα, η εκτέλεσή της να επιτυγχάνει λύση που ελαχιστοποιεί περαιτέρω την τιμή NLL. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $2.295391 \cdot 10^4$ .

Κατόπιν, για την επιτυγχανόμενη ελάχιστη τιμή NLL, οι τιμές των παραμέτρων του μοντέλου SLI είναι ίσες με  $m = 7.8290$ ,  $\lambda = 1.1051$ ,  $c1 = 2,603.33$ ,  $\mu = 2.0097$ . Άρα, διακρίνεται ότι οι βέλτιστες τιμές των παραμέτρων  $m, \mu$  έχουν μεταβληθεί ελάχιστα συγκριτικά με τις αρχικές τιμές τους. Οι βέλτιστες τιμές των παραμέτρων  $c1, \lambda$  θεωρούνται σημαντικές και παρατηρήσιμες συγκριτικά με τις αρχικές τιμές τους. Αυτό συμβαίνει διότι, η ελαχιστοποίηση της NLL εξαρτάται κυρίως από τον λόγο  $c1/\lambda$ .

Για την τοπική μέθοδο fmincon αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση της μεθόδου ΠΑ για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:



#### 4.1. Περιγραφή του προβλήματος

---

Πίνακας 4.2: Παράμετροι ελέγχου του αλγόριθμου ΠΑ για τα χωρικά πραγματικά δεδομένα

Παράμετρος	Επιλογή
MaxFunctionEvaluations	2,000
MaxIterations	1,500
ObjectiveLimit	22,954
FunctionTolerance	$1.00 \cdot 10^{-04}$
InitialTemperature	500
MaxStallIterations	500
TemperatureFcn	temperatureexp
ReannealInterval	20
AnnealingFcn	annealingfast
Display	iter
DisplayInterval	10



## Κεφάλαιο 5

# Μέθοδος Καθολικής Αναζήτησης (Matlab GlobalSearch)

### 5.1 Εισαγωγή

Η καθολική αναζήτηση (GS) παρουσιάζει ορισμένα κοινά χαρακτηριστικά με το μοντέλο του OptQuest/Nonlinear Local Programming (OQNLP) αλγόριθμου [5]. Η GS είναι ευρετική μέθοδος σχεδιασμένη για την εύρεση του καθολικού ελαχίστου σε μη γραμμικά προβλήματα. Αναλυτικότερα, εκκινεί μια μέθοδο τοπικής βελτιστοποίησης από πολλαπλά σημεία εκκίνησης του συνόλου. Τα σημεία εκκίνησης δημιουργούνται από τον αλγόριθμο **αναζήτησης διασποράς (scatter-search)**. Στη συνέχεια, η GS αναλύει τα σημεία εκκίνησης και απορρίπτει εκείνα που είναι απίθανο να βελτιώσουν το βέλτιστο τοπικό ελάχιστο μέχρι στιγμής [32]. Ακόμη, η τιμή του καθολικού ελαχίστου που εξάγεται, είναι το εφικτό σημείο που έχει τη χαμηλότερη τιμή συνάρτησης κόστους. Τέλος, η GS επιτρέπει στον χρήστη αποκλειστικά και μόνο τη χρήση της συνάρτησης **fmincon** για τη μέθοδο τοπικής βελτιστοποίησης.

Η μέθοδος **αναζήτησης διασποράς (SS)** αναπτύχθηκε για πρώτη φορά από τον Glover το 1977 ως ευρετική μέθοδος για τον ακέραιο προγραμματισμό [33]. Αναλυτικότερα, συνιστά μετά-ευρετικό αλγόριθμο, σχεδιασμένο για την επίλυση μη γραμμικών και συνδυαστικών προβλημάτων βελτιστοποίησης [34]. Επιπλέον, αντλεί τα θεμέλιά της από στρατηγικές που απαιτούνταν για το συνδυασμό κανόνων απόφασης καθώς και περιορισμών στο πλαίσιο του προβλήματος. Η μέθοδος SS διαθέτει ενοποιητικές αρχές για το συνδυασμό σημείων που βασίζονται σε κατασκευές διαδρομών σε ευκλείδειους χώρους, συγκριτικά με άλλες προσεγγίσεις που εφαρμόζουν την τυχαιοποίηση [35]. Έτσι, οι συστηματικοί σχεδιασμοί και οι μέθοδοι για τη δημιουργία νέων σημείων παρουσιάζουν σημαντικά κέρδη σε σχέση με αυτά που δίνει η τυχαιοποίηση.

### 5.1. Εισαγωγή

Η μέθοδος SS είναι σχεδιασμένη για να λειτουργεί σε ένα σύνολο σημείων που ονομάζεται σύνολο αναφοράς ή πληθυσμού. Συγκεκριμένα, τα σημεία του συνόλου αναφοράς διατηρούνται και ενημερώνονται σε κάθε επανάληψη. Το κύριο χαρακτηριστικό της είναι ότι συνδυάζει ένα σύνολο αρχικών σημείων και το βελτιώνει δια επαναλήψεως χρησιμοποιώντας «ντετερμινιστικούς» συνδυασμούς για τη δημιουργία των νέων σημείων του συνόλου αναφοράς [5]. Ειδικά, οι συνδυασμοί συνοδεύονται από διαδικασίες που προσαρμοστικά επιβάλλουν συνθήκες εφικτότητας και διακριτότητας [36].

Ο αλγόριθμος SS αποτελείται από πέντε μεθόδους. Αρχικά, περιλαμβάνει τη **μέθοδο δημιουργίας διαφοροποίησης** για τη δημιουργία μιας σειράς από διαφορετικά δοκιμαστικά σημεία, ξεκινώντας από ένα ή περισσότερα αυθαίρετα δοκιμαστικά σημεία. Έπειτα, η **μέθοδος βελτίωσης** είναι υπεύθυνη για τη μετατροπή ενός δοκιμαστικού σημείου σε ένα ή περισσότερα βελτιωμένα δοκιμαστικά σημεία. Εάν δεν προκύψει καμία βελτίωση, το βελτιωμένο σημείο θεωρείται ότι είναι ισοδύναμο με το δοκιμαστικό σημείο εισόδου. Στη συνέχεια, η **μέθοδος ενημέρωσης συνόλου αναφοράς** είναι υπεύθυνη για τη δημιουργία και συντήρηση του συνόλου αναφοράς που αποτελείται από τα βέλτιστα σημεία. Ακόμη, η **μέθοδος δημιουργίας υποσυνόλων** λειτουργεί στο σύνολο αναφοράς προκειμένου να δημιουργήσει υποσύνολα σημείων που εν συνεχεία θα αποτελέσουν τη βάση για τη δημιουργία των συνδυασμένων σημείων. Τέλος, η **μέθοδος συνδυασμού** μετατρέπει ένα υποσύνολο σημείων, που έχει αποκτηθεί από τη μέθοδο δημιουργίας υποσυνόλων, σε ένα ή περισσότερα συνδυασμένα διανύσματα σημείου [37].

Η περιγραφή της συνάρτησης `fmincon` αναφέρεται στην ενότητα (3.4). Για τον αλγόριθμο GS, η εκτέλεση της τοπικής μεθόδου βελτιστοποίησης περιλαμβάνεται σε δύο στάδια. Συγκεκριμένα, στο πρώτο στάδιο, εκκινεί με το αρχικό σημείο και εφόσον το αρχικό σημείο δεν είναι εφικτό, ερευνά ένα εφικτό σημείο εφαρμόζοντας διάφορες ελαχιστοποιήσεις στην τιμή των παραβιασμένων περιορισμών. Αν βρεθεί ένα εφικτό σημείο από το πρώτο στάδιο, τότε στο δεύτερο στάδιο αξιοποιείται ως σημείο εκκίνησης και η συνάρτηση `fmincon` συνεχίζει με την ελαχιστοποίηση του. Ακόμη, και τα δύο στάδια, περιλαμβάνουν μια ακολουθία γραμμικών αναζητήσεων. Ο σκοπός είναι η παραγωγή εφικτού σημείου με τη χαμηλότερη τιμή συνάρτησης κόστους [5]. Γενικώς, όταν πραγματοποιείται η εκτέλεση της τοπικής μεθόδου, η λύση που επιστρέφεται αξιολογείται από τον αλγόριθμο GS. Ο αλγόριθμος πραγματοποιεί ελέγχους και αποφεύγει την επανεξέταση σημείων που βρίσκονται σε ίδιες λεκάνες έλξης [32].

Στην συγκεκριμένη εργασία, τα προβλήματα βελτιστοποίησης που επιλύονται από την εφαρμογή της μεθόδου GS, υλοποιούνται στο προγραμματιστικό περιβάλλον της Matlab. Οι τελικές τιμές που επιστρέφονται από τον αλγόριθμο GS δεν είναι οι

## 5.2. Η περιγραφή της μεθόδου GS

τιμές που λαμβάνονται από την τελευταία εκτέλεσή του. Αντιθέτως, είναι οι επιτυγχάνομενες ελάχιστες τιμές που προκύπτουν κατά τη διάρκεια της διαδικασίας βελτιστοποίησης.

## 5.2 Η περιγραφή της μεθόδου GS

Παρακάτω παρουσιάζεται αναλυτικά η διαδικασία βελτιστοποίησης του αλγόριθμου GS [32]:

**Βήμα 1:** Δημιουργία αρχικού σημείου και εκτέλεση μεθόδου τοπικής βελτιστοποίησης από το συγκεκριμένο σημείο. Εάν υπάρξει σύγκλιση, τότε παρέχεται πληροφορία στον αλγόριθμο που αφορά στην αρχική εκτίμηση της ακτίνας μιας λεκάνης έλξης. Παρακάτω παρατίθενται ορισμένες έννοιες για την καλύτερη κατανόηση της λειτουργίας του αλγόριθμου.

Η ακτίνα μιας λεκάνης έλξης ορίζει την απόσταση μεταξύ ενός σημείου έναρξης μέχρι το σημείο λύσης (το σημείο στο οποίο συγκλίνει η εκτέλεση της τοπικής μεθόδου).

Η συνάρτηση βαθμολογίας είναι το άθροισμα της τιμής συνάρτησης κόστους σε ένα σημείο με έναν πολλαπλασιαστή του αθροίσματος των παραβιασμένων περιορισμών.

Η πολλαπλασιαστική σταθερά των παραβιασμένων περιορισμών έχει αρχικά υψηλή τιμή και στη συνέχεια ενημερώνεται δυναμικά από τον αλγόριθμο. Συγκεκριμένα, μέσω αυτής της ποινής γίνεται αντιληπτό αν μία λύση επιτυγχάνει ή αποτυγχάνει να καλύψει τους περιορισμούς του προβλήματος. Το σημείο θεωρείται εφικτό όταν οι τιμές των παραβιασμένων περιορισμών είναι μηδενικές και η συνάρτηση βαθμολογίας σε αυτήν την περίπτωση είναι ίση με την τιμή συνάρτησης κόστους.

**Βήμα 2:** Καθορισμός του αριθμού συνόλου δοκιμαστικών σημείων, τα οποία συνιστούν πιθανά σημεία έναρξης του αλγόριθμου.

**Βήμα 3:** Δημιουργία ενός μικρότερου συνόλου δοκιμαστικών σημείων (βλέπε ενότητα (5.2.1)), που αποτελούν μέρος του αρχικού. Ο αλγόριθμος GS αξιολογεί τη συνάρτηση βαθμολογίας των δοκιμαστικών σημείων του συγκεκριμένου συνόλου. Έτσι, λαμβάνεται το σημείο με την καλύτερη βαθμολογία και πραγματοποιείται η εκκίνηση της τοπικής μεθόδου βελτιστοποίησης από το συγκεκριμένο σημείο. Η καλύτερη βαθμολογία αναφέρεται στην ελάχιστη τιμή ενός εφικτού σημείου. Έπειτα, αφαιρείται το συγκεκριμένο σύνολο από την λίστα των συνολικών σημείων.

## 5.2. Η περιγραφή της μεθόδου GS

Βήμα 4: Αρχικοποίηση λεκανών, μετρητών και ορίου. Αναλυτικότερα, οι λεκάνες έλξης είναι σφαιρικές. Το σύνολο των σημείων εντός αυτής της λεκάνης, όταν χρησιμοποιούνται ως σημεία έναρξης, έχει ως αποτέλεσμα να συγκλίνουν στο ίδιο ελάχιστο. Ακόμη, το όριο είναι η ελάχιστη τιμή από τις τιμές συνάρτησης κόστους στα σημεία λύσης. Τα σημεία λύσης προκύπτουν από την εκτέλεση της τοπικής μεθόδου βελτιστοποίησης από το αρχικό σημείο και από το σημείο εκκίνησης του βήματος 3. Επιπρόσθετα, πραγματοποιείται η αρχικοποίηση δύο τύπων μετρητών (διαδοχικά σημεία που βρίσκονται εντός μιας λεκάνης έλξης και διαδοχικά σημεία που η τιμή συνάρτησης βαθμολογίας τους είναι μεγαλύτερη από το όριο).

Βήμα 5: Κύριος βρόγχος για τον έλεγχο εκτέλεσης της τοπικής μεθόδου βελτιστοποίησης κάθε δοκιμαστικού σημείου που απομένει στη λίστα. Έστω  $p$  το δοκιμαστικό σημείο που περιέχεται στη λίστα. Η εκτέλεση της τοπικής βελτιστοποίησης συμβαίνει όταν:

- Το δοκιμαστικό σημείο  $p$  δεν βρίσκεται σε καμία υπάρχουσα λεκάνη έλξης.

$$|p - \text{center}(i)| > \text{DistanceThresholdFactor} \cdot \text{radius}(i) \quad (5.1)$$

•

$$\text{score function of the trial point } (p) < \text{threshold} \quad (5.2)$$

- Το δοκιμαστικό σημείο  $p$  ικανοποιεί τους περιορισμούς ορίων ή τους περιορισμούς ορίων και ανισοτήτων.

Στην περίπτωση που συμβαίνει η εκτέλεση της τοπικής βελτιστοποίησης τότε πραγματοποιούνται ενημερώσεις και έλεγχοι για τη βελτίωση της αναζήτησης. Αρχικά, ο αλγόριθμος επαναφέρει τους μετρητές για το όριο και τις λεκάνες ίσες με μηδέν. Στη συνέχεια, το δοκιμαστικό σημείο προστίθενται στο διάνυσμα αντικειμένων `GlobalOptimSolution` αν η απόστασή του ή η διαφορά στην τιμή συνάρτησης κόστους από οποιοδήποτε άλλο σημείο του συνόλου είναι μεγαλύτερη από συγκεκριμένα όρια ανοχής. Αντιθέτως, θεωρείται ισοδύναμο με κάποιο υπάρχον σημείο και τροποποιείται το διάνυσμα αντικειμένων `GlobalOptimSolution` αφού προστίθενται ως αρχικό σημείο. Επιπλέον, ο αλγόριθμος ενημερώνει την ακτίνα λεκάνης έλξης και την τιμή ορίου για το συγκεκριμένο δοκιμαστικό σημείο  $p$ .

Στην περίπτωση που δεν εκτελείται η τοπική βελτιστοποίηση, ο αλγόριθμος πραγματοποιεί αλλαγές για να βελτιώσει τη στρατηγική αναζήτησή του. Συγκεκριμένα, αυξάνεται ο μετρητής κάθε λεκάνης που περιέχει το σημείο  $p$ . Έπειτα, αν η τιμή συνάρτησης βαθμολογίας του σημείου  $p$  είναι μεγαλύτερη

## 5.2. Η περιγραφή της μεθόδου GS

---

ή ίση από την τιμή του ορίου, τότε ο μετρητής του ορίου αυξάνεται. Σε αντίθετη περίπτωση, ο μετρητής επαναφέρεται σε τιμή ίση με μηδέν. Ακόμη, όταν τα δοκιμαστικά σημεία για ένα καθορισμένο αριθμό διαδοχικών επαναλήψεων βρίσκονται εντός μιας λεκάνης έλξης, τότε η ακτίνα μειώνεται. Επιπλέον, όταν η τιμή της συνάρτησης βαθμολογίας των σημείων είναι μεγαλύτερη του ορίου για ένα καθορισμένο αριθμό διαδοχικών επαναλήψεων, τότε αυξάνεται το όριο.

Βήμα 6: Τερματισμός της διαδικασίας όταν ο αλγόριθμος υπερβεί σε δευτερόλεπτα τον μέγιστο χρόνο ή όταν αναλύει όλα τα δοκιμαστικά σημεία. Ο αλγόριθμος GS δημιουργεί το διάνυσμα αντικείμενων `GlobalOptimSolution` και ταξινομεί τις τιμές συνάρτησης κόστους από τη χαμηλότερη (καλύτερη) προς την υψηλότερη (χειρότερη) τιμή.

## 5.2. Η περιγραφή της μεθόδου GS

---

### Algorithm 1 : The pseudocode of the GS algorithm

---

```
1: INITIALIZATION
2: Read_Parameters (number of variables, bounds, starting point);
3: Set_GS_Control_Parameters_and_Options_for_SS (setting GS algorithm control parameters and SS algorithm
   options, such as stage 1 and 2 iteration limits);
4: Initialize_SS_Population;
5: Stage 1 iterations  $\leftarrow 0$ ;
6: Stage 2 iterations  $\leftarrow 0$ ;
7:
8: STAGE 1: Execute best start point among the total trial points (NumStageOnePoints)
9: while Stage 1 iterations < Stage 1 iteration limit do
10:   Get (trial point from SS);
11:   Evaluate (cost function and multiple of the sum of the constraint violations at trial point);
12:   Put (trial point, cost function and sum of the constraint violations to SS database);
13:   Stage 1 iterations  $\leftarrow$  Stage 1 iterations + 1;
14: end while
15: Get_Best_Point_from_SS_database (starting point);
16: Call_fmincon (starting point, local solution);
17: if both solution points (fmincon solutions from x0 and Stage 1 starting point) exist and are feasible then
18:   threshold  $\leftarrow$  min(cost_function_value(solution_point_x0), cost_function_value(Stage_1_solution_point))
19: else if one of the solution points exists and is feasible then
20:   threshold  $\leftarrow$  cost_function_value(existing_solution_point)
21: else
22:   threshold  $\leftarrow$  penalty_function_value(Stage_1_start_point)
23: end if
24:
25: STAGE 2: MAIN ITERATIVE LOOP (Examine Trial Point to See if fmincon Runs)
26: while Stage 2 iterations < Stage 2 iteration limit do
27:   Get (trial point from SS);
28:   Evaluate (cost function and multiple of the sum of the constraint violations at trial point);
29:   Put (trial point, cost function and sum of the constraint violations to SS database);
30:   if  $|p - \text{center}(\text{basin})| > \text{DistanceThresholdFactor} \cdot \text{radius}(\text{basin})$  AND score_function(p) < threshold AND p satisfies
      bound and/or inequality constraints then
31:     Call_fmincon (trial point (p), local solution);
32:     if local solution feasible then
33:       Insert local solution in GlobalOptimSolution object;
34:       threshold  $\leftarrow$  calculate score function of p trial point;
35:       update the value of the basin radius;
36:     end if
37:   else if score function of trials points > threshold for MaxWaitCycle consecutive iterations then
38:     increase threshold;
39:   else if trial points for a MaxWaitCycle number of consecutive iterations are within a basin of attraction then
40:     decrease radius;
41:   end if
42:   Stage 2 iterations  $\leftarrow$  Stage 2 iterations + 1;
43: end while
```

---



## 5.2. Η περιγραφή της μεθόδου GS

---

### 5.2.1 Η περιγραφή της μεθόδου SS

Σε αυτή την ενότητα, περιγράφονται οι μέθοδοι που περιλαμβάνει ο αλγόριθμος SS για τη δημιουργία των δοκιμαστικών σημείων. Τα δοκιμαστικά σημεία δημιουργούνται δυναμικά κατά τη διάρκεια εκτέλεσης του GS.

**Μέθοδος Δημιουργίας Διαφοροποίησης:** Η μέθοδος είναι υπεύθυνη για τη δημιουργία ενός μεγάλου συνόλου αναφοράς που αποτελείται από διαφορετικά και διακριτά δοκιμαστικά σημεία [33]. Ειδικότερα, για την επίτευξη αυτού του συνόλου διαιρείται το εύρος της κάθε μεταβλητής σε  $n$  υπό-εύρη ίσου μεγέθους. Έπειτα, το σημείο δημιουργείται σε δύο βήματα. Στο πρώτο βήμα γίνεται τυχαία η επιλογή ενός υπό-εύρους ενώ, στο δεύτερο, γίνεται η τυχαία επιλογή μιας τιμής από το επιλεγμένο υπό-εύρος του πρώτου βήματος. Το αρχικό σύνολο αναφοράς στο οποίο χρησιμοποιείται η συγκεκριμένη μέθοδος, περιλαμβάνει ένα σύνολο τριών ή τεσσάρων σημείων. Αναλυτικότερα, περιλαμβάνεται το πρώτο που όλες οι μεταβλητές τίθενται ίσες στο κατώτερο όριο, το δεύτερο που τίθενται ίσες στο ανώτερο όριο και το τρίτο που τίθενται ίσες στη μέση τιμή μεταξύ των ορίων. Εάν είναι ορισμένο ένα αρχικό σημείο, τότε προστίθενται στο αρχικό σύνολο αναφοράς [34]. Τα σημεία χρησιμοποιούνται ως είσοδος στη διαδικασία αναζήτησης του αλγόριθμου και είναι ομοιόμορφα κατανεμημένα στο χώρο [38]. Η μέθοδος χρησιμοποιείται και για την ανασυγκρότηση του συνόλου αναφοράς.

**Μέθοδος Βελτίωσης:** Η μέθοδος είναι υπεύθυνη για τη μετατροπή του δοκιμαστικού σημείου σε ένα ή περισσότερα [39]. Στόχος είναι η μετατροπή του σημείου σε ένα καλύτερο-βελτιωμένο ως προς την τιμή συνάρτησης κόστους και την εφικτότητα (μέτρηση τιμής των παραβιασμένων περιορισμών). Αυτό πραγματοποιείται μέσω των διαδικασιών τοπικής αναζήτησης ή μεταβλητής κατάβασης γειτονίας (variable neighborhood descent). Η είσοδος της μεθόδου αποτελείται από δοκιμαστικά σημεία τόσο εφικτά όσο και μη. Η έξοδος της μεθόδου αποτελείται από βελτιωμένα ή μη δοκιμαστικά σημεία συγκριτικά με το αρχικό. Στην περίπτωση που δεν υπάρχει βελτίωση, το σημείο θεωρείται ίδιο με το αρχικό [40]. Η μέθοδος εφαρμόζεται σε μεγάλο αριθμό σημείων καθώς δέχεται σημεία από τις μεθόδους συνδυασμού και δημιουργίας διαφοροποίησης.

**Μέθοδος Ενημέρωσης Συνόλου Αναφοράς:** Η μέθοδος χρησιμοποιείται σε δύο διαφορετικούς τομείς του αλγόριθμου [34]. Πιο συγκεκριμένα, είναι υπεύθυνη για τη δημιουργία του πρώτου ενημερωμένου συνόλου αναφοράς που προέρχεται από τα σημεία των μεθόδων δημιουργίας διαφοροποίησης και βελτίωσης. Η τυπική σχεδίαση της μεθόδου για το πρώτο ενημερωμένο σύνολο αναφοράς σε αυτό το πεδίο είναι η συλλογή από σημεία τόσο υψηλής ποιότητας όσο και από ποικίλα. Έπειτα, οι επόμενες κλήσεις στη μέθοδο αφορούν στη συντήρηση αυτού του συνόλου. Καθώς

## 5.2. Η περιγραφή της μεθόδου GS

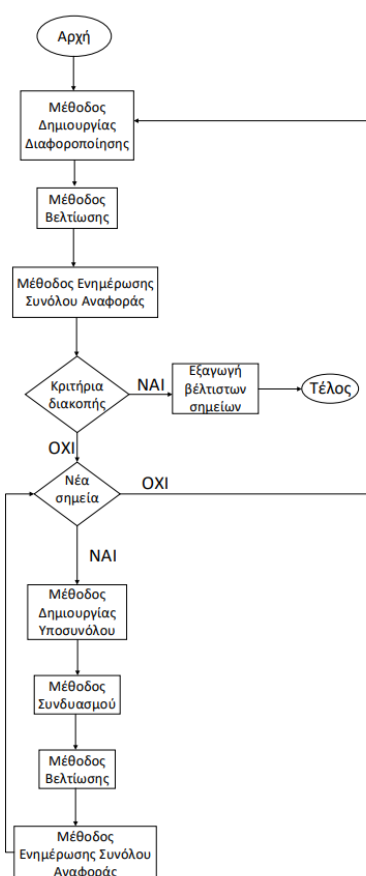
---

προχωρά η διαδικασία, δημιουργούνται νέα σημεία από τις μεθόδους συνδυασμού και βελτίωσης. Το σύνολο αναφοράς είναι οργανωμένο για την παροχή αποδοτικής πρόσβασης στις άλλες μεθόδους του αλγόριθμου. Έτσι, τα σημεία ταξινομούνται σύμφωνα με την ποιότητα τους, δηλαδή το καλύτερο σημείο βρίσκεται πρώτο στη λίστα [40].

**Μέθοδος Δημιουργίας Υποσυνόλων:** Η μέθοδος είναι υπεύθυνη για τη δημιουργία υποσυνόλων διαφορετικών μεγεθών, τα οποία αποτελούν την είσοδο για τη μέθοδο συνδυασμού. Η μέθοδος πραγματοποιεί έλεγχο του συνολικού αριθμού των υποσυνόλων που δημιουργεί [33]. Επιπλέον, ο αριθμός των σημείων που δημιουργούνται, εξαρτάται από τη στρατηγική που εφαρμόζεται στη μέθοδο ενημέρωσης συνόλου αναφοράς. Η τυπική υλοποίηση περιλαμβάνει τη δημιουργία όλων των δυνατών ζευγών. Ωστόσο, η δημιουργία υποσυνόλων πραγματοποιείται με μεθόδους διαφορετικών μεγεθών ορίζοντας τύπους υποσυνόλων. Αναλυτικά, οι τύποι υποσυνόλων είναι: α) Τύπος-I: περιέχει υποσύνολα δύο σημείων, β) Τύπος-II: περιέχει υποσύνολα τριών σημείων που αποτελούνται από υποσύνολα δύο σημείων, γ) Τύπος-III: περιέχει υποσύνολα τεσσάρων σημείων που αποτελούνται από υποσύνολα τριών σημείων, δ) Τύπος-IV: περιέχει υποσύνολα που αποτελούνται από τα  $k$ -καλύτερα σημεία (δηλαδή, το  $k$  βρίσκεται στο εύρος του [5, Συνόλου Αναφοράς]) [39].

**Μέθοδος Συνδυασμού:** Η μέθοδος αξιοποιεί τα υποσύνολα που δημιουργούνται από τη μέθοδο δημιουργίας υποσυνόλων για τη δημιουργία νέων σημείων [39]. Συγκεκριμένα, τα νέα δοκιμαστικά σημεία προκύπτουν από το συνδυασμό δύο ή περισσότερων σημείων του συνόλου αναφοράς. Ο αλγόριθμος SS παρέχει τη δυνατότητα παραγωγής των συνδυασμένων σημείων και στο βήμα που πραγματοποιείται η δημιουργία των υποσυνόλων [33]. Τέλος, όλα τα σημεία που προκύπτουν από αυτή τη μέθοδο ελέγχονται από τη μέθοδο βελτίωσης και από τη μέθοδο ενημέρωσης του συνόλου αναφοράς.

## 5.2. Η περιγραφή της μεθόδου GS



Σχήμα 5.1: Διάγραμμα ροής του αλγόριθμου SS



### 5.3. Παράμετροι ελέγχου της μεθόδου GS

**NumStageOnePoints:** Η παράμετρος αυτή ορίζει τον αριθμό των δοκιμαστικών σημείων στην αρχική φάση του αλγόριθμου. Συγκεκριμένα, η αξιολόγηση των σημείων σε αυτό το στάδιο γίνεται για την εύρεση του καλύτερου σημείου με σκοπό την εκτέλεση της τοπικής βελτιστοποίησης. Η επιλογή καλύτερου σημείου αναφέρεται στο σημείο με την καλύτερη βαθμολογία συνάρτησης. Στα συγκεκριμένα προβλήματα βελτιστοποίησης, συνεπάγεται το σημείο με τη χαμηλότερη τιμή συνάρτησης κόστους και τη χαμηλότερη τιμή του αθροίσματος των παραβιασμένων περιορισμών. Ο αλγόριθμος **SS** είναι υπεύθυνος για τη δημιουργία των συγκεκριμένων δοκιμαστικών σημείων.

**MaxWaitCycle:** Η παράμετρος ορίζει το μέγιστο καθορισμένο αριθμό διαδοχικών επαναλήψεων που ο αλγόριθμος απορρίπτει τα δοκιμαστικά σημεία και δεν πραγματοποιεί την εκτέλεση τοπικής βελτιστοποίησης. Ο αλγόριθμος, για να βελτιώσει την αναζήτησή του, μεταβάλλει ορισμένες παραμέτρους. Αναλυτικότερα, όταν τα δοκιμαστικά σημεία για τον καθορισμένο αριθμό διαδοχικών επαναλήψεων βρίσκονται εντός μιας λεκάνης έλξης, τότε η ακτίνα μειώνεται σύμφωνα με την παράμετρο **BasinRadiusFactor**. Επιπλέον, όταν η τιμή συνάρτησης βαθμολογίας των σημείων για τον καθορισμένο αριθμό διαδοχικών επαναλήψεων είναι μεγαλύτερη από την τρέχουσα τιμή ορίου, τότε αυξάνεται η τιμή του ορίου σύμφωνα με την παράμετρο **PenaltyThresholdFactor**.

**BasinRadiusFactor:** Η παράμετρος καθορίζει τον συντελεστή μείωσης της ακτίνας λεκάνης. Συγκεκριμένα, με τη δημιουργία κάθε δοκιμαστικού σημείου, πραγματοποιείται έλεγχος αν το σημείο βρίσκεται εντός μιας υπάρχουσας λεκάνης. Στην περίπτωση που τα δοκιμαστικά σημεία για τον καθορισμένο αριθμό διαδοχικών επαναλήψεων (βάση της παραμέτρου **MaxWaitCycle**) βρίσκονται εντός μιας λεκάνης έλξης, τότε η ακτίνα μειώνεται. Η ακτίνα πολλαπλασιάζεται με την τιμή  $1 - \text{BasinRadiusFactor}$ . Στην αντίθετη περίπτωση, η ακτίνα του δοκιμαστικού σημείου ισούται με το μέγιστο μεταξύ της υπάρχουσας ακτίνας (εφόσον υπάρχει) και της απόστασης μεταξύ του δοκιμαστικού σημείου και της θέσης της τοπικής λύσης.

**PenaltyThresholdFactor:** Η παράμετρος χρησιμοποιείται για την αύξηση της τιμής του ορίου. Ειδικότερα, με τη δημιουργία κάθε δοκιμαστικού σημείου, ελέγχεται η τιμή συνάρτησης βαθμολογίας του και συγκρίνεται με την τρέχουσα τιμή του ορίου. Στην περίπτωση που η τιμή συνάρτησης βαθμολογίας των σημείων είναι μεγαλύτερη για τον καθορισμένο αριθμό διαδοχικών επαναλήψεων (βάση της παραμέτρου **MaxWaitCycle**), τότε αυξάνεται η τιμή του ορίου σύμφωνα με την εξίσωση:

$$\text{new threshold} = \text{threshold} + \text{PenaltyThresholdFactor} \cdot (1 + |\text{threshold}|) \quad (5.3)$$

Στην αντίθετη περίπτωση, εκτελείται η τοπική βελτιστοποίηση και η τιμή του ορίου μειώνεται και γίνεται ίση με την τιμή συνάρτησης βαθμολογίας του σημείου.

### 5.3. Παράμετροι ελέγχου της μεθόδου GS

**DistanceThresholdFactor:** Η παράμετρος είναι ένας πολλαπλασιαστής που επηρεάζει τη συνθήκη (5.1) για το αν ένα δοκιμαστικό σημείο βρίσκεται εντός μιας υπάρχουσας λεκάνης. Κάθε δοκιμαστικό σημείο θεωρείται ότι βρίσκεται εκτός της λεκάνης, εάν η απόσταση του σημείου από τη θέση μιας οποιαδήποτε τοπικής λύσης που έχει βρεθεί, είναι μεγαλύτερη από το γινόμενο της ακτίνας της λεκάνης και της παραμέτρου **DistanceThresholdFactor**. Στην περίπτωση που το δοκιμαστικό σημείο βρίσκεται εντός μιας λεκάνης έλξης, τότε ο αλγόριθμος GS δεν εκτελεί την τοπική βελτιστοποίηση και λαμβάνει το επόμενο δοκιμαστικό σημείο από τη λίστα. Αντιθέτως, εκτελείται η τοπική βελτιστοποίηση.

**MaxTime:** Η παράμετρος καθορίζει το μέγιστο χρόνο (σε δευτερόλεπτα) που εκτελείται ο αλγόριθμος GS. Ο συνολικός χρόνος βελτιστοποίησης μπορεί να υπερβεί το μέγιστο χρόνο που έχει οριστεί. Αυτό συμβαίνει διότι, ο αλγόριθμος δεν επιβάλλει απότομο τερματισμό των διαδικασιών τοπικής βελτιστοποίησης.

**StartPointsToRun:** Η παράμετρος καθορίζει τα σημεία εκκίνησης για την εκτέλεση των τοπικών βελτιστοποιήσεων. Αναλυτικότερα, παρέχει τη δυνατότητα τριών επιλογών (**all**, **bounds**, **bounds-ineqs**). Ειδικά, η επιλογή **all** επιτρέπει στον αλγόριθμο να αποδεχτεί όλα τα σημεία εκκίνησης που δίνονται από τον χρήστη. Η επιλογή **bounds** επιτρέπει στον αλγόριθμο να αποδεχτεί τα σημεία εκκίνησης που βρίσκονται εντός των ορίων του προβλήματος. Η επιλογή **bounds-ineqs** επιτρέπει στον αλγόριθμο να αποδεχτεί τα σημεία εκκίνησης που βρίσκονται εντός των ορίων και των περιορισμών ανισότητας του κάθε προβλήματος. Ο αλγόριθμος ελέγχει τη συγκεκριμένη παράμετρο μόνο κατά τη διάρκεια εκτέλεσης του κύριου βρόγχου.

Ο αλγόριθμος GS ενημερώνει το σύνολο λύσεων του κατά τη διαδικασία βελτιστοποίησης με σκοπό τη διάκριση των λύσεων ως πανομοιότυπες ή διακριτές. Σε εκτενέστερη ανάλυση, παρουσιάζονται δύο παράμετροι που σχετίζονται άμεσα με τη διαχείριση, διάκριση και βελτίωση του συνόλου λύσεων. Η παράμετρος **XTolerance** συνιστά όριο που καθορίζει την ελάχιστη επιτρεπόμενη απόσταση μεταξύ δύο σημείων του χώρου λύσης για να θεωρηθούν διακριτές. Η παράμετρος **FunctionTolerance** συνιστά όριο που καθορίζει την ελάχιστη επιτρεπόμενη διαφορά μεταξύ των τιμών συναρτήσεων κόστους των δύο λύσεων για να θεωρηθούν διακριτές. Παρακάτω, αναφέρονται σε εκτενή ανάλυση οι συνθήκες για τη διάκριση των λύσεων.

Περιγραφή των συνθηκών: Έστω ότι η τοπική βελτιστοποίηση εκτελείται για το τυχαίο σημείο  $p$ , και γίνεται η θεώρηση ότι συγκλίνει με σημείο λύσης το  $x_p$  και τιμή συνάρτησης κόστους  $f_p$ . Η θετική σημαία εξόδου υποδηλώνει ότι η λύση που βρέθηκε ικανοποιεί τα κριτήρια σύγκλισης. Για οποιοδήποτε άλλο σημείο λύσης  $x_q$  με τιμή συνάρτησης κόστους  $f_q$ :

### 5.3. Παράμετροι ελέγχου της μεθόδου GS

---

Αν:

$$|xq - xp| \leq XTolerance * \max(1, |xp|)$$

ΚΑΙ

$$|fq - fp| \leq FunctionTolerance * \max(1, |fp|)$$

τότε οι δύο λύσεις θεωρούνται πανομοιότυπες.

Αν:

$$|xq - xp| > XTolerance * \max(1, |xp|)$$

Ή

$$|fq - fp| > FunctionTolerance * \max(1, |fp|)$$

τότε οι δύο λύσεις θεωρούνται διακριτές.





## Κεφάλαιο 6

# Εφαρμογή Καθολικής Αναζήτησης σε Συναρτήσεις Ελέγχου

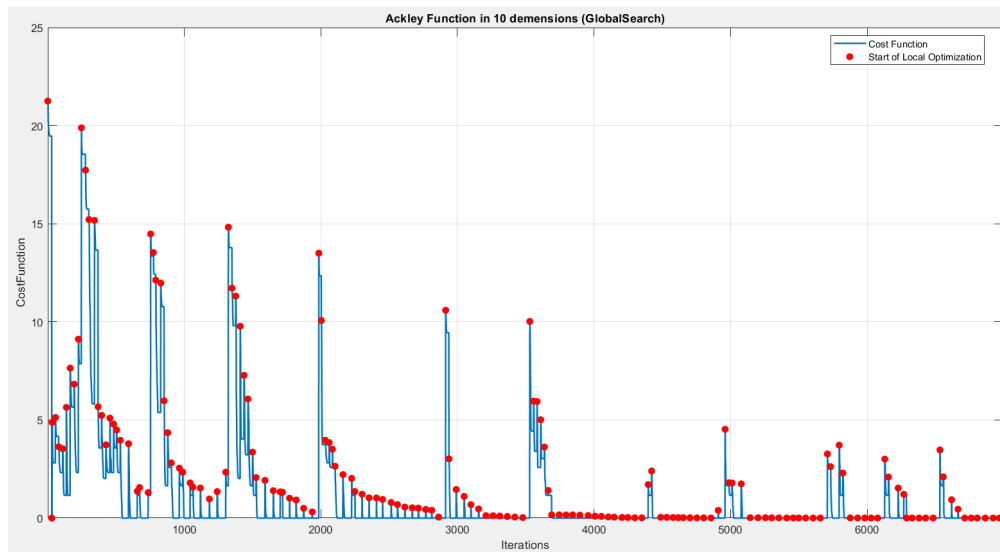
### 6.1 Παράμετροι ελέγχου της μεθόδου τοπικής βελτιστοποίησης `fmincon`

Τα κριτήρια τερματισμού της συνάρτησης `fmincon` επηρεάζουν άμεσα τη διαδικασία βελτιστοποίησης. Στα προβλήματα βελτιστοποίησης της συνάρτησης ελέγχου Ackley στο χώρο των 10 και 30 διαστάσεων χρησιμοποιείται το κριτήριο τερματισμού `StepTolerance` για τη συνολική βελτίωση της απόδοσης. Η ανοχή καθορίζει το ελάχιστο αποδεκτό μέγεθος βήματος των μεταβλητών  $x$  μεταξύ των επαναλήψεων. Εάν η αλλαγή στο μέγεθος βήματος του  $x$  μεταξύ δύο επαναλήψεων είναι μικρότερη από αυτήν την ανοχή, τότε η διαδικασία βελτιστοποίησης τερματίζει. Για αυτόν το λόγο, η ανοχή είναι ορισμένη σε χαμηλή τιμή [20]. Στα προβλήματα βελτιστοποίησης της συνάρτησης ελέγχου Cross-In-Tray στο χώρο των 10 και 30 διαστάσεων, αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων.

### 6.2 Συνάρτηση Ackley σε χώρο 10 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Ackley σε χώρο 10 διαστάσεων από την εφαρμογή του αλγόριθμου GS:

## 6.2. Συνάρτηση Ackley σε χώρο 10 διαστάσεων



**Σχήμα 6.1:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon).

Από το Σχήμα 6.1 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος βελτιστοποίησης GS ελαχιστοποιεί τη συνάρτηση Ackley με την πάροδο του χρόνου. Οι κόκκινες ακίδες στη γραφική παράσταση υποδεικνύουν την τιμή συνάρτησης κόστους των σημείων που πραγματοποιείται η εκκίνηση της τοπικής βελτιστοποίησης. Η συνεχής γραμμή μπλε χρώματος αντιπροσωπεύει την εξέλιξη της τιμής συνάρτησης κόστους κάθε τοπικής βελτιστοποίησης συναρτήσει των επαναλήψεών της.

Ο αλγόριθμος GS εκκινεί την αναζήτησή του από τυχαίο αρχικό σημείο με υψηλή τιμή συνάρτησης κόστους. Η τοπική βελτιστοποίηση επιτυγχάνει μείωση στην τιμή αλλά ο αλγόριθμος βρίσκεται ακόμη μακριά από το καθολικό ελάχιστο. Στη συνέχεια, από το συγκεκριμένο αριθμό δοκιμαστικών σημείων (NumStageOnePoints), επιλέγεται αυτό με την χαμηλότερη τιμή συνάρτησης κόστους και εκτελείται η τοπική βελτιστοποίηση. Το σημείο που επιστρέφεται ως το καθολικό ελάχιστο του προβλήματος είναι το σημείο που λαμβάνεται από την εκτέλεση του σταδίου 1.

Από το Σχήμα 6.1 παρατηρείται ότι καθ' όλη τη διάρκεια της διαδικασίας βελτιστοποίησης τα σημεία εκκίνησης βρίσκονται σε πολύ κοντινές αποστάσεις μεταξύ τους. Συγκεκριμένα, η επιλογή της παραμέτρου DistanceThresholdFactor είναι ορισμένη σε χαμηλή τιμή. Αυτό το γεγονός επιτρέπει την εκτέλεση τοπικών βελτιστοποιήσεων ακόμη και για τα νέα σημεία που βρίσκονται πολύ κοντά σε τοπικές λύσεις που έχουν ήδη βρεθεί σύμφωνα με τη συνθήκη (5.1). Αυτό είναι ωφέλιμο στο συγκεκριμένο πρόβλημα καθώς υπάρχουν πολλά τοπικά ακρότατα σε μικρές αποστάσεις μεταξύ τους.

## 6.2. Συνάρτηση Ackley σε χώρο 10 διαστάσεων

Από το Σχήμα 6.1 παρατηρούνται περιοδικές κορυφές με τα δοκιμαστικά σημεία εκκίνησης να λαμβάνουν υψηλότερες τιμές και να βρίσκονται μακριά από το καθολικό ελάχιστο της συνάρτησης. Στη συνέχεια, μετά την παρουσία των κορυφών διακρίνεται πτωτική τάση στα σημεία εκκίνησης. Η παρουσία, λοιπόν, αυτών των κορυφών οφείλεται στο γεγονός της απόρριψης των δοκιμαστικών σημείων για διαδοχικές επαναλήψεις ίσες με την τιμή `MaxWaitCycle`. Στη συγκεκριμένη περίπτωση, αυξάνεται η τιμή του ορίου (σύμφωνα με την εξίσωση (5.3)) για να προσαρμοστεί η αναζήτηση σε νέες περιοχές. Ακόμη, σημαντικό ρόλο διαδραματίζει και η παράμετρος `PenaltyThresholdFactor`. Είναι ορισμένη σε μέτρια τιμή για τα δοκιμαστικά σημεία που παραβιάζουν τους περιορισμούς. Έτσι, επιτρέπει στον αλγόριθμο την αποδοχή δοκιμαστικών σημείων που χαρακτηρίζονται από υψηλότερες τιμές στη συνάρτηση βαθμολογίας τους. Αυτό έχει ως αποτέλεσμα την αποδοχή σημείων χαμηλότερης ποιότητας που βρίσκονται μακριά από το καθολικό ελάχιστο. Ο σκοπός της συγκεκριμένης διερεύνησης συνιστά την αποφυγή του αλγόριθμου από πρόωρη σύγκλιση σε τοπικό ελάχιστο.

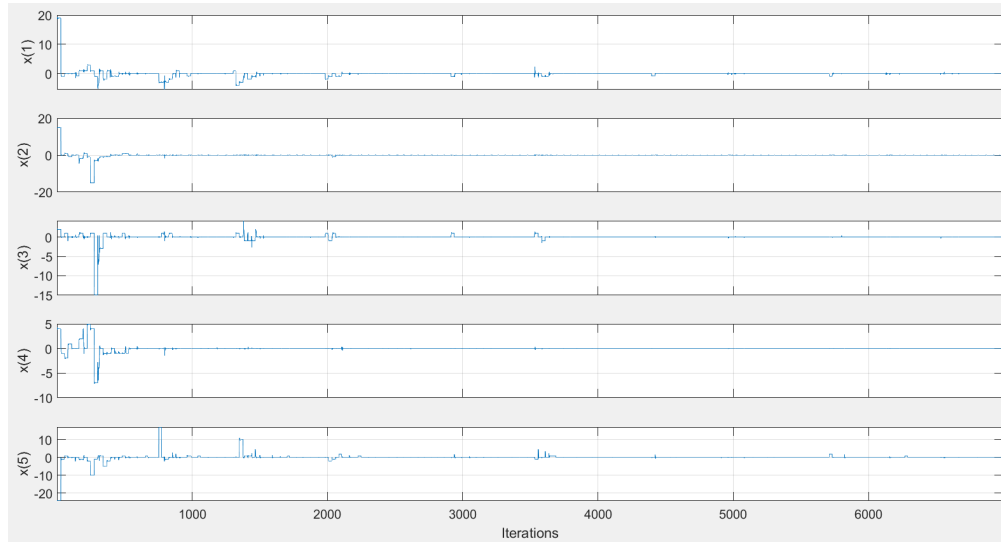
Από το Σχήμα 6.1 διακρίνεται ότι με την πάροδο των επαναλήψεων, η τοπική βελτιστοποίηση εκκινεί από σημεία υψηλότερης ποιότητας. Αυτό συμβαίνει διότι ο αλγόριθμος δίνει προτεραιότητα στα σημεία εκκίνησης με χαμηλή τιμή στη συνάρτησης βαθμολογίας τους σύμφωνα με τη συνθήκη (5.2). Επιπλέον, με τη συνεχή εκτέλεση τοπικών βελτιστοποιήσεων, μειώνεται συνεχώς η τιμή του ορίου. Αυτό έχει ως αποτέλεσμα, να πραγματοποιείται η επιλογή σημείων που είναι πιο κοντά στο καθολικό ελάχιστο. Ακόμη, όταν απορρίπτονται τα δοκιμαστικά σημεία για διαδοχικές επαναλήψεις ίσες με την τιμή `MaxWaitCycle` (σύμφωνα με τη συνθήκη (5.1)), τότε μειώνεται η ακτίνα της λεκάνης έλξης. Σε αυτή την περίπτωση, τα διαδοχικά δοκιμαστικά σημεία βρίσκονται εντός μιας λεκάνης έλξης και ο αλγόριθμος προσαρμόζει την αναζήτηση του. Με την πάροδο των επαναλήψεων, επιτρέπεται η σταδιακή εκτέλεση περισσότερων δοκιμαστικών σημείων γύρω από τις μειωμένες λεκάνες.

Ακολούθως, σε κάθε επανάληψη δημιουργούνται νέα δοκιμαστικά σημεία που προκύπτουν από το συνδυασμό των προηγούμενων που βρίσκονται στο σύνολο αναφοράς (βλέπε ενότητα (5.2.1)). Το σύνολο αναφοράς αποτελείται από σημεία υψηλής ποιότητας και ενημερώνεται συνεχώς με νέα βελτιωμένα. Έτσι, πραγματοποιείται σταδιακή βελτίωση της ποιότητας των σημείων εκκίνησης. Η πτωτική τάση στην τιμή συνάρτησης κόστους έχει ως αποτέλεσμα ο αλγόριθμος να συγκλίνει και να οδηγείται σε κατάσταση ισορροπίας.

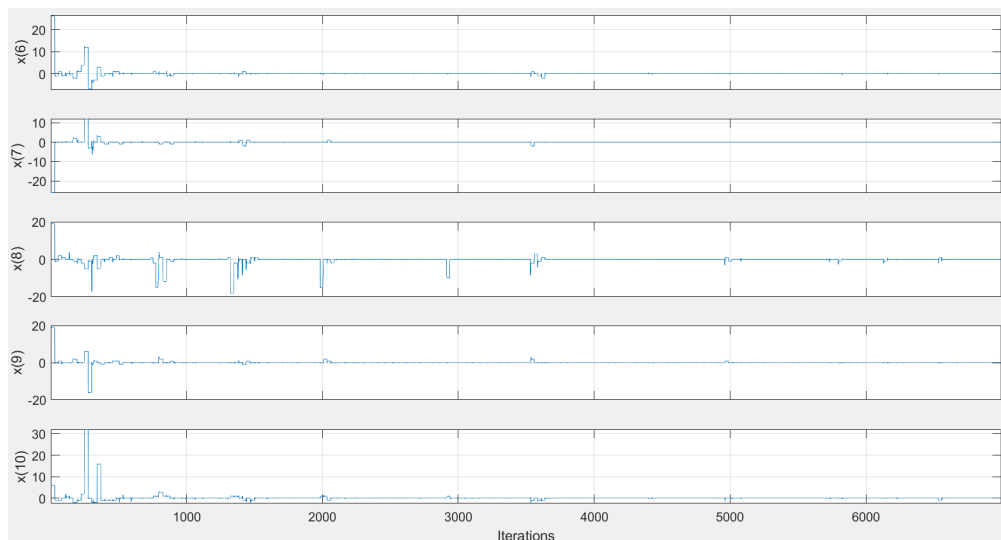
Η διαδικασία βελτιστοποίησης τερματίζει, επειδή ο αλγόριθμος GS αναλύει όλα τα δοκιμαστικά σημεία. Η επιτυγχανόμενη ελάχιστη τιμή είναι  $8.88 \cdot 10^{-16}$  με τιμές των σημείων λύσης ίσες με μηδέν.

## 6.2. Συνάρτηση Ackley σε χώρο 10 διαστάσεων

---



**Σχήμα 6.2:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS.



**Σχήμα 6.3:** Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Ackley σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS.

## 6.2. Συνάρτηση Ackley σε χώρο 10 διαστάσεων

Τα σχήματα 6.2 και 6.3 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής εισόδου  $x$  σε κάθε διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Σε κάθε διάσταση διακρίνεται ότι στις πρώτες περίπου 3,500 επαναλήψεις, υπάρχουν μεγαλύτερες διακυμάνσεις και συμβαίνουν συνεχώς αλλαγές στις τιμές. Αυτό συμβαίνει γιατί στις αρχικές επαναλήψεις, το σύνολο αναφοράς περιλαμβάνει σημεία που καλύπτουν διαφορετικές περιοχές του χώρου αναζήτησης. Αντιθέτως, στο διάστημα από τις 3,500 επαναλήψεις, μέχρι τον τερματισμό της διαδικασίας βελτιστοποίησης, παρατηρούνται μικρότερες αλλαγές. Αυτό οφείλεται στο γεγονός ότι με την πάροδο των επαναλήψεων, το σύνολο αναφοράς ενημερώνεται συνεχώς με βελτιωμένα σημεία που βρίσκονται σε περιοχές πιο κοντά στο καθολικό ελάχιστο. Τα άνω και κάτω όρια της μεταβλητής  $x$  σε κάθε διάσταση προσδιορίζονται από το εύρος  $[-32.7680, 32.7680]$ .

Οι παραμετροποιήσεις που αξιοποιούνται για τα αποτελέσματα των παραπάνω σχημάτων για τον αλγόριθμο GS και την τοπική μέθοδο βελτιστοποίησης `fmincon` φαίνονται παρακάτω:

**Πίνακας 6.1:** Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων

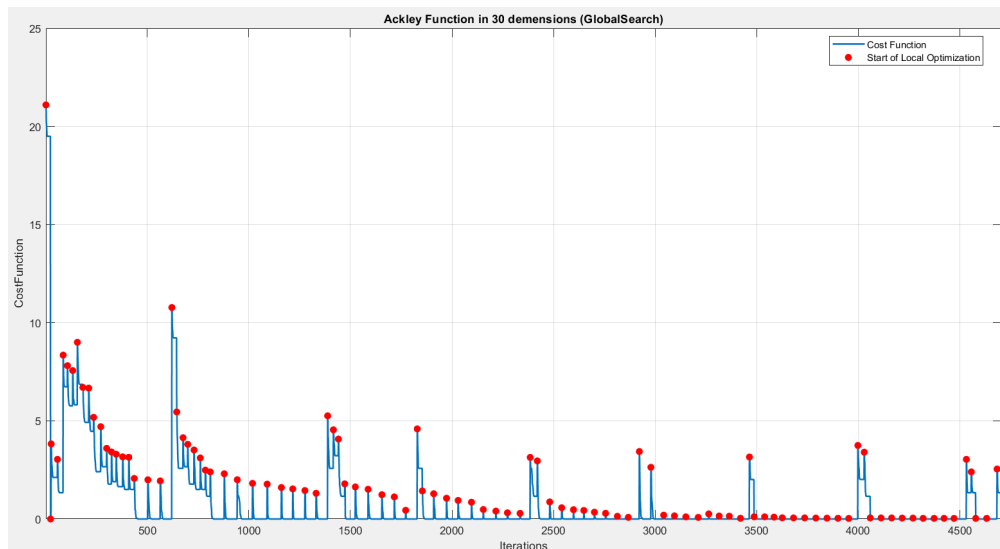
Παράμετρος	Επιλογή
NumTrialPoints	5,000
NumStageOnePoints	700
PenaltyThresholdFactor	0.5
DistanceThresholdFactor	$1.00 \cdot 10^{-04}$
BasinRadiusFactor	0.3
MaxWaitCycle	40
StartPointsToRun	bounds
MaxTime	60
FunctionTolerance	$1.00 \cdot 10^{-06}$
XTolerance	$1.00 \cdot 10^{-06}$
Display	iter

**Πίνακας 6.2:** Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης `fmincon` για τη συνάρτηση Ackley σε χώρο 10 διαστάσεων

Παράμετρος	Επιλογή
Algorithm	interior-point
StepTolerance	$1.00 \cdot 10^{-15}$

## 6.3 Συνάρτηση Ackley σε χώρο 30 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Ackley σε χώρο 30 διαστάσεων από την εφαρμογή του αλγόριθμου GS:



**Σχήμα 6.4:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Ackley σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon).

Από το Σχήμα 6.4 παρατηρείται πτωτική πορεία στην τιμή συνάρτησης κόστους, επισημαίνοντας ότι ο αλγόριθμος βελτιστοποίησης GS ελαχιστοποιεί τη συνάρτηση Ackley με την πάροδο του χρόνου. Οι κόκκινες ακίδες στη γραφική παράσταση υποδεικνύουν την τιμή συνάρτησης κόστους των σημείων που πραγματοποιείται η εκκίνηση της τοπικής βελτιστοποίησης. Η συνεχής γραμμή μπλε χρώματος αντιπροσωπεύει την εξέλιξη της τιμής συνάρτησης κόστους κάθε τοπικής βελτιστοποίησης συναρτήσει των επαναλήψεών της.

Η μορφή και περιγραφή της γραφικής παράστασης του Σχήματος 6.4 παρουσιάζει κοινή δομή με τη γραφική παράσταση του Σχήματος 6.1. (βλέπε ενότητα 6.2)

Από το Σχήμα 6.4 διακρίνεται ότι ο αλγόριθμος εκτελεί λιγότερες τοπικές βελτιστοποιήσεις συγκριτικά με αυτές που έχουν συμβεί στο σχήμα 6.1. Αυτό συμβαίνει γιατί η παράμετρος DistanceThresholdFactor είναι αυξημένη και η επίδρασή της στη συνθήκη (5.1) γίνεται πιο αυστηρή. Συγκρίνοντας τα δύο σχήματα, παρατηρείται ότι τα σημεία εκκίνησης του σχήματος 6.4 βρίσκονται πιο κοντά στο καθολικό ελάχιστο. Αυτό καθορίζεται από την επιλογή της παραμέτρου MaxWaitCycle. Η υψηλότερη τιμή της απορρίπτει ένα μεγαλύτερο αριθμό δοκιμαστικών σημείων. Αυτό επηρεάζει την αναζήτηση του αλγόριθμου καθώς εστιάζει στη συνεχή βελτίωση των λύσεων του εντός εφικτών περιοχών. Επιπλέον, με λιγότερες προσαρμογές της εξίσωσης (5.3)

#### 6.4. Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων

μειώνεται η ποικιλομορφία των σημείων εκκίνησης.

Η διαδικασία βελτιστοποίησης τερματίζει, επειδή ο αλγόριθμος GS αναλύει όλα τα δοκιμαστικά σημεία. Η επιτυγχανόμενη ελάχιστη τιμή είναι  $8.88 \cdot 10^{-16}$  με τιμές των σημείων λύσης ίσες με μηδέν.

Οι παραμετροποιήσεις που αξιοποιούνται για τα αποτελέσματα των παραπάνω σχημάτων για τον αλγόριθμο GS και την τοπική μέθοδο βελτιστοποίησης fmincon φαίνονται παρακάτω:

**Πίνακας 6.3:** Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων

Παράμετρος	Επιλογή
NumTrialPoints	5,000
NumStageOnePoints	700
PenaltyThresholdFactor	0.6
DistanceThresholdFactor	$1.00 \cdot 10^{-02}$
BasinRadiusFactor	$1.00 \cdot 10^{-01}$
MaxWaitCycle	70
StartPointsToRun	bounds
MaxTime	60
FunctionTolerance	$1.00 \cdot 10^{-06}$
XTolerance	$1.00 \cdot 10^{-06}$
Display	iter

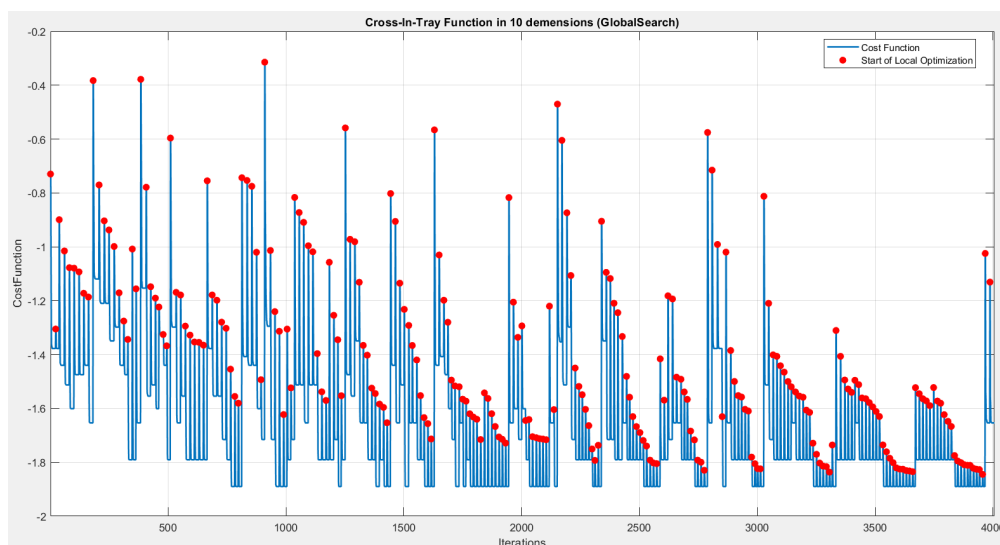
**Πίνακας 6.4:** Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης fmincon για τη συνάρτηση Ackley σε χώρο 30 διαστάσεων

Παράμετρος	Επιλογή
Algorithm	interior-point
StepTolerance	$1.00 \cdot 10^{-15}$

#### 6.4 Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Cross-In-Tray σε χώρο 10 διαστάσεων από την εφαρμογή του αλγόριθμου GS:

#### 6.4. Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων



**Σχήμα 6.5:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon).

Στο Σχήμα 6.5 διακρίνεται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος βελτιστοποίησης GS ελαχιστοποιεί τη συνάρτηση Cross-In-tray με την πάροδο του χρόνου. Οι κόκκινες ακίδες στη γραφική παράσταση υποδεικνύουν την τιμή συνάρτησης κόστους των σημείων που πραγματοποιείται η εκκίνηση της τοπικής βελτιστοποίησης. Η συνεχής γραμμή μπλε χρώματος αντιπροσωπεύει την εξέλιξη της τιμής συνάρτησης κόστους κάθε τοπικής βελτιστοποίησης συναρτήσει των επαναλήψεων της.

Στο κεφάλαιο 3 έχει αναφερθεί ότι για τη συνάρτηση Cross-In-Tray για διάσταση μεγαλύτερη από δύο, δεν είναι γνωστή η θέση ή οι θέσεις και η τιμή του καθολικού ελαχίστου (βλέπε ενότητα (3.3)). Η συγκεκριμένη παραμετροποίηση έχει σκοπό τη διερεύνηση διαφορετικών περιοχών για την εύρεση της ελάχιστης λύσης.

Από το Σχήμα 6.5 διακρίνεται ότι σε μεγάλη διάρκεια της διαδικασίας βελτιστοποίησης, τα σημεία εκκίνησης διαφοροποιούνται και καλύπτουν μεγάλο μέρος του χώρου αναζήτησης. Σύμφωνα με τη συνθήκη (5.1) για υψηλότερες τιμές του γινομένου  $DistanceThresholdFactor \cdot radius(i)$  απαιτούνται μεγαλύτερες αποστάσεις μεταξύ σημείου και θέσης μιας οποιαδήποτε τοπικής λύσης για την εκτέλεση της τοπικής βελτιστοποίησης. Ωστόσο, προς το τέλος της διαδικασίας βελτιστοποίησης, τα σημεία εκκίνησης βρίσκονται σε κοντινότερη απόσταση μεταξύ τους. Αυτό συμβαίνει διότι, όταν απορρίπτονται τα δοκιμαστικά σημεία για διαδοχικές επαναλήψεις ίσες με την τιμή  $MaxWaitCycle$ , τότε μειώνεται η ακτίνα της λεκάνης έλξης (σύμφωνα με την παράμετρο  $BasinRadiusFactor$ ). Σε αυτή την περίπτωση, το γινόμενο  $DistanceThresholdFactor \cdot radius(i)$  με την πάροδο των επαναλήψεων, γίνεται



#### 6.4. Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων

---

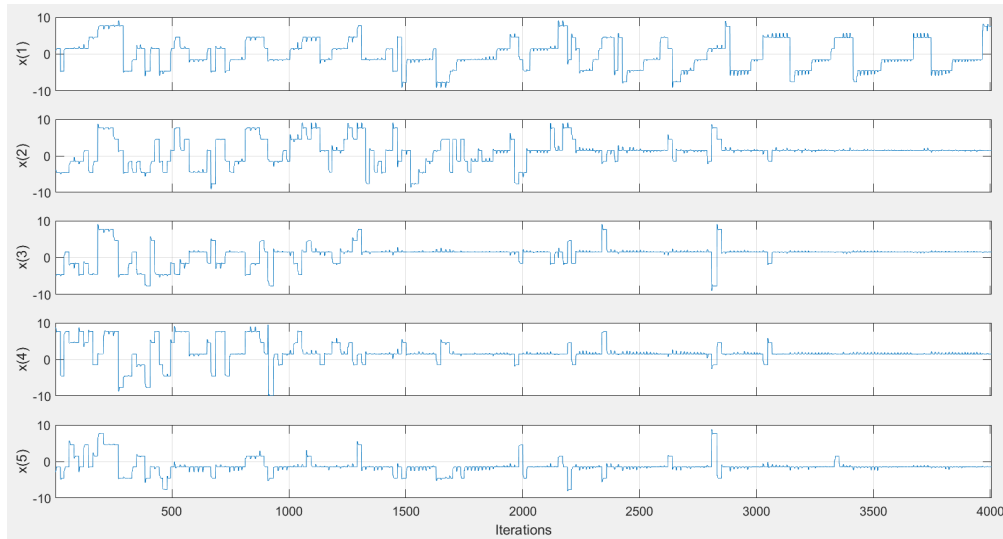
συνεχώς μικρότερο.

Από το Σχήμα 6.5 παρατηρούνται περιοδικές κορυφές με τα δοκιμαστικά σημεία εκκίνησης να λαμβάνουν υψηλότερες τιμές. Η παρουσία των κορυφών οφείλεται στο γεγονός της απόρριψης των δοκιμαστικών σημείων για διαδοχικές επαναλήψεις ίσες με την τιμή `MaxWaitCycle`. Στη συγκεκριμένη περίπτωση, αυξάνεται η τιμή του ορίου (σύμφωνα με την εξίσωση (5.3)) για να προσαρμοστεί η αναζήτηση σε νέες περιοχές. Επιπρόσθετα, η παράμετρος `PenaltyThresholdFactor` είναι ορισμένη σε μία μέτρια τιμή για τα δοκιμαστικά σημεία που παραβιάζουν τους περιορισμούς. Επομένως, επιτρέπει στον αλγόριθμο την αποδοχή δοκιμαστικών σημείων που χαρακτηρίζονται από υψηλότερες τιμές στη συνάρτηση βαθμολογίας τους. Άρα, πραγματοποιείται η αποδοχή σημείων χαμηλότερης ποιότητας που βρίσκονται μακριά από τη βέλτιστη λύση.

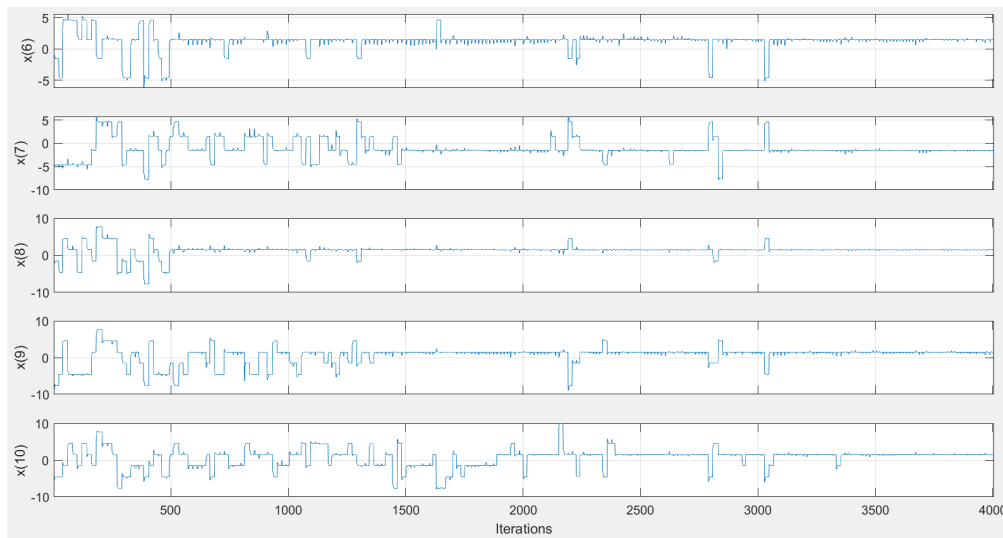
Από το Σχήμα 6.5 διακρίνεται ότι τα σημεία εκκίνησης, προς το τέλος της διαδικασίας βελτιστοποίησης, βρίσκονται πιο κοντά στην ελάχιστη τιμή συνάρτησης κόστους. Σύμφωνα με τη συνθήκη (5.2), ο αλγόριθμος επιλέγει τα σημεία εκκίνησης βασιζόμενος στη συνάρτηση βαθμολογίας τους. Με την πάροδο των επαναλήψεων, εκτελείται η τοπική βελτιστοποίηση για τα σημεία με τις χαμηλότερες τιμές συναρτήσεων βαθμολογίας. Εν συνεχεία, το σύνολο αναφοράς αποτελείται από σημεία υψηλής ποιότητας και ενημερώνεται συνεχώς με νέα βελτιωμένα (βλέπε ενότητα (5.2.1)). Έτσι, πραγματοποιείται η σταδιακή βελτίωση της ποιότητας των σημείων εκκίνησης. Η πτωτική τάση στην τιμή συνάρτησης κόστους καθ' όλη τη διαδικασία βελτιστοποίησης έχει ως αποτέλεσμα ο αλγόριθμος να συγκλίνει και να οδηγείται σε κατάσταση ισορροπίας.

Η διαδικασία βελτιστοποίησης τερματίζει, επειδή ο αλγόριθμος GS αναλύει όλα τα δοκιμαστικά σημεία. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στο στάδιο 2, που πολλές από τις τοπικές βελτιστοποιήσεις, επιστρέφουν το ίδιο ελάχιστο. Η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $-1.89$ .

#### 6.4. Συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων



Σχήμα 6.6: Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 1 έως το 5 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS.



Σχήμα 6.7: Η εξέλιξη της μεταβλητής  $x$  σε κάθε διάσταση από το 6 έως το 10 (συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS.

### 6.5. Συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων

Τα σχήματα 6.6 και 6.7 περιέχουν τη γραφική παράσταση εξέλιξης της μεταβλητής  $x$  σε κάθε διάσταση συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Σε κάθε διάσταση διακρίνεται ότι, για μεγάλο μέρος της διαδικασίας βελτιστοποίησης, υπάρχουν μεγαλύτερες διακυμάνσεις και συμβαίνουν συνεχώς αλλαγές στις τιμές. Αυτό συμβαίνει διότι η συγκεκριμένη παραμετροποίηση έχει στόχο τη διερεύνηση διαφορετικών περιοχών για την εύρεση της ελάχιστης λύσης. Αντιθέτως, προς το τέλος της διαδικασίας βελτιστοποίησης παρατηρούνται μικρότερες αλλαγές. Πιο συγκεκριμένα, με την πάροδο των επαναλήψεων, το σύνολο αναφοράς ενημερώνεται συνεχώς με βελτιωμένα σημεία που βρίσκονται σε περιοχές πιο κοντά στην ελάχιστη τιμή. Τα άνω και κάτω όρια της μεταβλητής  $x$  σε κάθε διάσταση προσδιορίζονται από το εύρος  $[-10, 10]$  και οι τελικές τιμές των  $x$  είναι πολύ κοντά στις τιμές 1.4705 και  $-1.4705$ .

Για την τοπική μέθοδο βελτιστοποίησης `fmincon` αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση του αλγόριθμου GS για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:

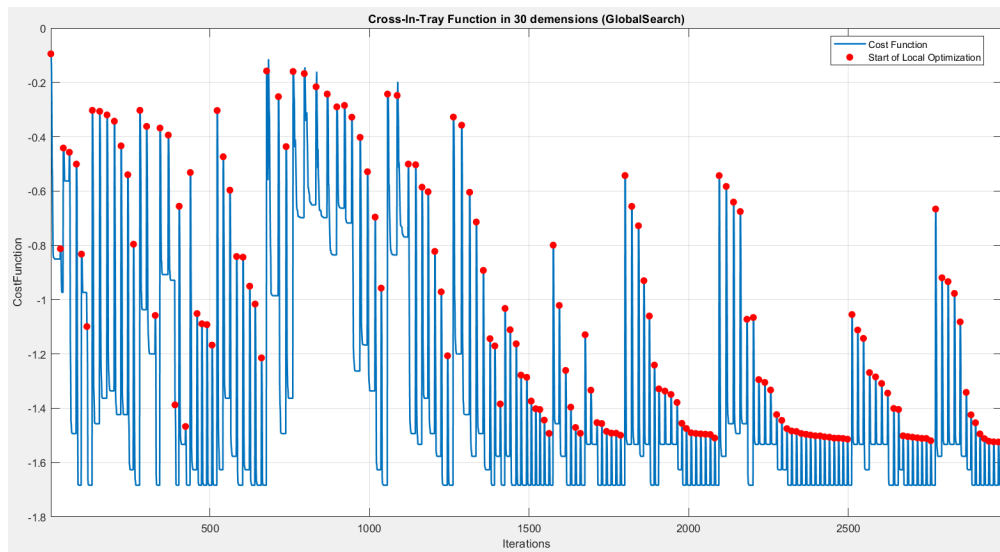
**Πίνακας 6.5:** Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Cross-In-Tray σε χώρο 10 διαστάσεων

Παράμετρος	Επιλογή
NumTrialPoints	6,000
NumStageOnePoints	600
PenaltyThresholdFactor	0.5
DistanceThresholdFactor	0.5
BasinRadiusFactor	0.4
MaxWaitCycle	70
StartPointsToRun	bounds
MaxTime	60
FunctionTolerance	$1.00 \cdot 10^{-06}$
XTolerance	$1.00 \cdot 10^{-06}$
Display	iter

### 6.5 Συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων

Παρακάτω φαίνονται οι γραφικές παραστάσεις της συνάρτησης Cross-In-Tray σε χώρο 30 διαστάσεων από την εφαρμογή του αλγόριθμου GS:

## 6.5. Συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων



**Σχήμα 6.8:** Η εξέλιξη της τιμής συνάρτησης κόστους (συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων) συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon).

Από το Σχήμα 6.8 παρατηρείται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος βελτιστοποίησης GS ελαχιστοποιεί τη συνάρτηση Cross-In-tray με την πάροδο του χρόνου. Οι κόκκινες ακίδες στη γραφική παράσταση υποδεικνύουν την τιμή συνάρτησης κόστους των σημείων που πραγματοποιείται η εκκίνηση της τοπικής βελτιστοποίησης. Η συνεχής γραμμή μπλε χρώματος αντιπροσωπεύει την εξέλιξη της τιμής συνάρτησης κόστους κάθε τοπικής βελτιστοποίησης συναρτήσει των επαναλήψεων της.

Στο κεφάλαιο 3 έχει αναφερθεί ότι για τη συνάρτηση Cross-In-Tray για διάσταση μεγαλύτερη από δύο, δεν είναι γνωστή η θέση ή οι θέσεις και η τιμή του καθολικού ελαχίστου (βλέπε ενότητα (3.3)). Στις δύο διαστάσεις, η συνάρτηση διακρίνεται από πολύπλοκο τοπίο με πολλά τοπικά ελάχιστα και περιορίζεται σε μικρό χώρο αναζήτησης. Άρα, με την αύξηση του αριθμού των διαστάσεων σε τριάντα, αυξάνεται εκθετικά η πολυπλοκότητα. Η συγκεκριμένη παραμετροποίηση έχει ως σκοπό τη διερεύνηση διαφορετικών περιοχών ώστε να μη χαθεί κάποια που ενδεχομένως να καταλήγει στη βέλτιστη λύση.

Η μορφή και η περιγραφή της γραφικής παράστασης του σχήματος 6.8 παρουσιάζει κοινή δομή με τη γραφική παράσταση του σχήματος 6.5. (βλέπε ενότητα 6.4)

Από το Σχήμα 6.8 παρατηρείται ότι ο αλγόριθμος εκτελεί λιγότερες τοπικές βελτιστοποιήσεις συγκριτικά με αυτές που πραγματοποιούνται στο χώρο των 10 διαστάσεων. Η υψηλότερη τιμή της παραμέτρου MaxWaitCycle απορρίπτει ένα μεγαλύτερο αριθμό δοκιμαστικών σημείων. Έτσι, με λιγότερες προσαρμογές του ορίου,

### 6.5. Συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων

ο αλγόριθμος εστιάζει στη συνεχή βελτίωση των λύσεων του εντός διαφορετικών ε-φικτών περιοχών.

Η διαδικασία βελτιστοποίησης τερματίζει, επειδή ο αλγόριθμος GS αναλύει όλα τα δοκιμαστικά σημεία. Η επιτυγχανόμενη ελάχιστη τιμή λαμβάνεται στο στάδιο 2, που πολλές από τις τοπικές βελτιστοποιήσεις, επιστρέφουν το ίδιο ελάχιστο. Η επιτυγχανόμενη ελάχιστη τιμή είναι ίση με  $-1.6835$ .

Για την τοπική μέθοδο βελτιστοποίησης `fmincon` αξιοποιούνται οι προκαθορισμένες επιλογές των παραμέτρων. Η παραμετροποίηση του αλγόριθμου GS για τα αποτελέσματα των παραπάνω σχημάτων φαίνεται παρακάτω:

**Πίνακας 6.6:** Παράμετροι ελέγχου του αλγόριθμου GS για τη συνάρτηση Cross-In-Tray σε χώρο 30 διαστάσεων

Παράμετρος	Επιλογή
NumTrialPoints	6,000
NumStageOnePoints	600
PenaltyThresholdFactor	0.4
DistanceThresholdFactor	0.35
BasinRadiusFactor	0.3
MaxWaitCycle	150
StartPointsToRun	bounds
MaxTime	60
FunctionTolerance	$1.00 \cdot 10^{-06}$
XTolerance	$1.00 \cdot 10^{-06}$
Display	iter



## Κεφάλαιο 7

# Μεγιστοποίηση Συνάρτησης Πιθανοφάνειας με Εφαρμογή Καθολικής Αναζήτησης

### 7.1 Παράμετροι ελέγχου της μεθόδου τοπικής βελτιστοποίησης `fmincon`

Στη διαδικασία βελτιστοποίησης, η συνάρτηση `fmincon` αξιολογεί πολλές φορές τη συνάρτηση που υπολογίζεται η τιμή NLL. Αυτό έχει ως συνέπεια, την αύξηση του υπολογιστικού χρόνου βελτιστοποίησης. Έτσι, χρησιμοποιείται η τεχνική της χρήσης πολλαπλών επεξεργαστών, δηλαδή ορίζεται η παράμετρος `UseParallel` να είναι ενεργή. Οι αξιολογήσεις της συνάρτησης γίνονται ταυτόχρονα σε πολλαπλούς επεξεργαστές με αποτέλεσμα την επιτάχυνση της διαδικασίας βελτιστοποίησης [20].

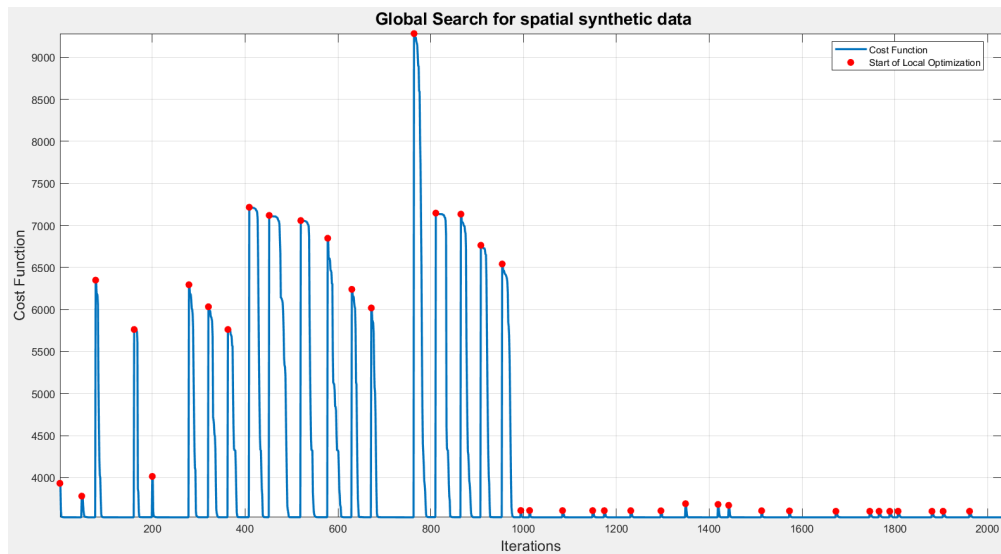
### 7.2 Περιγραφή του προβλήματος

Η περιγραφή του προβλήματος αναφέρεται στην ενότητα (4.1).

#### 7.2.1 Αποτελέσματα για το μοντέλο SLI και χωρικά συνθετικά δεδομένα

Παρακάτω φαίνονται οι γραφικές παραστάσεις από την εφαρμογή της μεθόδου GS στα χωρικά συνθετικά δεδομένα:

## 7.2. Περιγραφή του προβλήματος



**Σχήμα 7.1:** Η εξέλιξη της τιμής NLL στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Οι κόκκινες ακίδες υποδηλώνουν τις επαναλήψεις στις οποίες εκκινεί η διαδικασία τοπικής βελτιστοποίησης (fmincon). Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $3.5285127 \cdot 10^3$  και λαμβάνεται στην 735η επανάληψη.

Από το Σχήμα 7.1 παρατηρείται πτωτική πορεία στην τιμή συνάρτησης κόστους, γεγονός ότι ο αλγόριθμος GS ελαχιστοποιεί την τιμή NLL με την πάροδο του χρόνου. Ο αλγόριθμος συγκλίνει κοντά σε τιμή ίση με  $3.528 \cdot 10^3$ . Οι κόκκινες ακίδες στη γραφική παράσταση υποδεικνύουν την τιμή συνάρτησης κόστους των σημείων που πραγματοποιείται η εκκίνηση της τοπικής βελτιστοποίησης. Η τυχαία αρχική τιμή είναι ίση με  $3.9329 \cdot 10^3$  και η τυχαία αρχική τιμή του διανύσματος **Param0** είναι ίση με  $[2.7293 \ 1 \ 1,000 \ 1.5]$ .

Από το Σχήμα 7.1 διακρίνεται ότι, μέχρι το μέσον της διαδικασίας βελτιστοποίησης, τα σημεία εκκίνησης βρίσκονται μακριά από την ελαχιστοποιημένη τιμή NLL. Η υψηλή τιμή της παραμέτρου *DistanceThresholdFactor* διασφαλίζει ότι τα σημεία είναι διαφοροποιημένα και δεν τοποθετούνται πολύ κοντά σε οποιαδήποτε τοπική λύση. Έτσι, για την εκτέλεση της τοπικής βελτιστοποίησης οι αποστάσεις μεταξύ νέου σημείου και της θέσης μιας οποιαδήποτε τοπικής λύσης απαιτείται να είναι μεγαλύτερη του γινομένου  $DistanceThresholdFactor \cdot radius(i)$ .

Από το Σχήμα 7.1 παρατηρούνται περιοδικές κορυφές όπου τα δοκιμαστικά σημεία εκκίνησης λαμβάνουν υψηλότερες τιμές. Έπειτα, διακρίνεται πτωτική πορεία στις τιμές των σημείων εκκίνησης. Η παρουσία των κορυφών οφείλεται στο γεγονός της απόρριψης των δοκιμαστικών σημείων για διαδοχικές επαναλήψεις ίσες με την τιμή *MaxWaitCycle*. Στη συγκεκριμένη περίπτωση, αυξάνεται η τιμή του ορίου (σύμφωνα με την εξίσωση (5.3)) για να προσαρμοστεί η αναζήτηση σε νέες περιοχές.

Από το Σχήμα 7.1 διακρίνεται ότι, μετά την πάροδο των 1,000 επαναλήψεων, τα



## 7.2. Περιγραφή του προβλήματος

---

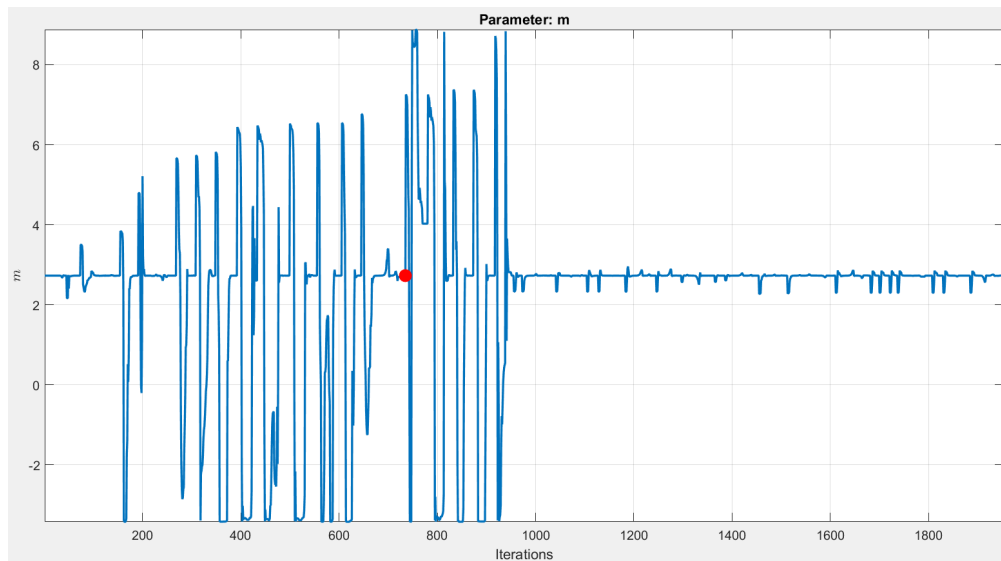
σημεία εκκίνησης βρίσκονται πιο κοντά στην ελαχιστοποιημένη τιμή NLL. Αναλυτικότερα, με την πάροδο των επαναλήψεων, το σύνολο αναφοράς αποτελείται από σημεία υψηλής ποιότητας και ενημερώνεται συνεχώς με νέα βελτιωμένα (βλέπε ενότητα (5.2.1)). Αυτή η επαναληπτική διαδικασία επιτρέπει τη σταδιακή βελτίωση της ποιότητας των σημείων εκκίνησης. Επιπρόσθετα, από τη συνθήκη (5.2) επιλέγονται τα σημεία εκκίνησης βάσει της συνάρτησης βαθμολογίας τους. Έτσι, με την πάροδο των επαναλήψεων, επιλέγονται τα σημεία με μηδενικές τιμές του αθροίσματος των παραβιασμένων περιορισμών.

Συνακολούθως, σύμφωνα με τη συνθήκη (5.1), όταν απορρίπτονται τα δοκιμαστικά σημεία για διαδοχικές επαναλήψεις ίσες με την τιμή `MaxWaitCycle`, τότε μειώνεται η ακτίνα της λεκάνης έλξης. Άρα, με την πάροδο των επαναλήψεων, επιτρέπεται η σταδιακή εκτέλεση περισσότερων δοκιμαστικών σημείων γύρω από αυτές τις μειωμένες λεκάνες. Αυτό έχει ως συνέπεια, να βρίσκονται πιο κοντά στην ελάχιστη τιμή NLL.

Από το Σχήμα 7.1 διακρίνεται ότι, πριν την κάθε εκ νέου τοπική βελτιστοποίηση, ο αλγόριθμος συγκλίνει σε λύσεις πολύ κοντά στην τιμή που επέστρεψε ως ελάχιστη. Ο αλγόριθμος θεωρείται ότι φτάνει σε κατάσταση ισορροπίας σχετικά νωρίς στη διαδικασία βελτιστοποίησης και όλες οι περαιτέρω εκ νέου αναζητήσεις επιστρέφουν σε αυτή την κατάσταση.

Η διαδικασία βελτιστοποίησης τερματίζει επειδή ο αλγόριθμος GS αναλύει όλα τα δοκιμαστικά σημεία. Η επιτυγχανόμενη ελάχιστη τιμή NLL είναι ίση με  $3.5285127 \cdot 10^3$ . Η συνολική μείωση είναι σημαντική και περίπου ίση με 404.4. Επομένως, επιδεικνύεται η αποτελεσματικότητα του αλγόριθμου GS στην ελαχιστοποίηση της NLL για το σύνολο των χωρικών συνθετικών δεδομένων.

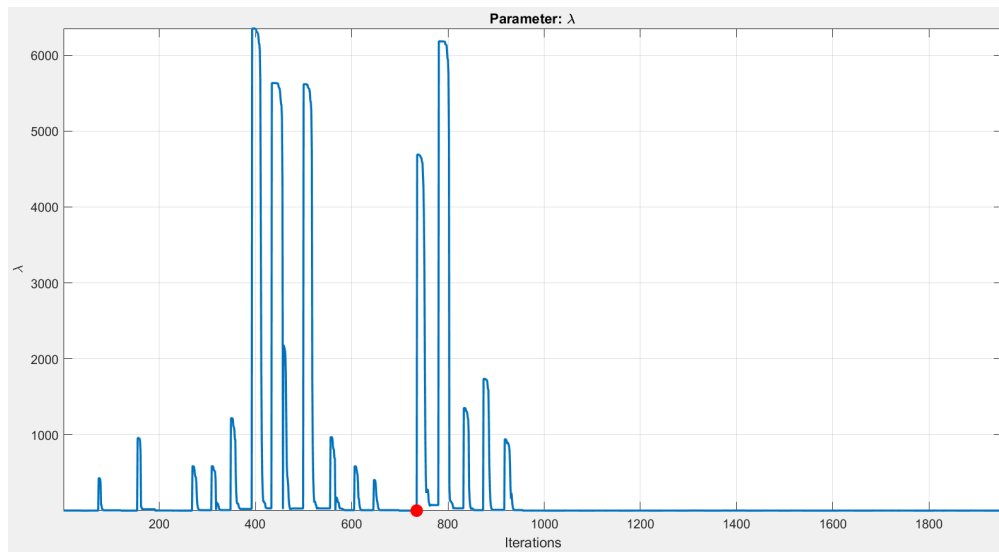
## 7.2. Περιγραφή του προβλήματος



**Σχήμα 7.2:** Η εξέλιξη της τιμής παραμέτρου  $m$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 7.2 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $m$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Η αρχική τιμή της είναι ίση με 2.7293, ενώ η βελτιστοποιημένη είναι ίση με 2.7293. Το άνω και κάτω όριο είναι 8.88 και  $-3.4214$  αντίστοιχα. Η βέλτιστη τιμή της είναι ίση με την αρχική. Η αρχική τιμή της παραμέτρου  $m$  ορίζεται ως ο μέσος όρος των δειγματοληπτικών τιμών. Αυτό το γεγονός, συνιστά καλή εκτίμηση του πραγματικού μέσου όρου των δεδομένων. Επομένως, η αρχική τιμή της βρίσκεται σε ευνοϊκή περιοχή και είναι ακριβής.

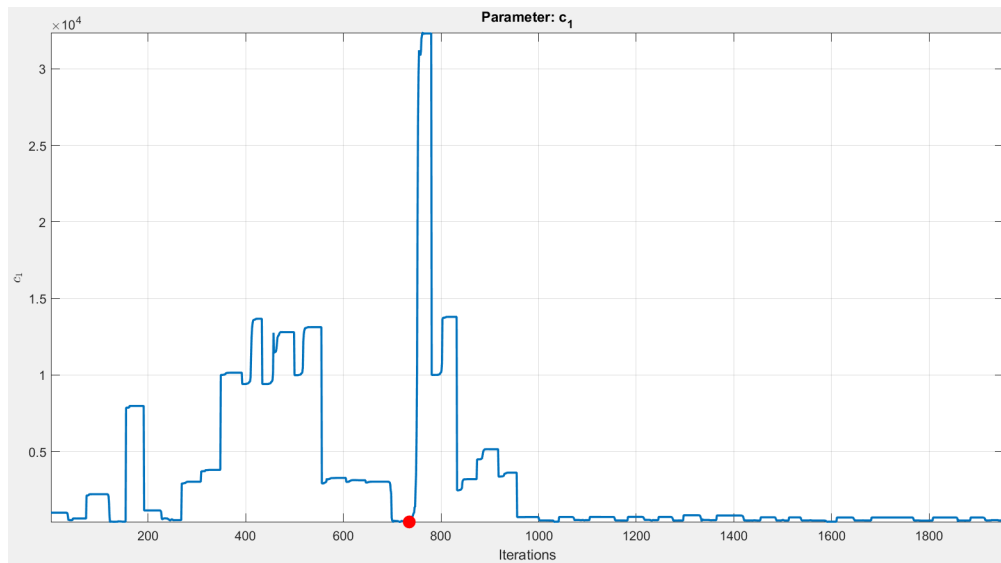
## 7.2. Περιγραφή του προβλήματος



**Σχήμα 7.3:** Η εξέλιξη της τιμής παραμέτρου  $\lambda$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 7.3 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\lambda$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Η αρχική τιμή της είναι ίση με 1, ενώ η βελτιστοποιημένη είναι ίση με 0.9049. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Ο συντελεστής κλίμακας  $\lambda$  είναι ανάλογος με τη συνολική διασπορά στα δεδομένα. Στην αρχική φάση βελτιστοποίησης καλύπτεται σχετικά μεγάλο εύρος του χώρου αναζήτησης. Αντιθέτως, προς το τέλος της διαδικασίας η τιμή της παραμέτρου κυμαίνεται γύρω από μία συγκεκριμένη περιοχή. Επομένως, οι υψηλότερες τιμές της προκαλούν αύξηση στην τιμή NLL, ενώ οι χαμηλότερες προκαλούν μείωση. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής είναι παρατηρήσιμη καθώς υπάρχει μεταβολή στην συνολική διασπορά των δεδομένων.

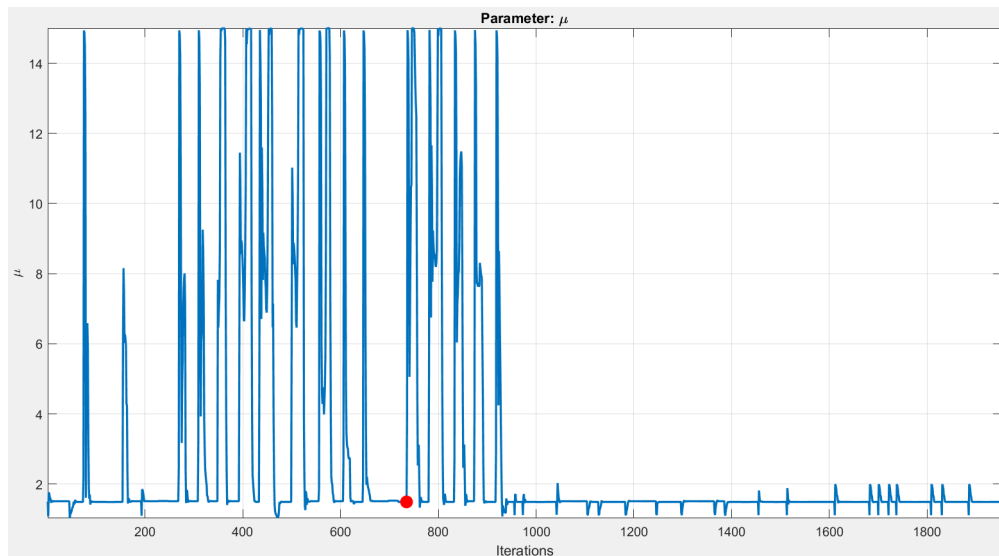
## 7.2. Περιγραφή του προβλήματος



**Σχήμα 7.4:** Η εξέλιξη της τιμής παραμέτρου  $c_1$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 7.4 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $c_1$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Η αρχική τιμή της είναι ίση με 1,000, ενώ η βελτιστοποιημένη είναι ίση με 395.6649. Το άνω και κάτω όριο είναι  $Inf$  και  $2.2204 \cdot 10^{-16}$  αντίστοιχα. Οι υψηλότερες τιμές της προκαλούν αύξηση στην τιμή NLL, ενώ οι χαμηλότερες προκαλούν μείωση. Αυτό το γεγονός υποδεικνύει ότι η βέλτιστη τιμή της για την ελαχιστοποίηση της NLL βρίσκεται σε χαμηλότερες τιμές συγκριτικά με την αρχική. Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής είναι σημαντική.

## 7.2. Περιγραφή του προβλήματος



**Σχήμα 7.5:** Η εξέλιξη της τιμής παραμέτρου  $\mu$  του μοντέλου SLI στα χωρικά συνθετικά δεδομένα συναρτήσει των επαναλήψεων από την εφαρμογή του GS. Η κόκκινη κουκκίδα υποδηλώνει την επανάληψη στην οποία βρίσκεται η βέλτιστη τιμή της παραμέτρου που ελαχιστοποιεί τη NLL.

Το Σχήμα 7.5 περιέχει τη γραφική παράσταση εξέλιξης της παραμέτρου  $\mu$  συναρτήσει των επαναλήψεων από την εκτέλεση του αλγόριθμου GS. Η αρχική τιμή της είναι ίση με 1.5, ενώ η βελτιστοποιημένη είναι ίση με 1.4906. Το άνω και κάτω όριο είναι 15 και 0.5 αντίστοιχα. Αναλυτικότερα, οι υψηλότερες τιμές της παραμέτρου  $\mu$  προκαλούν αύξηση στην τιμή NLL, ενώ οι χαμηλότερες προκαλούν μείωση. Η βέλτιστη τιμή της βρίσκεται πιθανόν στο διάστημα  $[1.4, 2.5]$ . Η μεταβολή μεταξύ της αρχικής και βέλτιστης τιμής θεωρείται σχετικά μικρή. Αυτό υποδεικνύει ότι η αρχική εκτίμηση για την παράμετρο  $\mu$  βρίσκεται σε περιοχή κοντά στη βέλτιστη τιμή της.

Οι παραμετροποιήσεις που αξιοποιούνται για τα αποτελέσματα των παραπάνω σχημάτων για τον αλγόριθμο GS και την τοπική μέθοδο βελτιστοποίησης `fmincon` φαίνονται παρακάτω:

## 7.2. Περιγραφή του προβλήματος

**Πίνακας 7.1:** Παράμετροι ελέγχου του αλγόριθμου GS για τα χωρικά συνθετικά δεδομένα

Παράμετρος	Επιλογή
NumTrialPoints	1,500
NumStageOnePoints	300
PenaltyThresholdFactor	0.35
DistanceThresholdFactor	0.5
BasinRadiusFactor	0.3
MaxWaitCycle	80
StartPointsToRun	bounds
MaxTime	7,200
FunctionTolerance	$1.00 \cdot 10^{-06}$
XTolerance	$1.00 \cdot 10^{-06}$
Display	iter

**Πίνακας 7.2:** Παράμετροι ελέγχου της τοπικής μεθόδου βελτιστοποίησης fmincon για τα χωρικά συνθετικά δεδομένα

Παράμετρος	Επιλογή
Algorithm	interior-point
UseParallel	true

### 7.2.2 Αποτελέσματα για το μοντέλο SLI και χωρικά πραγματικά δεδομένα

Για την ελαχιστοποίηση της τιμής NLL του συνόλου των χωρικών πραγματικών δεδομένων ως προς τις παραμέτρους του μοντέλου SLI αξιοποιώντας τον αλγόριθμο GS δεν προέκυψαν αποτελέσματα. Συγκεκριμένα, δεν υπήρξε σύγκλιση για χρονικό διάστημα πάνω από δύο ώρες και η διαδικασία βελτιστοποίησης δεν τερμάτισε εντός αυτού του χρονικού πλαισίου. Αυτό το γεγονός ενδεχομένως οφείλεται στις υψηλές τιμές του πίνακα που περιέχει τις χωρικές θέσεις των πραγματικών δεδομένων και του πίνακα που περιλαμβάνει τις τιμές των δεδομένων.

## Κεφάλαιο 8

# Συμπεράσματα

Στην παρούσα μελέτη, χρησιμοποιήθηκαν οι μέθοδοι της προσομοιωμένης ανόπτησης (ΠΑ) και καθολικής αναζήτησης (GS) σε δύο διακριτά προβλήματα βελτιστοποίησης. Το πρώτο πρόβλημα αφορούσε στην εύρεση καθολικού βέλτιστου σε συναρτήσεις ελέγχου υψηλών διαστάσεων (χώροι 10 και 30 διαστάσεων) με πολλαπλά τοπικά ακρότατα. Οι συναρτήσεις ελέγχου που ελαχιστοποιήθηκαν ήταν η Ackley και Cross-In-Tray. Η διερεύνηση των δύο συναρτήσεων έγινε διότι παρουσιάζουν διαφορετικά χαρακτηριστικά. Έτσι, διαφορετικά χαρακτηριστικά δημιουργούν διαφορετικές προκλήσεις στις μεθόδους καθολικής βελτιστοποίησης. Ειδικά, η συνάρτηση Ackley χαρακτηρίζεται από ομαλότερο τοπίο με πολλαπλά τοπικά ελάχιστα διάσπαρτα στο χώρο αναζήτησης. Αντίθετα, η συνάρτηση Cross-In-Tray χαρακτηρίζεται από πιο περίπλοκο και μη ομαλό τοπίο με πολλές κορυφές και βαθιές κοιλάδες σε ένα μικρότερο χώρο αναζήτησης ( $[-10, 10]$ ). Η καθολική βελτιστοποίηση των συγκεκριμένων συναρτήσεων σε παραμετρικούς χώρους υψηλών διαστάσεων αποτελεί απαιτητική διαδικασία. Ο σκοπός ήταν να εξεταστεί η αποτελεσματικότητα της κάθε μεθόδου ως προς την ακρίβεια εντοπισμού του καθολικού βέλτιστου. Τα αποτελέσματα συγκρίνονται ως προς τον υπολογιστικό χρόνο, την ακρίβεια λύσης και την ευαισθησία των μεθόδων στις διάφορες παραμετροποιήσεις τους.

Με βάση τα αποτελέσματα στο χώρο των 10 και 30 διαστάσεων για τη συνάρτηση ελέγχου Ackley και οι δύο μέθοδοι πέτυχαν εξίσου καλή ακρίβεια λύσης ως προς την τιμή συνάρτησης κόστους. Στο χώρο των 10 διαστάσεων για τη συνάρτηση Cross-In-Tray και οι δύο μέθοδοι πέτυχαν ισότιμη βελτίωση της συνάρτησης κόστους συγκριτικά με την τοπική βελτιστοποίηση. Ωστόσο, η ΠΑ απαιτούσε υψηλότερο υπολογιστικό χρόνο ως προς τις επιλεγμένες παραμετροποιήσεις των δύο μεθόδων. Στο χώρο των 30 διαστάσεων για τη συνάρτηση ελέγχου Cross-In-Tray οι μέθοδοι ΠΑ και GS πέτυχαν βελτίωση στη τιμή της συνάρτησης κόστους συγκριτικά με την τοπική βελτιστοποίηση. Παρ' όλα αυτά, η μέθοδος GS σημείωσε λύση υψηλότερης

---

ακρίβειας από την ΠΑ. Η μέθοδος ΠΑ έδειχνε συνεχώς κατευθυνόμενη τάση προς λύσεις χαμηλότερης ακρίβειας στις διάφορες παραμετροποιήσεις της.

Συμπερασματικά, η εφαρμογή της μεθόδου GS στις συναρτήσεις ελέγχου Ackley και Cross-In-Tray ήταν αποδοτικότερη καθώς επιτύγχανε λύσεις υψηλότερης ακρίβειας σε συνδυασμό με χαμηλότερο υπολογιστικό χρόνο. Αντιθέτως, η μέθοδος ΠΑ χρήζει περαιτέρω βελτίωσης. Αυτές οι βελτιώσεις είναι προτιμότερο να εστιάσουν στον σχεδιασμό καταλληλότερων παραμέτρων ελέγχου που να ταιριάζουν στα χαρακτηριστικά των συγκεκριμένων συναρτήσεων σε χώρους υψηλών διαστάσεων.

Στο δεύτερο πρόβλημα ασχοληθήκαμε με τη μεγιστοποίηση της πιθανοφάνειας ενός συνόλου χωρικών δεδομένων (δείγμα χωρικής στοχαστικής διαδικασίας) ως προς τις παραμέτρους του χωρικού στοχαστικού μοντέλου τοπικών αλληλεπιδράσεων (SLI). Πιο συγκεκριμένα, υπήρχε η επιλογή ανάμεσα σε δύο χωρικά σύνολα δεδομένων, στα πραγματικά ή συνθετικά. Σε αυτήν την περίπτωση χρησιμοποιήθηκε ένας παραμετρικός χώρος τεσσάρων διαστάσεων ( $m$ ,  $\lambda$ ,  $c1$ ,  $\mu$ ). Ο σκοπός της παρούσας μελέτης ήταν να παρατηρηθεί η συμπεριφορά της κάθε μεθόδου στα χωρικά δεδομένα και να εξεταστεί η αποτελεσματικότητά της ως προς την ακρίβεια μεγιστοποίησης της πιθανοφάνειας.

Εν συνεχεία, βάση των αποτελεσμάτων στα χωρικά συνθετικά δεδομένα και οι δύο μέθοδοι επιτύγχαναν παρόμοιες λύσεις για τη μεγιστοποίηση της πιθανοφάνειας. Ωστόσο, καμία από τις δύο μεθόδους δεν επιτύγχανε σημαντική βελτίωση συγκριτικά με τα αποτελέσματα της τοπικής βελτιστοποίησης. Η μέθοδος GS απαιτούσε υψηλότερο υπολογιστικό χρόνο ως προς τις επιλεγμένες παραμετροποιήσεις των δύο μεθόδων. Στα χωρικά πραγματικά δεδομένα υπήρξαν αποτελέσματα μόνο από την εφαρμογή της μεθόδου ΠΑ. Η μέθοδος ΠΑ δεν επιτύγχανε σημαντική βελτίωση για τη μεγιστοποίηση της πιθανοφάνειας σε σύγκριση με την τοπική βελτιστοποίηση. Με τη μέθοδο GS δεν υπήρξε σύγκλιση για χρονικό διάστημα μεγαλύτερο των δύο ωρών και η διαδικασία βελτιστοποίησης δεν τερμάτισε εντός αυτού του χρονικού πλαισίου.

Ακολούθως, σε καμία από τις δύο μεθόδους δεν διαπιστώθηκε ευαισθησία σε διαφοροποίηση των παραμετροποιήσεων. Αυτό είχε ως αποτέλεσμα, για κάθε μέθοδο οι διαφορετικές παραμετροποιήσεις να επιτύγχαναν παρόμοια βελτίωση της πιθανοφάνειας. Συνεπώς, η μέθοδος ΠΑ ήταν προτιμότερη καθώς απαιτούσε λιγότερο υπολογιστικό χρόνο συγκριτικά με την GS. Αντιθέτως, η μέθοδος GS χρήζει περαιτέρω βελτίωσης. Αυτές οι βελτιώσεις είναι προτιμότερο να εστιάσουν στη διερεύνηση τεχνικών για μείωση του υπολογιστικού χρόνου σε προβλήματα με μεγάλα χωρικά σύνολα δεδομένων.

Εν κατακλείδι, οι δύο μέθοδοι απαιτούσαν υψηλότερο υπολογιστικό χρόνο συ-



---

γκριτικά με την τοπική βελτιστοποίηση. Συνοπτικά, επιβεβαιώνεται η επάρκεια της τοπικής βελτιστοποίησης στο πρόβλημα μεγιστοποίησης της πιθανοφάνειας του μοντέλου SLI και για τα δύο χωρικά σύνολα δεδομένων.

Σύμφωνα με την παρούσα μελέτη, το γενικό συμπέρασμα που προκύπτει είναι ότι η απόδοση της κάθε μεθόδου διαφέρει ως προς την ακρίβεια της επιτυγχανόμενης λύσης και ως προς τον υπολογιστικό χρόνο ανάλογα το πρόβλημα βελτιστοποίησης που επιλύεται.

---

# Βιβλιογραφία

- [1] X. Jiankai, S. Bo, Dung beetle optimizer: a new meta-heuristic algorithm for global optimization, *Journal of Supercomputing* 79 (2023) 7305–7336. doi:[10.1007/s11227-022-04959-6](https://doi.org/10.1007/s11227-022-04959-6).
- [2] D. F. Siarry Patrick, Berthiau Gérard, H. Jacques, Enhanced simulated annealing for globally minimizing functions of many-continuous variables, *ACM Trans. Math. Softw.* 23 (2) (1997) 209–228. doi:[10.1145/264029.264043](https://doi.org/10.1145/264029.264043).
- [3] B. Dimitris, T. John, Simulated annealing, *Statistical Science* 8 (1) (1993) 10–15.
- [4] B. Franco, Simulated annealing overview, *Research Gate* (2001) 1–11.
- [5] J. P. Zsolt Ugray, Leon Lasdon, F. Glover, J. Kelly, R. Marti, Scatter search and local nlp solvers: A multistart framework for global optimization, *INFORMS Journal on Computing* 19 (3) (2007) 328–340.
- [6] A. W. J. Darrall Henderson, Sheldon H. Jacobson, The theory and practice of simulated annealing, in: F. Glover, G. Kochenberger (Eds.), *Handbook of Meta-heuristics*, Vol. 57 of *International Series in Operations Research and Management Science*, Springer, 2003, pp. 287–319.
- [7] N. C. Sarkar, Performance test of simulated annealing algorithm to search global optima of a surface function, <https://www.linkedin.com/pulse/performance-test-simulated-annealing-algorithm-search-sarkar>, last accessed 9 December 2020 (December 2020).
- [8] H. Miao, Y.-C. Tian, Robot path planning in dynamic environments using a simulated annealing based approach, in: *2008 10th International Conference on Control, Automation, Robotics and Vision*, Hanoi, Vietnam, 2008, pp. 1253–1258. doi:[10.1109/ICARCV.2008.4795701](https://doi.org/10.1109/ICARCV.2008.4795701).
- [9] Simulated annealing explained, <https://www.baeldung.com/cs/simulated-annealing>, last accessed 18 March 2024 (March 2024).

- [10] L. Ingber, Adaptive simulated annealing (asa), <https://www.ingber.com/ASA-README.html>, last accessed 09 April 2024 (2023).
- [11] L. Ingber, Adaptive simulated annealing (asa): Lessons learned, *Control and Cybernetics* 25 (1) (1996) 33–54.
- [12] R. Iglesias-Marzoa, M. López-Morales, M. J. A. Morales, The rvfit code: A detailed adaptive simulated annealing code for fitting binaries and exoplanets radial velocities, *Publications of the Astronomical Society of the Pacific* 127 (952) (2015) 1–50.
- [13] I. Lester, Asa options, *Journal of Research Gate* (05 2012).
- [14] M. Amirabdollahian, B. Datta, Application of simulated annealing and adaptive simulated annealing in search for efficient optimal solutions of a groundwater contamination related problem, in: H. Peyvandi (Ed.), *Computational Optimization in Engineering*, IntechOpen, 2017, Ch. 7, pp. 134—147.
- [15] R. E. Banchs, Simulated annealing, Research progress report, University of Texas at Austin (1997).
- [16] The MathWorks Inc., Simulated annealing options, <https://www.mathworks.com/help/gads/simulated-annealing-options.html> (2018).
- [17] The MathWorks Inc., How simulated annealing works, <https://www.mathworks.com/help/gads/how-simulated-annealing-works.html> (2018).
- [18] D. Delahaye, S. Chaimatanan, M. Mongeau, Simulated annealing: From basics to applications, in: G. M., P. J.Y. (Eds.), *Handbook of Metaheuristics*, Vol. 272 of International Series in Operations Research and Management Science, Springer, Cham, 2019, pp. 1–35. doi:10.1007/978-3-319-91086-4\_1.
- [19] O. L. Vincent Kelner, Florin Capitanescu, L. Wehenkel, A hybrid optimization technique coupling an evolutionary and a local search algorithm, *Journal of Computational and Applied Mathematics* 215 (2) (2008) 448–456.
- [20] The MathWorks Inc., fmincon, <https://www.mathworks.com/help/optim/ug/fmincon.html> (2018).
- [21] A. M. Albaghdadi, M. B. Baharom, S. A. bin Sulaiman, Parameter design optimization of the crank-rocker engine using the fmincon function in matlab, *IOP Conference Series: Materials Science and Engineering* 1088 (1) (2021) 012072.

- [22] The MathWorks Inc., Choosing the algorithm, <https://www.mathworks.com/help/optim/ug/choosing-the-algorithm.html> (2018).
- [23] The MathWorks Inc., Nonlinear constraints with gradients, <https://www.mathworks.com/help/optim/ug/nonlinear-constraints-with-gradients.html> (2018).
- [24] N. V. Findler, C. Lo, R. Lo, Pattern search for optimization, *Mathematics and Computers in Simulation* 29 (1) (1987) 41–50.
- [25] C. Audet, J. E. Dennis, Pattern search algorithms for mixed variable programming, *SIAM Journal on Optimization* 11 (3) (2001) 573–594.
- [26] The MathWorks Inc., Pattern search, <https://www.mathworks.com/help/gads/patternsearch.html> (2018).
- [27] A. Thevenot, Optimization eye pleasure: 78 benchmark test functions for single-objective optimization, <https://towardsdatascience.com/optimization-eye-pleasure-78-benchmark-test-functions-for-single-objective-optimization-92e7ed1d1f12> (2020).
- [28] S. Surjanovic, D. Bingham, Ackley function, <https://www.sfu.ca/~ssurjano/ackley.html> (2013).
- [29] S. Surjanovic, D. Bingham, Cross-in-tray function, <https://www.sfu.ca/~ssurjano/crossit.html> (2013).
- [30] D. T. Hristopulos, V. D. Agou, Stochastic local interaction model with sparse precision matrix for space–time interpolation, *Spatial Statistics* 40 (2020) 1–24.
- [31] V. D. A. Dionissios T. Hristopulos, Andrew Pavlides, P. Gkafa, Stochastic local interaction model: An alternative to kriging for massive datasets, *Mathematical Geosciences* 53 (2021) 1907–1949.
- [32] The MathWorks Inc., How globalsearch and multistart work, <https://www.mathworks.com/help/gads/how-globalsearch-and-multistart-work.html> (2018).
- [33] R. Martí, M. Laguna, F. Glover, Principles of scatter search, *European Journal of Operational Research* 169 (2) (2006) 359–372.
- [34] B. J. R. Egea Jose A., Rodríguez-Fernández María, M. Rafael, Scatter search for chemical and bio-process optimization, *Journal of Global Optimization* 37 (2006) 481–503.

- [35] F. Glover, M. Laguna, R. Marti, Scatter search, in: A. Ghosh, S. Tsutsui (Eds.), *Advances in Evolutionary Computing*, Natural Computing Series, Springer, 2003, pp. 519–537. doi:10.1007/978-3-642-18965-4\_20.
- [36] F. Glover, M. Laguna, R. Marti, Scatter search and path relinking: Advances and applications, in: F. Glover, G. A. Kochenberger (Eds.), *Handbook of Metaheuristics*, Vol. 57 of International Series in Operations Research and Management Science, Springer, 2003, pp. 1–36. doi:10.1007/0-306-48056-5\_1.
- [37] F. Glover, A template for scatter search and path relinking, in: J. K. Hao, E. Lut-ton, E. Ronald, M. Schoenauer, D. Snyers (Eds.), *Artificial Evolution. AE 1997*, Vol. 1363 of Lecture Notes in Computer Science, Springer, 1998, pp. 1–51.
- [38] M. de Athayde Costa e Silva, C. E. Klein, V. C. Mariani, L. dos Santos Coelho, Multiobjective scatter search approach with new combination scheme applied to solve environmental/economic dispatch problem, *Energy* 53 (2013) 14–21.
- [39] M. Kalra, S. Tyagi, V. Kumar, M. Kaur, W. K. Mashwani, H. Shah, K. Shah, A comprehensive review on scatter search: Techniques, applications, and challenges, *Mathematical Problems in Engineering* 2021 (2021) 1–21.
- [40] M. Laguna, R. Marti, Scatter search, in: E. Alba, R. Marti (Eds.), *Metaheuristic Procedures for Training Neural Networks*, Vol. 36 of Operations Research/-Computer Science Interfaces Series, Springer, Boston, MA, 2006, pp. 139–152. doi:10.1007/0-387-33416-5\_7.
- [41] J. J. Pantrigo, Ángel Sánchez, K. Gianikellis, A. S. Montemayor, Combining particle filter and population-based metaheuristics for visual articulated motion tracking, *Electronic Letters on Computer Vision and Image Analysis* 5 (3) (2005) 68–83.
- [42] The MathWorks Inc., Global search, <https://www.mathworks.com/help/gads/globalsearch.html> (2018).

## Παράρτημα Α΄

# Κώδικας Matlab

Συνάρτηση δημιουργίας νέων σημείων για την προσομοιωμένη απόσπηση

Συνάρτηση annealingfast

```
1 currentx = optimValues.x;
2 nvar = numel(currentx);
3 newx = currentx;
4 y = randn(nvar,1);
5 y = y./norm(y);
6 newx(:) = currentx(:) + optimValues.temperature.*y;
7
8 newx = sahonorbounds(newx,optimValues,problem);
```

Συνάρτηση annealingboltz

```
1 currentx = optimValues.x;
2 nvar = numel(currentx);
3 newx = currentx;
4 y = randn(nvar,1);
5 y = y./norm(y);
6 newx(:) = currentx(:) + sqrt(optimValues.temperature).*y;
7
8 newx = sahonorbounds(newx,optimValues,problem);
```

## Α.1 Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

Βελτιστοποίηση της συνάρτησης Ackley σε χώρο 10 διαστάσεων χρησιμοποιώντας την υβριδική προσέγγιση με την προσομοιωμένη ανόπτηση και την τοπική μέθοδο fmincon

```

1 clear all;
2 close all;
3
4 %Define the objective function to be minimized
5 ObjectiveFunction = @ackley;
6
7 % Creating a random starting point within the bounds [-32.768, 32.768]
8 x0 = -32.768 + (32.768 - (-32.768)) * rand(1, 10);
9 % Lower and upper bounds for the variables
10 lb = ones(1, 10)*-32.768;
11 ub = ones(1, 10)*32.768;
12
13 % Options for the hybrid optimization algorithm
14 hybridopts = optimoptions(@fmincon, 'Algorithm', 'interior-point', ...
15     'Display', 'iter', 'StepTolerance', 1e-15, ...
16     'SpecifyObjectiveGradient', true, 'OutputFcn', @hybrid_output_function);
17
18 % Options for the simulated annealing algorithm
19 options = optimoptions(@simulannealbnd, 'Display', 'iter', ...
20     'MaxFunctionEvaluations', 75000, 'MaxIterations', 70000, ...
21     'ObjectiveLimit', 1e-02, 'FunctionTolerance', 1e-04, ...
22     'InitialTemperature', 100, 'DisplayInterval', 1000, ...
23     'MaxStallIterations', 15000, 'TemperatureFcn', @temperaturefast, ...
24     'ReannealInterval', 650, 'AnnealingFcn', @annealingboltz, ...
25     'OutputFcns', @plot_progress, 'HybridFcn', {@fmincon, hybridopts});
26
27 % Run the optimization using simulated annealing with a hybrid function
28 [x, fval, exitflag, output] = ...
29     simulannealbnd(ObjectiveFunction, x0, lb, ub, options);
30
31 % Display the optimal solutions found by the simulated annealing algorithm
32 % and by the hybrid optimization algorithm
33 disp(['Best SA Solution: x = ', num2str(bestx_SA)]);
34 disp(['Best SA f(x) = ', num2str(bestfval_SA)]);
35 disp(['Best Hybrid Solution: x = ', num2str(bestx_hybrid)]);
36 disp(['Best Hybrid f(x) = ', num2str(bestfval_hybrid)]);
37
38 % Ackley function definition with gradient calculation
39 function [y, grady] = ackley(xx)

```



### A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
40 % Constants
41 a = 20;
42 b = 0.2;
43 c = 2*pi;
44 d = length(xx);
45
46 % Ackley Function Calculation
47 sum1 = sum(xx.^2);
48 sum2 = sum(cos(c*xx));
49
50 term1 = -a * exp(-b*sqrt(sum1/d));
51 term2 = -exp(sum2/d);
52
53 y = term1 + term2 + a + exp(1);
54
55 % Gradient calculation
56 if nargout > 1
57     grady = zeros(1, d);
58     for i = 1:d
59         grady(i) = a*b*xx(i)/(d*sqrt(sum1))*exp(-b*sqrt(sum1/d)) + ...
60                 c*sin(c*xx(i))/d*exp(sum2/d);
61     end
62 end
63 end
64
65 % Output function containing the graphs during
66 % simulated annealing algorithm optimization
67 function [stop,options,optchanged] = plot_progress(options,optimvalues,flag)
68 % Persistent variables to store data across iterations
69 persistent fval_history x_history temperature_history reanneal_count ...
70             last_temperature reanneal_iters reanneal_fvals reanneal_temps ...
71             relevant_fval bestx_SA bestfval_SA;
72
73 % Switch statement to handle different states of the optimization
74 switch flag
75     % Init case: define initial values for persistent variables
76     case 'init'
77         % Initialize an empty array
78         % to stores the values of the cost function
79         fval_history = [];
80
81         % Initialize an empty array
82         % to stores the x values
83         x_history = [];
84
85         % Initialize the temperature value
86         temperature_history = 100;
87
```

#### A'.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

---

```
88         % counter for re-annealing events
89         reanneal_count = 0;
90         last_temperature = [];
91
92         %Initialize an empty array
93         % to store iterations at which re-annealing events occurred
94         reanneal_iters = [];
95
96         % Initialize an empty array
97         % to store cost function values at re-annealing events
98         reanneal_fvals = [];
99
100        % Initialize an empty array
101        % to store temperatures values at re-annealing events
102        reanneal_temps = [];
103
104        % Initialize an empty array
105        % to calculating the logarithmic value of the cost function
106        relevant_fval=[];
107        bestx_SA = [];
108        bestfval_SA = Inf;
109
110        % Iter case: update historical data with current values
111        case 'iter'
112
113            fval_history = [fval_history; optimvalues.fval];
114            x_history = [x_history; optimvalues.x];
115            temperature_history = [temperature_history; optimvalues.
116temperature(1)];
117
118            % Check if the re-annealing event occurs
119            if ~isempty(last_temperature) && optimvalues.temperature(1) > ...
120                last_temperature
121                reanneal_count = reanneal_count + 1;
122                reanneal_iters = [reanneal_iters; optimvalues.iteration];
123                reanneal_fvals = [reanneal_fvals; optimvalues.fval];
124                reanneal_temps = [reanneal_temps; optimvalues.temperature(1)
125];
126
127            end
128            % Update last_temperature
129            last_temperature = optimvalues.temperature(1);
130
131            % Update best cost function value and x value
132            % in case they are improved
133            if isempty(bestfval_SA) || optimvalues.fval < bestfval_SA
134                bestx_SA = optimvalues.x;
135                bestfval_SA = optimvalues.fval;
136            end
```

```

134
135     % Done case: final processing and plotting
136     case 'done'
137
138         % Plot cost function value history with reanneal points
139         f=figure('Position', [100, 100, 800, 600]);
140         plot(fval_history);
141         hold on;
142         scatter(reanneal_iters, reanneal_fvals, 'o', 'filled', ...
143             'MarkerFaceColor', 'r');
144         title('Ackley function in 10 dimensions');
145         xlabel('Iterations');
146         ylabel('costFunction');
147         set(gca, 'FontSize', 12);
148         grid on;
149         xlim([1, length(fval_history)]);
150         hold off;
151
152         export_fig(f, 'ackley_function_progress.png', '-png', '-m2');
153
154         % Plot x history for dimensions 1 to 5
155         f=figure;
156
157         for i = 1:5
158             subplot(5,1,i);
159             plot(x_history(:, i));
160             xlim([1 size(x_history, 1)]);
161             ylabel(['x(', num2str(i), ')']);
162             set(gca, 'FontSize', 12);
163             grid on;
164
165             if i == 5
166                 xlabel('Iterations');
167             end
168
169             if i < 5
170                 set(gca, 'XTickLabel', []);
171             end
172         end
173         export_fig(f, 'x_history_1_to_5.png', '-png', '-m2');
174
175         % Plot x history for dimensions 6 to 10
176         f=figure;
177
178         for i = 6:10
179             subplot(6,1,i-5);
180             plot(x_history(:, i));
181             xlim([1 size(x_history, 1)]);

```

```

182         ylabel(['x(', num2str(i), ')']);
183         set(gca, 'FontSize', 12);
184         grid on;
185
186         if i == 10
187             xlabel('Iterations');
188         end
189         if i < 10
190             set(gca, 'XTickLabel', []);
191         end
192     end
193
194     export_fig(f, 'x_history_6_to_10.png', '-png', '-m2');
195
196     % Plot temperature history with reanneal points
197     figure;
198     plot(temperature_history);
199     xlabel('Iterations');
200     ylabel('Temperature');
201     set(gca, 'FontSize', 12);
202     grid on;
203     xlim([1, length(temperature_history)]);
204     ylim([0,100]);
205
206     % Export the figure using export_fig
207     export_fig('temperature_history.png', ...
208         '-png', '-transparent', '-m2');
209
210
211     % Check the length of fval_history
212     max_iter = length(fval_history);
213     start_iter = 1;
214     end_iter = max_iter;
215
216     % Extract the entire fval_history
217     relevant_fval = fval_history(start_iter:end_iter);
218
219     % logarithmic transformation
220     log_fval = log(relevant_fval);
221
222     % Plot the logarithmic cost function value history
223     f_log_fval=figure('Position', [100, 100, 800, 600]);
224     plot(start_iter:end_iter, log_fval);
225     title('Logarithm Ackley Function in 10 dimensions');
226     xlabel('Iterations');
227     ylabel('Log(costFunction)');
228     set(gca, 'FontSize', 12);
229     grid on;

```

## A'.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
230     if start_iter < end_iter
231         xlim([start_iter, end_iter]);
232     end
233
234     export_fig(f_log_fval, 'log_ackley_function_progress.png', ...
235         '-png', '-m2');
236
237     % Display the number of re-annealing events
238     disp(['Number of ReannealInterval events: ', ...
239         num2str(reanneal_count)]);
240
241     % Assign variables to base workspace
242     assignin('base','reanneal_count',reanneal_count);
243     assignin('base','fval_history',fval_history);
244     assignin('base','x_history',x_history);
245     assignin('base','temperature_history',temperature_history);
246     assignin('base','bestx_SA',bestx_SA);
247     assignin('base','bestfval_SA',bestfval_SA);
248 end
249
250 stop = false;
251 optchanged = false;
252 end
253
254 % Output function containing the graphs during hybrid optimization algorithm
255 function [stop] = hybrid_output_function(x,optimvalues,state)
256     % Persistent variables to store data across iterations
257     persistent fval_history x_hist bestx_hybrid bestfval_hybrid;
258
259
260     switch state
261         % Init case: initialization of persistent variables
262         % based on final values from simulated annealing
263         case 'init'
264
265             fval_history = optimvalues.fval;
266             x_hist = x;
267
268             bestx_hybrid = x;
269             bestfval_hybrid = optimvalues.fval;
270
271         % Iter case: update historical data with current values
272         case 'iter'
273
274             fval_history = [fval_history; optimvalues.fval];
275             x_hist = [x_hist; x];
276
277             % Update best cost function value and
```

```

278         % x value in case they are improved
279         if optimvalues.fval < bestfval_hybrid
280             bestx_hybrid = x;
281             bestfval_hybrid = optimvalues.fval;
282         end
283
284     % Done case: final processing and plotting
285     case 'done'
286
287         % Plot the history of cost function values
288         % using the hybrid function
289         f_hybrid=figure;
290         plot(fval_history, 'o');
291         hold on;
292         scatter(1:length(fval_history), fval_history, 'filled');
293         hold off;
294         xlabel('Iterations(hybrid)');
295         xlim([1, length(fval_history)]);
296         ylabel('CostFunction(hybrid)');
297         set(gca, 'FontSize', 12);
298
299         export_fig(f_hybrid, 'hybrid_fval_history.png', '-png', '-m2');
300
301     % Plot x history for dimensions 1 to 5 using the hybrid function
302     f=figure;
303     for i = 1:5
304         subplot(5,1,i);
305         plot(x_hist(:, i));
306         title(['x(', num2str(i), ')-Hybrid']);
307         xlim([1 size(x_hist, 1)]);
308         ylabel(['x(', num2str(i), ')]');
309         set(gca, 'FontSize', 12);
310
311         if i == 5
312             xlabel('Iterations');
313         end
314
315         if i < 5
316             set(gca, 'XTickLabel', []);
317         end
318     end
319     export_fig(f, 'x_history_hybrid_1_to_5.png', '-png', '-m2');
320
321     % Plot x history for dimensions 6 to 10 using the hybrid function
322     f=figure;
323     for i = 6:10
324         subplot(5,1,i-5);
325         plot(x_hist(:, i));

```

```

326         title(['x(', num2str(i), ')-Hybrid']);
327         xlim([1 size(x_hist, 1)]);
328         ylabel(['x(', num2str(i), ')]');
329         set(gca, 'FontSize', 12);
330
331         if i == 10
332             xlabel('Iterations');
333         end
334
335         if i < 10
336             set(gca, 'XTickLabel', []);
337         end
338     end
339
340     export_fig(f, 'x_history_hybrid_6_to_10.png', '-png', '-m2');
341
342     % Assign variables to base workspace
343     assignin('base','bestx_hybrid',bestx_hybrid);
344     assignin('base','bestfval_hybrid',bestfval_hybrid);
345
346 end
347
348 stop = false;
349 end
350
351

```

**Βελτιστοποίηση της συνάρτησης Ackley σε χώρο 30 διαστάσεων χρησιμοποιώντας την υβριδική προσέγγιση με την προσομοιωμένη ανόπτηση και την τοπική μέθοδο fmincon**

```

1 clear all;
2 close all;
3
4 %Define the objective function to be minimized
5 ObjectiveFunction = @ackley;
6
7 %Creating a random starting point within the bounds [-32.768, 32.768]
8 x0 = -32.768 + (32.768 - (-32.768)) * rand(1, 30);
9 % Lower and upper bounds for the variables
10 lb = ones(1, 30)*-32.768;
11 ub = ones(1, 30)*32.768;
12
13 % Options for the hybrid optimization algorithm
14 hybridopts = optimoptions(@fmincon,'Algorithm','interior-point', ...
15     'Display','iter','StepTolerance', 1e-15, ...

```

### A'.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
16     'SpecifyObjectiveGradient', true, 'OutputFcn', @hybrid_output_function);
17
18 % Options for the simulated annealing algorithm
19 options = optimoptions(@simulannealbnd, 'Display', 'iter', ...
20     'MaxFunctionEvaluations', 220000, 'MaxIterations', 200000, ...
21     'ObjectiveLimit', 1e-01, 'FunctionTolerance', 1e-04, ...
22     'InitialTemperature', 1000, 'DisplayInterval', 1000, ...
23     'MaxStallIterations', 40000, 'TemperatureFcn', @temperaturefast, ...
24     'ReannealInterval', 11500, 'AnnealingFcn', @annealingboltz, ...
25     'OutputFcns', @plot_progress, 'HybridFcn', {@fmincon, hybridopts});
26
27 % Run the optimization using simulated annealing with a hybrid function
28 [x, fval, exitflag, output] = ...
29     simulannealbnd(ObjectiveFunction, x0, lb, ub, options);
30
31 % Display the optimal solutions found by the simulated annealing algorithm
32 % and by the hybrid optimization algorithm
33 disp(['Best SA Solution: x = ', num2str(bestx_SA)]);
34 disp(['Best SA f(x) = ', num2str(bestfval_SA)]);
35 disp(['Best Hybrid Solution: x = ', num2str(bestx_hybrid)]);
36 disp(['Best Hybrid f(x) = ', num2str(bestfval_hybrid)]);
37
38 % Ackley function definition with gradient calculation
39 function [y, grady] = ackley(xx)
40     % Constants
41     a = 20;
42     b = 0.2;
43     c = 2*pi;
44     d = length(xx);
45
46     % Ackley Function Calculation
47     sum1 = sum(xx.^2);
48     sum2 = sum(cos(c*xx));
49
50     term1 = -a * exp(-b*sqrt(sum1/d));
51     term2 = -exp(sum2/d);
52
53     y = term1 + term2 + a + exp(1);
54
55     % Gradient calculation
56     if nargout > 1
57         grady = zeros(1, d);
58         for i = 1:d
59             grady(i) = a*b*xx(i)/(d*sqrt(sum1))*exp(-b*sqrt(sum1/d)) + ...
60                 c*sin(c*xx(i))/d*exp(sum2/d);
61         end
62     end
63
```



```

64 end
65
66 % Output function containing the graphs during
67 % simulated annealing algorithm optimization
68 function [stop,options,optchanged] = plot_progress(options,optimvalues,flag)
69     % Persistent variables to store data across iterations
70     persistent fval_history temperature_history reanneal_count ...
71         last_temperature reanneal_iters reanneal_fvals reanneal_temps ...
72         relevant_fval bestx_SA bestfval_SA;
73
74     switch flag
75         % Init case: define initial values for persistent variables
76         case 'init'
77             % Initialize an empty array
78             % to stores the values of the cost function
79             fval_history = [];
80
81             % Initialize the temperature value
82             temperature_history = 1000;
83
84             % counter for re-annealing events
85             reanneal_count = 0;
86             last_temperature = [];
87
88             %Initialize an empty array
89             % to store iterations at which re-annealing events occurred
90             reanneal_iters = [];
91
92             % Initialize an empty array
93             % to store cost function values at re-annealing events
94             reanneal_fvals = [];
95
96             % Initialize an empty array
97             % to store temperatures values at re-annealing events
98             reanneal_temps = [];
99
100            % Initialize an empty array
101            % to calculating the logarithmic value of the cost function
102            relevant_fval=[];
103            bestx_SA = [];
104            bestfval_SA = Inf;
105
106            % Iter case: update historical data with current values
107            case 'iter'
108
109                fval_history = [fval_history; optimvalues.fval];
110                temperature_history = ...
111                    [temperature_history; optimvalues.temperature(1)];

```

```

112
113     % Check if the re-annealing event occurs
114     if ~isempty(last_temperature) && optimvalues.temperature(1) > ...
115         last_temperature
116         reanneal_count = reanneal_count + 1;
117         reanneal_iters = [reanneal_iters; optimvalues.iteration];
118         reanneal_fvals = [reanneal_fvals; optimvalues.fval];
119         reanneal_temps = [reanneal_temps; optimvalues.temperature(1)];
120     end
121     % Update last_temperature
122     last_temperature = optimvalues.temperature(1);
123
124     if isempty(bestfval_SA) || optimvalues.fval < bestfval_SA
125         bestx_SA = optimvalues.x;
126         bestfval_SA = optimvalues.fval;
127     end
128     % Done case: final processing and plotting
129     case 'done'
130
131         % Plot cost function value history with reanneal points
132         f=figure('Position', [100, 100, 800, 600]);
133         plot(fval_history);
134         hold on;
135         scatter(reanneal_iters, reanneal_fvals, 'o', 'filled', ...
136             'MarkerFaceColor', 'r');
137         title('Ackley function in 30 dimensions');
138         xlabel('Iterations');
139         ylabel('costFunction');
140         set(gca, 'FontSize', 12);
141         grid on;
142         xlim([1, length(fval_history)]);
143         hold off;
144
145         export_fig(f, 'ackley_function_progress.png', '-png', '-m2');
146
147         % Plot temperature history with reanneal points
148         figure;
149         plot(temperature_history);
150         xlabel('Iterations');
151         ylabel('Temperature');
152         set(gca, 'FontSize', 12);
153         grid on;
154         xlim([1, length(temperature_history)]);
155         ylim([0,1000]);
156
157         % Export the figure using export_fig
158         export_fig('temperature_history.png', '-png', ...
159             '-transparent', '-m2');

```

```

160
161
162     % Check the length of fval_history
163     max_iter = length(fval_history);
164     start_iter = 1;
165     end_iter = max_iter;
166
167     % Extract the entire fval_history
168     relevant_fval = fval_history(start_iter:end_iter);
169
170     % Logarithmic transformation
171     log_fval = log(relevant_fval);
172
173     % Plot the logarithmic cost function value history
174     f_log_fval=figure('Position', [100, 100, 800, 600]);
175     plot(start_iter:end_iter, log_fval);
176     title('Logarithm Ackley Function in 30 dimensions');
177     xlabel('Iterations');
178     ylabel('Log(costFunction)');
179     set(gca, 'FontSize', 12);
180     grid on;
181     if start_iter < end_iter
182         xlim([start_iter, end_iter]);
183     end
184
185     export_fig(f_log_fval, 'log_ackley_function_progress.png', ...
186         '-png', '-m2');
187
188     % Display the number of re-annealing events
189     disp(['Number of ReannealInterval events: ', ...
190         num2str(reanneal_count)]);
191
192     % Assign variables to base workspace
193     assignin('base','reanneal_count',reanneal_count);
194     assignin('base','fval_history',fval_history);
195     assignin('base','temperature_history',temperature_history);
196     assignin('base','bestx_SA',bestx_SA);
197     assignin('base','bestfval_SA',bestfval_SA);
198 end
199 stop = false;
200 optchanged = false;
201 end
202
203 % Output function containing the graphs
204 % during hybrid optimization algorithm
205 function [stop] = hybrid_output_function(x,optimvalues,state)
206     % Persistent variables to store data across iterations
207     persistent fval_history bestx_hybrid bestfval_hybrid;

```

```

208
209     switch state
210         % Init case: initialization of persistent variables based
211         % on final values from simulated annealing
212         case 'init'
213
214             fval_history = optimvalues.fval;
215
216             bestx_hybrid = x;
217             bestfval_hybrid = optimvalues.fval;
218
219         % Iter case: update historical data with current values
220         case 'iter'
221
222             fval_history = [fval_history; optimvalues.fval];
223
224             if optimvalues.fval < bestfval_hybrid
225                 bestx_hybrid = x;
226                 bestfval_hybrid = optimvalues.fval;
227             end
228         % Done case: final processing and plotting
229         case 'done'
230
231             % Plot the history of cost function values
232             % using the hybrid function
233             f_hybrid = figure;
234             plot(fval_history, 'o');
235             hold on;
236             scatter(1:length(fval_history), fval_history, 'filled');
237             hold off;
238             xlabel('Iterations(hybrid)');
239             xlim([1, length(fval_history)]);
240             ylabel('costFunction(hybrid)');
241             set(gca, 'FontSize', 12);
242
243             export_fig(f_hybrid, 'hybrid_fval_history.png', '-png', '-m2');
244
245             % Assign variables to base workspace
246             assignin('base','bestx_hybrid',bestx_hybrid);
247             assignin('base','bestfval_hybrid',bestfval_hybrid);
248         end
249     stop = false;
250 end
251
252
253

```

**Βελτιστοποίηση της συνάρτησης Cross-In-Tray σε χώρο 10 διαστάσεων χρησιμοποιώντας την υβριδική προσέγγιση με την προσομοιωμένη ανόπτηση και την τοπική μέθοδο patternsearch**

```

1 clear all;
2 close all;
3
4 %Define the objective function to be minimized
5 ObjectiveFunction = @crossit;
6
7 %Creating a random starting point within the bounds [-10, 10]
8 x0 = -10 + (10 - (-10)) * rand(1, 10);
9 % Lower and upper bounds for the variables
10 lb = ones(1, 10)*-10;
11 ub = ones(1, 10)*10;
12
13 % Options for the hybrid optimization algorithm
14 patternsearch_opts = optimoptions(@patternsearch, 'Display', 'iter', ...
15     'PlotFcn',@hybrid_output_function);
16
17 % Options for the simulated annealing algorithm
18 options = optimoptions(@simulannealbnd,'Display','iter', ...
19     'MaxFunctionEvaluations',85000,'MaxIterations',80000, ...
20     'ObjectiveLimit',-1.80,'FunctionTolerance',1e-04, ...
21     'InitialTemperature',100,'DisplayInterval',1000, ...
22     'MaxStallIterations',25000,'TemperatureFcn',@temperaturefast, ...
23     'ReannealInterval',1300,'AnnealingFcn',@annealingboltz, ...
24     'OutputFcns', @plot_progress, ...
25     'HybridFcn',{@patternsearch,patternsearch_opts});
26
27 % Run the optimization using simulated annealing with a hybrid function
28 [x, fval, exitflag, output] = ...
29     simulannealbnd(ObjectiveFunction, x0, lb, ub, options);
30
31 % Display the optimal solutions found by the simulated annealing algorithm
32 % and by the hybrid optimization algorithm
33 disp(['Best SA Solution: x = ', num2str(bestx_SA)]);
34 disp(['Best SA f(x) = ', num2str(bestfval_SA)]);
35 disp(['Best Hybrid Solution: x = ', num2str(bestx_hybrid)]);
36 disp(['Best Hybrid f(x) = ', num2str(bestfval_hybrid)]);
37
38 % Cross-In-Tray function definition
39 function [y] = crossit(xx)
40 % Cross-In-Tray Function Calculation
41 n = 10;
42 fact1 = 1;
43 sum_squares = 0;

```

### A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
44 for i = 1:n
45     fact1 = fact1 * sin(xx(i));
46     sum_squares = sum_squares + xx(i)^2;
47 end
48 fact2 = exp(abs(100 - sqrt(sum_squares)/pi));
49
50 y = -0.0001 * (abs(fact1*fact2)+1)^0.1;
51
52 end
53
54 % Output function containing the graphs
55 % during simulated annealing algorithm optimization
56 function [stop,options,optchanged] = plot_progress(options,optimvalues,flag)
57     % Persistent variables to store data across iterations
58     persistent fval_history x_history temperature_history ...
59         reanneal_count last_temperature reanneal_iters reanneal_fvals ...
60         reanneal_temps bestx_SA bestfval_SA;
61
62     % Switch statement to handle different states of the optimization
63     switch flag
64         % Init case: define initial values for persistent variables
65         case 'init'
66
67             % Initialize an empty array
68             % to stores the values of the cost function
69             fval_history = [];
70
71             % Initialize an empty array
72             % to stores the x values
73             x_history = [];
74
75             % Initialize an empty array
76             % to stores the temperature values
77             temperature_history = [];
78
79             % counter for re-annealing events
80             reanneal_count = 0;
81             last_temperature = [];
82
83             %Initialize an empty array
84             % to store iterations at which re-annealing events occurred
85             reanneal_iters = [];
86
87             % Initialize an empty array
88             % to store cost function values at re-annealing events
89             reanneal_fvals = [];
90
91             % Initialize an empty array
```

## A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
92         % to store temperatures values at re-annealing events
93         reanneal_temps = [];
94         bestx_SA = [];
95         bestfval_SA = Inf;
96
97         % Iter case: update historical data with current values
98         case 'iter'
99
100             fval_history = [fval_history; optimvalues.fval];
101             x_history = [x_history; optimvalues.x];
102             temperature_history = ...
103                 [temperature_history; optimvalues.temperature];
104
105             % Check if the re-annealing event occurs
106             if ~isempty(last_temperature) && optimvalues.temperature(1) > ...
107                 last_temperature
108                 reanneal_count = reanneal_count + 1;
109                 reanneal_iters = [reanneal_iters; optimvalues.iteration];
110                 reanneal_fvals = [reanneal_fvals; optimvalues.fval];
111                 reanneal_temps = [reanneal_temps; optimvalues.temperature(1)];
112             end
113             % Update last_temperature
114             last_temperature = optimvalues.temperature(1);
115
116             % Update best cost function value and
117             % x value in case they are improved
118             if isempty(bestfval_SA) || optimvalues.fval < bestfval_SA
119                 bestx_SA = optimvalues.x;
120                 bestfval_SA = optimvalues.fval;
121             end
122
123         % Done case: final processing and plotting
124         case 'done'
125
126             % Plot cost function value history with reanneal points
127             f=figure('Position', [100, 100, 800, 600]);
128             plot(fval_history);
129             hold on;
130             scatter(reanneal_iters, reanneal_fvals, 'o', 'filled', ...
131                 'MarkerFaceColor', 'r');
132             title('Cross-In-Tray function in 10 dimensions');
133             xlabel('Iterations');
134             ylabel('costFunction');
135             set(gca, 'FontSize', 12);
136             grid on;
137             xlim([1, length(fval_history)]);
138             hold off;
```

```

140     export_fig(f, 'Cross-In-Tray_function_progress.png', ...
141               '-png', '-m2');
142
143     % Plot x history for dimensions 1 to 5
144     f=figure;
145     for i = 1:5
146         subplot(5,1,i);
147         plot(x_history(:, i));
148         xlim([1 size(x_history, 1)]);
149         ylabel(['x(', num2str(i), ')']);
150         set(gca, 'FontSize', 12);
151         grid on;
152
153         if i == 5
154             xlabel('Iterations');
155         end
156         if i < 5
157             set(gca, 'XTickLabel', []);
158         end
159     end
160     export_fig(f, 'x_history_1_to_5.png', '-png', '-m2');
161
162     % Plot x history for dimensions 6 to 10
163     f=figure;
164     for i = 6:10
165         subplot(6,1,i-5);
166         plot(x_history(:, i));
167         xlim([1 size(x_history, 1)]);
168         ylabel(['x(', num2str(i), ')']);
169         set(gca, 'FontSize', 12);
170         grid on;
171
172         if i == 10
173             xlabel('Iterations');
174         end
175         if i < 10
176             set(gca, 'XTickLabel', []);
177         end
178     end
179
180     export_fig(f, 'x_history_6_to_10.png', '-png', '-m2');
181
182     % Plot temperature history with reanneal points
183     figure;
184     plot(temperature_history);
185     xlabel('Iterations');
186     ylabel('Temperature');
187     set(gca, 'FontSize', 12);

```



```

188     grid on;
189     xlim([1, length(temperature_history)]);
190
191     % Export the figure using export_fig
192     export_fig('temperature_history.png', '-png', ...
193         '-transparent', '-m2');
194
195     % Display the number of re-annealing events
196     disp(['Number of ReannealInterval events: ', ...
197         num2str(reanneal_count)]);
198
199     % Assign variables to base workspace
200     assignin('base','reanneal_count',reanneal_count);
201     assignin('base','fval_history',fval_history);
202     assignin('base','x_history',x_history);
203     assignin('base','temperature_history',temperature_history);
204     assignin('base','bestx_SA',bestx_SA);
205     assignin('base','bestfval_SA',bestfval_SA);
206 end
207 stop = false;
208 optchanged = false;
209 end
210
211 % Output function containing the graphs during
212 % hybrid optimization algorithm
213 function stop = hybrid_output_function(optimvalues,flag)
214     % Persistent variables to store data across iterations
215     persistent fval_history x_hist bestx_hybrid bestfval_hybrid;
216
217     % Switch statement to handle different states of the optimization
218     switch flag
219         % Init case: initialization of persistent variables based
220         % on final values from simulated annealing
221         case 'init'
222
223             fval_history = optimvalues.fval;
224             x_hist = optimvalues.x;
225
226             bestx_hybrid = optimvalues.x;
227             bestfval_hybrid = optimvalues.fval;
228
229         % Iter case: update historical data with current values
230         case 'iter'
231
232             fval_history = [fval_history; optimvalues.fval];
233             x_hist = [x_hist; optimvalues.x];
234
235             % Update best cost function value and

```

```

236         % x value in case they are improved
237         if optimvalues.fval < bestfval_hybrid
238             bestx_hybrid = optimvalues.x;
239             bestfval_hybrid = optimvalues.fval;
240         end
241
242     % Done case: final processing and plotting
243     case 'done'
244
245         % Plot the history of cost function values
246         % using the hybrid function
247         f_hybrid = figure;
248         plot(fval_history, 'o');
249         hold on;
250         scatter(1:length(fval_history), fval_history, 'filled');
251         hold off;
252         xlabel('Iterations(hybrid)');
253         xlim([1, length(fval_history)]);
254         ylabel('fval(hybrid)');
255         set(gca, 'FontSize', 12);
256
257         export_fig(f_hybrid, 'hybrid_fval_history.png', '-png', '-m2');
258
259         f=figure;
260         % Plot x history for dimensions 1 to 5 using the hybrid function
261         for i = 1:5
262             subplot(5,1,i);
263             plot(x_hist(:, i));
264             title(['x(', num2str(i), ')-Hybrid']);
265             xlim([1 size(x_hist, 1)]);
266             ylabel(['x(', num2str(i), ')']);
267             set(gca, 'FontSize', 12);
268
269             if i == 5
270                 xlabel('Iterations');
271             end
272
273             if i < 5
274                 set(gca, 'XTickLabel', []);
275             end
276         end
277         export_fig(f, 'x_history_hybrid_1_to_5.png', '-png', '-m2');
278
279         % Plot x history for dimensions 6 to 10 using the hybrid function
280         f=figure;
281         for i = 6:10
282             subplot(5,1,i-5);
283             plot(x_hist(:, i));

```

```

284         title(['x(', num2str(i), ')-Hybrid']);
285         xlim([1 size(x_hist, 1)]);
286         ylabel(['x(', num2str(i), ')']);
287         set(gca, 'FontSize', 12);
288
289         if i == 10
290             xlabel('Iterations');
291         end
292
293         if i < 10
294             set(gca, 'XTickLabel', []);
295         end
296     end
297     export_fig(f, 'x_history_hybrid_6_to_10.png', '-png', '-m2');
298
299     % Assign variables to base workspace
300     assignin('base','bestx_hybrid',bestx_hybrid);
301     assignin('base','bestfval_hybrid',bestfval_hybrid);
302
303 end
304 stop = false;
305 end
306
307

```

Βελτιστοποίηση της συνάρτησης Cross-In-Tray σε χώρο 30 διαστάσεων χρησιμοποιώντας την υβριδική προσέγγιση με την προσομοιωμένη ανόπτηση και την τοπική μέθοδο patternsearch

```

1
2 clear all;
3 close all;
4
5 %Define the objective function to be minimized
6 ObjectiveFunction = @crossit;
7
8 %Creating a random starting point within the bounds [-10, 10]
9 x0 = -10 + (10 - (-10)) * rand(1, 30);
10 % Lower and upper bounds for the variables
11 lb = ones(1, 30)*-10;
12 ub = ones(1, 30)*10;
13
14 % Options for the hybrid optimization algorithm
15 patternsearch_opts = optimoptions(@patternsearch,'Display', 'iter', ...
16     'PlotFcn',@hybrid_output_function);
17

```

### A'.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
18 % Options for the simulated annealing algorithm
19 options = optimoptions(@simulannealbnd,'Display','iter', ...
20     'MaxFunctionEvaluations',160000,'MaxIterations',150000, ...
21     'ObjectiveLimit',-1.3,'FunctionTolerance',1e-06, ...
22     'InitialTemperature',100,'DisplayInterval',1000, ...
23     'MaxStallIterations',70000,'TemperatureFcn',@temperaturefast, ...
24     'ReannealInterval',1650,'AnnealingFcn',@annealingfast, ...
25     'OutputFcns', @plot_progress, ...
26     'HybridFcn',{@patternsearch,patternsearch_opts});
27
28 % Run the optimization using simulated annealing with a hybrid function
29 [x, fval, exitflag, output] = ...
30     simulannealbnd(ObjectiveFunction, x0, lb, ub, options);
31
32 % Display the optimal solutions found by the simulated annealing algorithm
33 % and by the hybrid optimization algorithm
34 disp(['Best SA Solution: x = ', num2str(bestx_SA)]);
35 disp(['Best SA f(x) = ', num2str(bestfval_SA)]);
36 disp(['Best Hybrid Solution: x = ', num2str(bestx_hybrid)]);
37 disp(['Best Hybrid f(x) = ', num2str(bestfval_hybrid)]);
38
39 % Cross-In-Tray function definition
40 function [y] = crossit(xx)
41 % Cross-In-Tray Function Calculation
42 n = 30;
43 fact1 = 1;
44 sum_squares = 0;
45 for i = 1:n
46     fact1 = fact1 * sin(xx(i));
47     sum_squares = sum_squares + xx(i)^2;
48 end
49 fact2 = exp(abs(100 - sqrt(sum_squares)/pi));
50
51 y = -0.0001 * (abs(fact1*fact2)+1)^0.1;
52
53 end
54
55 % Output function containing the graphs during
56 % simulated annealing algorithm optimization
57 function [stop,options,optchanged] = plot_progress(options,optimvalues,flag)
58 % Persistent variables to store data across iterations
59 persistent fval_history x_history temperature_history ...
60     reanneal_count last_temperature reanneal_iters reanneal_fvals ...
61     reanneal_temps bestx_SA bestfval_SA;
62
63 % Switch statement to handle different states of the optimization
64 switch flag
65     % Init case: define initial values for persistent variables
```

```

66     case 'init'
67
68         % Initialize an empty array
69         % to stores the values of the cost function
70         fval_history = [];
71
72         % Initialize the temperature value
73         temperature_history = 100;
74
75         % counter for re-annealing events
76         reanneal_count = 0;
77         last_temperature = [];
78
79         %Initialize an empty array
80         % to store iterations at which re-annealing events occurred
81         reanneal_iters = [];
82
83         % Initialize an empty array
84         % to store cost function values at re-annealing events
85         reanneal_fvals = [];
86
87         % Initialize an empty array
88         % to store temperatures values at re-annealing events
89         reanneal_temps = [];
90         bestx_SA = [];
91         bestfval_SA = Inf;
92
93     % Iter case: update historical data with current values
94     case 'iter'
95
96         fval_history = [fval_history; optimvalues.fval];
97         x_history = [x_history; optimvalues.x];
98         temperature_history = ...
99             [temperature_history; optimvalues.temperature(1)];
100
101         % Check if the re-annealing event occurs
102         if ~isempty(last_temperature) && optimvalues.temperature(1) > ...
103             last_temperature
104             reanneal_count = reanneal_count + 1;
105             reanneal_iters = [reanneal_iters; optimvalues.iteration];
106             reanneal_fvals = [reanneal_fvals; optimvalues.fval];
107             reanneal_temps = [reanneal_temps; optimvalues.temperature(1)];
108         end
109         % Update last_temperature
110         last_temperature = optimvalues.temperature(1);
111
112         % Update best cost function value and
113         % x value in case they are improved

```

```

114         if isempty(bestfval_SA) || optimvalues.fval < bestfval_SA
115             bestx_SA = optimvalues.x;
116             bestfval_SA = optimvalues.fval;
117         end
118
119     % Done case: final processing and plotting
120     case 'done'
121
122         % Plot cost function value history with reanneal points
123         f=figure('Position', [100, 100, 800, 600]);
124         plot(fval_history);
125         hold on;
126         scatter(reanneal_iters, reanneal_fvals, 'o', 'filled', ...
127             'MarkerFaceColor', 'r');
128         title('Cross-In-Tray function in 30 dimensions');
129         xlabel('Iterations');
130         ylabel('costFunction');
131         set(gca, 'FontSize', 14);
132         grid on;
133         xlim([1, length(fval_history)]);
134         hold off;
135
136         export_fig(f, 'Cross-In-Tray_function_progress.png', ...
137             '-png', '-m2');
138
139         % Plot temperature history with reanneal points
140         figure;
141         plot(1:length(temperature_history), temperature_history);
142         hold on;
143         xlabel('Iterations');
144         ylabel('Temperature');
145         set(gca, 'FontSize', 12);
146         grid on;
147         xlim([1, length(temperature_history)]);
148         ylim([0,100]);
149         hold off;
150
151         % Export the figure using export_fig
152         export_fig('temperature_history.png', '-png', ...
153             '-transparent', '-m2');
154
155         % Display the number of re-annealing events
156         disp(['Number of ReannealInterval events: ', ...
157             num2str(reanneal_count)]);
158
159         % Assign variables to base workspace
160         assignin('base','reanneal_count',reanneal_count);
161         assignin('base','fval_history',fval_history);

```

### A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
162     assignin('base','x_history',x_history);
163     assignin('base','temperature_history',temperature_history);
164     assignin('base','bestx_SA',bestx_SA);
165     assignin('base','bestfval_SA',bestfval_SA);
166 end
167
168 stop = false;
169 optchanged = false;
170 end
171
172 % Output function containing the graphs during
173 % hybrid optimization algorithm
174 function stop = hybrid_output_function(optimvalues,flag)
175     % Persistent variables to store data across iterations
176     persistent fval_history x_hist bestx_hybrid bestfval_hybrid;
177
178     % Switch statement to handle different states of the optimization
179     switch flag
180         % Init case: initialization of persistent variables based
181         % on final values from simulated annealing
182         case 'init'
183
184             fval_history = optimvalues.fval;
185             x_hist = optimvalues.x;
186
187             bestx_hybrid = optimvalues.x;
188             bestfval_hybrid = optimvalues.fval;
189
190         % Iter case: update historical data with current values
191         case 'iter'
192
193             fval_history = [fval_history; optimvalues.fval];
194             x_hist = [x_hist; optimvalues.x];
195
196             % Update best cost function value and
197             % x value in case they are improved
198             if optimvalues.fval < bestfval_hybrid
199                 bestx_hybrid = optimvalues.x;
200                 bestfval_hybrid = optimvalues.fval;
201             end
202
203         % Done case: final processing and plotting
204         case 'done'
205
206             % Plot the history of cost function values
207             % using the hybrid function
208             f_hybrid = figure;
209             plot(fval_history, 'o');
```

#### A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
210         hold on;
211         scatter(1:length(fval_history), fval_history, 'filled');
212         hold off;
213         xlabel('Iterations(hybrid)');
214         xlim([1, length(fval_history)]);
215         ylabel('costFunction(hybrid)');
216         set(gca, 'FontSize', 12);
217
218         export_fig(f_hybrid, 'hybrid_fval_history.png', '-png', '-m2');
219
220         % Assign variables to base workspace
221         assignin('base','bestx_hybrid',bestx_hybrid);
222         assignin('base','bestfval_hybrid',bestfval_hybrid);
223
224     end
225     stop = false;
226 end
227
228
```

**Βελτιστοποίηση της πιθανοφάνειας του συνόλου χωρικών δεδομένων (πραγματικά ή συνθετικά) χρησιμοποιώντας την υβριδική προσέγγιση με την προσομοιωμένη ανόπτηση και την τοπική μέθοδο fmincon**

```
1
2 % Specify kernel function and
3 % near-neighbor order for kernel bandwidths for the SLI model
4 ker.fun = 'Quad';
5 ker.nnb = 3;
6
7 % Two problems are examined.
8 % 1. Synthetic data
9 % 2. Campbell coal data (thickness)
10 % For both see:
11 % Hristopulos, D.T., Pavlides, A., Agou, V.D. et al.
12 % Stochastic Local Interaction Model: An Alternative to Kriging for Massive
13 % Datasets. Math Geosci 53, -19071949 (2021).
14 % https://doi.org/10.1007/s11004-021-09957-7
15
16 dataset = 1;
17
18 switch dataset
19
20     case 1
21
22         % This dataset contains 139 simulations of a chi-squared
```



### A.1. Προσομοιωμένη ανόπτηση για τους δύο τύπους προβλημάτων

```
23 % random field sampled at locations drawn from the Campbell dataset
24 load('Data/Simul_ChiSquare_Campbell_locs.mat');
25 locs = [X2 Y2];
26 datall = zol;
27 indsim = 1; % This selects a specific simulated state
28 z = zol(:, indsim);
29
30 % Initial values for the SLI parameters. These will be used
31 % for the minimization algorithm.
32 Param0 = [ mean(z) 1 1000 1.5];
33
34 % Define lower and upper bounds for the parameters
35 lb = [ mean(z)-2*std(z) eps eps 0.5];
36 ub = [ mean(z)+2*std(z) inf inf 15];
37
38 % Calculation of the negative log-likelihood
39 % for the initial parameters
40 [y0] = sli_mle_function(Param0, locs, z, ker);
41
42 % Definition of the objective function
43 % (sli_mle_function) to be minimized
44 objFun = @(Param) sli_mle_function(Param, locs, z, ker);
45
46 % Options for the hybrid optimization algorithm
47 hybridopts = optimoptions(@fmincon, ...
48     'Algorithm', 'interior-point','Display', 'iter', ...
49     'OutputFcn', @hybrid_output_function);
50
51 % Options for the simulated annealing algorithm
52 options = optimoptions(@simulannealbnd,'Display','iter', ...
53     'MaxFunctionEvaluations',3500,'MaxIterations',3000, ...
54     'ObjectiveLimit',3528.6,'FunctionTolerance',1e-04, ...
55     'InitialTemperature',700,'DisplayInterval',10, ...
56     'MaxStallIterations',1000,'TemperatureFcn',@temperatureexp, ...
57     'ReannealInterval',20,'AnnealingFcn',@annealingfast, ...
58     'OutputFcns', @plot_progress, ...
59     'HybridFcn',{@fmincon,hybridopts});
60
61 % Run the optimization using simulated annealing
62 % with a hybrid function
63 [optimalParam, fval, exitflag, output] = ...
64     simulannealbnd(objFun, Param0, lb, ub, options);
65
66 % Display the optimal parameters and the minimum NLL value
67 disp(['Optimal Parameters for Case 1: ', num2str(optimalParam)]);
68 disp(['Cost Function Value: ', num2str(fval)]);
69
70
```

```

71
72     case 2
73
74         % This dataset contains measurements of coal thickness in the
75         % Campbell field sampled at 11461 locations
76         load('Data/Campbell_normalized.mat');
77         locs = [X2 Y2];
78         z = TD1;
79
80         % Initial values for the SLI parameters. These will be used
81         % for the minimization algorithm.
82         Param0 = [ mean(z) 1 1000 1.5];
83
84         % Define lower and upper bounds for the parameters
85         lb = [ mean(z)-2*std(z) eps eps 0.5];
86         ub = [ mean(z)+2*std(z) inf inf 15];
87
88         % Calculation of the negative log-likelihood
89         % for the initial parameters
90         [y0] = sli_mle_function(Param0, locs, z, ker);
91
92         % Definition of the objective function
93         % (sli_mle_function) to be minimized
94         objective_function = @(Param) sli_mle_function(Param, locs, z, ker);
95
96         % Options for the hybrid optimization algorithm
97         hybridopts = optimoptions(@fmincon, ...
98             'Algorithm', 'interior-point','Display', 'iter', ...
99             'OutputFcn', @hybrid_output_function);
100
101         % Options for the simulated annealing
102         options = optimoptions(@simulannealbnd,'Display','iter', ...
103             'MaxFunctionEvaluations',2000,'MaxIterations',1500, ...
104             'FunctionTolerance',1e-04,'InitialTemperature',500, ...
105             'DisplayInterval',10,'MaxStallIterations',500, ...
106             'TemperatureFcn',@temperatureexp,'ReannealInterval',20, ...
107             'AnnealingFcn',@annealingfast, 'ObjectiveLimit',22954, ...
108             'OutputFcns', @plot_progress, ...
109             'HybridFcn',{@fmincon,hybridopts});
110
111         % Run the optimization using simulated annealing
112         % with a hybrid function
113         [OptimalParam, fval, exitflag, output] = ...
114             simulannealbnd(objective_function, Param0, lb, ub, options);
115
116         % Display the optimal parameters and the cost function value
117         disp(['Optimal Parameters for Case 2: ', num2str(OptimalParam)]);
118         disp(['Objective Function Value: ', num2str(fval)]);

```

```

119
120
121 end
122
123 % Function to plot the figures during
124 % simulated annealing algorithm optimization
125 function [stop,options,optchanged] = plot_progress(options,optimvalues,flag)
126     % Persistent variables to store data across iterations
127     persistent param_history fval_history relevant_fval;
128
129     switch flag
130
131         % Init case: define initial values for persistent variables
132         case 'init'
133
134             % Initialize an empty array
135             %to store the history of parameter values
136             param_history = [];
137
138             % Initialize an empty array
139             %to store the history of cost function values
140             fval_history = [];
141
142             % Initialize an empty array
143             %to calculating the logarithmic value of the cost function
144             relevant_fval=[];
145
146         % Iter case: update data with current values
147         case 'iter'
148
149             param_history = [param_history; optimvalues.x];
150             fval_history = [fval_history; optimvalues.fval];
151
152         % Done case: final processing and plotting
153         case 'done'
154
155             % Plot evolution of the value of the negative log-likelihood
156             figure;
157             plot(fval_history, 'LineWidth', 2);
158             title('Simulated Annealing for spatial real data', ...
159                 'FontWeight', 'bold', 'FontSize', 16);
160             xlabel('Iterations', 'FontSize', 14);
161             ylabel('Cost Function', 'FontSize', 14);
162             xlim([1, length(fval_history)]);
163             ylim([min(fval_history), max(fval_history)]);
164             grid on;
165
166             export_fig('cost_function_SA_real_data.png');

```

```

167
168     % Check the length of fval_history
169     max_iter = length(fval_history);
170     start_iter = 1;
171     end_iter = max_iter;
172
173     % Extract the entire fval_history
174     relevant_fval = fval_history(start_iter:end_iter);
175
176     % logarithmic transformation
177     log_fval = log(relevant_fval);
178
179     % Plot logarithmic scale of the NLL value
180     figure;
181     plot(start_iter:end_iter, log_fval, 'LineWidth', 2);
182     title({'Logarithmic Scale of Cost Function', ...
183           'for spatial real data'}, ...
184           'FontWeight', 'bold', 'FontSize', 14);
185     xlabel('Iterations', 'FontSize', 14);
186     ylabel('Log(Cost Function)', 'FontSize', 14);
187     set(gca, 'FontSize', 12);
188     grid on;
189     if start_iter < end_iter
190         xlim([start_iter, end_iter]);
191     end
192     ylim([min(log_fval(relevant_fval > 0)), max(log_fval)]);
193
194     % Find the iteration with the minimum value of the NLL
195     [~, minIndex] = min(fval_history);
196
197     % Optimal parameters at the minimum value of the NLL
198     optimalParam = param_history(minIndex, :);
199
200     % Create separate plots for each parameter
201     parameter_names = {'m', '\lambda', 'c_{1}', '\mu'};
202     for i = 1:size(param_history,2)
203         figure;
204         plot(param_history(:,i), 'LineWidth', 2);
205         hold on;
206         plot(minIndex, optimalParam(i), 'ro', 'MarkerSize', 10, ...
207              'LineWidth', 2, 'MarkerFaceColor', 'r');
208         hold off;
209         title(['Parameter: ', parameter_names{i}], ...
210              'FontWeight', 'bold', 'FontSize', 16);
211         xlabel('Iterations', 'FontSize', 14);
212         ylabel(['$', parameter_names{i}, '$'], ...
213              'Interpreter', 'latex', 'FontSize', 14);
214         set(gca, 'FontSize', 12);

```

```

215         grid on;
216         xlim([1, length(fval_history)]);
217         ylim([min(param_history(:,i)), max(param_history(:,i))]);
218     end
219
220     % Assign variables to base workspace
221     assignin('base', 'param_history', param_history);
222     assignin('base', 'fval_history', fval_history);
223 end
224 stop = false;
225 optchanged = false;
226 end
227
228 % Hybrid function to plot the figures during fmincon optimization
229 function [stop] = hybrid_output_function(x,optimvalues,state)
230 % Persistent variable to store data across iterations
231 persistent fval_history;
232
233 switch state
234     case 'init'
235         % Init case: initialization of persistent variable
236         %based on final value from simulated annealing
237         fval_history = optimvalues.fval;
238
239     case 'iter'
240         % Iter case: update data with current values
241         fval_history = [fval_history; optimvalues.fval];
242
243     case 'done'
244         % Done case: final processing and plotting
245
246         % Plot the evolution of NLL value during hybrid optimization
247         figure;
248         plot(fval_history, 'o');
249         hold on;
250         scatter(1:length(fval_history), fval_history, 'filled');
251         hold off;
252         xlabel('Iterations(hybrid)');
253         xlim([1, length(fval_history)]);
254         ylabel('costFunction(hybrid)');
255         set(gca, 'FontSize', 12);
256     end
257     stop = false;
258 end
259
260

```

## Α'.2 Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

Βελτιστοποίηση της συνάρτησης Ackley σε χώρο 10 διαστάσεων χρησιμοποιώντας την καθολική αναζήτηση

```
1
2
3 %Define the objective function (Ackley) to be minimized
4 ObjectiveFunction = @ackley;
5 % Definition of global variables
6 global funVals xVals optimStarts optimEnds;
7
8 % Initialize an empty array to store cost function values
9 funVals = [];
10 % Initialize an empty array to store x vectors
11 xVals = [];
12 % Initialize an empty array to track the start of local optimizations
13 optimStarts = [];
14
15 % Creating a random starting point within the bounds [-32.768, 32.768]
16 x0 = -32.768 + (32.768 - (-32.768)) * rand(1, 10);
17 % Lower and upper bounds for the variables
18 lb = ones(1, 10)*-32.768;
19 ub = ones(1, 10)*32.768;
20
21 % Options for the local solver fmincon
22 options = optimoptions(@fmincon, 'Algorithm', 'interior-point', ...
23     'Display', 'off', 'StepTolerance', 1e-15, 'OutputFcn', @outfun);
24
25 % Definition of the optimization problem for Global Search
26 problem = createOptimProblem('fmincon','objective',...
27     ObjectiveFunction,'x0',x0,'lb',lb,'ub',ub,'options',options);
28
29 % Options for Global Search algorithm
30 gs = GlobalSearch('MaxTime',60,'NumTrialPoints',5000, ...
31     'Display','iter', 'NumStageOnePoints', 700, ...
32     'PenaltyThresholdFactor', 0.5,'StartPointsToRun','bounds', ...
33     'MaxWaitCycle', 40, 'XTolerance', 1e-06, ...
34     'DistanceThresholdFactor', 1e-04, 'BasinRadiusFactor', 0.3, ...
35     'FunctionTolerance', 1e-06);
36
37 % Run Global Search algorithm
38 [x,fval] = run(gs, problem);
39
40 % Determine the end points of local optimizations
```

## Α.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
41 if ~isempty(optimStarts)
42     % Last starting point defines the end of the optimization process
43     optimEnds = [optimStarts(2:end) - 1; numel(funVals)];
44 else
45     optimEnds = [];
46 end
47
48 % Plot the evolution of cost function value with markers
49 % which they depict the start of each local optimization
50 figure;
51 plot(funVals, 'LineWidth', 1.5);
52 hold on;
53 if ~isempty(optimStarts)
54     plot(optimStarts, funVals(optimStarts), 'ro', 'MarkerFaceColor', 'r');
55 end
56 hold off;
57 xlabel('Iterations');
58 ylabel('CostFunction');
59 title('Ackley Function in 10 demensions (GlobalSearch)');
60 legend('Cost Function', 'Start of Local Optimization', 'Location', 'Best');
61 grid on;
62 xlim([1 length(funVals)]);
63
64 % Convert the cell array of x vectors to a matrix for plotting
65 xMatrix = cell2mat(xVals);
66
67 % Plot the evolution of x for dimensions 1 to 5
68 figure;
69 for i = 1:5
70     subplot(5, 1, i);
71     plot(xMatrix(:,i));
72     ylabel(['x(', num2str(i), ')']);
73     set(gca, 'FontSize', 12);
74     grid on;
75
76     xlim([1 size(xMatrix, 1)]);
77
78     if i == 5
79         xlabel('Iterations');
80     end
81     if i < 5
82         set(gca, 'XTickLabel', []);
83     end
84 end
85
86
87 % Plot the evolution of x for dimensions 6 to 10
88 figure;
```

## A'.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
89 for i = 6:10
90     subplot(5, 1, i-5);
91     plot(xMatrix(:,i));
92     ylabel(['x(', num2str(i), ')']);
93     set(gca, 'FontSize', 12);
94     grid on;
95
96     xlim([1 size(xMatrix, 1)]);
97
98     if i == 10
99         xlabel('Iterations');
100    end
101    if i < 10
102        set(gca, 'XTickLabel', []);
103    end
104 end
105
106 % Clear global variables
107 clear funVals xVals optimStarts;
108
109 % Display the position and value of the global minimum
110 disp('Global minimum found:');
111 disp(x);
112 disp('Cost function value at global minimum:');
113 disp(fval);
114
115 % Output function to track the optimization process
116 % Records cost function values, x vectors
117 % and marks the start of each local optimization.
118 function stop = outfun(x, optimValues, state)
119     stop = false;
120     global funVals xVals optimStarts;
121     funVals = [funVals; optimValues.fval];
122     xVals = [xVals; {x}];
123
124     switch state
125     case 'init'
126         % Record the start of each local optimization
127         optimStarts = [optimStarts; numel(funVals)];
128     case 'iter'
129
130     case 'done'
131
132     end
133 end
134
135 function [y] = ackley(xx)
136     % Constants
```



## Α.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
137     a = 20;
138     b = 0.2;
139     c = 2*pi;
140     d = length(xx);
141
142     % Ackley Function Calculation
143     sum1 = sum(xx.^2);
144     sum2 = sum(cos(c*xx));
145
146     term1 = -a * exp(-b*sqrt(sum1/d));
147     term2 = -exp(sum2/d);
148
149     y = term1 + term2 + a + exp(1);
150
151 end
152
153
```

**Βελτιστοποίηση της συνάρτησης Ackley σε χώρο 30 διαστάσεων χρησιμοποιώντας την καθολική αναζήτηση**

```
1
2
3
4 %Define the objective function (Ackley) to be minimized
5 ObjectiveFunction = @ackley;
6 % Definition of global variables
7 global funVals optimStarts optimEnds;
8
9 % Initialize an empty array to store cost function values
10 funVals = [];
11 % Initialize an empty array to track the start of local optimizations
12 optimStarts = [];
13
14 % Creating a random starting point within the bounds [-32.768, 32.768]
15 x0 = -32.768 + (32.768 - (-32.768)) * rand(1, 30);
16 % Lower and upper bounds for the variables
17 lb = ones(1, 30)*-32.768;
18 ub = ones(1, 30)*32.768;
19
20 % Options for the local solver fmincon
21 options = optimoptions(@fmincon, 'Algorithm', 'interior-point', ...
22     'Display', 'off', 'StepTolerance', 1e-15, 'OutputFcn', @outfun);
23
24 % Definition of the optimization problem for Global Search
25 problem = createOptimProblem('fmincon','objective',...
```

## A.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
26     ObjectiveFunction,'x0',x0,'lb',lb,'ub',ub,'options',options);
27
28 % Options for Global Search algorithm
29 gs = GlobalSearch('MaxTime',60,'NumTrialPoints',5000, ...
30     'Display','iter', 'NumStageOnePoints', 700, ...
31     'PenaltyThresholdFactor', 0.6,'StartPointsToRun','bounds', ...
32     'MaxWaitCycle', 70, 'XTolerance', 1e-06, ...
33     'DistanceThresholdFactor', 0.01, 'BasinRadiusFactor', 0.1, ...
34     'FunctionTolerance', 1e-06);
35
36 % Run Global Search algorithm
37 [x,fval] = run(gs, problem);
38
39 % Determine the end points of local optimizations
40 if ~isempty(optimStarts)
41     % Last starting point defines the end of the optimization process
42     optimEnds = [optimStarts(2:end) - 1; numel(funVals)];
43 else
44     optimEnds = [];
45 end
46
47 % Plot the evolution of cost function value with markers
48 % which they depict the start of each local optimization
49 figure;
50 plot(funVals, 'LineWidth', 1.5);
51 hold on;
52 if ~isempty(optimStarts)
53     plot(optimStarts, funVals(optimStarts), 'ro', 'MarkerFaceColor', 'r');
54 end
55 hold off;
56 xlabel('Iterations');
57 ylabel('CostFunction');
58 title('Ackley Function in 30 demensions (GlobalSearch)');
59 legend('Cost Function', 'Start of Local Optimization', 'Location', 'Best');
60 grid on;
61 xlim([1 length(funVals)]);
62
63 % Clear global variable
64 clear global funVals;
65
66 % Display the position and value of the global minimum
67 disp('Global minimum found:');
68 disp(x);
69 disp('Cost function value at global minimum:');
70 disp(fval);
71
72 % Output function to track the optimization process
73 % Records cost function values
```

## Α.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
74 % and marks the start of each local optimization.
75 function stop = outfun(x, optimValues, state)
76     stop = false;
77     global funVals optimStarts;
78     funVals = [funVals; optimValues.fval];
79
80     switch state
81         case 'init'
82             % Record the start of each local optimization
83             optimStarts = [optimStarts; numel(funVals)];
84         case 'iter'
85
86         case 'done'
87
88     end
89 end
90
91 function [y] = ackley(xx)
92     % Constants
93     a = 20;
94     b = 0.2;
95     c = 2*pi;
96     d = length(xx);
97
98     % Ackley Function Calculation
99     sum1 = sum(xx.^2);
100     sum2 = sum(cos(c*xx));
101
102     term1 = -a * exp(-b*sqrt(sum1/d));
103     term2 = -exp(sum2/d);
104
105     y = term1 + term2 + a + exp(1);
106
107 end
108
109
```

**Βελτιστοποίηση της συνάρτησης Cross-In-Tray σε χώρο 10 διαστάσεων χρησιμοποιώντας την καθολική αναζήτηση**

```
1
2 %Define the objective function (Cross-In-Tray) to be minimized
3 ObjectiveFunction = @crossit;
4 % Definition of global variables
5 global funVals xVals optimStarts optimEnds;
6 % Initialize an empty array to store cost function values
```

## A.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
7 funVals = [];  
8 % Initialize an empty array to store x vectors  
9 xVals = [];  
10 % Initialize an empty array to track the start of local optimizations  
11 optimStarts = [];  
12  
13 % Creating a random starting point within the bounds [-10, 10]  
14 x0 = -10 + (10 - (-10)) * rand(1, 10);  
15 % Lower and upper bounds for the variables  
16 lb = ones(1, 10)*-10;  
17 ub = ones(1, 10)*10;  
18  
19 % Options for the local solver fmincon  
20 options = optimoptions(@fmincon,'Algorithm','interior-point', ...  
21     'Display','off','OutputFcn', @outfun);  
22  
23 % Definition of the optimization problem for Global Search  
24 problem = createOptimProblem('fmincon','objective',...  
25     ObjectiveFunction,'x0',x0,'lb',lb,'ub',ub,'options',options);  
26  
27 % Options for Global Search algorithm  
28 gs = GlobalSearch('FunctionTolerance',1e-06,'MaxTime',60, ...  
29     'NumTrialPoints', 6000,'Display','iter', 'NumStageOnePoints', 600, ...  
30     'PenaltyThresholdFactor', 0.5,'StartPointsToRun','bounds', ...  
31     'XTolerance', 1e-06, 'MaxWaitCycle', 70, ...  
32     'DistanceThresholdFactor', 0.5, 'BasinRadiusFactor', 0.4);  
33  
34 % Run Global Search algorithm  
35 [x,fval] = run(gs, problem);  
36  
37 % Determine the end points of local optimizations  
38 if ~isempty(optimStarts)  
39     % Last starting point defines the end of the optimization process  
40     optimEnds = [optimStarts(2:end) - 1; numel(funVals)];  
41 else  
42     optimEnds = [];  
43 end  
44  
45 % Plot the evolution of cost function value with markers  
46 % which they depict the start of each local optimization  
47 figure;  
48 plot(funVals, 'LineWidth', 1.5);  
49 hold on;  
50 if ~isempty(optimStarts)  
51     plot(optimStarts, funVals(optimStarts), 'ro', 'MarkerFaceColor', 'r');  
52 end  
53 hold off;  
54 xlabel('Iterations');
```

## A'.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
55 ylabel('CostFunction');
56 title('Cross-In-Tray Function in 10 demensions (GlobalSearch)');
57 legend('Cost Function', 'Start of Local Optimization', 'Location', 'Best');
58 grid on;
59 xlim([1 length(funVals)]);
60
61 % Convert the cell array of x vectors to a matrix for plotting
62 xMatrix = cell2mat(xVals);
63
64 % Plot the evolution of x for dimensions 1 to 5
65 figure;
66 for i = 1:5
67     subplot(5, 1, i);
68     plot(xMatrix(:,i));
69     ylabel(['x(', num2str(i), ')']);
70     set(gca, 'FontSize', 12);
71     grid on;
72
73     xlim([1 size(xMatrix, 1)]);
74
75     if i == 5
76         xlabel('Iterations');
77     end
78     if i < 5
79         set(gca, 'XTickLabel', []);
80     end
81 end
82
83
84 % Plot the evolution of x for dimensions 6 to 10
85 figure;
86 for i = 6:10
87     subplot(5, 1, i-5);
88     plot(xMatrix(:,i));
89     ylabel(['x(', num2str(i), ')']);
90     set(gca, 'FontSize', 12);
91     grid on;
92
93     xlim([1 size(xMatrix, 1)]);
94
95     if i == 10
96         xlabel('Iterations');
97     end
98     if i < 10
99         set(gca, 'XTickLabel', []);
100     end
101 end
102
```

## A.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
103 % Clear global variables
104 clear funVals xVals optimStarts;
105
106 % Display the position and value of the global minimum
107 disp('Global minimum found:');
108 disp(x);
109 disp('Cost function value at global minimum:');
110 disp(fval);
111
112 % Output function to track the optimization process
113 % Records cost function values, x vectors
114 % and marks the start of each local optimization.
115 function stop = outfun(x, optimValues, state)
116     stop = false;
117     global funVals xVals optimStarts;
118     funVals = [funVals; optimValues.fval];
119     xVals = [xVals; {x}];
120
121     switch state
122     case 'init'
123         % Record the start of each local optimization
124         optimStarts = [optimStarts; numel(funVals)];
125     case 'iter'
126
127     case 'done'
128
129     end
130 end
131
132 % Cross-In-Tray function definition
133 function [y] = crossit(xx)
134 % Cross-In-Tray Function Calculation
135     n = 10;
136     fact1 = 1;
137     sum_squares = 0;
138     for i = 1:n
139         fact1 = fact1 * sin(xx(i));
140         sum_squares = sum_squares + xx(i)^2;
141     end
142     fact2 = exp(abs(100 - sqrt(sum_squares)/pi));
143
144     y = -0.0001 * (abs(fact1*fact2)+1)^0.1;
145
146 end
147
148
```

## Βελτιστοποίηση της συνάρτησης Cross-In-Tray σε χώρο 30 διαστάσεων χρησιμοποιώντας την καθολική αναζήτηση

```

1
2 %Define the objective function (Cross-In-Tray) to be minimized
3 ObjectiveFunction = @crossit;
4 % Definition of global variables
5 global funVals optimStarts optimEnds;
6 % Initialize an empty array to store cost function values
7 funVals = [];
8 % Initialize an empty array to track the start of local optimizations
9 optimStarts = [];
10
11 % Creating a random starting point within the bounds [-10, 10]
12 x0 = -10 + (10 - (-10)) * rand(1, 30);
13 % Lower and upper bounds for the variables
14 lb = ones(1, 30)*-10;
15 ub = ones(1, 30)*10;
16
17 % Options for the local solver fmincon
18 options = optimoptions(@fmincon,'Algorithm','interior-point', ...
19     'OutputFcn', @outfun);
20
21 % Definition of the optimization problem for Global Search
22 problem = createOptimProblem('fmincon','objective',...
23     ObjectiveFunction,'x0',x0,'lb',lb,'ub',ub,'options',options);
24
25 % Options for Global Search algorithm
26 gs = GlobalSearch('FunctionTolerance',1e-06,'MaxTime',60, ...
27     'NumTrialPoints', 6000,'Display','iter','NumStageOnePoints', 600, ...
28     'PenaltyThresholdFactor', 0.4,'StartPointsToRun','bounds', ...
29     'XTolerance', 1e-06, 'MaxWaitCycle', 150, ...
30     'DistanceThresholdFactor', 0.35, 'BasinRadiusFactor', 0.3);
31
32 % Run Global Search algorithm
33 [x,fval] = run(gs, problem);
34
35 % Determine the end points of local optimizations
36 if ~isempty(optimStarts)
37     % Last starting point defines the end of the optimization process
38     optimEnds = [optimStarts(2:end) - 1; numel(funVals)];
39 else
40     optimEnds = [];
41 end
42
43 % Plot the evolution of cost function value with markers
44 % which they depict the start of each local optimization
45 figure;

```

## A.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
46 plot(funVals, 'LineWidth', 1.5);
47 hold on;
48 if ~isempty(optimStarts)
49     plot(optimStarts, funVals(optimStarts), 'ro', 'MarkerFaceColor', 'r');
50 end
51 hold off;
52 xlabel('Iterations');
53 ylabel('CostFunction');
54 title('Cross-In-Tray Function in 30 demensions (GlobalSearch)');
55 legend('Cost Function', 'Start of Local Optimization', 'Location', 'Best');
56 grid on;
57 xlim([1 length(funVals)]);
58
59 % Clear global variables
60 clear funVals optimStarts;
61
62 % Display the position and value of the global minimum
63 disp('Global minimum found:');
64 disp(x);
65 disp('Function value at global minimum:');
66 disp(fval);
67
68 % Output function to track the optimization process
69 % Records cost function values
70 % and marks the start of each local optimization.
71 function stop = outfun(x, optimValues, state)
72     stop = false;
73     global funVals optimStarts;
74     funVals = [funVals; optimValues.fval];
75
76     switch state
77         case 'init'
78             % Record the start of each local optimization
79             optimStarts = [optimStarts; numel(funVals)];
80         case 'iter'
81
82         case 'done'
83
84     end
85 end
86
87 % Cross-In-Tray function definition
88 function [y] = crossit(xx)
89 % Cross-In-Tray Function Calculation
90
91     n = 30;
92     fact1 = 1;
93     sum_squares = 0;
```



## A'.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
94     for i = 1:n
95         fact1 = fact1 * sin(xx(i));
96         sum_squares = sum_squares + xx(i)^2;
97     end
98     fact2 = exp(abs(100 - sqrt(sum_squares)/pi));
99
100     y = -0.0001 * (abs(fact1*fact2)+1)^0.1;
101
102 end
103
104
```

**Βελτιστοποίηση της πιθανοφάνειας του συνόλου χωρικών δεδομένων (πραγματικά ή συνθετικά) χρησιμοποιώντας την καθολική αναζήτηση**

```
1
2 % Specify kernel function and
3 % near-neighbor order for kernel bandwidths for the SLI model
4 ker.fun = 'Quad';
5 ker.nnb = 3;
6
7 % Two problems are examined.
8 % 1. Synthetic data
9 % 2. Campbell coal data (thickness)
10 % For both see:
11 % Hristopulos, D.T., Pavlides, A., Agou, V.D. et al.
12 % Stochastic Local Interaction Model: An Alternative to Kriging for Massive
13 % Datasets. Math Geosci 53, -19071949 (2021).
14 % https://doi.org/10.1007/s11004-021-09957-7
15
16 % Definition of global variables
17 global funVals meanVals lambdaVals ciVals muVals optimStarts optimEnds ;
18
19 dataset = 1;
20
21 switch dataset
22
23     case 1
24
25         % This dataset contains 139 simulations of a chi-squared
26         % random field sampled at locations drawn from the Campbell dataset
27         load('Data/Simul_ChiSquare_Campbell_locs.mat');
28         locs = [X2 Y2];
29         datall = zol;
30         indsim = 1; % This selects a specific simulated state
31         z = zol(:, indsim);
```

```

32
33     % Initial values for the SLI parameters. These will be used
34     % for the minimization algorithm.
35     Param0 = [ mean(z) 1 1000 1.5];
36
37     % Define lower and upper bounds for the parameters
38     lb = [ mean(z)-2*std(z) eps eps 0.5];
39     ub = [ mean(z)+2*std(z) inf inf 15];
40
41     % Calculation of the negative log-likelihood
42     % for the initial parameters
43     [y0] = sli_mle_function(Param0, locs, z, ker);
44
45     % Definition of the objective function
46     % (sli_mle_function) to be minimized
47     fun = @(Param) sli_mle_function(Param, locs, z, ker);
48
49     % Initialize an empty array to store cost function values
50     funVals = [];
51     % Initialize an empty array to store the values
52     % of the mean parameter
53     meanVals = [];
54     % Initialize an empty array to store the values
55     % of the lambda parameter
56     lambdaVals = [];
57     % Initialize an empty array to store the values of the c1 parameter
58     c1Vals = [];
59     % Initialize an empty array to store the values of the mu parameter
60     muVals = [];
61     % Initialize an empty array to track
62     % the start of local optimizations
63     optimStarts = [];
64
65     % Options for the local solver fmincon
66     opts = optimoptions('fmincon','Display', 'off', ...
67         'UseParallel',true,'OutputFcn', @outfun);
68
69     % Definition of the optimization problem for Global Search
70     problem = createOptimProblem('fmincon', 'objective', fun, ...
71         'x0', Param0, 'lb', lb, 'ub', ub,'options',opts);
72
73     % Options for Global Search algorithm
74     gs = GlobalSearch('FunctionTolerance', 1e-06, ...
75         'NumTrialPoints', 1500, 'Display', 'iter', ...
76         'NumStageOnePoints', 300, ...
77         'MaxTime',7200, ...
78         'PenaltyThresholdFactor', 0.35, ...
79         'StartPointsToRun', 'bounds', ...

```

## A'.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
80         'XTolerance', 1e-06, 'MaxWaitCycle', 80, ...
81         'DistanceThresholdFactor', 0.5, ...
82         'BasinRadiusFactor', 0.3);
83
84 % Run Global Search algorithm
85 [xOpt, fOpt] = run(gs, problem);
86
87 % Determine the end points of local optimizations
88 if ~isempty(optimStarts)
89     % Last starting point defines the end
90     % of the optimization process
91     optimEnds = [optimStarts(2:end) - 1; numel(funVals)];
92 else
93     optimEnds = [];
94 end
95
96 % Plot the evolution of the value of the negative log-likelihood
97 % with markers which they depict
98 % the start of each local optimization
99 figure;
100 plot(funVals, 'LineWidth', 2);
101 hold on;
102 if ~isempty(optimStarts)
103     plot(optimStarts, funVals(optimStarts), 'ro', ...
104          'MarkerFaceColor', 'r');
105 end
106 hold off;
107 title('Global Search for spatial synthetic data', ...
108       'FontWeight', 'bold', 'FontSize', 16);
109 legend('Cost Function', 'Start of Local Optimization', ...
110        'Location', 'Best');
111 xlabel('Iterations', 'FontSize', 14);
112 ylabel('Cost Function', 'FontSize', 14);
113 grid on;
114 xlim([1, length(funVals)]);
115 ylim([min(funVals), max(funVals)]);
116
117 % Extract the optimized parameters
118 [~, minIdx] = min(funVals);
119 optimizedMean = xOpt(1);
120 optimizedLambda = xOpt(2);
121 optimizedC1 = xOpt(3);
122 optimizedMu = xOpt(4);
123
124 % Find the index of the optimized parameters
125 [~, optimalIdx] = min(abs(meanVals - optimizedMean) + ...
126                       abs(lambdaVals - optimizedLambda) + ...
127                       abs(c1Vals - optimizedC1) + abs(muVals - optimizedMu));
```

## A'.2. Καθολική αναζήτηση για τους δύο τύπους προβλημάτων

```
128
129 % Names of parameters for plotting
130 parameter_names = {'m', '\lambda', 'c_{1}', '\mu'};
131 optimalParams = [optimizedMean, optimizedLambda, ...
132     optimizedC1, optimizedMu];
133 paramHistories = {meanVals, lambdaVals, c1Vals, muVals};
134
135 % Create separate plots for each parameter
136 for i = 1:length(parameter_names)
137     figure;
138     plot(paramHistories{i}, 'LineWidth', 2);
139     hold on;
140     plot(optimalIdx, optimalParams(i), 'ro', 'MarkerSize', 10, ...
141         'LineWidth', 2, 'MarkerFaceColor', 'r');
142     hold off;
143     title(['Parameter: ', parameter_names{i}], ...
144         'FontWeight', 'bold', 'FontSize', 16);
145     xlabel('Iterations', 'FontSize', 14);
146     ylabel(['$', parameter_names{i}, '$'], ...
147         'Interpreter', 'latex', 'FontSize', 14);
148     set(gca, 'FontSize', 12);
149     grid on;
150     xlim([1, length(paramHistories{i})]);
151     ylim([min(paramHistories{i}), max(paramHistories{i})]);
152 end
153
154 % Clear global variables
155 clear global funVals meanVals lambdaVals c1Vals muVals optimStarts;
156
157 % Display the optimal parameters and the minimum NLL value
158 disp('Optimized Parameters:');
159 disp(xOpt);
160 disp(['Minimum cost function Value: ', num2str(fOpt)]);
161
162
163 case 2
164
165 load('Data/Campbell_normalized.mat');
166 % This dataset contains measurements of coal thickness in the
167 % Campbell field sampled at 11461 locations
168 locs = [X2 Y2];
169 z = TD1;
170
171 % Initial values and bounds for the SLI parameters.
172 % These will be used for the minimization algorithm.
173 Param0 = [ mean(z) 1 1000 1.5];
174
175 lb = [ mean(z)-2*std(z) eps eps 0.5];
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
176         ub = [ mean(z)+2*std(z)   inf         inf         15];
177
178         [y0] = sli_mle_function(Param0, locs, z, ker);
179
180
181     end
182
183
184 % Output function to track and store results of the optimization process
185 function stop = outfun(x, optimValues, state)
186     stop = false;
187     global funVals meanVals lambdaVals c1Vals muVals optimStarts;
188     funVals = [funVals; optimValues.fval];
189     switch state
190         case 'init'
191             % Record the start of each local optimization
192             optimStarts = [optimStarts; numel(funVals)];
193         case 'iter'
194             % Save current value of the parameters in global variables
195             % This helps to observe how
196             % each parameter changes over the iterations.
197             meanVals = [meanVals; x(1)];
198             lambdaVals = [lambdaVals; x(2)];
199             c1Vals = [c1Vals; x(3)];
200             muVals = [muVals; x(4)];
201         case 'done'
202
203     end
204 end
205
206
```

### A.3 Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
1
2 function [y] = sli_mle_function(Param, locs, z, ker)
3 %   SLI_MLE_FUNCTION is the MLE cost functional (Negative Log-Likelihood)
4 %   used to determine the SLI parameters.
5 %=====
6 %   Input parameters
7 %=====
8 %   Param:   SLI parameters (mean, lambda, c1, mu)
9 %   locs:    Data locations (Array N x d)
10 %   z:       Data values (Array N x 1).
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
11 %   ker:      Structure array containing kernel function information
12 %   ker.fun:   Type of kernel function that will be used
13 %   ker.nnb:   Near-neighbor order for kernel bandwidths
14 %=====
15 %   Output
16 %=====
17 % y:         Value of the MLE cost functional (Negative log-likelihood)
18 %=====
19 % EXAMPLE
20 %=====
21 %
22 %
23 %=====
24 %   FUNCTIONS USED
25 %   calc_dist_g: Estimate matrix of spatial distances
26 %   estim_J2_gra: Precision matrix calculation
27 %   logdet2:      It calculates the logarithm of the determinant of the
28 %                  large sparse precision matrix
29 %   kernel_s:      Kernel function
30 %=====
31 %   Copyright (C) 2023 Dionisis Hristopulos
32 %   email: dchristopoulos@tuc.gr
33 %=====
34 %   This program is free software; you can redistribute it and/or modify it
35 %   under the terms of the GNU General Public License as published by the
36 %   Free Software Foundation (version 2 of the License or later version).
37 %
38 %   This program is distributed WITHOUT ANY WARRANTY.
39 %   See the GNU General Public License for more details.
40 %   It has been tested under Matlab 2023
41 %=====
42 % REFERENCES (applications to parameter inference for random fields)
43 %=====
44 % Hristopulos, D.T., Pavlides, A., Agou, V.D. et al.
45 % Stochastic Local Interaction Model: An Alternative to Kriging for Massive
46 % Datasets. Math Geosci 53, €"19071949 (2021).
47 % https://doi.org/10.1007/s11004-021-09957-7
48 %
49 % Dionissios T. Hristopulos, Stochastic Local Interaction (SLI) model:
50 % Bridging machine learning and geostatistics, Computers & Geosciences,
51 % Volume 85, Part B, 2015, pages 26-37, doi: 10.1016/j.cageo.2015.05.018.
52 %=====
53
54 if nargin ~= 4
55     disp('Incorrect number of arguments in sli_mle_function');
56 end
57
58 fnan=0; finf=0;
```

### Α.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
59
60 mz = Param(1);
61 lambda = Param(2);
62 Param1 = Param(3: 4); % Param1 includes SLI parameters without lambda
63
64 dd = size(locs,2); % Dimensionality of space
65 kernel = ker.fun; % Kernel function to be used
66 K = ker.nnb; % Near-neighbor order for kernel bandwidth estimation
67
68 %=====
69 %   Estimation of distance matrix
70 %=====
71
72 [rr, h1] = calc_dist_g(dd, locs, K);
73
74 distms{1} = rr;
75 distms{2} = h1;
76
77 clear rr;
78 %=====
79 %   Estimation of SLI precision matrix
80 %=====
81
82 J = estim_J2_gra(Param1, distms, kernel);
83 if any(isnan(J)), disp('Nan Values in J');
84     fnan=1;
85 end
86
87 if any(isinf(J)), disp('Inf Values in J');
88     finf=1;
89 end
90
91 %=====
92 if fnan==1
93     return;
94 end
95 zn = z - mz;
96 zn = zn(:);
97 %size(zn), size(J),
98 HH = (zn' * J * zn) / 2 ;
99 if HH<0 || finf==1
100     y = realmax;
101 else
102
103 %=====
104 %   The following lines show how the Negative Log Likelihood is calculated
105 % L = exp(-H)/Z => logL = - H - logZ => NLL = -logL = H +logZ
106 % J= J1/lambda : Z=(2pi)^(N/2) |detJ|^{-1/2} =
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```

107 %          (2pi lambda)^(N/2) * (detJ1)^{-1/2}
108 % => NLL = H + N/2 * log(lambda) - 1/2 log(det J1) =
109 %          (zn' * J * zn) / (2 * lambda) + N/2 * log(lambda) - 1/2 log(det J1)
110
111     N = length(locs);
112
113     y1 = HH/lambda;
114     y2 = (N*log(2*pi*lambda) - logdet2(J))/2 ;
115     y = y1 + y2;
116 end
117
118 end
119 %=====
120 function [ r, h1 ] = calc_dist_g(dd, locs, K)
121 %   CALC_DIST_G Calculates the distance matrix and local bandwidths aspect
122 %   ratios. The common scaling factor MU is required to fully determine
123 %   bandwidths
124 %===== INPUT PARAMETERS =====
125 %   DD:          Dimension index (integer)
126 %   LOCS:        Array containing the coordinates of points ( N x DD)
127 %   K:           Number of near neighbors to take into consideration (integer)
128 %               NOTE: for kernels 'Expo' and 'Gaus' works best with K = 2
129 %===== OUTPUT PARAMETERS =====
130 %   RR:          Matrix of point-point distances ( N x N)
131 %   H1:          Vector of gradient bandwidth ratios ( N x 1)
132 %===== EXAMPLE =====
133 %   locs = rand(10,3); [ r, h1 ] = calc_dist_g(3, locs, 'Quad', 3);
134 %===== EXAMPLE =====
135 % Copyright (C) 2017 Dionisis Hristopoulos
136 % email: dionisi@mred.tuc.gr
137 %
138 % This program is free software; you can redistribute it and/or modify it
139 % under the terms of the GNU General Public License as published by the
140 % Free Software Foundation (version 2 of the License or later version).
141 %
142 % This program is distributed WITHOUT ANY WARRANTY.
143 % See the GNU General Public License for more details.
144 % This software bundle also includes code developed by other people.
145 % DISTMAT: Developed by Joseph Kirk
146 %=====
147 N = length(locs);
148
149 if N <= K
150     disp('Number of sampling locations should be > K');
151     r = nan; h1 = nan;
152     return;
153 end
154

```



### Α.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
155 if size(locs,2) ~= dd
156     locs = locs';
157 end
158
159 r = zeros(N);
160 h1 = zeros(N,1);
161
162 if N <=30, opt=1;
163 elseif (30< N) && (N<=300), opt=2;
164 else
165     opt=3;
166 end
167
168 switch dd
169     case rem(dd,1)~=0
170         disp('Non-integer dimension');
171         return;
172     case dd<0
173         disp('Negative dimension');
174         return;
175     case 2
176         r = distmat(locs, opt);
177     case 1
178         xs=locs;
179         r = abs( xs * ones(1, N) - ones(N, 1) * xs' );
180     otherwise
181         r = distmat(locs, opt);
182 end
183
184 %=====
185 %   Determine the local bandwidth aspect ratios
186 %=====
187 %   Use nearest neighbors for infinitely supported kernels and second
188 %   nearest neighbors for compactly supported kernels
189 %=====
190
191 % if strcmp(kernel,'Expo') || strcmp(kernel,'Gaus')
192 %     K=2;
193 % else
194 %     K=4;
195 % end
196 K = K+1; % +1 because K = 1 is the same point --> distance 0
197
198 switch dd
199     case 1
200         x = locs(:);
201         [~, D] = knnsearch(x, x, 'k', K);
202     case 2
```

### A'.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
203         if size(locs, 2)~=2
204             disp('Dimension mismatch');
205             return;
206         end
207         [~, D] = knnsearch(locs, locs, 'k', K);
208
209     otherwise
210         [~, D] = knnsearch(locs, locs, 'k', K);
211 end
212 h1 = D(:,K);
213
214 end
215
216 %=====
217 function [ret] = logdet2(X)
218 % Safe calculation of natural logarithm of determinant for large sparse
219 % matrices using LU decomposition.
220 %
221 % function [ret] = logdet2(X)
222 % X - matrix [n,n]
223 % ret - logarithm of determinant [scalar]
224
225 L=chol(X);
226 ret = 2*sum(log(diag(L)));
227
228 end
229 %=====
230 function [Jmat, Jsub, Den] = estim_J2_gra(Param1, distms, kernel)
231 % ESTIM_J2_GRA calculates the SLI precision matrix between sampling points
232 %=====
233 % Input parameters
234 %=====
235 % Param1: SLI parameters (c1, mu)
236 % distms: Cell containing the matrix of pair distances and the vector of
237 %         gradient bandwidths
238 % kernel: Type of kernel function that will be used
239 %=====
240 % Output parameters
241 %=====
242 % Jmat: SLI Precision matrix
243 % Jsub: SLI precision submatrices
244 % Den: Denominator of gradient kernel averages
245 %=====
246 % EXAMPLE
247 %=====
248 %
249 %=====
250 % FUNCTIONS USED
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
251 %   KERNEL_S:   Kernel function
252 %=====
253 %   Copyright (C) 2023 Dionisis Hristopoulos
254 %   email: dchristopoulos@tuc.gr
255 %
256 %   This program is free software; you can redistribute it and/or modify it
257 %   under the terms of the GNU General Public License as published by the
258 %   Free Software Foundation (version 2 of the License or later version).
259 %
260 %   This program is distributed WITHOUT ANY WARRANTY.
261 %   See the GNU General Public License for more details.
262 %=====
263 %   Notes: Unlike previous versions of the code (Estim_J2_g) this version
264 %   assumes that the dimension factor d is absorbed in c1. This means that
265 %   values of c1 obtained with the old code should be multiplied with 2.
266
267 c1 = Param1(1);
268 mu = Param1(2);
269
270 r = distms{1};
271
272 h1 = mu * distms{2};
273
274 N = length(h1);
275
276 Jsub = cell(2,1);
277 Jsub{1} = zeros(N,N);
278 Jsub{2} = zeros(N,N);
279
280 %=====
281 %   Build the Precision matrix
282 %=====
283 J0=sparse(eye(N)/N); %sparse
284
285 %=====Gradient submatrix=====
286
287 hs1=repmat(h1(:),1,N);
288 Num1 = sparse(kernel_s( r./hs1, kernel));
289 Den1 = sum(sum((Num1)));
290
291 v1=sum(Num1,2);
292 v2=sum(Num1,1);
293
294 u1= -(Num1 + Num1') + diag( v1 + v2');
295
296 J1 = u1 ./ Den1;
297
298 if any(isnan(J1)), disp('Nan Values in J1');
```

### A'3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
299 end
300
301 %=====
302 Jmat = J0 + c1 * J1;
303 Jsub{1} = J0;
304 Jsub{2} = c1 * J1 ;
305 if any(isinf(Jmat))
306     disp('Inf Values in J');
307 end
308
309 Den = Den1;
310
311 %=====
312 function y=kernel_s(r,kernel,nu)
313 %=====
314 % The function KERNEL_S computes the kernel function at distance 'r'
315 %===== INPUT =====
316 % r:          Vector (or matrix) of distances
317 % KERNEL:     Type of kernel function used in the estimate of the
318 %             sample constraints
319 %             Supported kernel functions
320 %             Kernel choices:
321 %             bitriangular: 'Tria'
322 %             Epanechnikov: 'Quad'
323 %             Tricubic:     'Tric'
324 %             Exponential:  'Expo'
325 %             Gaussian:     'Gaus'
326 %             Cauchy:       'Cauc'
327 %             Spherical:    'Sphe'
328 %             Cosine:       'Cosi'
329 %             Sine wave:    'Sinc'
330 %             Triangular:   'Trie'
331 %             Dirac:        'Dira'
332 %             Uniform:      'Unif'
333 %             Biweight:     'Biwe'
334 %             Matern:       'Mate'
335 %
336 % NU:         Smoothness parameters (used only for Matern kernel)
337 %
338 %===== OUTPUT =====
339 % y: Values of the kernel function (vector or matrix, depending on r)
340 %=====
341 % Copyright (C) 2013 Samuel Elogne and Dionisis Hristopoulos
342 % email: dchristopoulos@tuc.gr
343 %
344 % This program is free software; you can redistribute it and/or modify it
345 % under the terms of the GNU General Public License as published by the
346 % Free Software Foundation (version 2 of the License or later version).
```

### Α.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
347 % This program is distributed WITHOUT ANY WARRANTY.
348 % See the GNU General Public License for more details.
349 %=====
350 % For kernel expressions and applications with
351 % stochastic local interaction models
352 %=====
353 % Dionissios T. Hristopulos: Random Fields for Spatial Data Modeling.
354 % A Primer for Scientists and Engineers, Springer Nature B.V. 2020,
355 % Springer, Dordrecht, ISBN: 978-94-024-1916-0
356 %=====
357 if strcmp(kernel, 'Sinc')
358     y= abs(1*(r==0)+sin(r)./(r+(r==0)));
359 elseif strcmp(kernel, 'Cosi')
360     y=abs(cos(r));
361 elseif strcmp(kernel, 'Tria')
362     y=((1-abs(r)).^2).*(abs(r)<=1) ;
363 elseif strcmp(kernel, 'Tric')
364 % Tricubic Kernel
365     y=((1-abs(r).^3).^3).*(abs(r)<=1);
366 elseif strcmp(kernel, 'Quad')
367     y=((1-r.^2)).*(abs(r)<=1);
368 elseif strcmp(kernel, 'Gaus')
369     y=exp(-r.^2);
370 elseif strcmp(kernel, 'Expo')
371     y=exp(-abs(r));
372 elseif strcmp(kernel, 'Exps')
373     y=exp(-(abs(r)).^(0.57));
374 elseif strcmp(kernel, 'Trie')
375 y=((1-abs(r))).*(abs(r)<=1) ;
376 elseif strcmp(kernel, 'Triw')
377 y=((1-abs(r)).^2).^3).*(abs(r)<=1) ;
378 elseif strcmp(kernel, 'Biwe')
379     y=((1-abs(r)).^2).^2).*(abs(r)<=1) ;
380 elseif strcmp(kernel, 'Sphe')
381     y=(1-1.5*abs(r)+0.5*abs(r).^3).*(abs(r)<=1);
382 elseif strcmp(kernel, 'Cauc')
383     % y=(1./(1+r.^2)).*(abs(r)<=10);
384     y=(1./(1+r.^2));
385 elseif strcmp(kernel, 'Dira')
386     y=1.*(abs(r)==0)+0.*(abs(r)~=0);
387 elseif strcmp(kernel, 'Unif')
388     y=(abs(r) <= 1);
389     elseif strcmp(kernel, 'Mate')
390     c0=(2^(nu-1))*gamma(nu);
391     j=r==0;
392     r(j)=eps;
393     bes=besselk(nu,r);
394     y=(1./c0).*(r.^nu).*bes;
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
395     else
396         sprintf('%s','Unrecognizable Kernel')
397     end
398
399 end
400
401 end
402 %=====
403
404 function [dmat,opt] = distmat(xy,varargin)
405 % DISTMAT Distance matrix for a set of points
406 % Returns the point-to-point distance between all pairs of points in XY
407 % (similar to PDIST in the Statistics Toolbox)
408 %
409 % DMAT = DISTMAT(XY) Calculates the distance matrix using an automatic
410 % option
411 % DMAT = DISTMAT(XY,OPT) Uses the specified option to compute the distance
412 % matrix
413 % [DMAT,OPT] = DISTMAT(XY) Also returns the automatic option used by the
414 % function
415 %
416 % Inputs:
417 % XY is an NxP matrix of coordinates for N points in P dimensions
418 % OPT (optional) is an integer between 1 and 4 representing the chosen
419 % method for computing the distance matrix (see note below)
420 %
421 % Outputs:
422 % DMAT is an NxN matrix, where the value of DMAT(i,j) corresponds to
423 % the distance from XY(i,:) to XY(j,:)
424 % OPT (optional) is an integer between 1 and 4 representing the method
425 % used to compute the distance matrix (see note below)
426 %
427 % Note:
428 % DISTMAT contains 4 methods for computing the distance matrix
429 % OPT=1 Usually fastest for small inputs. Takes advantage of the
430 % symmetric
431 % property of distance matrices to perform half as many
432 % calculations
433 % OPT=2 Usually fastest for medium inputs. Uses a fully vectorized
434 % method
435 % OPT=3 Usually fastest for large inputs. Uses a partially vectorized
436 % method with relatively small memory requirement
437 % OPT=4 Another compact calculation, but usually slower than the
438 % others
439 %
440 % Example:
441 % % Test computation times for the options
442 % n = [10 100 1000];
```

### Α.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

```
436 %         dmat = distmat(10*rand(10,3),1); % First call is always really slow
437 %         for k=1:3
438 %             for opt=1:4
439 %                 tic; [dmat,opt] = distmat(10*rand(n(k),3),opt); t=toc;
440 %                 disp(sprintf('n=%d, opt=%d, t=%0.6f', n(k), opt, t))
441 %             end
442 %         end
443 %
444 % Example:
445 %         xy = 10*rand(25,2); % 25 points in 2D
446 %         dmat = distmat(xy);
447 %         figure; plot(xy(:,1),xy(:,2),'.');
448 %         for k=1:25, text(xy(k,1),xy(k,2),[' ' num2str(k)]); end
449 %         figure; imagesc(dmat); set(gca,'XTick',1:25,'YTick',1:25); colorbar
450 %
451 % Example:
452 %         xyz = 10*rand(20,3); % 20 points in 3D
453 %         dmat = distmat(xyz);
454 %         figure; plot3(xyz(:,1),xyz(:,2),xyz(:,3),'.');
455 %         for k=1:20, text(xyz(k,1),xyz(k,2),xyz(k,3),[' ' num2str(k)]); end
456 %         figure; imagesc(dmat); set(gca,'XTick',1:20,'YTick',1:20); colorbar
457 %
458 % Author: Joseph Kirk
459 % Email: jdkirk630 at gmail dot com
460 % Release: 1.0
461 % Release Date: 5/29/07
462
463 % process inputs
464 narginchk(1,2);
465 [n,dims] = size(xy);
466 numel = n*n*dims;
467 opt = 2; if numel > 5e4, opt = 3; elseif n < 20, opt = 1; end
468 for var = varargin
469     if length(var{1}) == 1
470         opt = max(1, min(4, round(abs(var{1})))));
471     else
472         error('Invalid input argument.');
```

```
473     end
474 end
475
476 % distance matrix calculation options
477 switch opt
478     case 1 % half as many computations (symmetric upper triangular property)
479         [k,kk] = find(triu(ones(n),1));
480         dmat = zeros(n);
481         dmat(k+n*(kk-1)) = sqrt(sum((xy(k,:) - xy(kk,:)).^2,2));
482         dmat(kk+n*(k-1)) = dmat(k+n*(kk-1));
483     case 2 % fully vectorized calculation (very fast for medium inputs)
```

### A.3. Συνάρτηση υπολογισμού της αρνητικής λογαριθμικής πιθανοφάνειας

---

```
484     a = reshape(xy,1,n,dims);
485     b = reshape(xy,n,1,dims);
486     dmat = sqrt(sum((a(ones(n,1),:,:) - b(:,ones(n,1),:)).^2,3));
487 case 3 % partially vectorized (smaller memory requirement for large
inputs)
488     dmat = zeros(n,n);
489     for k = 1:n
490         dmat(k,:) = sqrt(sum((xy(k*ones(n,1),:) - xy).^2,2));
491     end
492 case 4 % another compact method, generally slower than the others
493     a = (1:n);
494     b = a(ones(n,1),:);
495     dmat = reshape(sqrt(sum((xy(b,:) - xy(b',:)).^2,2)),n,n);
496 end
497
498
```