Συστήματα Επιχειρηματικής Ευφυίας

Εισαγωγικές έννοιες Υπολογιστικής Νοημοσύνης

Διάρθρωση του μαθήματος

- Το μάθημα αποτελείται από τρείς τρίωρες διαλέξεις και ένα επαναληπτικό τρίωρο.
- Οι διαλέξεις αφορούν σε μια εισαγωγή στην Υπολογιστική Νοημοσύνη (ΥΝ). Περιγράφονται επιλεκτικά μερικοί γνωστοί αλγόριθμοι της ΥΝ και παρουσιάζονται κάποια προβλήματα εφαρμογής τους.

Πρώτη Διάλεξη – Περιεχόμενα (1)

- Προβλήματα βελτιστοποίησης.
- Περιορισμοί προβλημάτων βελτιστοποίησης.
- Λύσεις προβλημάτων βελτιστοποίησης.
- Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.
- Ή όλα ή τίποτε! Ή μήπως δεν είναι έτσι;…
- Υπολογιστική Νοημοσύνη: ένας νέος επιστημονικός κλάδος.
- Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.

Πρώτη Διάλεξη – Περιεχόμενα (2)

- Η τεχνική Hill climbing: εκ των ων ουκ άνευ της ΥΝ.
- Ένα ενδιαφέρον ζήτημα της τεχνικής Hill climbing!
- Test functions: Ορισμός-χρησιμότητα
- Παραδείγματα test optimization functions
- Αλγόριθμοι ΥΝ: αιτιοκρατικοί (ντετερμινιστικοί) ή στοχαστικοί (πιθανοτικοί);
- Πώς αξιολογούμε έναν αλγόριθμο της ΥΝ.

Προβλήματα βελτιστοποίησης (1)

- Πρόβλημα βελτιστοποίησης: στα Μαθηματικά και την Επιστήμη των ΗΥ ορίζεται ως η εύρεση της κάλλιστης λύσης από ένα σύνολο **εφικτών** λύσεων.
- Τα προβλήματα βελτιστοποίησης κατηγοριοποιούνται σε δύο κατηγορίες:
  - Συνδυαστικά (combinatorial) προβλήματα βελτιστοποίησης, των οποίων οι μεταβλητές είναι διακριτές.
  - Συνεχή (continuous) προβλήματα βελτιστοποίησης, των οποίων οι μεταβλητές είναι συνεχείς.

Προβλήματα βελτιστοποίησης (2)

- Στα συνδυαστικά προβλήματα αναζητούμε ένα αντικείμενο (λύση), το οποίο συνήθως είναι μια αντιμετάθεση (permutation) Ακεραίων αριθμών, ή έναν γράφο (graph) ανάμεσα σε ένα πεπερασμένο σύνολο ή άπειρο αλλά αριθμήσιμο σύνολο.

- Στα συνεχή προβλήματα βελτιστοποίησης αναζητούμε την λύση από ένα άπειρο σύνολο.

Προβλήματα βελτιστοποίησης (3)

- Τόσο τα συνδυαστικά όσο και τα συνεχή προβλήματα βελτιστοποίησης μπορεί να αφορούν σε μεγιστοποίηση ή ελαχιστοποίηση. Για παράδειγμα, η εύρεση του ολικού μεγίστου μιας πραγματικής συνάρτησης με την μέθοδο των παραγώγων συνιστά ένα πρόβλημα μεγιστοποίησης. Ακόμη, ο προσδιορισμός των παραμέτρων και συνθηκών κάτω από τις οποίες μια επιχείρηση ελαχιστοποιεί το κόστος λειτουργίας της ή το κόστος παραγωγής ενός προϊόντος συνιστά ένα πρόβλημα ελαχιστοποίησης.

Περιορισμοί προβλημάτων βελτιστοποίησης.

- Κάθε πρόβλημα εμπεριέχει κάποιους περιορισμούς. Αυτοί διακρίνονται σε:
- **Ανελαστικούς περιορισμούς** (Hard Constraints): είναι αυτοί που πρέπει να ικανοποιούνται οπωσδήποτε κατά την λύση του προβλήματος.
- **Ελαστικούς περιορισμούς** (Soft Constraints): είναι αυτοί των οποίων η ικανοποίηση είναι επιθυμητή αλλά όχι απαραίτητη.
- **Πεντάλεπτος προβληματισμός**: αναζητείστε ένα πρόβλημα και αναφέρετε κάποιους από τους ανελαστικούς και ελαστικούς περιορισμούς του.

Λύσεις προβλημάτων βελτιστοποίησης.(1)

- **Εφικτή λύση** (feasible solution): ορίζεται η λύση ενός προβλήματος η οποία ικανοποιεί όλους τους **ανελαστικούς περιορισμούς**.

- **Αποτελεσματική - ποιοτική λύση** (efficient solution): ορίζεται η λύση ενός προβλήματος, η οποία είναι εφικτή και επιπλέον ικανοποιεί όσο το δυνατόν περισσότερους από τους **ελαστικούς περιορισμούς** του προβλήματος.

Λύσεις προβλημάτων βελτιστοποίησης.(2)

- Ο βαθμός ικανοποίησης των ελαστικών περιορισμών από μια λύση είναι ευθέως ανάλογος με την ποιότητά της και αποτελεί ένα μέτρο σύγκρισης μεταξύ των αποτελεσματικών λύσεων.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(1)

- Η αναγκαιότητα χρήσης ΗΥ για την επίλυση προβλημάτων είναι αυτονόητη.
- Τα προς επίλυση με ΗΥ προβλήματα, ανάλογα με τον χρόνο επίλυσης, κατατάσσονται σε:
  - Εύκολα: όταν ο χρόνος επίλυσης φράσσεται άνω από ένα πολυώνυμο ως προς το μέγεθος του προβλήματος.
  - Δύσκολα: όταν ο χρόνος επίλυσης ΔΕΝ φράσσεται άνω από ένα πολυώνυμο ως προς το μέγεθος του προβλήματος, αλλά αντίθετα αυξάνει εκθετικά ή και παραγοντικά.
- Στην πρώτη περίπτωση αναφερόμαστε σε πολυωνυμικό χρόνο εκτέλεσης ενώ στην δεύτερη σε τουλάχιστον εκθετικό.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(2)

- Υπάρχουν προβλήματα για την λύση των οποίων έχουν επινοηθεί αλγόριθμοι, των οποίων ο χρόνος εκτέλεσης ακολουθεί μια συνάρτηση πολυωνυμικού τύπου ως προς το μέγεθος του προβλήματος. Παραδείγματα τέτοιων προβλημάτων αποτελούν το πρόβλημα της συντομότερης διαδρομής (***shortest path***) και ελάχιστου ζευγνύοντος δένδρου (***minimum spanning tree***) σε έναν γράφο, ή ακόμη το πρόβλημα της **μέγιστης ροής** σε ένα δίκτυο, κ.ά.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(3)

- Για τα προβλήματα αυτά, ο χρόνος εκτέλεσης των αλγορίθμων που τα επιλύουν φράσσεται από τα άνω με πολυωνυμικές συναρτήσεις, όπως η $\boldsymbol{n}$, $\boldsymbol{n^2}$, ή $\boldsymbol{n^3}$, όπου n είναι το μέγεθος του προβλήματος.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(4)

- Όπως αναφέρουν οι **Dasgupta κ.ά**. όλα τα προηγούμενα προβλήματα, θα μπορούσαν να επιλυθούν θεωρητικά με την εξέταση όλων των εναλλακτικών επιλογών. Για παράδειγμα, k αγόρια μπορούν να αποτελέσουν ζευγάρι με k κορίτσια με $\boldsymbol{k!}$ δυνατούς τρόπους, ενώ ένας γράφος με n κόμβους έχει $\boldsymbol{n^{n-2}}$ ζευγνύοντα δένδρα και ένας τυπικός γράφος έχει εκθετικό αριθμό διαδρομών από τον κόμβο t στον κόμβο s. Όμως, ένας τέτοιος αλγόριθμος, ο οποίος θα εξέταζε όλες τις πιθανές υποψήφιες λύσεις, θα ήταν στην **πράξη άχρηστος**, αφού ο χρόνος επίλυσης, πρακτικά, θα ήταν πολύ μεγάλος.

Dasgupta, S., Papadimitriou, C., & Vazirani, U. (2006). Algorithms. USA: McGraw-Hill Education.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(5)

- Ευτυχώς, όπως αναφέρθηκε στην αρχή της παραγράφου, έχουν επινοηθεί εξυπνότεροι αλγόριθμοι οι οποίοι επιλύουν αυτά τα προβλήματα σε πολυωνυμικό χρόνο. Βεβαίως, δεν υπάρχει η ίδια δυνατότητα για όλα τα προβλήματα.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(6)

- Αν ένα πρόβλημα επιλύεται με ΗΥ σε τουλάχιστον εκθετικό χρόνο, τότε μάλλον δεν υπάρχει ακριβής-ντετερμινιστικός αλγόριθμος επίλυσης αποδεκτού χρόνου. Ο χρόνος επίλυσης τέτοιων προβλημάτων με ακριβή αλγόριθμο ενδέχεται να ισούται με 10 φορές την ηλικία του σύμπαντος ή και περισσότερο (!).
- Υπάρχουν προβλήματα που, αν και επιλύονται σε πολυωνυμικό χρόνο, ο χρόνος επίλυσης μπορεί να ανέλθει σε 10000 χρόνια (!) ή και περισσότερο.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(7)

- Όταν αντιμετωπίζουμε προβλήματα των παραπάνω περιπτώσεων, προφανώς βρισκόμαστε σε πολύ δύσκολη θέση : οι γνωστοί ακριβείς αλγόριθμοι δεν είναι αποτελεσματικοί και ως εκ τούτου είναι άχρηστοι.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(8)

- Δυστυχώς υπάρχουν προβλήματα, τα οποία έχουν έναν εκθετικό αριθμό από πιθανές λύσεις και για τα οποία, παρά τις προσπάθειες των πιο λαμπρών επιστημόνων του χώρου, δεν έχει καταστεί δυνατόν να βρεθεί αλγόριθμος που να τα επιλύει σε πολυωνυμικό χρόνο. Όπως αναφέρεται από τους **Dasgupta κ.ά.**, το πρόβλημα της ικανοποιησιμότητας (***Satisfiability*** **ή** ***SAT*** **πρόβλημα**) παραμένει, υπό την παραπάνω έννοια, άλυτο για **50 χρόνια**, ενώ το **πρόβλημα του Περιοδεύοντος πωλητή** (***TSP***) περιμένει την λύση του για περισσότερο από **100 χρόνια**.

Dasgupta, S., Papadimitriou, C., & Vazirani, U. (2006). Algorithms. USA: McGraw-Hill Education.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(9)

- Επίσης, αν και το πρόβλημα του Γραμμικού Προγραμματισμού (***Linear Programming***) στην γενική του μορφή είναι θεωρητικά και πρακτικά επιλύσιμο σε πολυωνυμικό χρόνο, δεν συμβαίνει το ίδιο αν απαιτήσουμε οι λύσεις του να είναι ακέραιες – οπότε έχουμε το πρόβλημα του Ακέραιου Προγραμματισμού (***Integer Programming***). Σε αυτήν την περίπτωση, το γενικό πρόβλημα του Ακέραιου Προγραμματισμού απαιτεί εκθετικό χρόνο επίλυσης.

Επίλυση με ΗΥ - Εύκολα και δύσκολα προβλήματα.(10)

Παραδείγματα δύσκολων και εύκολων προβλημάτων	
Δύσκολα προβλήματα	Εύκολα προβλήματα
3SAT	2SAT, HORN SAT
TRAVELING SALESMAN PROBLEM	MINIMUM SPANNING TREE
LONGEST PATH	SHORTEST PATH
3D MATCHING	BIPARTITE MATCHING
KNAPSACK	UNARY KNAPSACK
INDEPENDENT SET	INDEPENDENT SET ON TREES
INTEGER LINEAR PROGRAMMING	LINEAR PROGRAMMING
RUDRATA PATH	EULER PATH
BALANCED CUT	MINIMUM CUT

Περισσότερες πληροφορίες για την υπολογιστική πολυπλοκότητα: Papadimitriou, C. H. (1994). Computational Complexity. New Jersey, USA: Pearson Education.

Ή όλα ή τίποτε! Ή μήπως δεν είναι έτσι;…(1)

- Ας υποτεθεί ότι αντιμετωπίζουμε το πρόβλημα του **Περιοδεύοντος πωλητή** (TSP). (ο πωλητής ξεκινάει από μια πόλη, επισκέπτεται όλες τις πόλεις μια μόνο φορά την κάθε μια και επιστρέφει στην πόλη εκκίνησης. Αναζητείται η ελάχιστη διαδρομή, δηλαδή η διαδρομή με την μικρότερη απόσταση). Είναι ένα γνωστό, δύσκολο πρόβλημα, εκθετικού χρόνου επίλυσης. Για ένα αρκετά μεγάλο πλήθος πόλεων, το πρόβλημα είναι πρακτικά άλυτο, αν ως λύση ορίζουμε την εύρεση της ακριβούς διαδρομής με την ελάχιστη απόσταση.

Ή όλα ή τίποτε! Ή μήπως δεν είναι έτσι;…(2)

- Συνεπώς δεν μπορούμε να τα έχουμε όλα.
- **Ερώτηση-προβληματισμός:** αν η εύρεση της ακριβούς διαδρομής για ένα μεγάλο πλήθος πόλεων απαιτεί για παράδειγμα 1000 χρόνια χρόνου εκτέλεσης (!), αλλά πρέπει οπωσδήποτε να δώσουμε κάποια απάντηση, μήπως μπορούμε να συμβιβαστούμε με μια περίπου βέλτιστη λύση, με δεδομένο ένα αποδεκτό σφάλμα, την οποία θα ανακαλύπταμε σε 10 λεπτά;

Υπολογιστική Νοημοσύνη: ένας νέος επιστημονικός κλάδος.(1)

- **Computational Intelligence (CI):** Πρόκειται για έναν κλάδο της Τεχνητής Νοημοσύνης (***Artificial Intelligence*** **–** ***AI***), που όμως έχει αναπτυχθεί σε τέτοιο βαθμό, έτσι ώστε να είναι δύσκολος ο εντοπισμός της συνάφειας μεταξύ *CI* και *AI*.

- Περιλαμβάνει ένα διευρυνόμενο σύνολο στοχαστικών αλγορίθμων, οι οποίοι δίνουν συνήθως καλές προσεγγιστικές λύσεις σε δύσκολα προβλήματα, σε αποδεκτό χρόνο.

Υπολογιστική Νοημοσύνη: ένας νέος επιστημονικός κλάδος.(2)

- Μερικοί από τους αλγορίθμους που περιλαμβάνει είναι τα Ασαφή Συστήματα (***Fuzzy Systems***), τα Νευρωνικά Δίκτυα (***Neural Networks***) και ο Εξελικτικός Προγραμματισμός (***Evolutionary Computation***). Κάποιοι από τους αλγορίθμους της μιμούνται διαδικασίες της φύσης, όπως οι Γενετικοί αλγόριθμοι (**Genetic Algorithms - GA**), Αλγόριθμος σμήνους σωματιδίων (**Particle Swarm Optimization - PSO**), Προσομοιωμένη Ανόπτηση (**Simulated Annealing - SA**), Αλγόριθμος Αποικίας Μυρμηγκιών (**Ant Colony Optimization - ACO**), Αλγόριθμος Αιλουροειδών (**Cat Swarm Optimization -CSO**), Αλγόριθμος Ψαριών (**Fish Swarm Optimization - FSO**), Αλγόριθμος του Μεγάλου Κατακλυσμού (**Great Deluge - GD**), Αλγόριθμος Απαγορευμένης Έρευνας (**Tabu Search**), Αλγόριθμος Μελισσών (**Bee Algorithm**), Αλγόριθμος των Πυροτεχνημάτων (**Fireworks Algorithm**) κ.ά.

Υπολογιστική Νοημοσύνη: ένας νέος επιστημονικός κλάδος.(3)

- Ο όρος Ελαστικός Προγραμματισμός (***Soft Computing***) χρησιμοποιείται συχνά ως εναλλακτικός όρος του όρου Υπολογιστική Νοημοσύνη.
- Τα πεδία εφαρμογής των μεθόδων της CI συνεχώς διευρύνονται και περιλαμβάνουν:
  - Λήψη Αποφάσεων
  - Ταξινόμηση (*Classification*)
  - Ηλεκτρονικές συσκευές καταναλωτών
  - Πρόβλεψη Χρονοσειρών
  - Συνδυαστική βελτιστοποίηση
  - Ιατρική, Βίο-ιατρική και Βίο-πληροφορική
  - Video Games

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(1)

- Χώρος έρευνας (**search space**): το ευρύτερο σύνολο μέσα από το οποίο αναζητάμε λύσεις του προβλήματος.
- (ενδιάμεση – μη τελική) λύση προβλήματος: κάθε απάντηση, όσο κακή και να είναι, στο πρόβλημα, η οποία παρέχεται από έναν αλγόριθμο (πριν τον τερματισμό του) ονομάζεται λύση του προβλήματος.
- Γείτονας (**neighbor**): με δεδομένη μια (ενδιάμεση – μη τελική) λύση του προβλήματος, κάθε αλλαγή-μετασχηματισμός επί της λύσης παράγει μια άλλη υποψήφια λύση, που καλείται «γείτονας» της αρχικής.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(2)

- Γειτονιά (**neighborhood**): το σύνολο των γειτόνων μιας λύσης αποτελούν μια γειτονιά. Ο όρος γειτονιά χρησιμοποιείται επίσης για να περιγράψει τον μηχανισμό ή την διαδικασία-μετασχηματισμό η οποία παράγει τους γείτονες μιας λύσης.
- Συνάρτηση ευρωστίας (**fitness function**): είναι μια συνάρτηση η οποία αντιστοιχίζει στις τιμές μιας ή περισσότερων μεταβλητών του προβλήματος έναν πραγματικό αριθμό, ο οποίος διαισθητικά αντιπροσωπεύει το «κόστος» μιας υποψήφιας λύσης. Συχνά αναφέρεται και ως «συνάρτηση κόστους» (**cost function**)

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(3)

- Σύγκριση υποψήφιων λύσεων και επιλογή: σε ένα πρόβλημα βελτιστοποίησης-ελαχιστοποίησης, μεταξύ δύο ή περισσότερων υποψηφίων λύσεων, καλύτερη είναι αυτή στην οποία αντιστοιχεί η μικρότερη τιμή της fitness function.
- Γενεά (**generation**): οι αλγόριθμοι της CI είναι επαναληπτικοί. Αυτό σημαίνει ότι εκτελούν έναν συγκεκριμένο αριθμό καλά καθορισμένων βημάτων (εντολών) επαναληπτικά, με την βοήθεια κάποιου βρόχου (loop). Κάθε κύκλος του βρόχου λέγεται «γενεά».

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(4)

- **Γενική Αρχή λειτουργίας αλγορίθμων CI:** αυτοί οι αλγόριθμοι τρέχουν για έναν αριθμό γενεών. Η εκκίνηση γίνεται με μια αρχική λύση ή ένα σύνολο αρχικών λύσεων (συνήθως τυχαίων και κακών, δηλαδή με μεγάλη τιμή fitness). Σε κάθε γενεά, εφαρμόζονται οι λειτουργίες του αλγορίθμου και οι λύσεις της τρέχουσας γενεάς μετασχηματίζονται, παράγοντας άλλες λύσεις. Από αυτές επιλέγουμε τις καλύτερες ή την καλύτερη και τις διοχετεύουμε στην επόμενη γενεά. Η ίδια διαδικασία επαναλαμβάνεται και συνήθως μετά το πέρας του αλγορίθμου το αποτέλεσμα είναι η τελική λύση του προβλήματος.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(5)

- Κριτήριο τερματισμού αλγορίθμου (**termination criterion**): ο αλγόριθμος τερματίζεται συνήθως όταν ικανοποιηθεί ένα από τα ακόλουθα κριτήρια:
  - Ο χρόνος εκτέλεσης έχει εξισωθεί με ένα προκαθορισμένο χρονικό όριο.
  - Ο αριθμός των γενεών που εκτελέστηκαν έχει εξισωθεί με ένα προκαθορισμένο πλήθος γενεών.
  - Ο αλγόριθμος έτρεξε για ένα συγκεκριμένο αριθμό γενεών χωρίς να μπορέσει να βελτιώσει την καλύτερη λύση που είχε ευρεθεί, δηλαδή έτρεξε χωρίς καμία ουσιαστική αλλαγή-βελτίωση.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(6)

- Τεχνική της αντιμετάθεσης (**swap**): κοινή τεχνική, εμφανιζόμενη σε όλους τους αλγορίθμους επίλυσης προβλημάτων συνδυαστικής βελτιστοποίησης. Σε τέτοια προβλήματα, κάθε λύση αναπαρίσταται συχνά με ένα διάνυσμα (**array-vector**) ή έναν πίνακα (**matrix**). Προκειμένου από μια λύση να παραχθεί ένας γείτονας, επιλέγονται (συνήθως) δύο κελιά, σύμφωνα με τα όσα καθορίζει η γειτονιά (δηλαδή ο μηχανισμός του μετασχηματισμού της λύσης) και τα κελιά ανταλλάσσουν το περιεχόμενό τους.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(7)

- Τοπική αναζήτηση (**local search**): από γενεά σε γενεά, στην τρέχουσα λύση εφαρμόζεται ένας μετασχηματισμός τοπικού χαρακτήρα, με σκοπό την παραγωγή ενός γείτονα. Αν ο γείτονας έχει καλύτερη (μικρότερη) τιμή fitness από την τρέχουσα λύση, τότε παίρνει αυτός τη θέση της καλύτερης τρέχουσας λύσης. Η διαδικασία επαναλαμβάνεται.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(8)

- Διαφοροποίηση έναντι εντατικοποίησης (**diversification vs intensification**): ένας ικανοποιητικός αλγόριθμος βελτιστοποίησης οφείλει στις πρώτες γενεές να «ψάχνει» σε μια ευρεία περιοχή του χώρου έρευνας (προφανώς δεν μπορεί να ψάξει σε κάθε σημείο του χώρου έρευνας). Αυτή η ιδιότητα λέγεται «**διαφοροποίηση**».

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(9)

- Αντίθετα, προς το τέλος του, οφείλει να εκλεπτύνει (**refine**) την τρέχουσα λύση, με αποτέλεσμα να την κάνει ακόμη καλύτερη. Η ιδιότητα αυτή λέγεται «**εντατικοποίηση**».
- Ένας καλός αλγόριθμος διαθέτει και τις δύο ιδιότητες στην ιδανική αναλογία. Προφανώς, ένας αλγόριθμος που δίνει υπέρμετρη βαρύτητα στην διαφοροποίηση δεν θα βρει σχεδόν ποτέ την καλή λύση. Αντίθετα, ένας αλγόριθμος με υπέρμετρη βαρύτητα στην εντατικοποίηση, θα εκλεπτύνει ενδεχομένως μια κακή λύση, η οποία εκ των πραγμάτων ίσως δεν μπορεί να γίνει πολύ καλή.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(10)

- Ολικό βέλτιστο και τοπικά βέλτιστα (**global and local optima**): ολικό βέλτιστο είναι η αληθινά καθολική βέλτιστη λύση ενός προβλήματος βελτιστοποίησης, η οποία, για τους λόγους που αναφέρθηκαν, σε ένα δύσκολο πρόβλημα πολύ σπάνια επιτυγχάνεται από τον αλγόριθμο της CI. Ο χώρος έρευνας ενός προβλήματος, εκτός από το ένα ή περισσότερα ολικά βέλτιστα, συνήθως έχει και λίγα ή αρκετά τοπικά βέλτιστα (ακρότατα). Αυτά είναι οι βέλτιστες λύσεις σε μια υπό-περιοχή του χώρου έρευνας.

Γενική και συνοπτική περιγραφή κάποιων βασικών εννοιών και τεχνικών της ΥΝ.(11)

- Εγκλωβισμός σε τοπικό ακρότατο - πρόωρη σύγκλιση (**premature convergence**): πρόκειται για την περίπτωση κατά την οποία μια λύση ταυτίζεται με ένα τοπικό ακρότατο, αλλά οι μηχανισμοί μετασχηματισμού λύσης που διαθέτει ο αλγόριθμος δεν είναι ικανοί να την μετασχηματίσουν αρκετά ώστε να μπορέσει ο αλγόριθμος να «ψάξει» σε άλλο σημείο, με σκοπό τον εντοπισμό μιας καλύτερης λύσης από αυτήν του προηγούμενου τοπικού ακρότατου. Αρκετοί αλγόριθμοι έχουν αυτό το αδύνατο σημείο, αφού σε πολλές περιπτώσεις εγκλωβίζονται και συγκλίνουν πρόωρα σε μια μη ικανοποιητική τοπικά σχεδόν βέλτιστη λύση.

Η τεχνική Hill climbing: εκ των ων ουκ άνευ της ΥΝ.

- Τεχνική ανάβασης λόφου (**Hill climbing**): εφαρμόζεται σε προβλήματα βελτιστοποίησης συχνότατα. Αν και σε ένα πρόβλημα ελαχιστοποίησης ο όρος που θα ταίριαζε καλύτερα θα ήταν «κατάβαση» λόφου, έχει επικρατήσει ο αρχικός όρος. Ανήκει στην οικογένεια των αλγορίθμων τοπικής αναζήτησης. Η τεχνική συνίσταται στην εκκίνηση από μια τυχαία επιλεγμένη λύση (ανεξάρτητα από την ποιότητά της) και στην εφαρμογή επί αυτής μικρών διαφοροποιήσεων, δηλαδή την παραγωγή ελαφρώς διαφοροποιημένων γειτόνων. Ένας γείτονας αντικαθιστά την τρέχουσα λύση εφόσον δεν είναι χειρότερος από αυτήν, όταν δηλαδή η τιμή fitness είναι μικρότερη ή ακόμη και ίση με αυτήν της τρέχουσας λύσης.

Ένα ενδιαφέρον ζήτημα της τεχνικής Hill climbing!

- **Πεντάλεπτος προβληματισμός:** είναι δυνατόν δύο διαφορετικές λύσεις να έχουν την ίδια τιμή fitness; Τι σκοπό εξυπηρετεί στην τεχνική **Hill climbing** η αποδοχή μιας λύσης η οποία έχει την ίδια τιμή fitness και όχι αναγκαστικά μικρότερη;

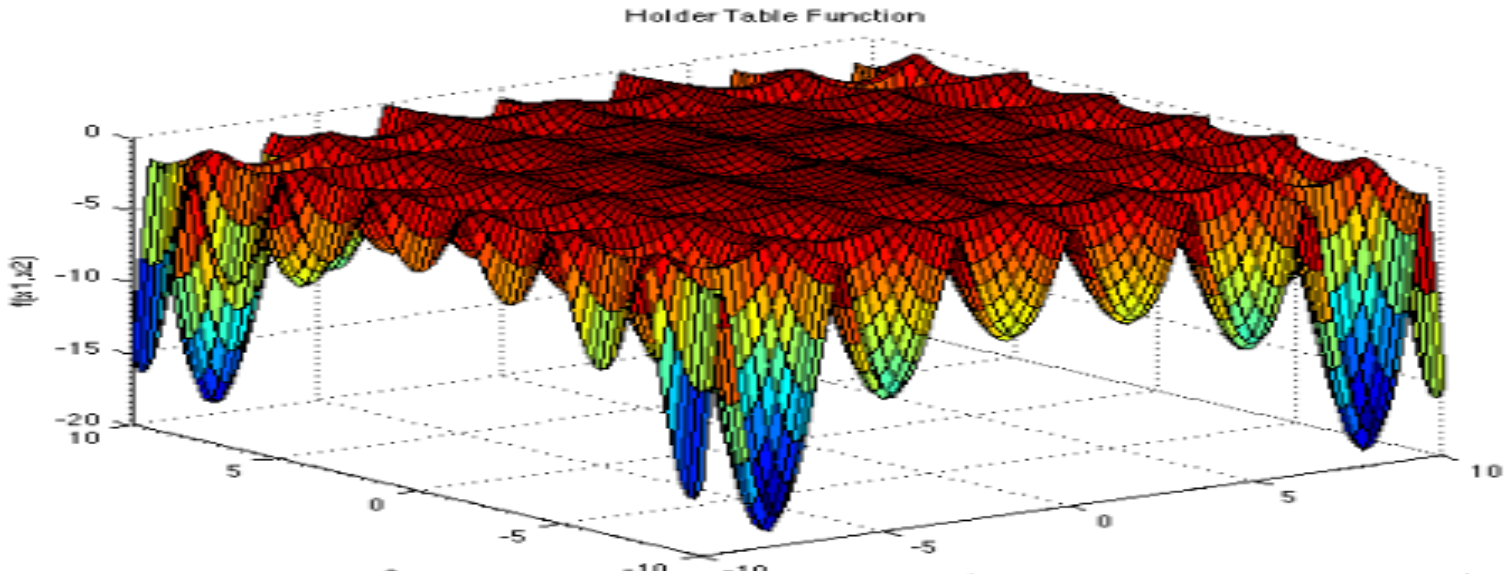
Test functions: Ορισμός-χρησιμότητα

- **Ορισμός:** πρόκειται για μαθηματικές συναρτήσεις, συνήθως περισσότερων της μιας μεταβλητής, οι οποίες είναι ιδιαίτερα πολύπλοκες. Θα πρέπει να εμφανίζουν κάποια ιδιομορφία, όπως:
  - Πολλά ολικά ακρότατα
  - Πολλά τοπικά ακρότατα, ομοιόμορφα ή μη ομοιόμορφα κατανεμημένα
  - Διάφορες ιδιαιτερότητες στο «τοπίο» τους.

- **Χρησιμότητα:** εκτός από το καθαρά Μαθηματικό ενδιαφέρον τους, μπορούν να χρησιμοποιηθούν για να πιστοποιηθεί το πόσο καλός είναι ένας αλγόριθμος βελτιστοποίησης και για την συγκριτική μελέτη απόδοσης περισσοτέρων αλγορίθμων.
  - Αν ένας αλγόριθμος προσεγγίζει το ολικό βέλτιστο της συνάρτησης με ικανοποιητική ακρίβεια, τότε έχουμε την ένδειξη ότι είναι καλά σχεδιασμένος.

Παραδείγματα test optimization functions (1)

- Συνάρτηση Holder table με 2 διαστάσεις όπου ο χώρος έρευνας έχει πολλά τοπικά βέλτιστα και 4 ολικά ελάχιστα.



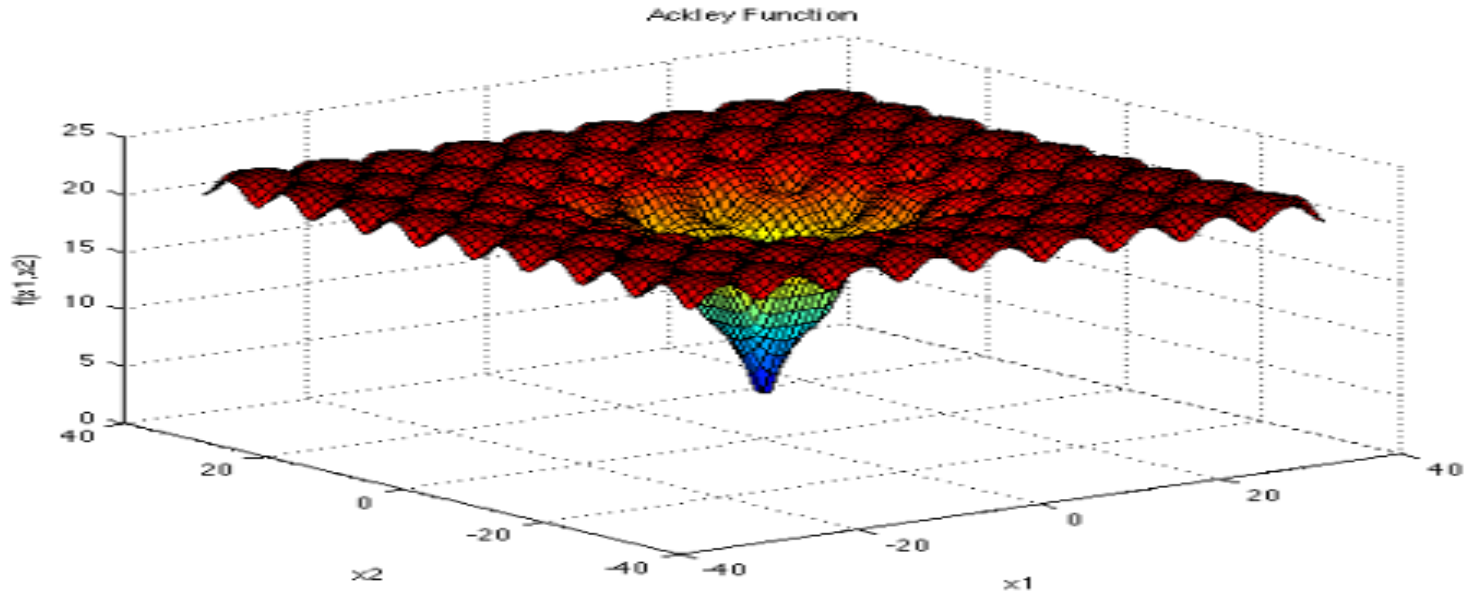
$$f(\mathbf{x}) = -\left| \sin(x_1)\cos(x_2)\exp\left(\left|1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right|\right)\right|$$

Global Minimum:

$f(\mathbf{x}^*) = -19.2085$, at $\mathbf{x}^* = (8.05502, 9.66459), (8.05502, -9.66459), (-8.05502, 9.66459)$ and $(-8.05502, -9.66459)$

Παραδείγματα test optimization functions (2)

- Συνάρτηση Ackley, με «κοιλάδα» και πολλά τοπικά ακρότατα και ένα ολικό ελάχιστο στο κέντρο. Ιδιαίτερα δύσκολη για Hill-Climbing.



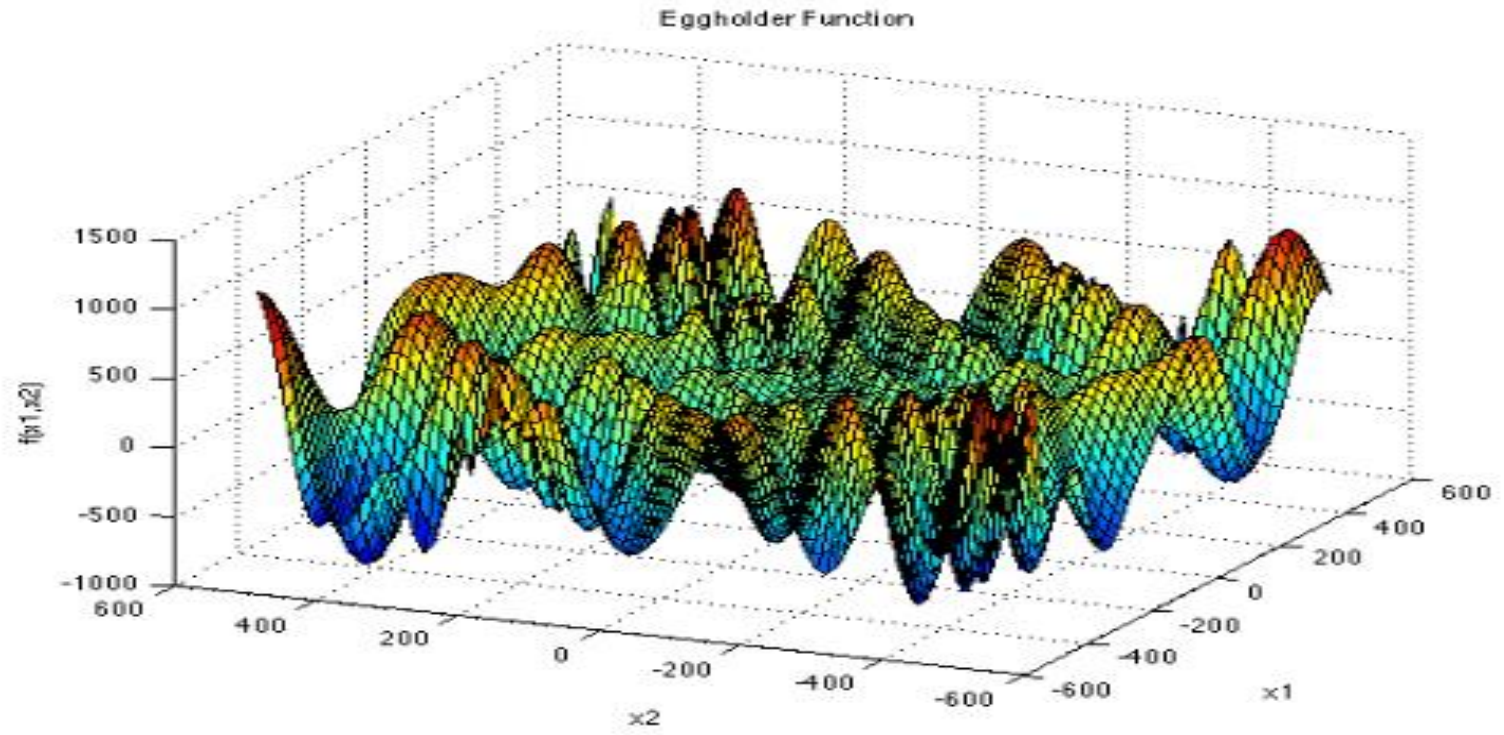
$$f(\mathbf{x}) = -a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right) + a + \exp(1)$$

Global Minimum:

$f(\mathbf{x}^*) = 0$, at $\mathbf{x}^* = (0, \ldots, 0)$

Παραδείγματα test optimization functions (3)

Συνάρτηση Egg-holder. Ιδιαίτερα δύσκολη λόγω των πολλών τοπικών ακροτάτων.



$$f(\mathbf{x}) = -(x_2 + 47)\sin\left(\sqrt{\left|x_2 + \frac{x_1}{2} + 47\right|}\right) - x_1\sin\left(\sqrt{|x_1 - (x_2 + 47)|}\right)$$

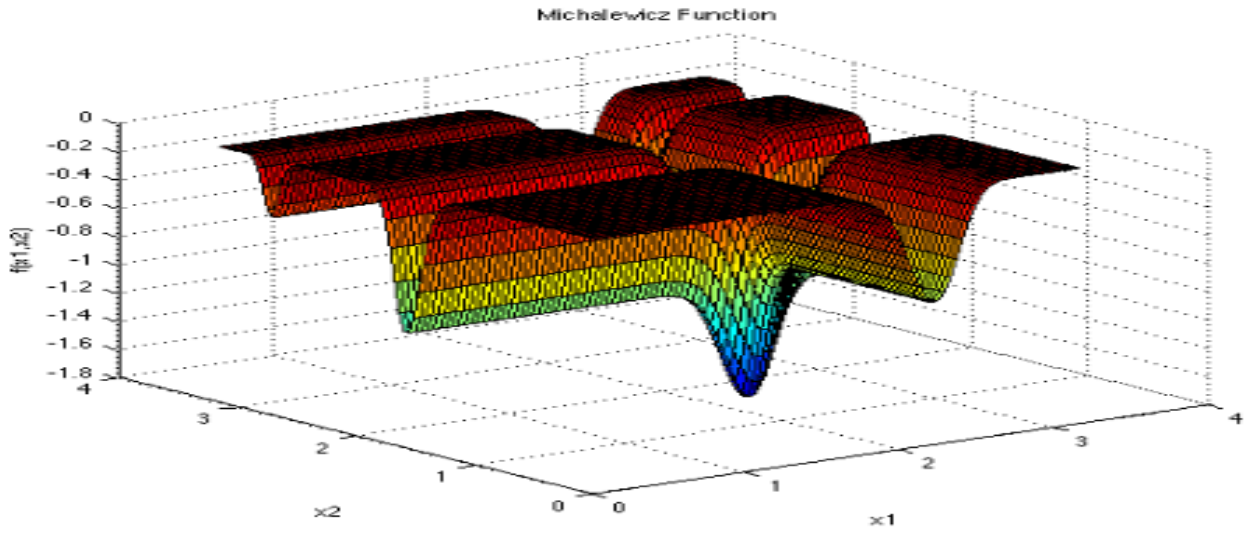
Global Minimum:

$f(\mathbf{x}^*) = -959.6407$, at $\mathbf{x}^* = (512, 404.2319)$

Παραδείγματα test optimization functions (4)

- Συνάρτηση Michalewicz.Επίπεδα τοπία με οξύ τοπικό ακρότατο.

MICHALEWICZ FUNCTION



$$f(\mathbf{x}) = -\sum_{i=1}^{d} \sin(x_i) \sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$$

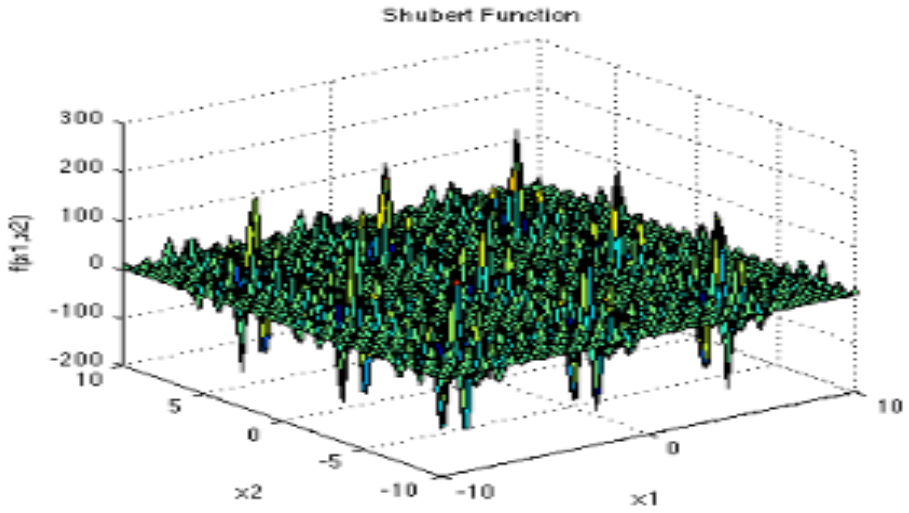
Global Minima:

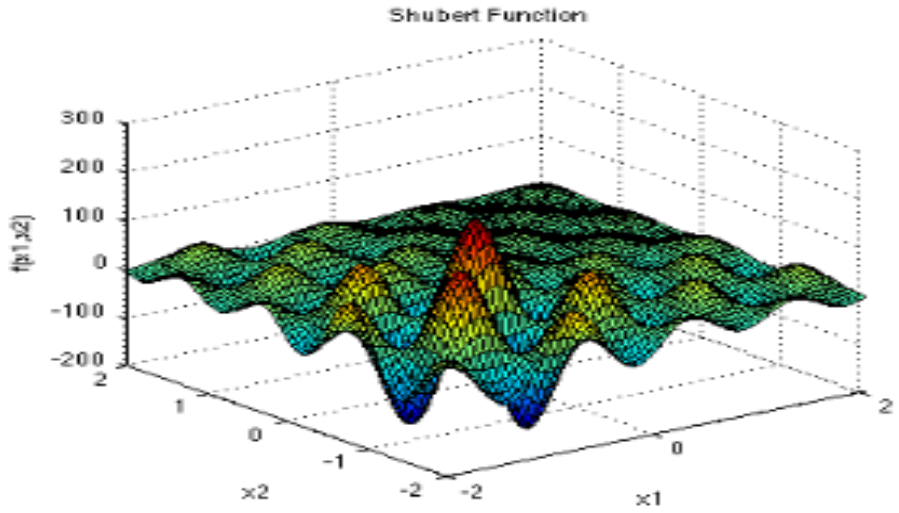
at $d = 2$: $f(\mathbf{x}^*) = -1.8013$, at $\mathbf{x}^* = (2.20, 1.57)$
at $d = 5$: $f(\mathbf{x}^*) = -4.687658$
at $d = 10$: $f(\mathbf{x}^*) = -9.66015$

Παραδείγματα test optimization functions (5)

- Συνάρτηση Shubert. Πολλά τοπικά ακρότατα.

SHUBERT FUNCTION





$$f(\mathbf{x}) = \left(\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right)$$

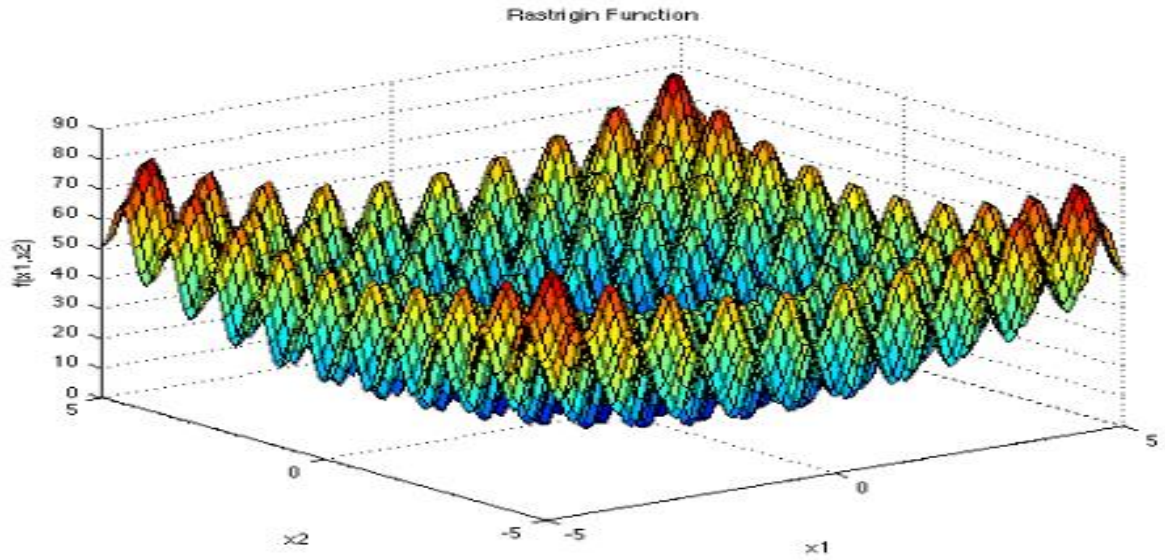
Global Minimum:

$$f(\mathbf{x}^*) = -186.7309$$

Παραδείγματα test optimization functions (6)

- Συνάρτηση **RASTRIGIN:** πολλά τοπικά ακρότατα, ομοιόμορφα κατανεμημένα.

RASTRIGIN FUNCTION



$$f(\mathbf{x}) = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)]$$

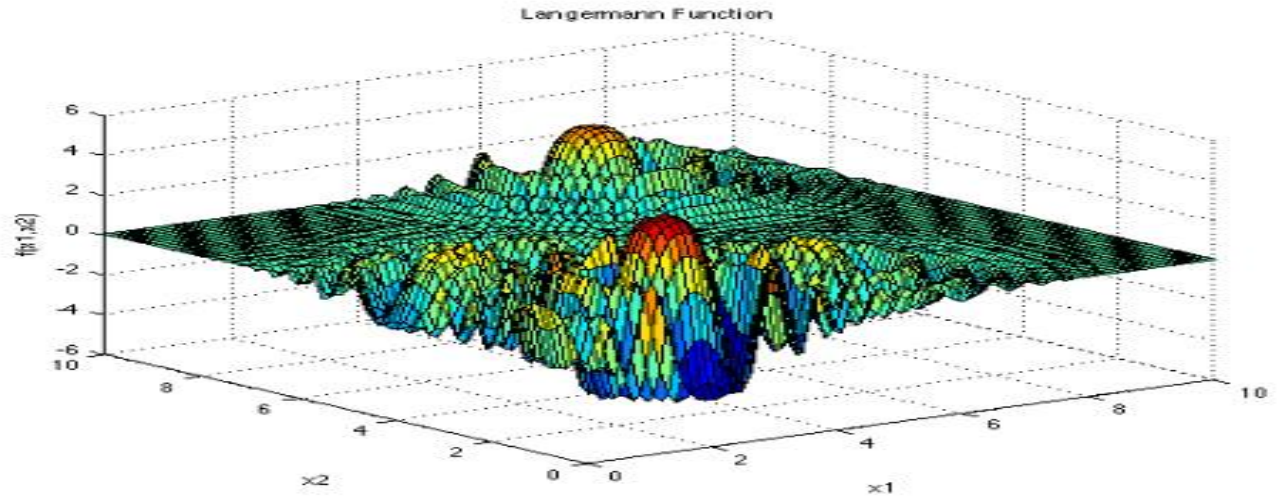
Global Minimum:

$f(\mathbf{x}^*) = 0$, at $\mathbf{x}^* = (0, \ldots, 0)$

Παραδείγματα test optimization functions (7)

- Συνάρτηση **LANGERMANN:** πολλά τοπικά ακρότατα **μη ομοιόμορφα** κατανεμημένα.

LANGERMANN FUNCTION



$$f(\mathbf{x}) = \sum_{i=1}^{m} c_i \exp\left(-\frac{1}{\pi}\sum_{j=1}^{d}(x_j - A_{ij})^2\right)\cos\left(\pi\sum_{j=1}^{d}(x_j - A_{ij})^2\right)$$

values of m, **c** and **A**, as given by Molga & Smutnicki (2005) are (for d = 2): m = 5, **c** = (1, 2, 5, 2, 3) and:

$$\mathbf{A} = \begin{pmatrix} 3 & 5 \\ 5 & 2 \\ 2 & 1 \\ 1 & 4 \\ 7 & 9 \end{pmatrix}$$

Αλγόριθμοι ΥΝ: αιτιοκρατικοί (ντετερμινιστικοί) ή στοχαστικοί (πιθανοτικοί); (1)

- Οι αλγόριθμοι της ΥΝ, επί της ουσίας, λειτουργούν με δειγματοληψία του χώρου έρευνας.
- Ερευνούν ένα υποσύνολο του συνόλου των λύσεων, «ελπίζοντας» να βρουν την σχεδόν βέλτιστη.
- Στην δειγματοληψία, η οποία είναι «στοχευμένη», έγκειται μερικώς η επιτυχία των εν λόγω αλγορίθμων.

Αλγόριθμοι ΥΝ: αιτιοκρατικοί (ντετερμινιστικοί) ή στοχαστικοί (πιθανοτικοί); (2)

- Σε κάθε σημείο του αλγορίθμου, όπου πρόκειται να ληφθεί μια απόφαση, αυτή λαμβάνεται με τυχαίο τρόπο ή με κάποια πιθανότητα. **Άρα είναι στοχαστικοί αλγόριθμοι.**
- Η σημασία της χρησιμοποιούμενης ακολουθίας τυχαίων αριθμών, κατά την διάρκεια εκτέλεσης του αλγορίθμου, είναι καθοριστική.
- Με την χρήση «πραγματικά» τυχαίων αριθμών, που ανήκουν σε δύο διαφορετικές ακολουθίες, σε διαδοχικές εκτελέσεις του αλγορίθμου, η τελική λύση είναι συνήθως διαφορετική.

Πώς αξιολογούμε έναν αλγόριθμο της ΥΝ. (1)

- Συνήθως εκτελούμε 20, 30 ή 100 φορές (πειράματα) τον αλγόριθμο, με διαφορετική κάθε φορά ακολουθία τυχαίων αριθμών. Όσον αφορά στην **τιμή fitness** της τελικής λύσης κάθε πειράματος και στον **χρόνο εκτέλεσης** του κάθε πειράματος:
- **Καταγράφουμε:**
  - το καλύτερο αποτέλεσμα των πειραμάτων(best),
  - το χειρότερο (worst).
- **Υπολογίζουμε:**
  - την μέση τιμή τους(average),
  - την τυπική απόκλιση (STD)
  - το συντελεστή μεταβλητότητας (CV)

Πώς αξιολογούμε έναν αλγόριθμο της ΥΝ. (2)

- **Πεντάλεπτος προβληματισμός:**
- Στο ίδιο πρόβλημα ελαχιστοποίησης, με γνωστό εκ των προτέρων ελάχιστο ίσο με το 0, σε 100 πειράματα, ένας αλγόριθμος Α πέτυχε :
- Best_A = 0.000001 (e -6)
- Worst_A = 1
- Average_A = 0.001.(e -3)

- Ένας άλλος αλγόριθμος Β, πέτυχε σε 20 πειράματα:
- Best_B= 0.00001 (e-5)
- Worst_B = 1
- Average_B = 0.0001. (e-4)
- Ποιόν αλγόριθμο θα επιλέγατε ως καλύτερο για την επίλυση του συγκεκριμένου προβλήματος και γιατί;

Ολοκλήρωση πρώτης διάλεξης.

- Στην επόμενη διάλεξη:

Γενετικοί αλγόριθμοι (GA) : Από τον Δαρβίνο (1859) στον J. Holland (1970).

(Ένα ταξίδι στον υπέροχο κόσμο της επιλογής, της διασταύρωσης και της μετάλλαξης).

Σας ευχαριστώ για την προσοχή σας.