Τεχνικές Βελτιστοποίησης

Project

Γενετικοί Αλγόριθμοι

Δεκέμβριος 2025

Ρεπάνης Γεώργιος-Δημήτριος

ΑΕΜ: 10588

Γενική Περιγραφή project:

Στην παρούσα εργασία μελετάμε την ταυτοποίηση ενός στατικού, μη γραμμικού συστήματος με δύο εισόδους u_1, u_2 και μια έξοδο y όπου $y = f(u_1, u_2)$.Η συνάρτηση f θεωρείται συνεχής και άγνωστη ενώ στόχος μας είναι να μοντελοποιήσουμε την f με μια αναλυτική μορφή χαμηλής πολυπλοκότητας χρησιμοποιώντας μετρήσεις εισόδου-εξόδου σαν δεδομένα εκπαίδευσης του αλγορίθμου.

Για την προσέγγιση αυτή χρησιμοποιούμε ένα μοντέλο συναρτήσεων βάσης, όπου η $\hat{f}(u_1, u_2)$ εκφράζεται από έναν γραμμικό συνδυασμό κάποιων Gaussianών συναρτήσεων. Κάθε γκαουσιανή έχει ένα κέντρο (c_1, c_2) και διασπορές (σ_1, σ_2) και τα βάρη του συνολικού μοντέλου w_i. Έτσι θα καταλήξουμε στην αναζήτηση των παραμέτρων για την επίλυση του προβλήματος, οι οποίες παράμετροι θα ελαχιστοποιούν το σφάλμα πρόβλεψης της $\hat{f}$ ως προς τις μετρήσεις.

Η βελτιστοποίηση των παραμέτρων θα γίνει με Γενετικό Αλγόριθμο (GA) που υλοποιείται από το μηδέν χωρίς έτοιμες συναρτήσεις Matlab όπως ga(), ώστε να διερευνήσουμε την αποτελεσματικότητα των εξελικτικών μεθόδων στο συγκεκριμένο μη γραμμικό, παραμετροποιημένο πρόβλημα. Η ποιότητα της προσεγγιστικής αναλυτικής έκφρασης θα την αξιολογήσουμε σε ξεχωριστό σύνολο δεδομένων από αυτό που θα χρησιμοποιηθεί για την εκπαίδευση, ώστε να ελέγξουμε την γενίκευση και να αποφύγουμε την υπερπροσαρμογή.

Διατύπωση δεδομένων

Έχουμε ένα σύστημα με δύο εισόδους u_1, u_2 και μία έξοδο y. Το σύστημα είναι στατικό δηλαδή η έξοδος δεν εξαρτάται από τον χρόνο άλλα μόνο από τις τρέχουσες τιμές των εισόδων. Δηλαδή:

$$y = f(u_1, u_2)$$

Η f είναι άγνωστη και συνεχής στο:

$$u_1 \in [-1,2], u_2 \in [-2,1]$$

Για να κάνουμε train την f, φτιάχνω ένα σύνολο δεδομένων:

$\{u_{1,k}, u_{2,k}, y_k\}$, με k=1,2,...N

Δηλαδή για κάθε δείγμα k:
Βάζουμε μία τιμή $u_{1,k}$ και μία τιμή $u_{2,k}$ και υπολογίζουμε την έξοδο y_k.

Διαχωρισμός train/set:
Χρησιμοποιούμε δύο ανεξάρτητα σύνολα δεδομένων:

- Train set: $N_{train} = 1000$ δείγματα (χρησιμοποιείται για εκπαίδευση)
- Test set: $N_{test} = 1000$ δείγματα (χρησιμοποιείται μόνο για αξιολόγηση).

Δίνεται η συνάρτηση:

$$f(u_1,u_2)=\sin(u_1+u_2)\sin(u_2^2)$$

και άρα με βάση αυτά που ορίζουμε

$$y_k=\sin(u_{1,k}+u_{2,k})\sin(u_{2,k}^2)$$

Εισάγω μια προσεγγιστική συνάρτηση $\hat{f}(u_1,u_2)$ που να προσεγγίζει στην πραγματική f.
Δηλαδή:

$\hat{f}(u_{1,k},u_{2,k})\approx y_k$ για όλα τα δείγματα train

και στη συνέχεια να δουλεύει σωστά και σε νέα σημεία test set.
Για να το κάνουμε αυτό θα μετατρέψουμε το πρόβλημα σε πρόβλημα βελτιστοποίησης εισάγοντας αρχικά τις παραμέτρους θ του μοντέλου που ελαχιστοποιούν το σφάλμα.

Ορίζω λοιπόν σφάλματα για τα δείγματα:

$$e_k=y_k-\hat{f}(u_{1,k},u_{2,k},\theta)$$

και ένα κριτήριο για το θ – Συνάρτηση Κόστους:

$$J(\theta)=MSE_{train}=\frac{1}{N_{train}}\sum_{k=1}^{N_{train}}e_k^2$$

Όπου θα βρούμε τα θ ώστε να ελαχιστοποιείται το J. Δηλαδή:

$$\theta^{\dot{\iota}}=arg\,min_\theta J(\theta)$$

που θα λυθεί με Γενετικό Αλγόριθμο.
Μετά την ολοκλήρωση της βελτιστοποίησης, αξιολογούμε την ποιότητα του τελικού μοντέλου με:

- MSE_{train} (προσαρμογή στο train)
- MSE_{test} (γενίκευση σε νέα δεδομένα)

$$MSE_{test}=\frac{1}{N_{test}}\sum_{k=1}^{N_{test}}e^2$$

Για αναπαραγωγιμότητα χρησιμοποιούμε rng(1) πριν τη δημιουργία δεδομένων.
Στην υλοποίηση ο GA είναι cost-based και σε κάθε επιλογή/σύγκριση χρησιμοποιείται το μικρότερο J.

Επιλογή Gaussian βάσης

Η βάση που μας δίνεται από την εκφώνηση είναι:

$$G(u_1,u_2)=\exp\left(-\left(\frac{(u_1-c_1)^2}{2\sigma_1^2}+\frac{(u_2-c_2)^2}{2\sigma_2^2}\right)\right)$$

$c_{1,2}$: είναι το κέντρο της καμπάνας στο επίπεδο
$\sigma_{1,2}$: είναι οι διασπορές πόσο απλωμένη είναι η καμπάνα σε κάθε άξονα
Το G παίρνει τιμές από 0 έως 1 με κορυφή κοντά στο 1 που είναι και το κέντρο.

Το μοντέλο μας θα είναι της μορφής ενός αθροίσματος από M Gaussians με $M \leq 15$ και βάρη w_i. Δηλαδή:

$$\hat{f}(u_1, u_2) = \sum_{i=1}^{M} w_i G_i(u_1, u_2)$$

όπου

$$G(u_1, u_2) = \exp\left(-\left(\frac{(u_1 - c_{1,i})^2}{2\sigma_{1,i}^2} + \frac{(u_2 - c_{2,i})^2}{2\sigma_{2,i}^2}\right)\right)$$

Άρα κάθε Gaussian πιάνει μια περιοχή του χώρου (u_1, u_2) και με βάση τα βάρη φτιάχνω το τελικό σχήμα.

Σε αυτή την παραμετροποίηση οι άγνωστοι παράμετροι είναι το βάρος w_i, το κέντρο $c_{1,i}, c_{2,i}$ και οι διασπορές $\sigma_{1,i}, \sigma_{2,i}$, οπότε:

$$\theta = \{w_i, c_{1,i}, c_{2,i}, \sigma_{1,i}, \sigma_{2,i}\} \text{ με i=1,2,...M=15}$$

Δηλαδή το μοντέλο μας έχει 5 παραμέτρους και για 15 Gaussians θα έχω συνολικά 15x5=75 παραμέτρους. Ο GA δηλαδή θα έχει να ψάξει μια λύση με 75 παραμέτρους.

Για χαμηλή πολυπλοκότητα βέβαια όπως θα δούμε και στη συνέχεια μπορεί να αφήνουμε κάποια βάρη w_i κοντά στο μηδέν και να μην συμμετέχουν στην εύρεση της λύσης δηλαδή να έχω λιγότερες από 15 ενεργές βάσεις.

Περιορισμοί για να μην ξεφεύγει η GA:

Domain εισόδων:

$$u_1 \in [-1,2], u_2 \in [-2,1]$$

Τα κέντρα :

$$c_{1,i} \in [-1,2], c_{2,i} \in [-2,1]$$

Οι διασπορές:

$$\sigma_{1,i} \in [0.1, 1.5], \sigma_{2,i} \in [0.1, 1.5]$$

Βάροι:

$$w_i \in [-2, 2]$$

Μετά από κάθε mutation εφαρμόζεται **clamping** ώστε κάθε γονίδιο να επιστρέφει εντός [lb,ub].

Για κάθε βήμα k:

$$\hat{y}_k = \hat{f}(u_{1,k}, u_{2,k}, \theta)$$

και

$$J(\theta) = \frac{1}{N} \sum_{k=1}^{N} (y_k - \hat{y}_k)^2$$

και ψάχνω το θ δηλαδή όλα τα w,c,σ που θα κάνει το J(θ) μικρό.

Κόστος και όρια παραμέτρων

Για κάθε δείγμα εκπαίδευσης του μοντέλου μας k=1,…,Ntrain.

- Πραγματική έξοδος: y_k
- Πρόβλεψη Μοντέλου: $\hat{y}_k = \hat{f}(u_{1,k}, u_{2,k}, x)$

Σφάλμα:

$$e_k = y_k - \hat{y}_k$$

Κόστος:

$$J(x) = MSE_{train} = \frac{1}{N_{train}} \sum_{k=1}^{N_{train}} e_k^2$$

Χαμηλή Πολυπλοκότητα:
Θα κάνουμε τον αλγόριθμο να μην αξιοποιεί άσκοπα και τις 15 Gaussians. Αυτό θα το κάνουμε προσθέτοντας ένα ακόμη άθροισμα-penalty όπου αν κάποια βάρη w_i είναι κοντά στο μηδέν η Gaussian να σβήνει. Κρατάμε το λ μικρό 10^{-4} μέχρι 10^{-2}.

$$J_\lambda(\theta) = \frac{1}{N} \sum_{k=1}^{N} e_k^2 + \lambda \sum_{i=1}^{M} |w_i|$$

Στον κώδικα έχουμε penalty υλοποιημένο (L1), αλλά στην main είναι usePenalty=false. Έχει υλοποιηθεί δηλαδή αλλά δεν ενεργοποιείται.
Στα αποτελέσματα που παρουσιάζονται, το cost είναι J=MSEtrain χωρίς penalty.

Κωδικοποίηση παραμέτρων θ σε χρωμόσωμα

Θα ορίσω πως ο Γενετικός Αλγόριθμος θα συγκρατεί μια λύση δηλαδή πώς θα γράφω όλες τις παραμέτρους θ σε ένα διάνυσμα αριθμών (χρωμόσωμα) ώστε να μπορεί να κάνει crossover/mutation.

Επιλογή M
Η εκφώνηση λέει έως 15 Gaussians παίρνω δηλαδή σταθερό M = 15 που θα είναι το μήκος του χρωμοσώματος.

Τι περιέχει μια Gaussian βάση
Για κάθε Gaussian i έχω 5 τιμές:

$$[w_i, c_{1,i}, c_{2,i}, \sigma_{1,i}, \sigma_{2,i}]$$

Άρα για M = 15 Gaussian έχω συνολικά :

5M = 75 γονίδια

Χρωμόσωμα:
Το χρωμόσωμα είναι ένα διάνυσμα :

$$x = [w_1, c_{1,1}, c_{2,1}, \sigma_{1,1}, \sigma_{2,1}, w_2, c_{1,2}, c_{2,2}, \sigma_{1,2}, \sigma_{2,2}, \ldots, w_{15}, c_{1,15}, c_{2,15}, \sigma_{1,15}, \sigma_{2,15}] \in \Re^{75}$$

και βάζω πακέτα των 5 το ένα μετά το άλλο.(Η αποκωδικοποίηση του θα γίνει στον κώδικα με την συνάρτηση decodeChromosome.)

Όρια ανά γονίδιο και clamping
Για να φτιάξω τον αρχικό πληθυσμό και πως διορθώνω μετά από mutation έχω τα παραπάνω όρια για τις παραμέτρους $w_i, c_{1,i}, c_{2,i}, \sigma_{1,i}, \sigma_{2,i}$ ώστε κάθε φορά που αλλάζω ένα γονίδιο να το κρατάω μέσα στα όρια:

$$x_j \leftarrow min\left(max\left(x_j, lowerbound_j\right), upperbound_j\right), j=1,\ldots,75$$

Αξιολόγηση χρωμοσώματος (cost)
Για κάθε χρωμόσωμα x υπολογίζουμε τις προβλέψεις $\hat{y}=\hat{f}\left(u_{1,k}, u_{2,k}, x\right)$ για όλα τα δείγματα train και το cost :

$$J(x)=MSE_{train}=\frac{1}{N_{train}}\sum_{k=1}^{N_{train}}\left(y_k-\hat{y}_k\right)^2$$

Η επιλογή (tournament) γίνεται με βάση το μικρότερο J(x) – cost-based υλοποίηση

Από χρωμόσωμα σε μοντέλο συγκεντρωτικά
Σε κάθε αξιολόγηση:
1. Παίρνω το x
2. Το σπάω σε 15 ομάδες των 5
3.Υπολογίζω την $\hat{f}\left(u_{1,k}, u_{2,k}\right)$ για όλα τα σημεία train
4. Βρίσκω την J(θ) (MSE) και το tournament επιλέγει το μικρότερο J.

Για την χαμηλή πολυπλοκότητα που μας ζητείται μπορούμε να μετράμε πόσες Gaussians είναι ενεργές κάθε φορά και έστω ε τέτοιο ώστε μια βάση να είναι ενεργή όταν $w_i > \varepsilon$ με ε=0.05.

<u>Αρχικοποίηση πληθυσμού</u>

Φτιάχνω P=80 άτομα uniform μέσα στα bounds.

Για κάθε άτομο επιλέγω μια τυχαία τιμή για τις παραμέτρους του, δηλαδή:
τυχαίο w_i μέσα στο [-2,2]
τυχαίο $c_{1,i}$ μέσα στο [-1,2]
τυχαίο $c_{2,i}$ μέσα στο [-2,1]
τυχαία $\sigma_{1,i}, \sigma_{2,i}$ μέσα στο [0.1,1.5]

Έτσι έχω τον 1ο πληθυσμό $\{\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(P)}\}$

Αξιολόγηση
Για κάθε χρωμόσωμα
1. Υπολογίζω $\hat{y}_k=\hat{f}\left(u_{1,k}, u_{2,k}, \theta\right)$ για όλα τα δείγματα train
2.Υπολογίζω το MSE:

$$J(\theta)=\frac{1}{N}\sum_{k=1}^{N}\left(y_k-\hat{y}_k\right)^2$$

Όσο μικρότερο MSE , τόσο μεγαλύτερο fitness, το tournament επιλέγει το μικρότερο cost J.

Επιλογή Γονέων

Θέλω να διαλέγω καλούς γονείς αλλά να μην αποκλείω τους κακούς. Για να το πετύχω αυτό θα υλοποιήσω αλγόριθμο Tournament Selection, δηλαδή
Tournament size k=3.
Για κάθε γονέα:
1. Διαλέγω τυχαία 3 χρωμοσώματα από τον πληθυσμό
2. Κρατάω το καλύτερο χρωμόσωμα με το μεγαλύτερο fitness και άρα το μικρότερο cost.
3. Το επαναλαμβάνω όσες φορές χρειάζεται για να σχηματίσω το σύνολο των γονέων για παραγωγή απογόνων.

Διασταύρωση (crossover)

Από 2 γονείς φτιάχνω 2 παιδιά
Πιθανότητα διασταύρωσης $p_c = 0.85$
Επειδή οι παράμετροι είναι πραγματικοί αριθμοί, χρησιμοποιούμε arithmetic crossover:
Οι γονείς είναι $x^{(A)}$ και $x^{(B)}$ διανύσματα ίδιου μήκους, τότε αν γίνει crossover παράγω δύο παιδιά:

$$x^{(child\,1)} = \alpha\, x^{(A)} + (1-\alpha)\, x^{(B)}$$
$$x^{(child\,2)} = (1-\alpha)\, x^{(A)} + \alpha x^{(B)}$$

όπου $\alpha \in [0,1]$ τυχαίο.
Αν δεν γίνει crossover με πιθανότητα $1 - p_c$ τα παιδιά είναι αντίγραφα των γονέων.

Μετάλλαξη (mutation)

Με μικρή πιθανότητα, αλλάζω λίγο κάποια γονίδια για να μην κολλήσει ο αλγόριθμος. Να αποφύγουμε δηλαδή την πρόωρη σύγκλιση, τροποποιούμε τα γονίδια με:
Πιθανότητα μετάλλαξης $p_m = 0.05$ (5% ανά γονίδιο ανά άτομο)
Προσθέτω μικρό θόρυβο:

$$x_j \leftarrow x_j + \beta (ub_j - lb_j) n$$

$n \sim N(0,1)$ και β=mutScale=0.08.
Μετά την μετάλλαξη εφαρμόζω clamping στα αποτέλεσματα, στα επιτρεπτά όρια.

$$x_j \leftarrow min(max(x_j, lb_j), ub_j)$$

Ελιτισμός (elitism)

Για να μην χάσω την καλύτερη λύση κρατάω τους 2 καλύτερους από την προηγούμενη γενιά και τους παίρνω αυτούσιους στην επόμενη.
Elites : E=2
δηλαδή οι δύο καλύτεροι της γενιάς g περνούν αυτούσιοι στην γενιά g+1 και τα υπόλοιπα άτομα είναι απόγονοι (crossover/mutation).

Νέα γενιά

Με τα παραπάνω φτιάχνω καινούργιο πληθυσμό και επαναλαμβάνω.
Η νέα γενιά σχηματίζεται ως εξής:
2 elites από την προηγούμενη γενιά
Οι υπόλοιποι P-E προκύπτουν από επιλογή γονέων crossover, mutation, clamping.

Τερματισμός

Σταματάω όταν:
Φτάσω μέγιστο αριθμό γενεών $G_{max} = 200$

Συγκεντρωτικά ο αλγόριθμος
1. αρχικοποίηση πληθυσμού
2. υπολογισμός fitness
3. επιλογή γονέων
4. κάνω crossover + mutation που δίνει παιδιά
5. ελιτισμός + νέα γενιά
6. επανάληψη μέχρι τον τερματισμό.

Δεδομένα (train/test) και πως αξιολογώ το μοντέλο

Αφού η GA θα μάθει τις παραμέτρους από δεδομένα πρέπει να δούμε πώς φτιάχνουμε αυτά τα δεδομένα train, πως φτιάχνω άλλο dataset για test και τι μετράω για να πω πως το μοντέλο είναι καλό.

Φτιάχνω dataset
Διαλέγω N τυχαία σημεία $(u_{1,k}, u_{2,k})$ μέσα στα όρια που έχω θέσει $u_1 \in [-1,2]$ και $u_2 \in [-2,1]$.
Για κάθε σημείο υπολογίζω την έξοδο από το πραγματικό σύστημα :

$$y_k = f(u_{1,k}, u_{2,k}) = \sin(u_{1,k} + u_{2,k}) \sin(u_{2,k}^2)$$

Άρα δημιουργώ το dataset:

$$\{(u_{1,k}, u_{2,k}, y_k)\}_{k=1}^{N}$$

Για αναπαραγωγή των αποτελεσμάτων θέτω seed τυχαιότητας rng(1).

Train set
Το train set είναι αυτό που χρησιμοποιεί ο Γενετικός αλγόριθμος για να προσαρμόσει τις παραμέτρους του μοντέλου (βάρη, κέντρα και διασπορές των Gaussian.
Χρησιμοποιώ:

$$N_{train} = 1000$$

τυχαία σημεία. Κατά την εκπαίδευση ο GA ελαχιστοποιεί το σφάλμα στο train set (MSE_{train})

Test set
Για την αξιολόγηση της γενίκευσης δημιουργώ ξεχωριστό σύνολο δεδομένων,(διαφορετικό τυχαίο δείγμα από το train, στο ίδιο domain) το οποίο δεν χρησιμοποιείται στην διαδικασία βελτιστοποίησης του GA.
Χρησιμοποιώ:

$$N_{test} = 1000$$

τυχαία σημεία uniform στο ίδιο domain.

Μέτρο απόδοσης (MSE) και αξιολόγηση
Μετά την ολοκλήρωση του GA επιλέγω το καλύτερο χρωμόσωμα θ* (βέλτιστες παράμετροι μοντέλου) και υπολογίζω:
Train MSE

$$MSE_{train} = \frac{1}{N_{train}} \sum_{k=1}^{N_{train}} (y_k - \hat{y}_k)^2$$

Test MSE

$$MSE_{test} = \frac{1}{N_{test}} \sum_{k=1}^{N_{test}} (y_k - \hat{y}_k)^2$$

Ένα μοντέλο θεωρείται καλύτερο όταν έχει χαμηλό MSE_{train}και ταυτόχρονα χαμηλό MSE_{test}, δηλαδή όταν γενικεύει καλά σε νέα δεδομένα που δεν είδε κατά την εκπαίδευση.
Για οπτική αξιολόγηση παρακάτω παρουσιάζονται επιφάνειες $f, \hat{f}$ και $e = f - \hat{f}$ πάνω στο domain, καθώς και η σύγκλιση του καλύτερου κόστους ανά γενιά.

Αναλυτικά τι θα υλοποιήσουμε στο MATLAB

Το πρόγραμμα:

1. δημιουργεί train/test δεδομένα
2. τρέχει τον Γενετικό Αλγόριθμο (GA) για να βρει τις παραμέτρους του μοντέλου Gaussian
3. κρατάει το καλύτερο χρωμόσωμα x_{best}
4. αξιολογεί την απόδοση σε train και test (MSE)
5. παράγει τα απαραίτητα γραφήματα (σύγκλιση, επιφάνειες)

Συναρτήσεις (αρχεία .m)

f_true.m
Σκοπός:Υλοποιεί το πραγματικό σύστημα της εκφώνησης

$$f(u_1, u_2) = \sin(u_1 + u_2)\sin(u_2^2)$$

Input: u_1, u_2
Output: y

generateData.m
Δημιουργεί δύο ανεξάρτητα dataset train/test με uniform δειγματοληψία μέσα στο domain.
Domain:

$$u_1 \in [-1,2], u_2 \in [-2,1]$$

Επιλέγω:

$$N_{train} = 1000, N_{test} = 1000$$

και υπολογίζω:

$$y_k = f(u_{1,k}, u_{2,k})$$

Outputs:

- Utrain $(N_{train} x 2)$
- ytrain $(N_{train} x 1)$
- Utest $(N_{test} x 2)$
- ytest $(N_{test} x 1)$

Αναπαραγωγιμότητα: χρησιμοποιείται rng(1)

makeBounds.m
Σκοπός: Φτιάχνει το άνω και το κάτω όριο lb, ub για όλα τα γονίδια του χρωμοσώματος.
Για κάθε Gaussian i:

- $w_i \in [-2,2]$
- $c_{1,i} \in [-1,2]$
- $c_{2,i} \in [-2,1]$
- $\sigma_{1,i}, \sigma_{2,i} \in [0.1, 1.5]$

Output: lb, ub (μήκους D=75)

decodeChromosome.m
Σκοπός: Αποκωδικοποιεί το χρωμόσωμα $x \in \Re^{75}$ σε 15 ομάδες παραμέτρων.
Input: x (1x75), M=15
Output: w, c_1, c_2, s_1, s_2 (όλα 15x1)

predictModel.m
Σκοπός: Υπολογίζει την πρόβλεψη $\hat{y}$ για δεδομένη είσοδο $U = [u_1, u_2]$ και χρωμόσωμα x.
Για κάθε Gaussian:

$$G(u_1, u_2) = \exp\left(-\left(\frac{(u_1 - c_{1,i})^2}{2\sigma_{1,i}^2} + \frac{(u_2 - c_{2,i})^2}{2\sigma_{2,i}^2}\right)\right)$$

και

$$\hat{y} = \sum_{i=1}^{15} w_i G_i(u_1, u_2)$$

Inputs: U (Nx2), x (1x75)
Outputs: yhat (Nx1)

costFunction.m
Σκοπός:Υπολογίζει το κόστος που ελαχιστοποιεί ο GA.
Βασικό κόστος:

$$J = MSE_{train} = \frac{1}{N_{train}} \sum_{k=1}^{N_{train}} (y_k - \hat{y}_k)^2$$

Inputs: x, Utrain, ytrain
Outputs: J

tournamentSelect.m
Σκοπός: Επιλογή γονέα με tournament selection μεγέθους:

k=3

Επιλέγονται 3 τυχαία άτομα και κρατάω αυτό με το μικρότερο cost J.

mutate.m
Σκοπός: Μετάλλαξη ανά γονίδιο με πιθανότητα:

$$p_m = 0.05$$

Μετάλλαξη με θόρυβο κλιμακωμένο από το εύρος του γονιδίου:

$$x_j \leftarrow x_j + mutScale\,(ub_j - lb_j)\,n, \quad n \sim N(0,1)$$

με mutScale =0.08
και μετά εφαρμόζω clamping:

$$x_j \leftarrow min\left(max\left(x_j, lb_j\right), ub_j\right)$$

GA_man.m – Κύριο πρόγραμμα / GA loop

1) Ορισμός υπερπαραμέτρων:

- Αριθμός Gaussian: M = 15
- Διάσταση χρωμοσώματος: D=5M=75
- Πληθυσμός: P=80
- Γενιές: G=200
- Πιθανότητα crossover: $p_c = 0.85$
- Πιθανότητα mutation: $p_m = 0.05$
- Elitism: E=2
- Tournament size: k= 3
- mutScale=0.08
- usePenalty=false
- rng(1) για αναπαραγωγιμότητα

2) Αρχικοποίηση Πληθυσμού

Δημιουργώ αρχικό πληθυσμό P χρωμοσωμάτων uniform μέσα στα bounds (lb,ub).

3)Επανάληψη ανά γενιά (GA loop)

Για g=1,…,G

1. Υπολογίζω cost J για όλα τα άτομα με costFunction (MSE στο train)
2. Εφαρμόζω Elitism : κρατάω τους E=2 καλύτερους (με το μικρότερο J)
3. Επιλογή γονέων με tournamentSelect (k=3)
4. Crossover (arithmetic) με πιθανότητα $p_c = 0.85$
5. Mutation με mutate (pm=0.05, mutScale=0.08) και clamping στα bounds.
6. Σχηματίζω τη νέα γενιά
7. Αποθηκεύω το καλύτερο $J_{best}(g)$ για το γράφημα σύγκλισης

4)Τελική λύση και αξιολόγηση

Στο τέλος κρατάω το καλύτερο χρωμόσωμα x_{best} και υπολογίζω:

- MSE_{train} στο train set
- MSE_{test} στο test set

Επιπλέον αναφέρω και τον αριθμό ενεργών Gaussian βάσεων με κριτήριο :

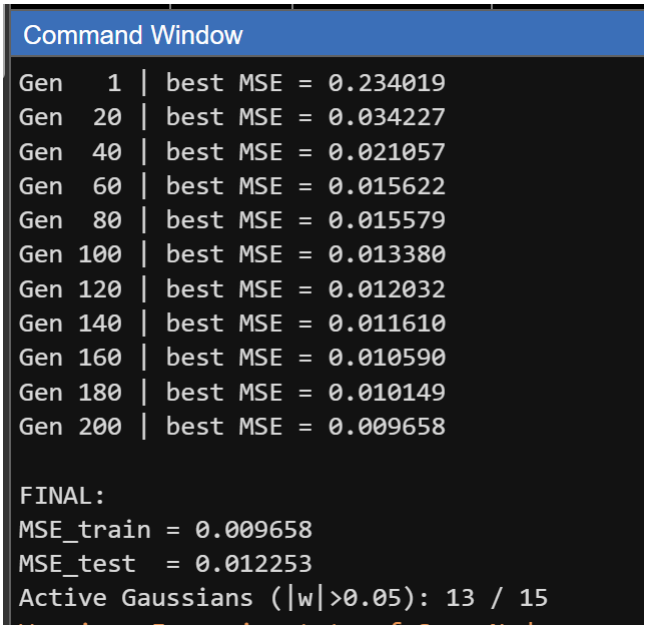
$$|w_i| > 0.05$$

5) Γραφήματα

Παράγω:

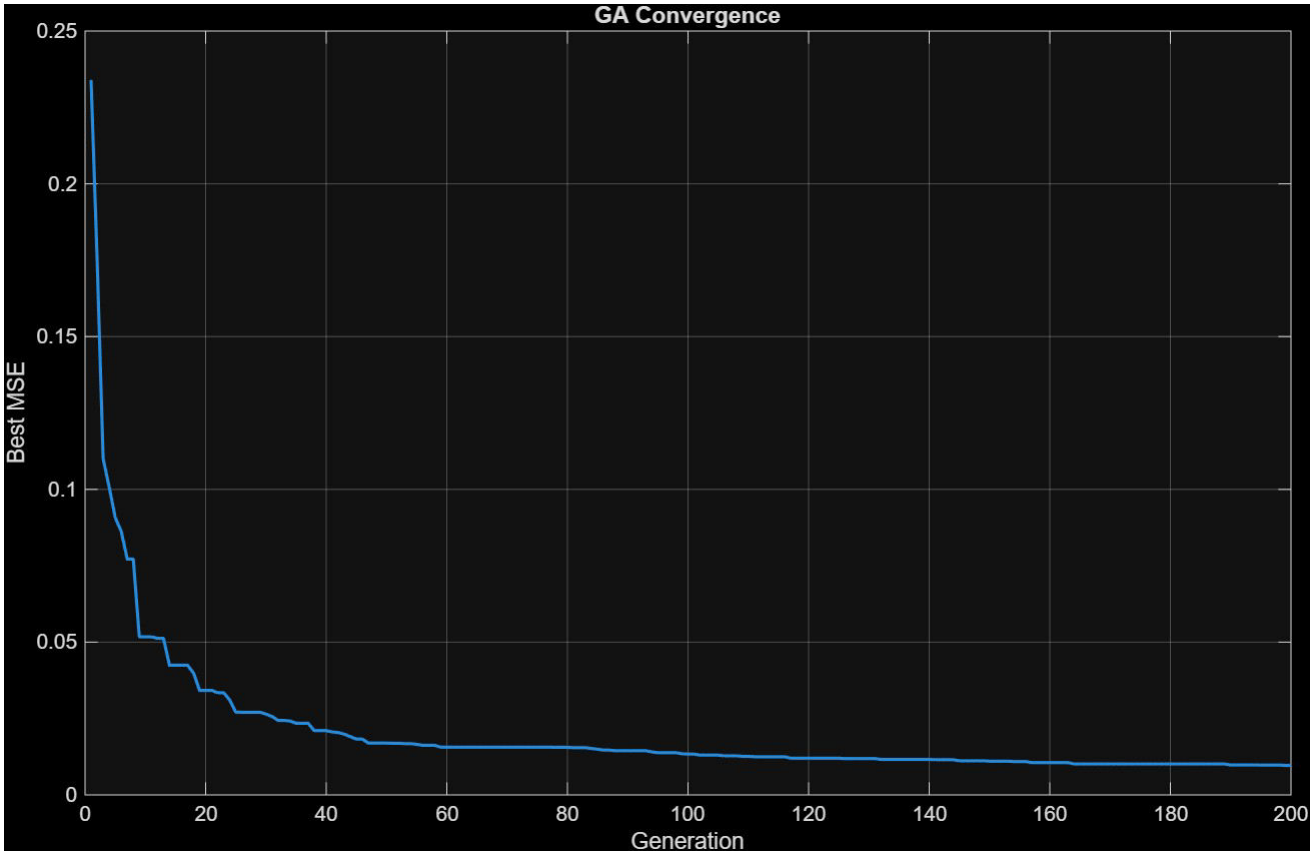
- Καμπύλη σύγκλισης J_{best} vs γενιά.
- Επιφάνειες $f(u_1,u_2), \hat{f}(u_1,u_2)$ και $e = f - \hat{f}$.

Αποτελέσματα – Σύγκλιση GA και μετρικές Απόδοσης



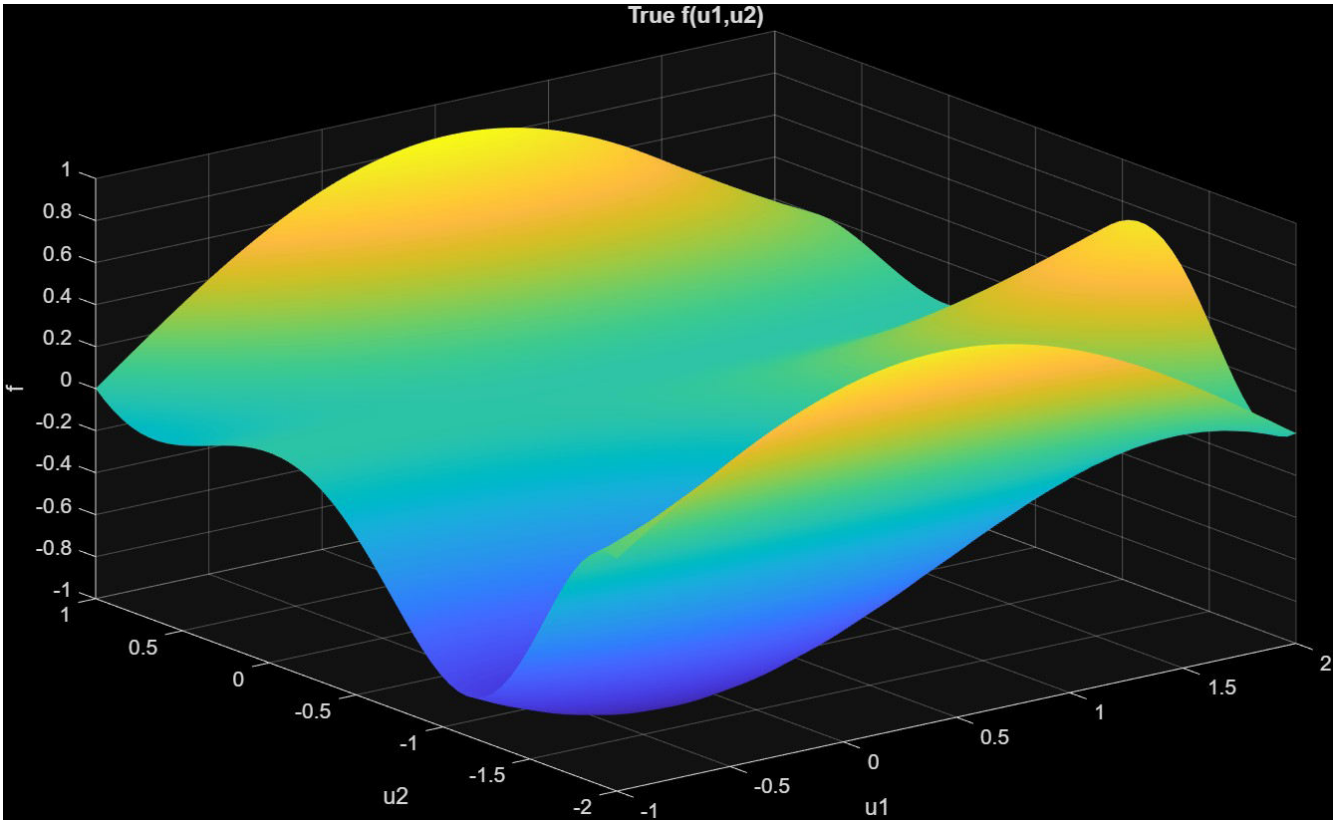
Σχολιασμός:

Οι γραμμές Gen ...| best MSE = … δείχνουν την καλύτερη τιμή του MSE που έχει βρεθεί μέχρι εκείνη τη γενιά.
Φαίνεται από τις τιμές ότι ο GA συγκλίνει από 0.234 στην 1η γενιά πέφτει στο 0.034 και συνεχίζει με μικρότερες βελτιώσεις μέχρι να φτάσει την γενιά 200 όπου παίρνει την τιμή 0.009658.
Στο τέλος το MSE_train = 0.009658 και MSE_test = 0.012253 είναι οι τελικές επιδόσεις στο train και στο test αντίστοιχα.
Το Active Gaussians : 13/15 δείχνει ότι από τις 15 διαθέσιμες βάσεις, οι 13 χρησιμοποιούνται ουσιαστικά ενώ οι υπόλοιπες δυο έχουν πολύ μικρο βάρους και πρακτικά δεν επηρεάζουν.
Αξιοποιούνται λιγότερες με στόχο την μείωση της πολυπλοκότητας του προγράμματος,



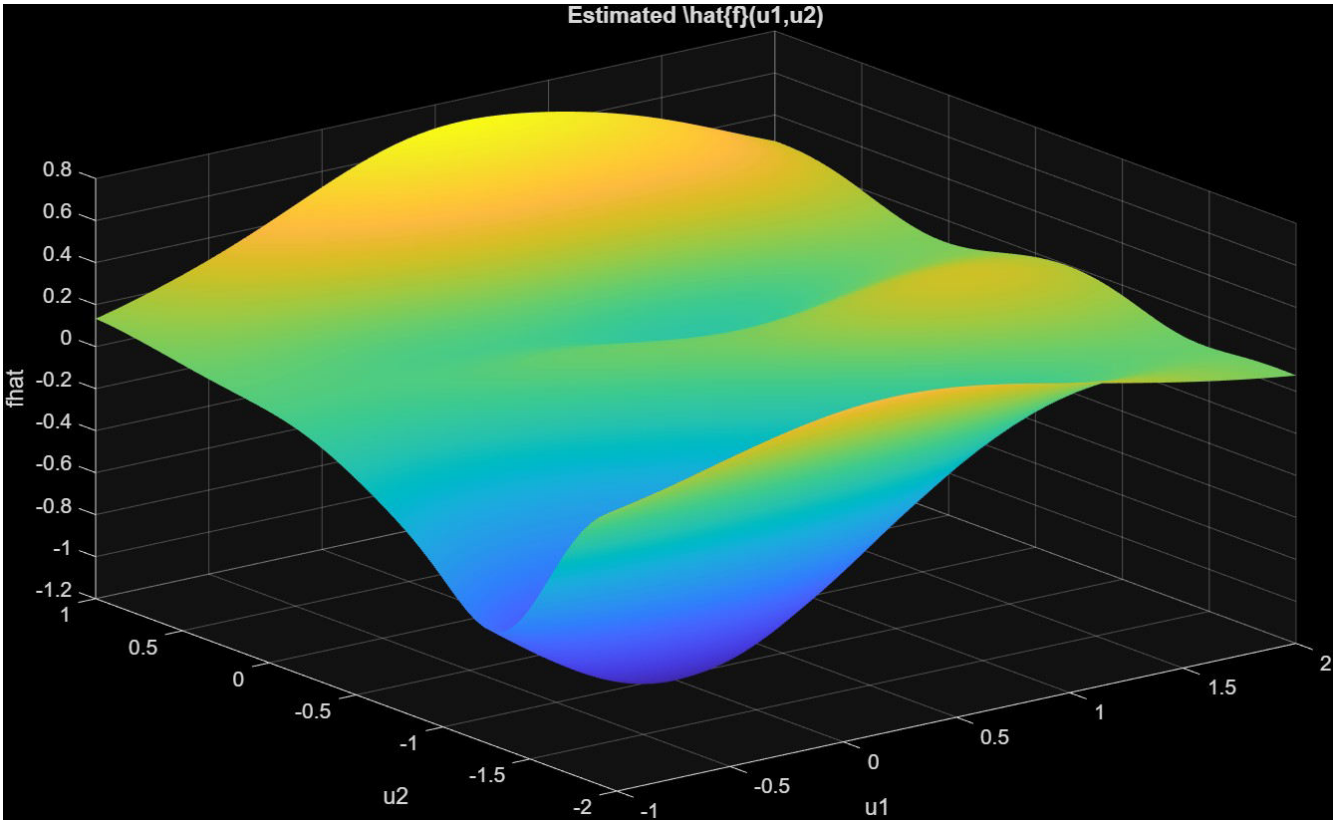
Σχολιασμός:

Το γράφημα δείχνει την σύγκλιση του γενετικού αλγορίθμου στον οριζόντιο άξονα έχουμε τις γενιές και στον κατακόρυφο το καλύτερο MSE που έχει βρεθεί μέχρι εκείνη τη γενιά
Βλέπω στην αρχή ότι το γράφημα πέφτει πολύ γρήγορα δηλαδή ο GA βρίσκει γρήγορα καλύτερες λύσεις και στην συνέχεια οι βελτιώσεις γίνονται όλο και μικρότερες καθώς πλησιάζει σε μια καλή βέλτιστη λύση.



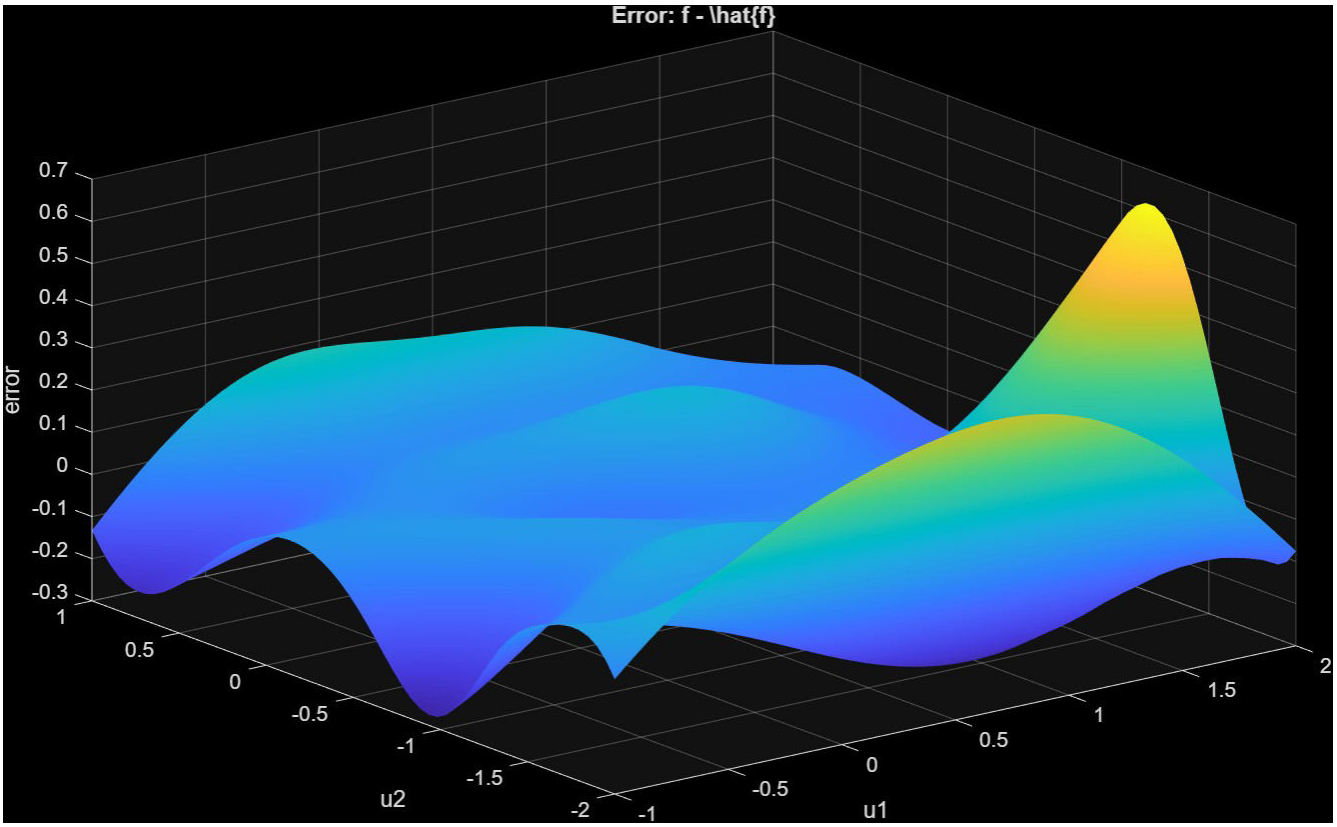
Σχολιασμός:

Στο παραπάνω διάγραμμα απεικονίζεται σε 3D αναπαράσταση η πραγματική συνάρτηση $f(u_1, u_2)$ στο πεδίο που ορίσαμε τα (u_1, u_2). Η επιφάνειά της είναι ομαλή και μη γραμμική με κοιλάδες με θετικές και αρνητικές τιμές στο [-1,1] λόγω των ημιτονοειδών όρων. Προσπαθεί να προσεγγίσει το μοντέλο μας με τις Gaussian.



<u>Σχολιασμός:</u>

Στο συγκεκριμένο διάγραμμα απεικονίζεται η εκτιμώμενη επιφάνεια της $\hat{f}(u_1, u_2)$ που έβγαλε το μοντέλο με τις Gaussian μετά τον GA. Φαίνεται ότι πλησιάζει την βασική μορφή της πραγματικής f αλλά τα πιο έντονα τοπικά χαρακτηριστικά της πραγματικής συνάρτησης δεν απεικονίζονται ακριβώς. Αυτό είναι κάτι αναμενόμενο αφού έχουμε περιορισμένο αριθμό βάσεων (έως 15) και ο αλγόριθμος προσπαθεί να βρει μια καλή προσέγγιση με μικρό MSE.



Σχολιασμός:

Αυτό το διάγραμμα δείχνει το σφάλμα προσέγγισης της μεθόδου. $e(u_1,u_2)=f(u_1,u_2)-\hat{f}(u_1,u_2)$
Όπου η επιφάνεια είναι κοντά στο μηδέν το μοντέλο προσεγγίζει καλά την πραγματική συνάρτηση. Θετικό σφάλμα πάνω από το μηδέν σημαίνει ότι η $\hat{f}$ υποεκτιμά την f ενώ αρνητικό σφάλμα σημαίνει ότι η $\hat{f}$ υπερεκτιμά την f.
Από το διάγραμμα φαίνεται ότι γενικά το σφάλμα είναι σχετικά μικρό στο μεγαλύτερο μέρος του domain, αλλά υπάρχει μία περιοχή προς τα δεξιά σε μεγάλα u_1 όπου εμφανίζεται πιο έντονο σφάλμα. Αυτό δείχνει ότι εκεί η πραγματική f έχει πιο δύσκολη μορφή και το μοντέλο με τις 15 Gaussian δεν την πιάνει τέλεια. Με καλύτερη τοποθέτηση των κέντρων και των πλατών ή περισσότερη τοπική ευελιξία ίσως να διορθώνεται.

Τελικά Συμπεράσματα

Στην εργασία ταυτοποιήσαμε ένα στατικό μη γραμμικό σύστημα δύο εισόδων προσεγγίζοντας την άγνωστη συνάρτηση $f(u_1,u_2)$ με έναν γραμμικό συνδυασμό έως 15 Γκαούσιαν βάσεων, όπου οι παράμετροι $(w_i,c_{1,i},c_{2,i},\sigma_{1,i},\sigma_{2,i})$ βρέθηκαν μέσω ενός Γενετικού αλγορίθμου που υλοποιήθηκε χωρίς έτοιμες συναρτήσεις MATLAB. Η εκπαίδευση έγινε σε σύνολο train και η αξιολόγηση σε διαφορετικό test set, δείχνοντας ότι η GA συγκλίνει σε λύση με χαμηλό MSE και ότι το τελικό μοντέλο αναπαράγει καλά την μορφή της πραγματικής επιφάνειας, με μικρό σφάλμα στο μεγαλύτερο μέρος.
Συνολικά η μέθοδος απέδειξε ότι ένας περιορισμένος αριθμός Gaussian μπορεί να δώσει μια αναλυτική έκφραση χαμηλής πολυπλοκότητας ενώ η προσέγγιση εξαρτάται από τις ρυθμίσεις του GA (πληθυσμός, γενιές, μετάλλαξη) και από τα όρια των παραμέτρων.