Εργαστηριακή Άσκηση Υπολογιστικής Νοημοσύνης
Μέρος Β

**Ονοματεπώνυμο: Νάνος Νικόλαος**
**ΑΜ:1054386**
**Έτος:4ο**

**Β1. Σχεδιασμός ΓΑ**

α) Η κωδικοποίηση που προτείνω είναι η ακέραια . Πιο συγκεκριμένα κάθε άτομο ενός πληθυσμού θα είναι ένα διάνυσμα όπου το κάθε γονίδιο θα είναι ένας ακέραιος αριθμός στο διάστημα [1,5] και θα αντιπροσωπεύει μια αξιολόγηση για μια ταινία (για τον χρήστη για τον οποίο εκτελείτε ο αλγόριθμος ).

β) Με την ακέραια κωδικοποίηση δεν προκύπτουν καθόλου πλεονάζουσες τιμές με την εφαρμογή των γενετικών τελεστών . Επομένως δεν δημιουργείται καν η ανάγκη για αντιμετώπιση τέτειων τιμών .

γ) Η διαδικασία δημιουργίας αρχικού πληθυσμού έχει ως εξής : Αρχικά δημιουργούμε άτομα (διανύσματα) των οποίων οι τιμές είναι ακέραιες και τυχαίες (εξάγονται από ομοιόμορφη κατανομή ) στο διάστημα [1,5] . Στη συνέχεια το κάθε άτομο επιδιορθώνεται με κατάλληλη διαδικασία η οποία καλείται έτσι ώστε οι τιμές των γονιδίων των ατόμων οι οποίες αντιστοιχούν σε αξιολογήσεις που έχουν ήδη γίνει και είναι εσφαλμένες ( λόγω της αντικατάστασής τους με τυχαίες τιμές ) να αντικαθίστανται με τις πραγματικές τιμές των αξιολογήσεων του χρήστη για τον οποίο εκτελείται ο αλγόριθμος.

δ) Η εναλλακτική η οποία αξιολογήθηκε για την επιδιόρθωση των μη νόμιμων λύσεων ήταν η υλοποίηση μιας διαδικασίας επιδιόρθωσης η οποία περιγράφεται από το script repair_population.py . Αυτή η διαδικασία επενεργεί σε κάθε άτομο ενός πληθυσμού απλά βάζοντας τη πραγματική αξιολόγηση (δηλαδή αυτή που έχει γίνει ήδη) στη κατάλληλη θέση των ατόμων όπου αυτή αντιστοιχεί (χρησιμοποιώντας πληροφορία του διανύσματος του προς εξέταση χρήστη) . Η διαδικασία εκτελείται αμέσως μετά τη δημιουργία αρχικού πληθυσμού (όπως αναφέρθηκε και παραπάνω ) και κάθε φορά μετά την εφαρμογή των γενετικών τελεστών .

ε) Για ένα συγκεκριμένο διάνυσμα χρήστη βρίσκουμε τη γειτονιά του χρησιμοποιώντας τη συσχέτιση Pearson ανάμεσα στους διάφορους χρήστες και στον χρήστη για τον οποίο εκτελούμε τον αλγόριθμο . Αυτή η μετρική είναι βασισμένη στα εσωτερικά γινόμενα και το αποτέλεσμά της είναι ένας αριθμός στο [-1,1] ο οποίος υποδεικνύει την ομοιότητα μεταξύ δύο διανυσμάτων (στην περίπτωσή μας τα διανύσματα των χρηστών) . Ακολουθείται η ίδια διαδικασία με αυτή του Cosine similarity με την διαφορά ότι στην Pearson τα διανύσματα εκτός από κανονικοποιημένα είναι και κεντραρισμένα . Το προηγούμενο trick μας δίνει το πλεονέκτημα της αμεταβλητότητας της μετρικής σε μετασχηματισμούς ολισθήσεων και κλιμάκωσης των διανυσμάτων (ενώ στο cosine η αμεταβλητότητα της μετρικής ισχύει μόνο ως προς την κλιμάκωση) . ‘Οσον αφορά την ευκλείδεια απόσταση μεταξύ δύο διανυσμάτων , δίνει το ίδιο βάρος και ιδιαίτερα στις μεγάλες τιμές των διαφορών των τιμών (ύψωση στο τετράγωνο) σε όλες τις κατευθύνσεις σε αντίθεση με την Pearson η οποία δεν λαμβάνει

υπόψιν περιπτώσεις μηδενικών τιμών των διανυσμάτων . Επομένως για το πρόβλημά μας το οποίο εμπεριέχει αραιά διανύσματα (λίγα μη μηδενικά στοιχεία ) πολλών διαστάσεων η συσχέτιση Pearson φαίνεται να είναι η καταλληλότερη . (Τις ελλιπείς τιμές τις αντικαθιστώ με μηδενικά ) . Η διαδικασία της εύρεσης γειτονιάς χρήστη υλοποιήθηκε από τη συνάρτηση get_neighborhood η οποία πέρνει ως όρισμα κάποιο διάνυσμα χρήστη και επιστρέφει τη γειτονιά του (10 πιο κοντινούς χρήστες κανονικοποιημένους και κεντραρισμένους )

στ) Η συνάρτηση καταλληλότητας που υλοποιήθηκε παίρνει ως όρισμα τη γειτονιά του ύπο εξέταση χρήστη και ένα άτομο του πληθυσμού της γενεάς , το κεντράρει και το κανονικοποίει και υπολογίζει την ομοιότητά του ( Pearson correlation) με τα 10 διανύσματα τις γειτονίας ( matrix (γειτονία) vector (χρήστης) multiplication ) . Τέλος κάνω scale τους συντελεστές προσθέτοντας 1 για να παίρνω μη αρνητικές τιμές και αθροίζω τους 10 συντελεστές συσχέτισης επιστρέφω το αποτέλεσμα . Η μέγιστη τιμή που μπορεί να πάρει η συνάρτηση καταληλλότητας είναι πλέον 20 (λόγω της ολίσθησης που έγινε στα correlation )

ζ) Για τη τον τελεστή επιλογής θεωρώ ότι θα αποδώσει καλύτερα στο πρόβλημά μας η tournament selection στρατιγική καθώς στηρίζεται μόνο σε συγκρίσεις των τιμών καταλληλότητας των διαφόρων ατόμων που συμμετέχουν στο tournament και είναι ανεξάρτητη από το scaling της fitness function . Σε αντίθεση με την Roulette wheele η οποία μπορεί να γίνει ευάλωτη στο στοχαστικό θόρυβο καθώς κάνουμε υπολογισμό των πιθανοτήτων επιλογής των ατόμων (οι οποίες είναι ανάλογες των τιμών καταλληλότητας όμως) . Αναφορικά με τον τελεστή διασταύρωσης επιλέξαμε διασταύρωση διπλού σημείου έτσι ώστε να χαλάσουμε λιγότερο κάποια καλά άτομα , να περιορίσουμε τη στοχαστικότητα των παιδιών που θα προκύψουν αλλά ταυτοχρόνως να εξερευνήσουμε και άλλες λύσεις στο χώρο αναζήτησης . Τέλος για τον τελεστή μετάλλαξης κάνουμε χρήση της ομοιόμορφης μετάλλαξης (σε αυτή κάθε αξιολόγηση παίρνει μια τυχαία τιμή στο [1,5] με κάποια πιθανότητα) διότι η χρήση του ελιτισμού θα μας περιόριζε στο να εξερευνήσουμε καινούριες λύσεις (αφού θα άλλαζε ένα άτομο με το καλύτερο της προηγούμενης γενιάς) .

**Β2. Υλοποίηση ΓΑ**

https://github.com/nnanos/Genetic-Algorithm-for-recommendation-systems.git

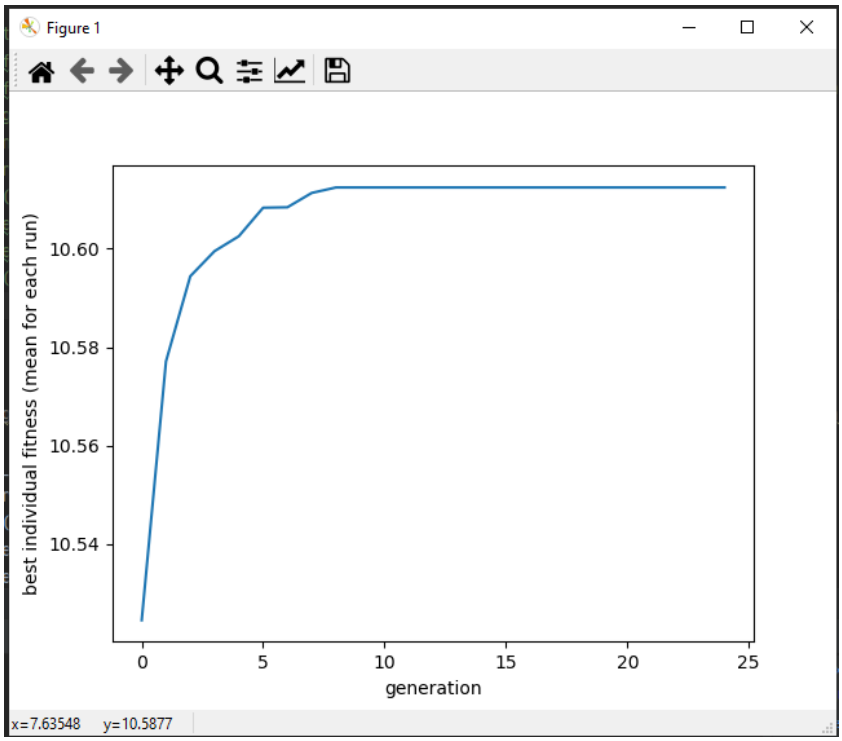
**Β3. Αξιολόγηση και Επίδραση Παραμέτρων**

| ΜΕΓΕΘΟΣ ΠΛΗΘΥΣΜΟΥ | ΠΙΘΑΝΟΤΗΤΑ ΔΙΑΣΤΑΥΡΩΣΗΣ | ΠΙΘΑΝΟΤΗΤΑ ΜΕΤΑΛΛΑΞΗΣ | ΜΕΣΗ ΤΙΜΗ ΒΕΛΤΙΣΤΟΥ | ΜΕΣΟΣ ΑΡΙΘΜΟΣ ΓΕΝΕΩΝ |
|---|---|---|---|---|
| 20 | 0.6 | 0.00 | 10.6096096298509 | 24 |
| 20 | 0.6 | 0.01 | 10.7071514686003 | 57 |
| 20 | 0.6 | 0.1 | 11.1944763003485 | 150 |
| 20 | 0.9 | 0.01 | 10.6999234407149 | 41 |
| 20 | 0.1 | 0.01 | 10.5825714053157 | 45 |
| 200 | 0.6 | 0.0 | 10.9818509710922 | 42 |

| | | | | |
|---|---|---|---|---|
| 200 | 0.6 | 0.01 | 11.3600184733190 | 138 |
| 200 | 0.1 | 0.01 | 11.2114109482911 | 144 |
| 200 | 0.9 | 0.01 | 11.3481627492832 | 112 |

Για να πάρω τις παραπάνω μέσες τιμές χρησιμοποίησα early stopping κατά το οποίο τερματίζουμε τον αλγόριθμο όταν η καλύτερη λύση της τρέχουσας γενιάς έχει σταματήσει να αυξάνεται για 20 γενιές . Ωστόσο , για να πάρω τις παρακάτω καμπύλες δεν χρησιμοποίησα early stopping διότι λόγω της φύσης του αλγορίθμου σε κάθε ένα από τα 10 τρεξίματα παίρνω και διαφορετικό πλήθος γενεών . Χρησιμοποίησα τη τιμή του μέσου αριθμού γενεών και έτρεξα τους αλγορίθμους ξανά με παγωμένο τον αριθμό των γενεών σε αυτόν τον αριθμό για την κάθε περίπτωση αντίστοιχα .

**Παρακάτω παραθέτουμε την κάθε περίπτωση καμπύλης της οποίας η σειρά συμπίπτει με αυτή των γραμμών του παραπάνω πίνακα** :



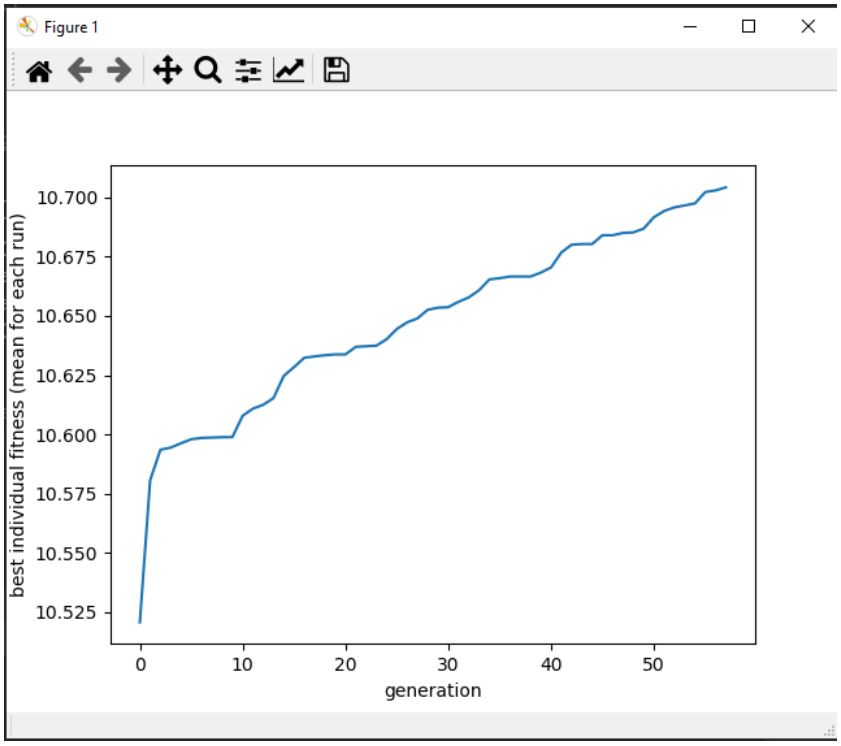


Figure 1
best individual fitness (mean for each run)
10.700
10.675
10.650
10.625
10.600
10.575
10.550
10.525
0
10
20
30
40
50
generation

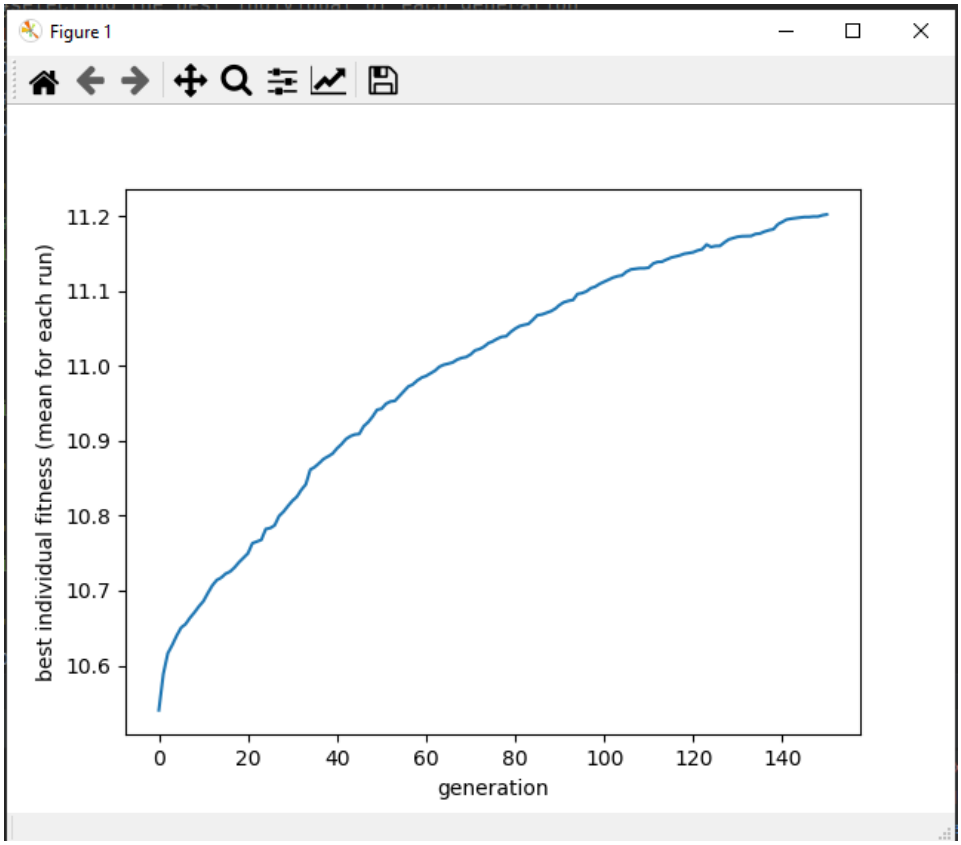


Figure 1
best individual fitness (mean for each run)
11.2
11.1
11.0
10.9
10.8
10.7
10.6
0
20
40
60
80
100
120
140
generation

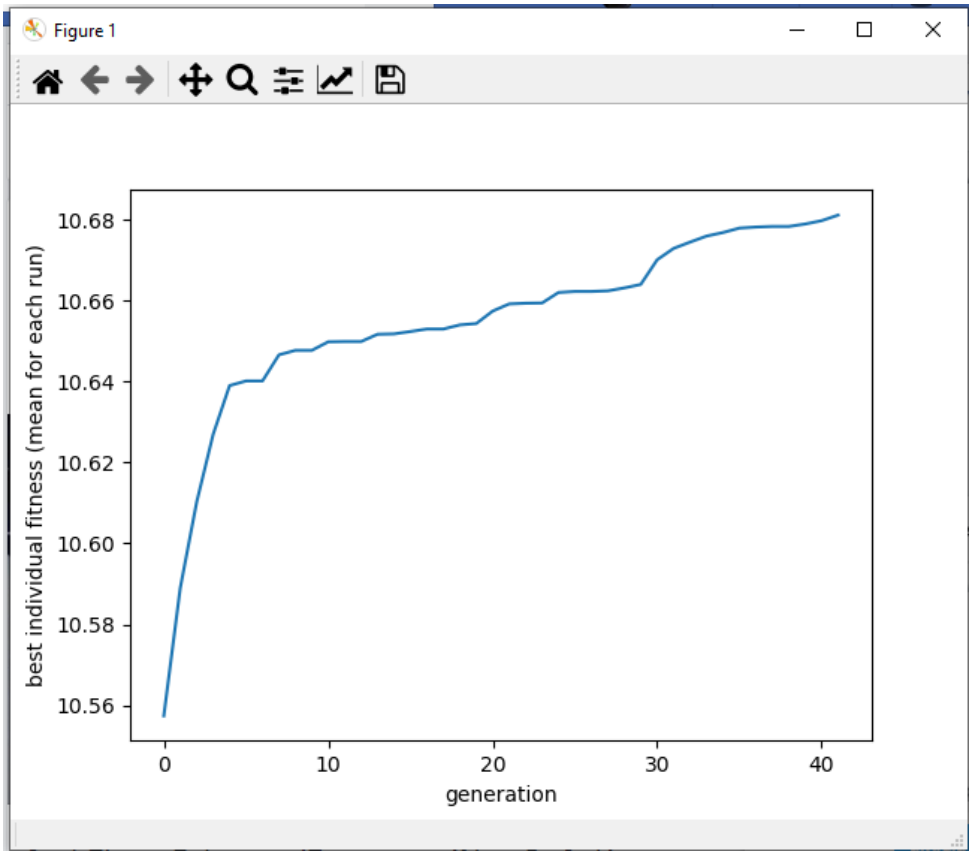


Figure 1
best individual fitness (mean for each run)
10.68
10.66
10.64
10.62
10.60
10.58
10.56
0
10
20
30
40
generation

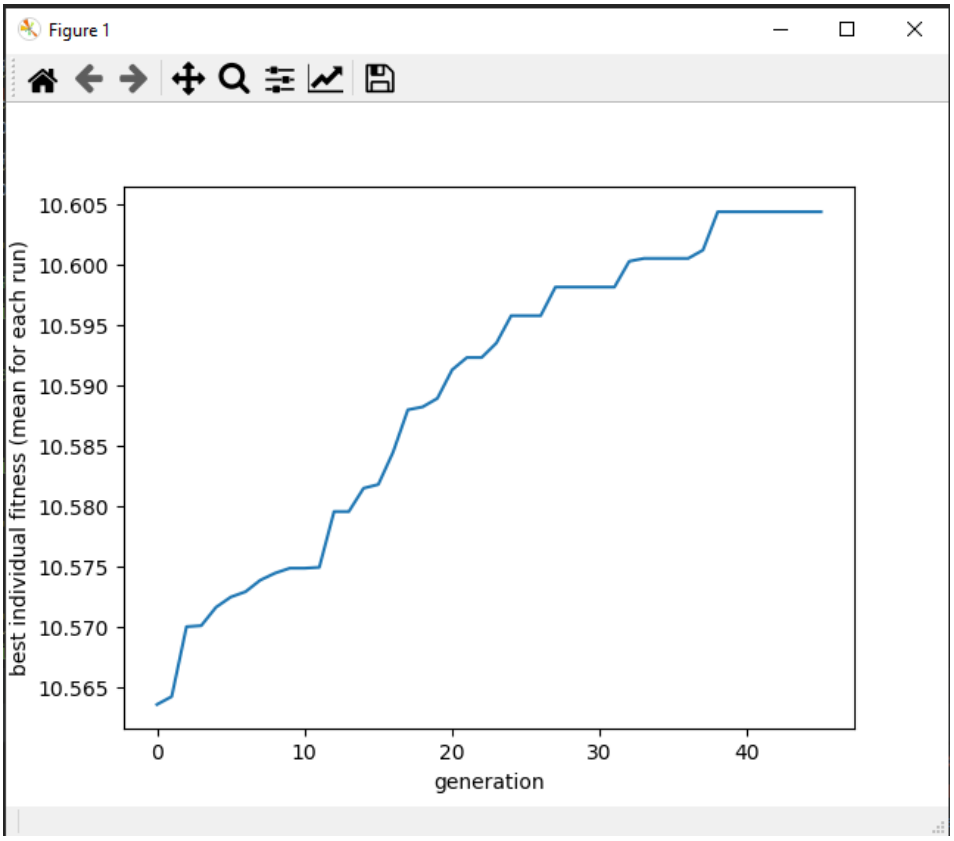


Figure 1
best individual fitness (mean for each run)
10.605
10.600
10.595
10.590
10.585
10.580
10.575
10.570
10.565
0
10
20
30
40
generation

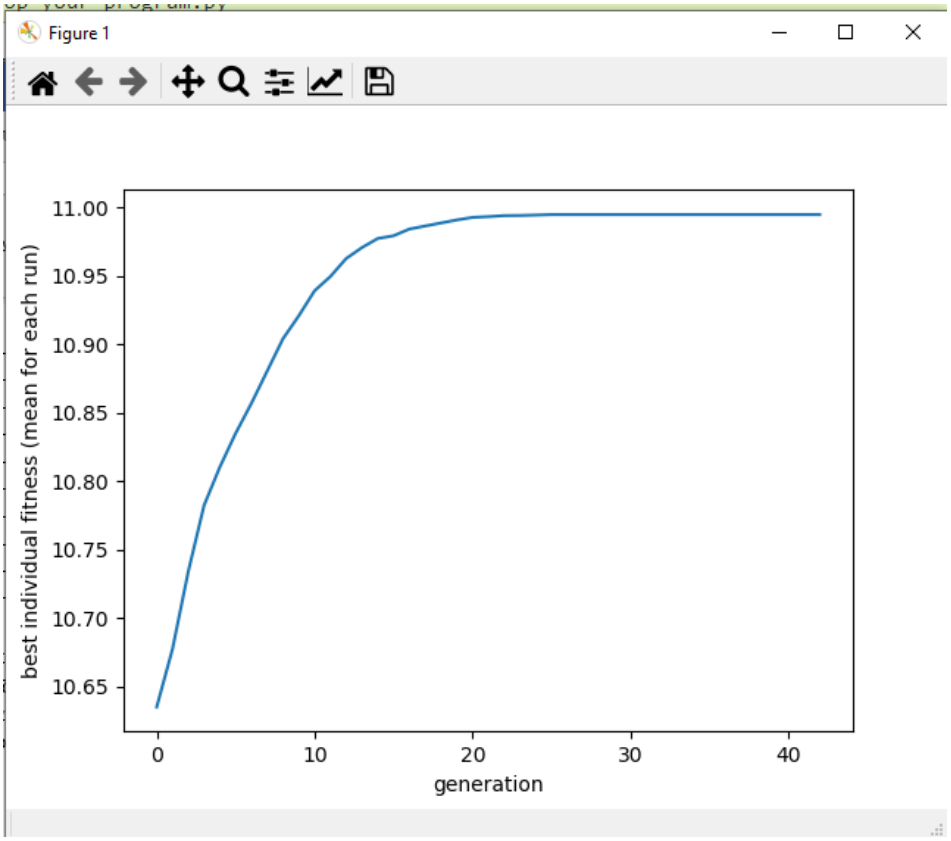


Figure 1
11.00
10.95
10.90
10.85
10.80
10.75
10.70
10.65
best individual fitness (mean for each run)
0
10
20
30
40
generation

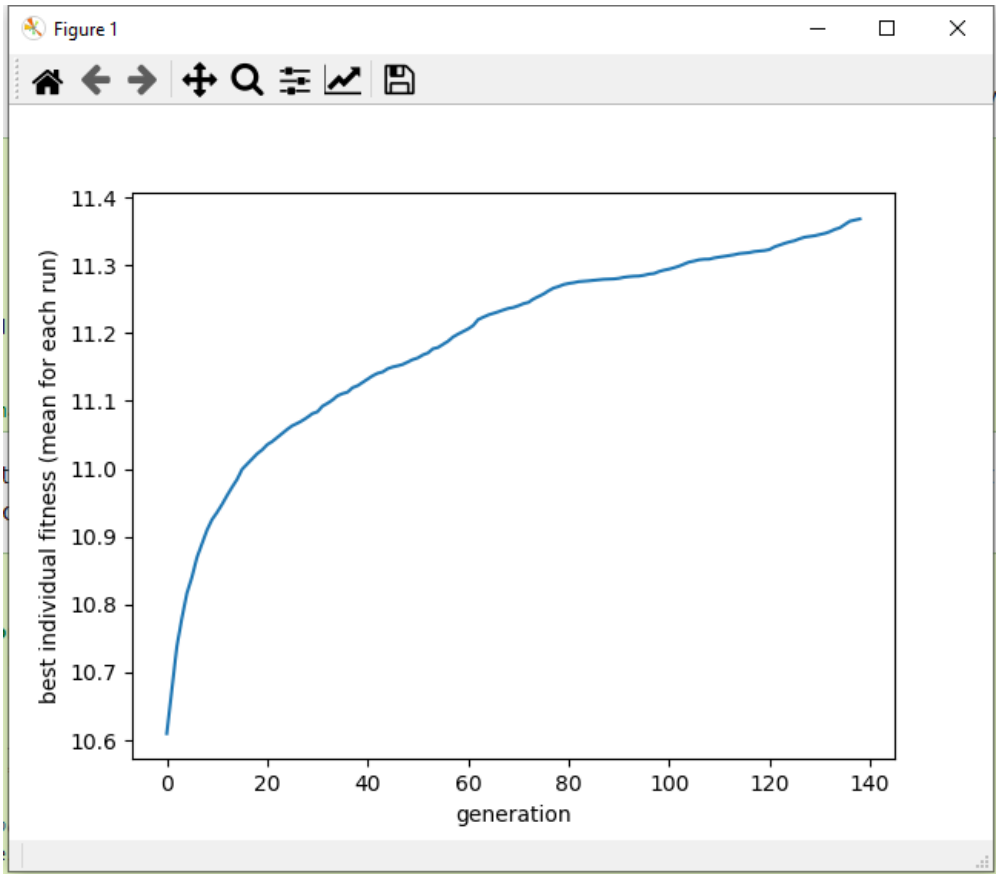


Figure 1
11.4
11.3
11.2
11.1
11.0
10.9
10.8
10.7
10.6
best individual fitness (mean for each run)
0
20
40
60
80
100
120
140
generation

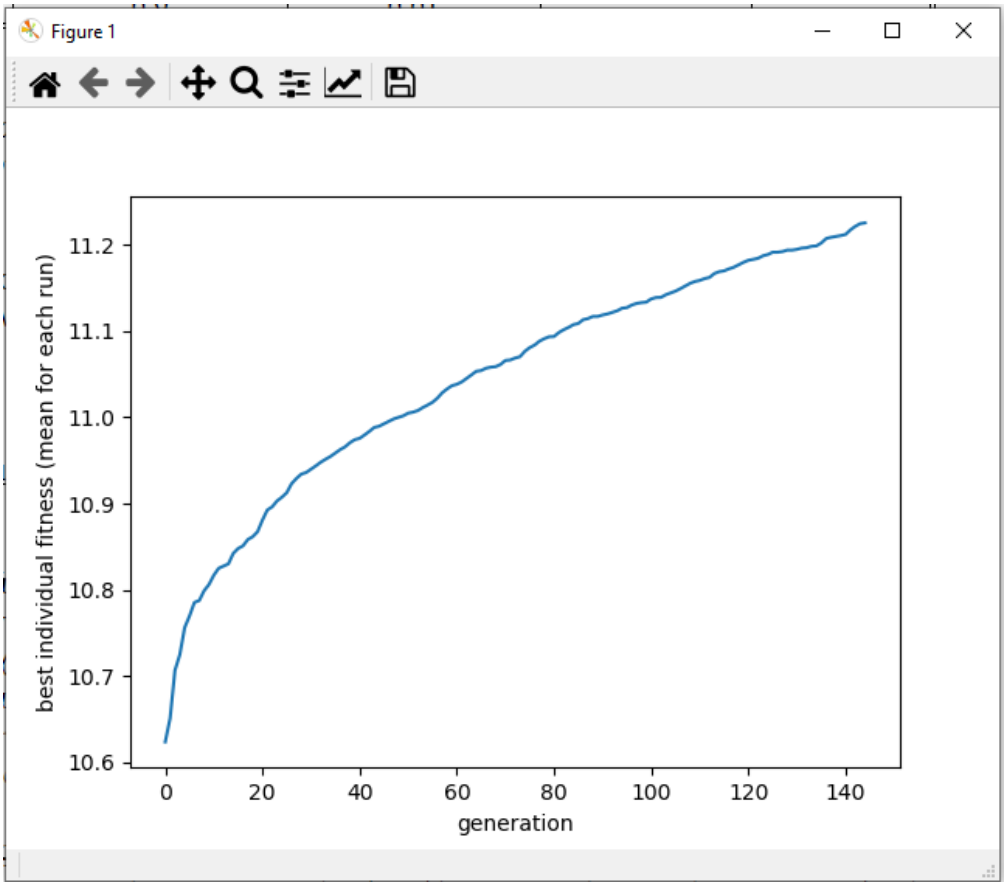


Figure 1
best individual fitness (mean for each run)
11.2
11.1
11.0
10.9
10.8
10.7
10.6
0
20
40
60
80
100
120
140
generation

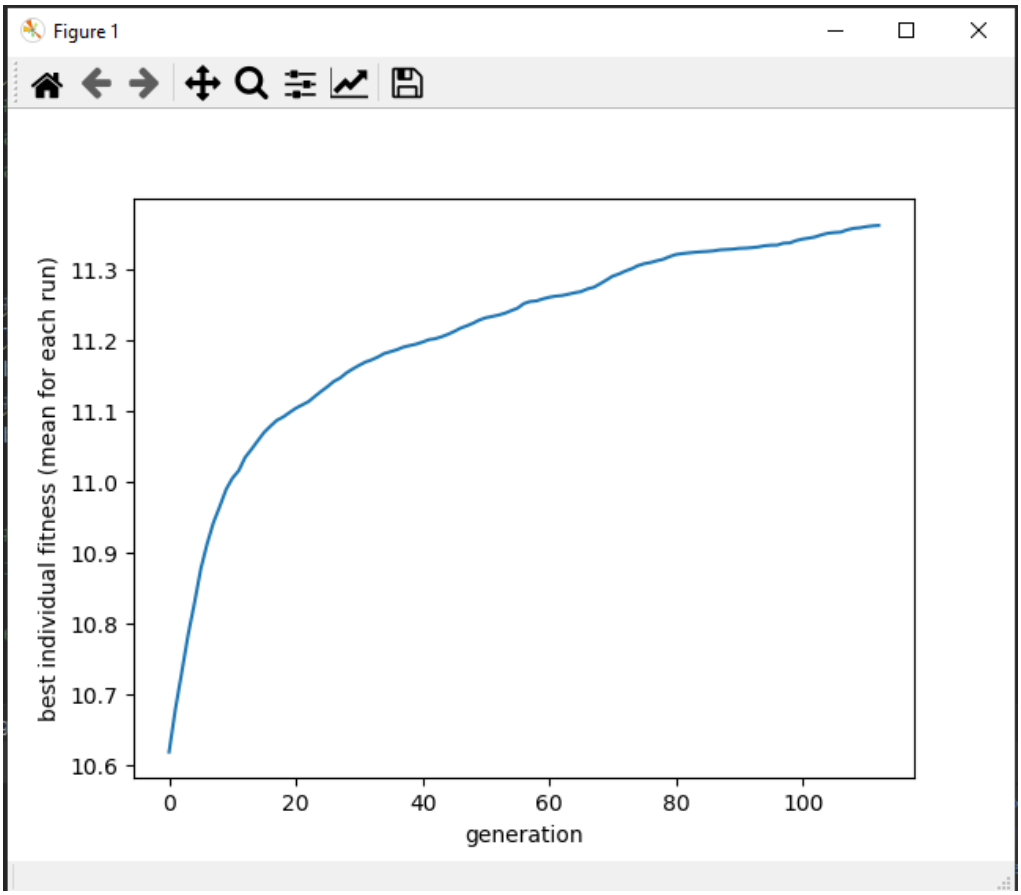


Figure 1
best individual fitness (mean for each run)
11.3
11.2
11.1
11.0
10.9
10.8
10.7
10.6
0
20
40
60
80
100
generation

Από τα αποτελέσματα είναι σαφές ότι όλοι οι παράμετροι που κάναμε tuning σε κάθε περίπτωση έχουν επίδραση τόσο στο κατά πόσο γίνεται μεγιστοποίηση της συνάρτησης καταλληλότητας (παίρνουμε καλύτερα άτομα) όσο και στο πλήθος των γενεών που χρειάζονται για να φτάσει (ο αλγόριθμος) σε κάποιο βαθμό μεγιστοποίησης . Ποιο συγκεκριμένα είναι εμφανές ότι με μεγαλύτερο μέγεθος πληθυσμού παίρνουμε καλύτερα άτομα ανεξάρτητα από την τιμή των υπόλοιπων υπερπαραμέτρων . Ωστόσο παρατηρούμε ότι καλή απόδοση (συγκριτικά με τον μεγάλο πληθυσμό) παίρνουμε και εάν αυξήσουμε τη πιθανότητα μετάλλαξης (στην περίπτωση του μικρού πληθυσμού ) . Όσον αφορά την πιθανότητα διασταύρωσης παρατηρούμε ότι παίρνουμε καλά αποτελέσματα στις ενδιάμεσες τιμές . Συμπεραίνω ότι θα μπορούσα να δοκίμαζα και στρατηγικές crossover οι οποίες ανταλλάσουν μεγαλύτερη γενετική πληροφορία ώστε να αυξηθεί η ποικιλομορφία των ατόμων (αύξηση της στοχαστικότητας του αλγορίθμου) .

**Με βάση τα αποτελέσματα επέλεξα να παγώσω τις παραμέτρους μου στις εξής τιμές:**

**Πλήθος ατόμων ανά γενιά = 20**

**Πιθανότητα διασταύρωσης = 0.6**

**Πιθανότητα μετάλλαξης = 0.1**

**Έκανα αυτήν την επιλογή διότι όταν το πλήθος των ατόμων είναι 20 τότε έχουμε μειωμένη πολυπλοκότητα (προσπάθησα να εκμεταλλευτώ την παραλληλία στο σημείο που υπολογίζουμε τις καταλληλότητες το οποίο είναι το πιο βαρύ σε υπολογισμούς αλλά βρήκα εμπόδια με τις υλοποιημένες βιβλιοθήκες και έτσι δεν μεγάλωσα τα άτομα στα 200 ) . Τέλος το πάγωμα των δύο παραπάνω παραμέτρων προέκυψε διότι μας έδιναν την καλύτερη λύση σε σχέση με τις υπόλοιπες (ήταν και αρκετά πιο κοντά στις λύσεις που εδίναν οι περιπτώσεις με τα 200 άτομα) .**

**Β4. Αξιολόγηση Συστάσεων**

Α)Για την αξιολόγηση των συστάσεων χρησιμοποίησα τα δύο datasets που είναι έτοιμα, το ua.base για το τρέξιμο του αλγορίθμου (train) και το ua.test για την αξιολόγηση των συστάσεων που έγινε πρόβλεψη (test) .

Β) Παρακάτω παρατίθενται τα ζητούμενα γραφήματα:

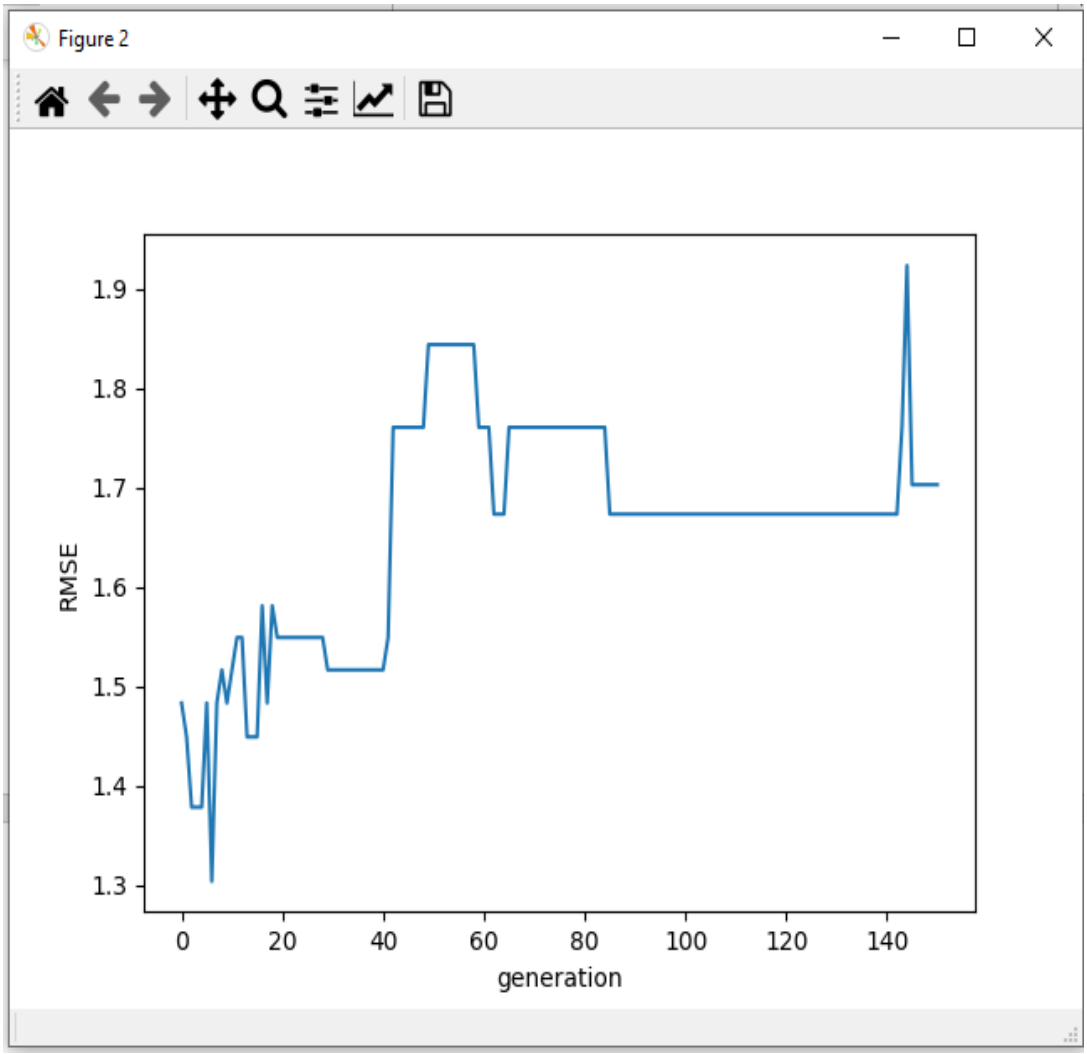


Figure 2
RMSE
1.9
1.8
1.7
1.6
1.5
1.4
1.3
0
20
40
60
80
100
120
140
generation

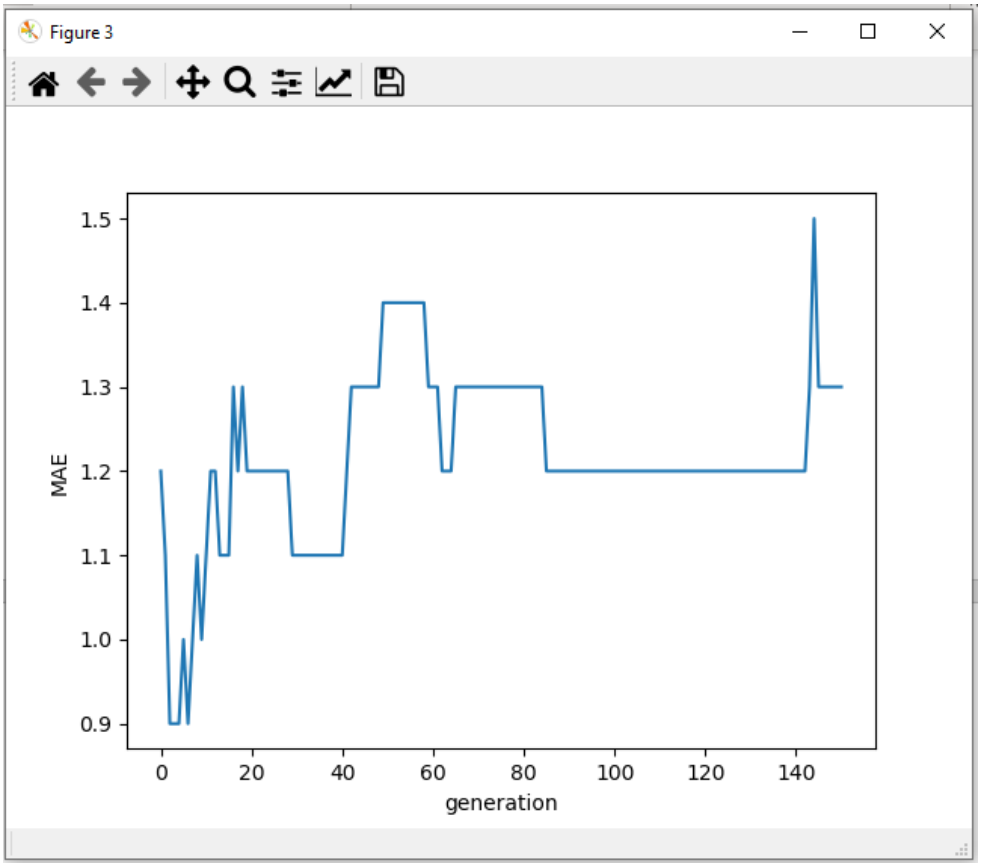


Figure 3
MAE
1.5
1.4
1.3
1.2
1.1
1.0
0.9
0
20
40
60
80
100
120
140
generation

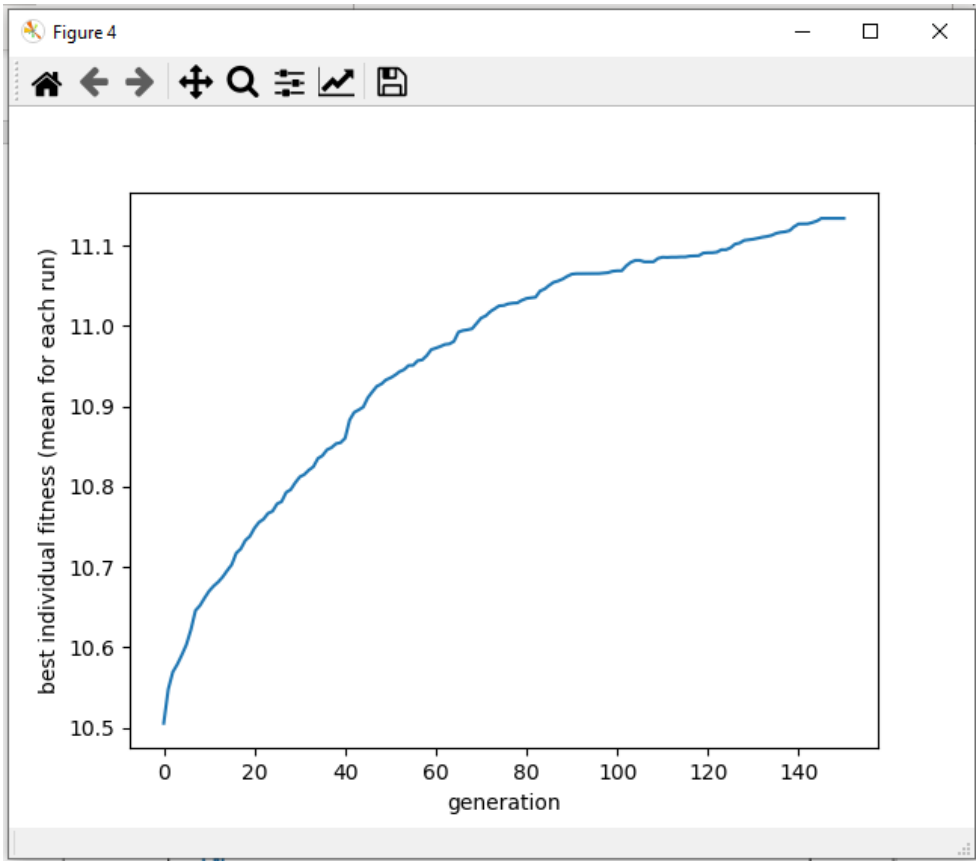
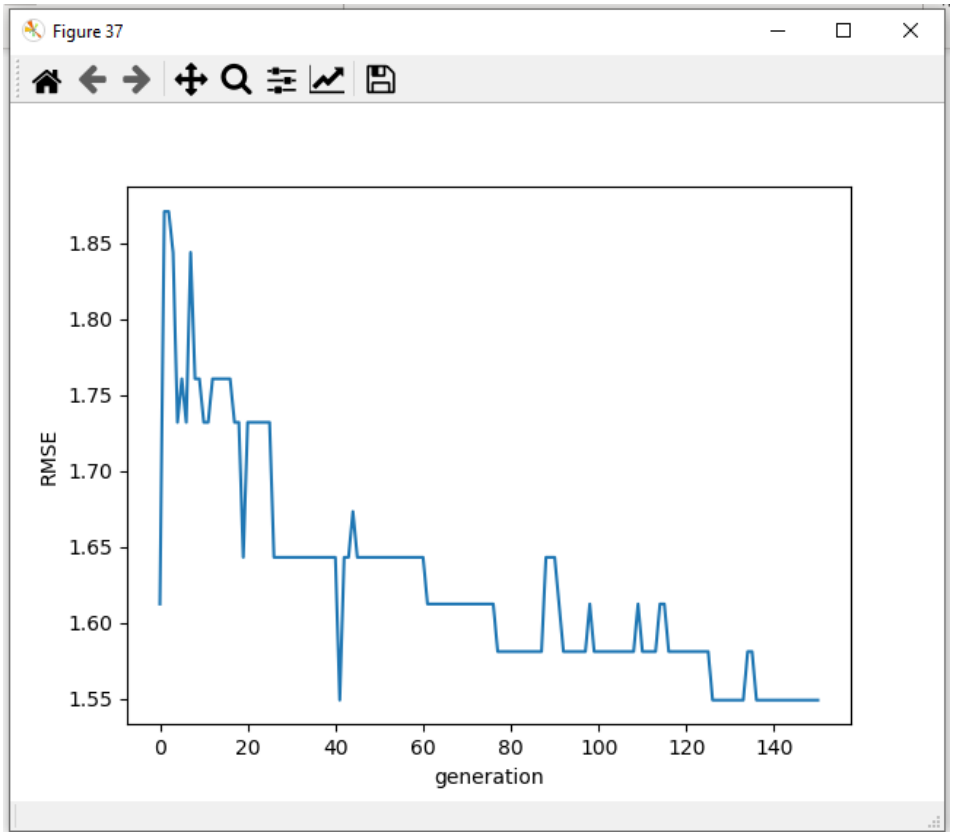


Figure 4
best individual fitness (mean for each run)
11.1
11.0
10.9
10.8
10.7
10.6
10.5
0
20
40
60
80
100
120
140
generation

Κοιτώντας τα παραπάνω γραφήματα παρατηρούμε ότι όσο περνάνε οι γενιές τα RMSE κ’ MAE μεγαλώνουν (με μη ομαλό τρόπο) γεγονός το οποίο αντιβαίνει με το επιθυμητό αποτέλεσμα (το οποίο είναι το ανάποδο) . Κανονικά από τη στιγμή που οι λύσεις μας γίνονται καλύτερες (φαίνεται στο τελευταίο γράφημα) θα έπρεπε να μειώνονται τα RMSE κ’ MAE διότι ένα καλύτερο άτομο (έτσι όπως έχουμε ορίσει το πρόβλημά μας) έχει και μεγαλύτερες πιθανότητες να κάνει καλύτερες προβλέψεις για τις αξιολογήσεις των χρηστών . Ωστόσο η κακή απόδοση (ως προς την αξιολόγηση των συστάσεων) του αλγορίθμου είναι πιθανό να οφείλεται σε έλλειψη δεδομένων (train και test).

Γ) Μετά την εφαρμογή της παραπάνω διαδικασίας (train κ’ test) για 50 διαφορετικούς χρήστες διαπίστωσα ότι σε κάποιες περιπτώσεις χρηστών ο αλγόριθμος αποτύγχανε να κάνει καλές προβλέψεις δηλαδή μεγάλωνε το RMSE παρόλο που πέρναμε καλύτερα άτομα σε κάθε γενιά ( τέτεια περίπτωση περιγράφεται παραπάνω ) . Ωστόσο παρατήρησα και αρκετές περιπτώσεις στις οποίες πέρναμε την επιθυμητή μείωση των μετρικών όσο πέρναγαν οι γενιές . Επίσης παρατήρησα και περιπτώσεις όπου οι μετρικές ταλαντώνονταν αρκετά .Τέλος να επισημάνω ότι σε πολλές περιπτώσεις χρηστών είδα και την παραγωγή καλύτερων λύσεων (που ξεπερνούν την καταλληλότητα της καλύτερης λύσης του χρήστη που παρέθεσα παραπάνω) .

Παραθέτω κάποιες γραφικές παραστάσεις για ορισμένους χρήστες :



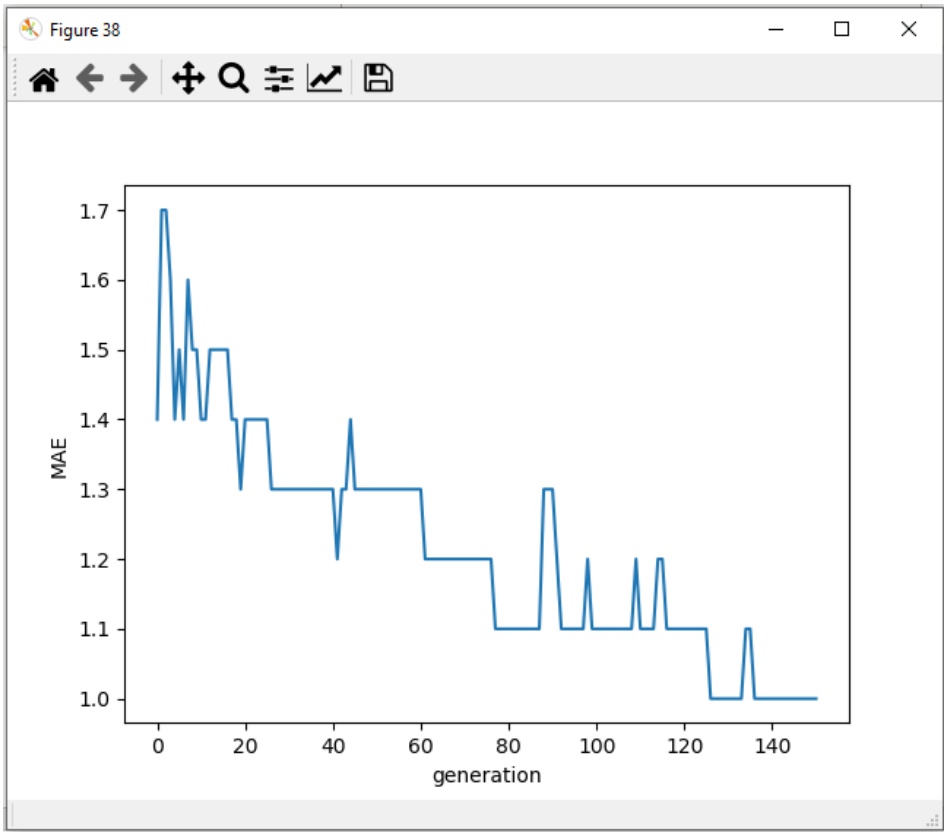


Figure 38
MAE
generation
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0
20
40
60
80
100
120
140

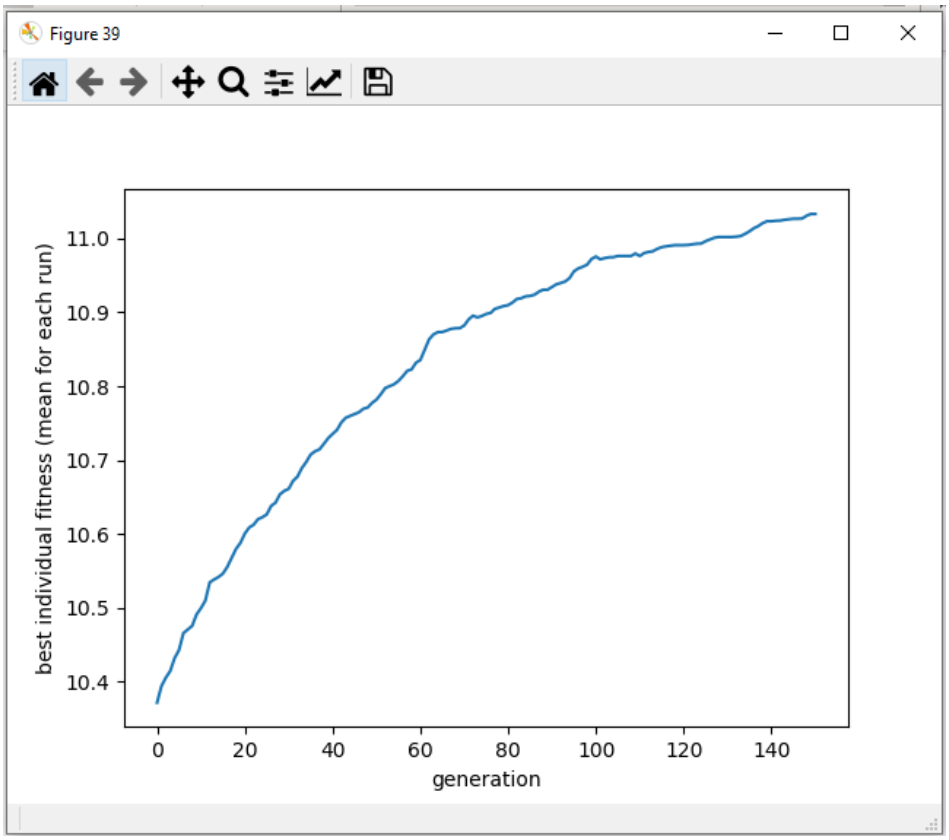


Figure 39
best individual fitness (mean for each run)
generation
11.0
10.9
10.8
10.7
10.6
10.5
10.4
0
20
40
60
80
100
120
140

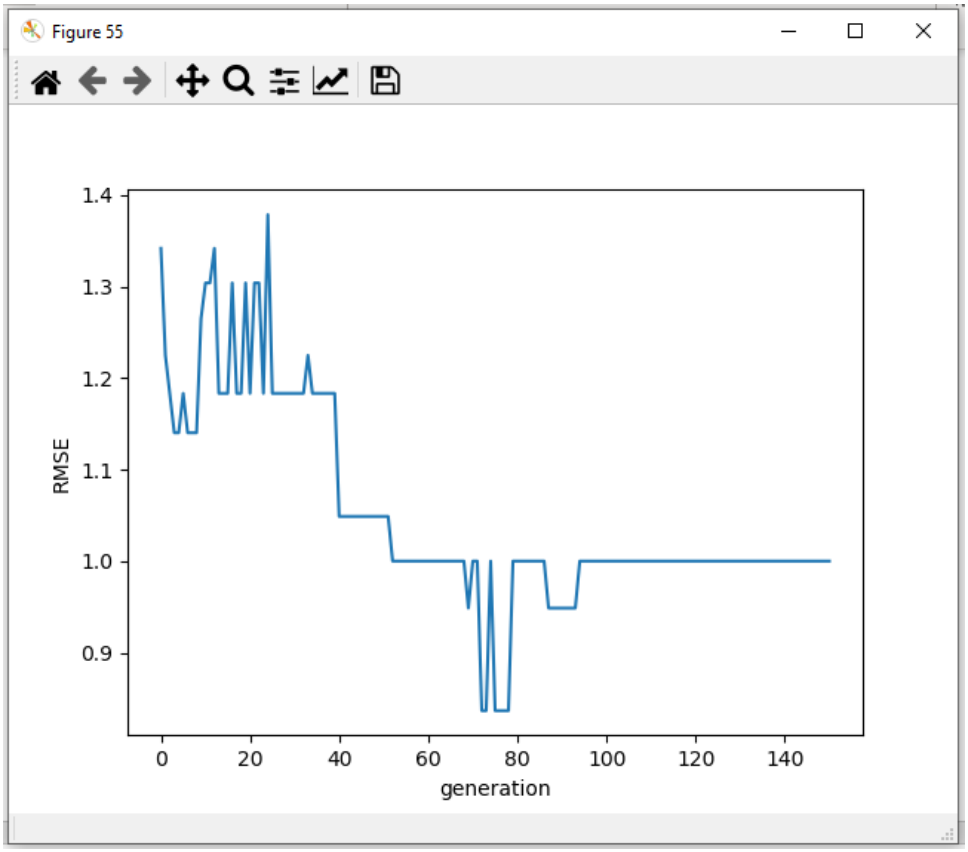


Figure 55
1.4
1.3
1.2
1.1
1.0
0.9
RMSE
0
20
40
60
80
100
120
140
generation

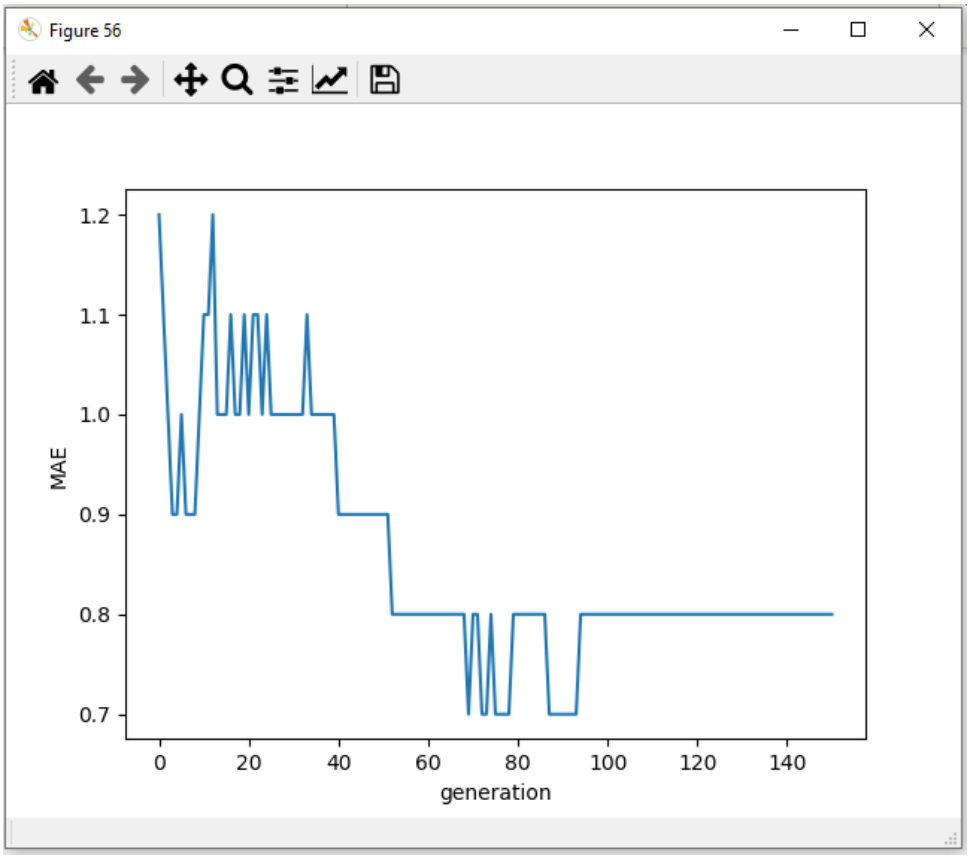


Figure 56
1.2
1.1
1.0
0.9
0.8
0.7
MAE
0
20
40
60
80
100
120
140
generation

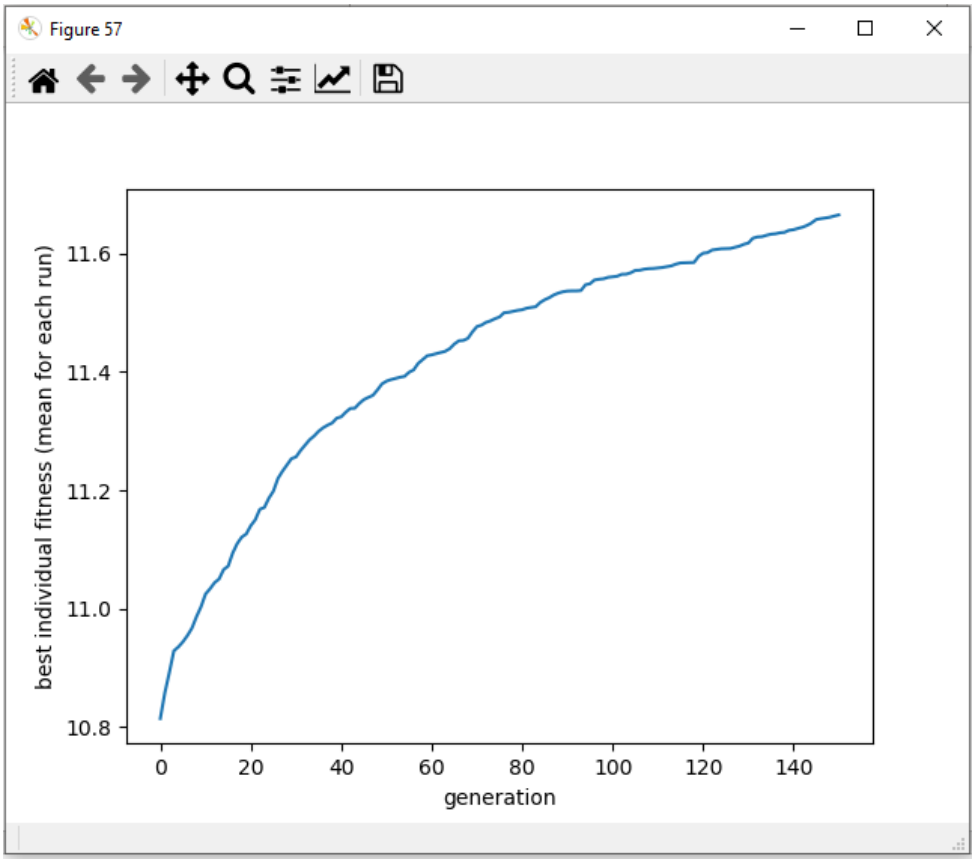


Figure 57
best individual fitness (mean for each run)
11.6
11.4
11.2
11.0
10.8
0
20
40
60
80
100
120
140
generation

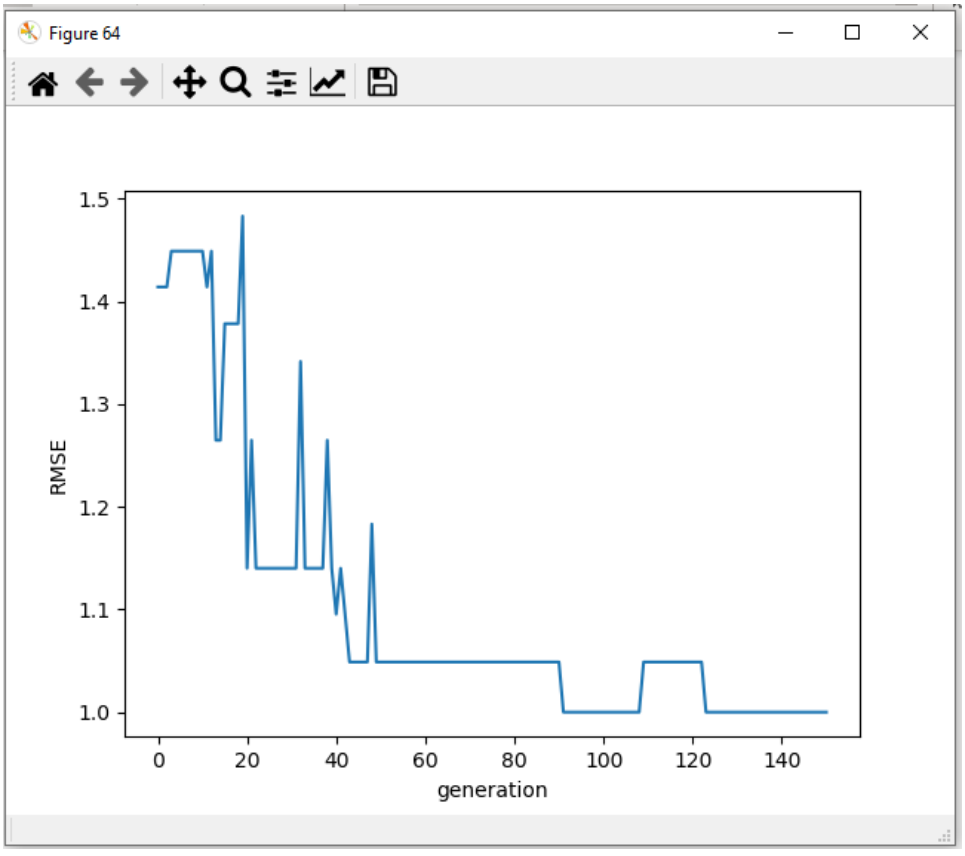


Figure 64
RMSE
1.5
1.4
1.3
1.2
1.1
1.0
0
20
40
60
80
100
120
140
generation

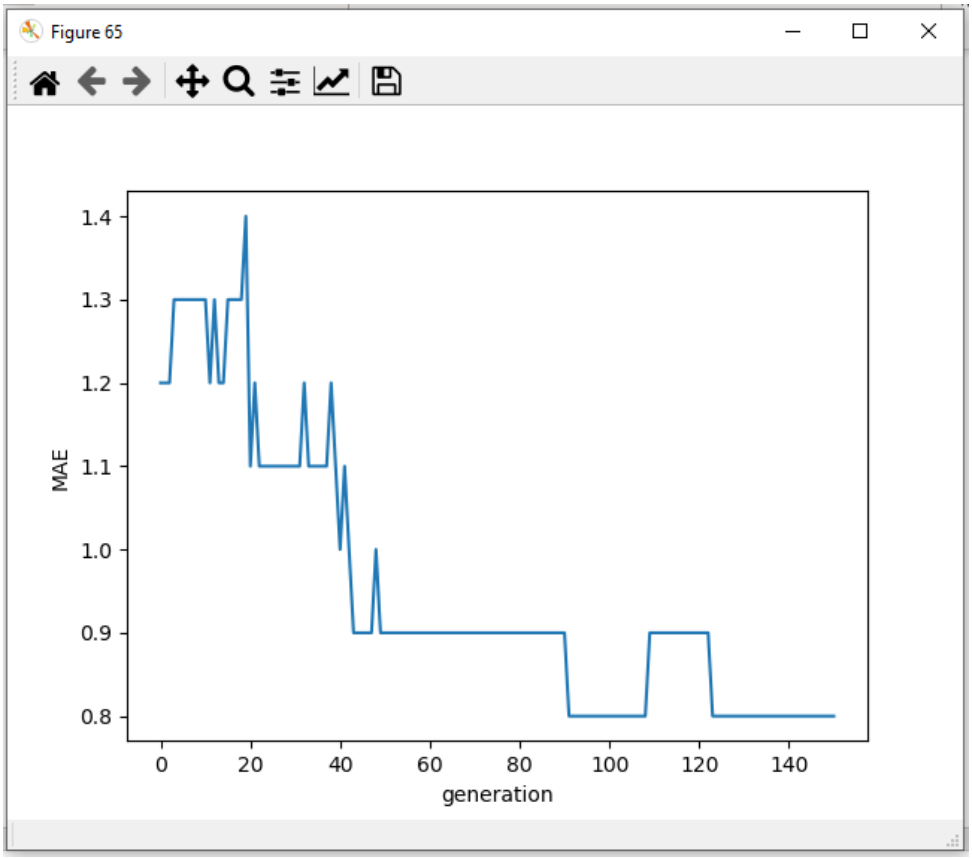


Figure 65
MAE
1.4
1.3
1.2
1.1
1.0
0.9
0.8
0
20
40
60
80
100
120
140
generation

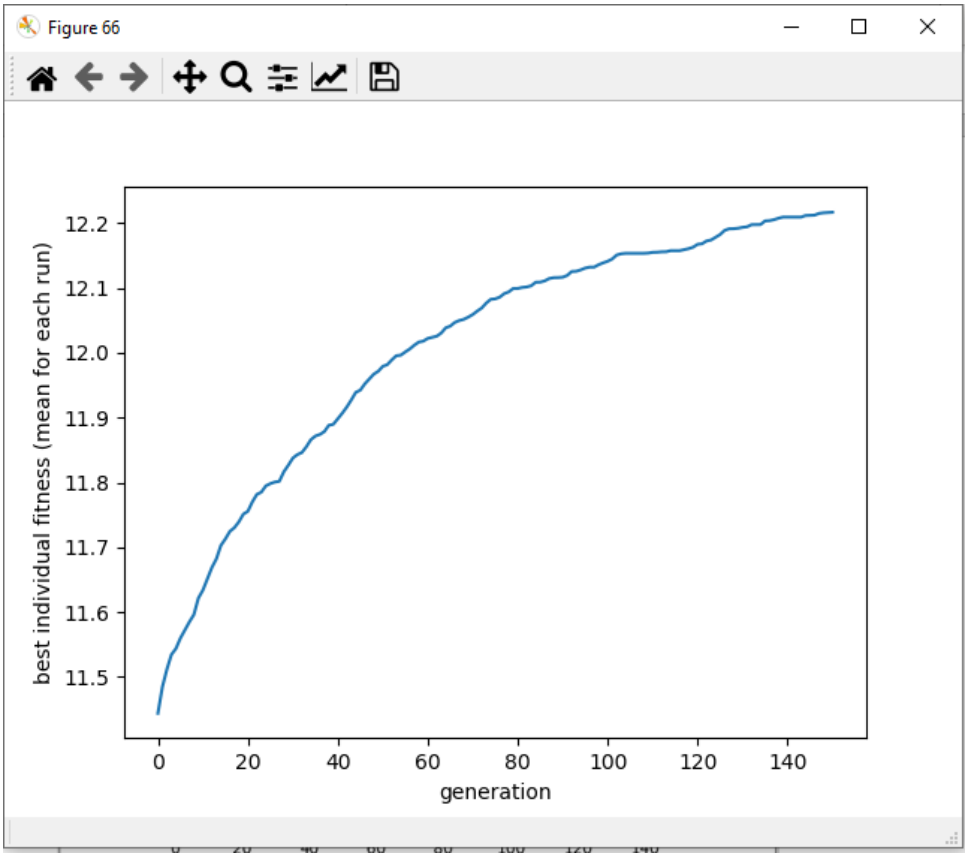


Figure 66
best individual fitness (mean for each run)
12.2
12.1
12.0
11.9
11.8
11.7
11.6
11.5
0
20
40
60
80
100
120
140
generation