SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU  
ELEKTROTEHNIČKI FAKULTET

Diplomski studij računarstva

*Laboratorijska vježba 3*

*Genetski algoritmi i Algoritam roja čestica*

*Rješavanje problema pronalaska minimuma n-dimenzionalne funkcije koristeći PSO i GA algoritme*

Denis Lazor, DRB

Osijek, 2020.

SADRŽAJ

[**1.** **UVOD** 1](#_Toc38202301)

[**2. PROBLEM PRONALASKA MINIMUMA N-DIMENZIONALNE FUNKCIJE** 2](#_Toc38202302)

[**2.1.** **Opis rješenja klasičnim algoritmom** 2](#_Toc38202303)

[**2.2.** **Genetski algoritam** 2](#_Toc38202304)

[**2.3.** **Opis rješenja genetskim algoritmom** 3](#_Toc38202305)

[**2.4.** **Algoritam roja čestica** 3](#_Toc38202306)

[**2.5.** **Opis rješenja algoritmom roja čestica** 4](#_Toc38202307)

[**2.6.** **Programski kod za automatsko izvođenje svih kombinacija** 5](#_Toc38202308)

[**3.** **REZULTATI** 10](#_Toc38202309)

[**3.1.** **Genetski algoritam** 11](#_Toc38202310)

[**3.1.1.** **Ovisnost o maksimalnoj apsolutnoj mutaciji** 11](#_Toc38202311)

[**3.1.2.** **Ovisnost o postotku mutacije** 13](#_Toc38202312)

[**3.1.3.** **Ovisnost o broju elitnih članova** 15](#_Toc38202313)

[**3.2.** **Algoritam roja čestica** 17](#_Toc38202314)

[**3.2.1.** **Ovisnost o inerciji** 17](#_Toc38202315)

[**3.2.2.** **Ovisnost o osobnom faktoru** 19](#_Toc38202316)

[**3.2.3.** **Ovisnost o socijalnom faktoru** 21](#_Toc38202317)

[**4.** **ZAKLJUČAK** 23](#_Toc38202318)

# **UVOD**

Na trećoj laboratorijskoj vježbi se koristeći genetski algoritam i algoritam roja čestica rješava problem pronalaska minimuma 5-D i 10-D Rastriginove funkcije.

Pri tome se mijenjaju parametri mutacije , elitizma i maksimalne apsolutne mutacije za genetski algoritam te inercija, osobni faktor i socijalni faktor za algoritam roja čestica.

# **2. PROBLEM PRONALASKA MINIMUMA N-DIMENZIONALNE FUNKCIJE**

## **Opis rješenja klasičnim algoritmom**

Jedno od mogućih rješenja je izračunavanje vrijednosti funkcije za sve točke u nekom intervalu i odabir najmanjeg rješenja što je ogroman broj točaka koji se povećava povećanjem dimenzija s čime se povećava i potreba za sve više resursa.

## **Genetski algoritam**

Genetski algoritam heuristička je metoda optimiranja koja imitira prirodni evolucijski proces. Evolucija je robustan proces pretraživanja prostora rješenja. Po načinu djelovanja ubraja se u metode usmjerenog slučajnog pretraživanja prostora rješenja u potrazi za globalnim optimumom.

Genetski algoritam najlakše je opisati njegovim elementima koji su objašnjeni u nastavku.

**Populacija** predstavlja skup jedinki, odnosno rješenja u i-tom koraku algoritma. Svaka jedinka predstavlja kromosom, odnosno jedno moguće rješenje zadanog problema. Svaka jedinka sastoji se od gena koji je nositelj jedne informacije. Geni se mogu kodirati na više načina, a najčešće se koristi binarno (gen može poprimiti samo dvije vrijednosti 0/1), cjelobrojno (gen poprima vrijednosti iz zadanog intervala) te permutacijsko kodiranje (gen poprima cjelobrojne vrijednosti iz zadanog intervala takve da kromosom sadrži sve vrijednosti iz intervala)

**Rekombinacija** predstavlja kombiniranje dva kromosoma (roditelja) u svrhu stvaranja novih i boljih kromosoma (potomka) . Rekombinacija može biti u jednoj točki, u više točaka ili uniformna. Za stvaranje nove populacije se najprije vrši selekcije stare populacije koja može biti generacijska (u rekombinaciji sudjeluju dobre jedinke) ili eliminacijska (u rekombinaciji sudjeluju lošiji kromosomi koje želimo eliminirati). Kako se najbolje jedinke populacije ne bi izgubile, moguće je uvesti mehanizam zaštite najboljih jedinki od bilo kakve izmjene ili eliminacije. Takav mehanizam se naziva **elitizam.**

**Mutacija** predstavlja promjenu nasumično odabranog jednog ili više gena kromosoma s ciljem unošenja nove informacije u populaciju koja će možda omogućiti izlazak iz lokalnog minimuma. U slučaju permutacijskog kodiranja, slučajno se odabire kromosom i zatim se ispremještaju njegovi geni.

Najvažniji element genetskog algoritma jest definiranje sustava vrijednosti koji će za svaku jedinku pokazati koliko je njegovo rješenje dobro. Takav element zove se funkcija dobrote ili **fitness funkcija**. Što je vrijednost fitness funkcije jedinke veća, jedinka će imati veću vjerojatnost preživljavanja i križanja. Dobro definirana fitness funkcija ključna je za proces selekcije.

## **Opis rješenja genetskim algoritmom**

Fitness vrijednost predstavlja vrijednost Rastriginove funkcije. Gen jedinke predstavlja jednu koordinatu u n-dimenzionalnom prostoru pri čemu se jedinka sastoji od n gena. Traži se skup od n koordinata za n-dimenzionalnu Rastriginovu funkciju koja će dati vrijednost funkcije približnu nuli (globalni minimum Rastriginove funkcije).

## **Algoritam roja čestica**

Algoritam roja čestica (engl**. Particle Swarm Optimization**, PSO) je biološki inspiriran metaheuristički algoritam za optimizaciju. PSO algoritmi jednako kao i genetski algoritmi (engl. Genetic Algorithms, GA) posjeduju populaciju sačinjenu od niza pojedinih mogućih rješenja koji se ovdje nazivaju čestice. Pošto PSO algoritmi nemaju mogućnost izravnog križanja pojedinih čestica kao što to mogu GA algoritmi putem operatora rekombinacije, čestice se ovdje ne dijele na pojedine nositelje informacije kao što su to geni kod genetskih algoritama. Čestice PSO algoritma se gibaju kroz područje pretraživanja koristeći informacije o vlastitom položaju u prostoru pretraživanja i brzini, te položaju trenutno najbolje čestice u roju. Pri tome u svom radu svaka čestica pamti slijedeće podatke:

**• Svoje do sada najbolje pronađeno rješenje problema**

**• Svoje trenutno rješenje problema**

**• Trenutno najbolje rješenje u roju kojemu pripada**

Na temelju ta tri podatka svaka čestica proračunava novu vlastitu brzinu koju dodaje trenutnom položaju i definira novi položaj promatrane čestice. Dakle, svaka čestica mijenja svoj položaj temeljem vlastitog iskustva, te iskustva bliskih susjeda (na taj se način modelira socijalna interakcija između čestica). Prethodno navedeni podaci se opisuju kao vektori n-dimenzionalnog prostora kojeg se pretražuje.

Također za izračunavanje vektora brzine potrebne su i sljedeće konstante:

**• c0 – mjera inercije – opisuje bitnost trenutnog smjera**

**• c1 – mjera individualnog faktora - opisuje mjeru individualnosti jedinke, potiče istraživanje prostora oko nekog prethodnog najboljeg rješenja promatrane čestice**

**• c2 – mjera socijalnog faktora - opisuje mjeru socijalnog utjecaja, potiče detaljnije istraživanje okoline trenutnog najboljeg rješenja pronađenog od svih čestica.**

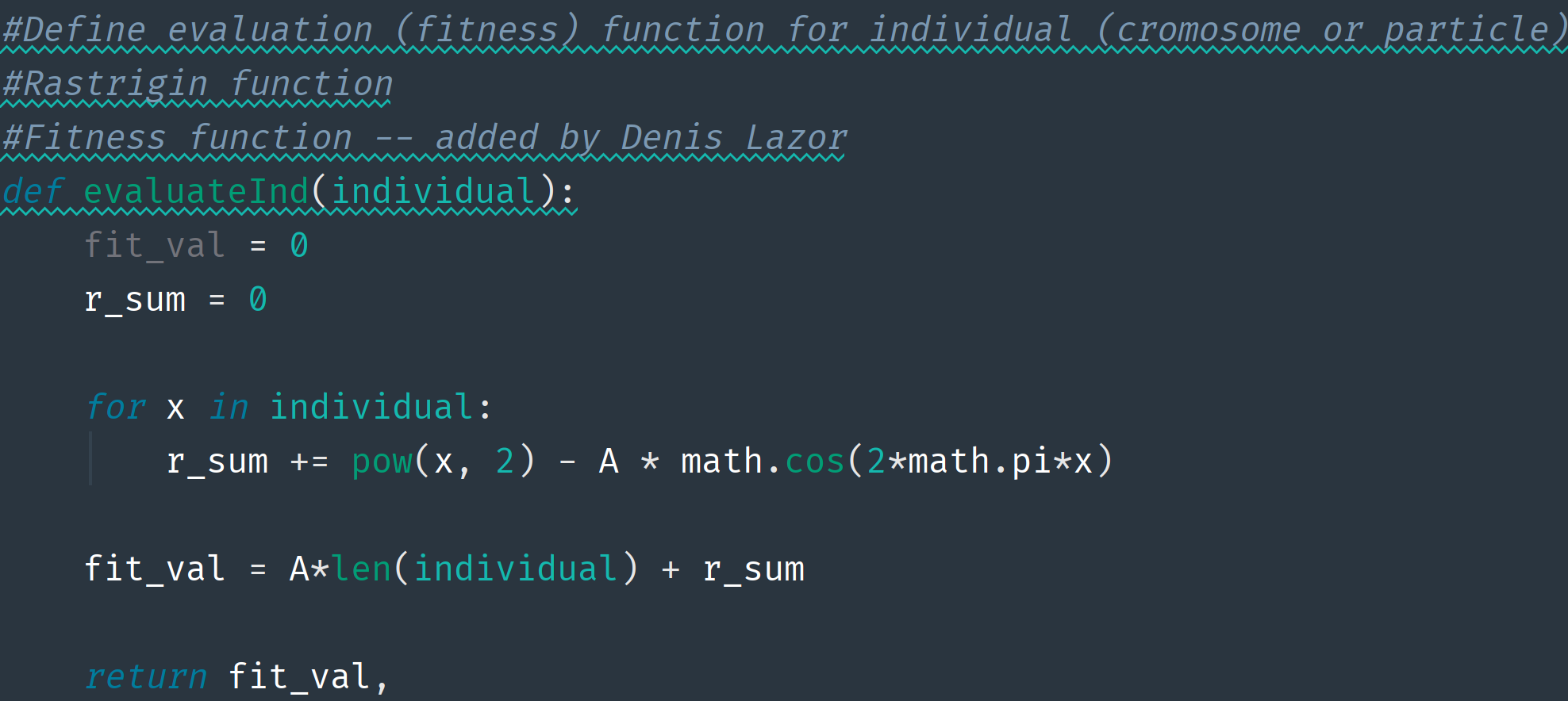
PSO algoritmi također zahtijevaju neku mjeru određivanja kvalitete pojedinog rješenja, fitnes funkciju tj. funkciju dobrote.

## **Opis rješenja algoritmom roja čestica**

PSO algoritam primijenjen na n-dimenzionalnu Rastriginovu funkciju predstavlja roj čestica koji pretražuju n-dimenzionalni prostor funkcije. Fitness vrijednost predstavlja vrijednost Rastriginove funkcije za položaj čestice u n-dimenzionalnom prostoru pretraživanja. Cilj je naći česticu što bližu globalnom minimumu funkcije.

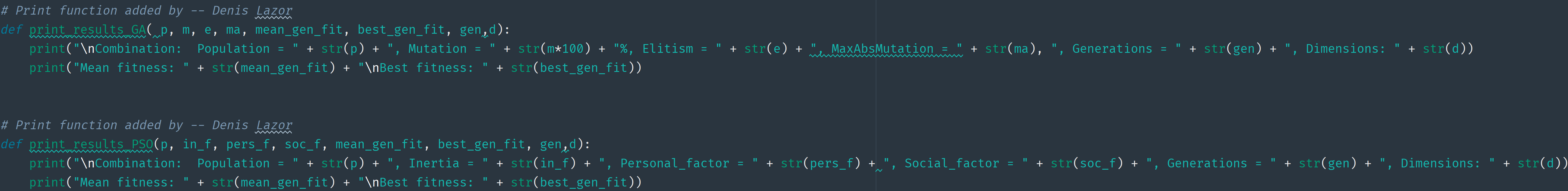
## **Programski kod za automatsko izvođenje svih kombinacija**

U postojeći kod dodane su različite funkcionalnosti koje su detaljno komentirane u kodu i sadrže ''Added by Denis Lazor'' komentare. Na slici 2.6.1. je prikazana funkcija za izračun fitness vrijednosti jedinke koja predstavlja kromosom kod genetskog algoritma i česticu kod algoritma roja čestica. Vrijednost fitness funkcije je jednaka vrijednosti Rastriginove funkcije za koordinate zadane jedinke.



Slika 2.6.1. Definiranje fitness funkcije

Funkcije prikazane na slici 2.6.2. predstavljaju funkcije za prikaz rezultata oba algoritma na ekranu.



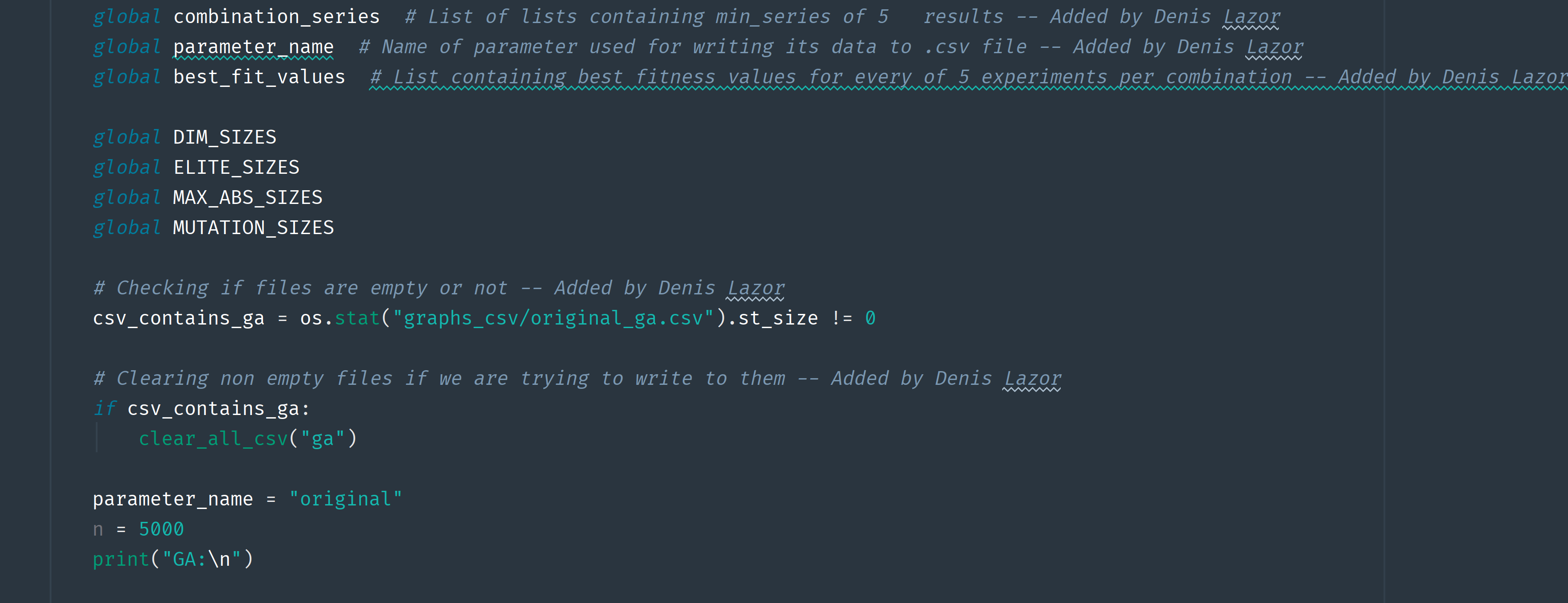
Slika 2.6.2. Prikaz print() funkcija

Funkcije prikazane na slici 2.6.3. prikazuju funkcije za brisanje sadržaja .csv datoteka te funkciju za pisanje sadržaja u .csv datoteke ovisno o tipu algoritma(GA i PSO) realiziranih dodavanjem nastavaka na ime datoteka za brisanje ili datoteke za pisanje ovisno o predanom argumentu funkciji pri tome koristeći biblioteku za rad s .csv datotekama.



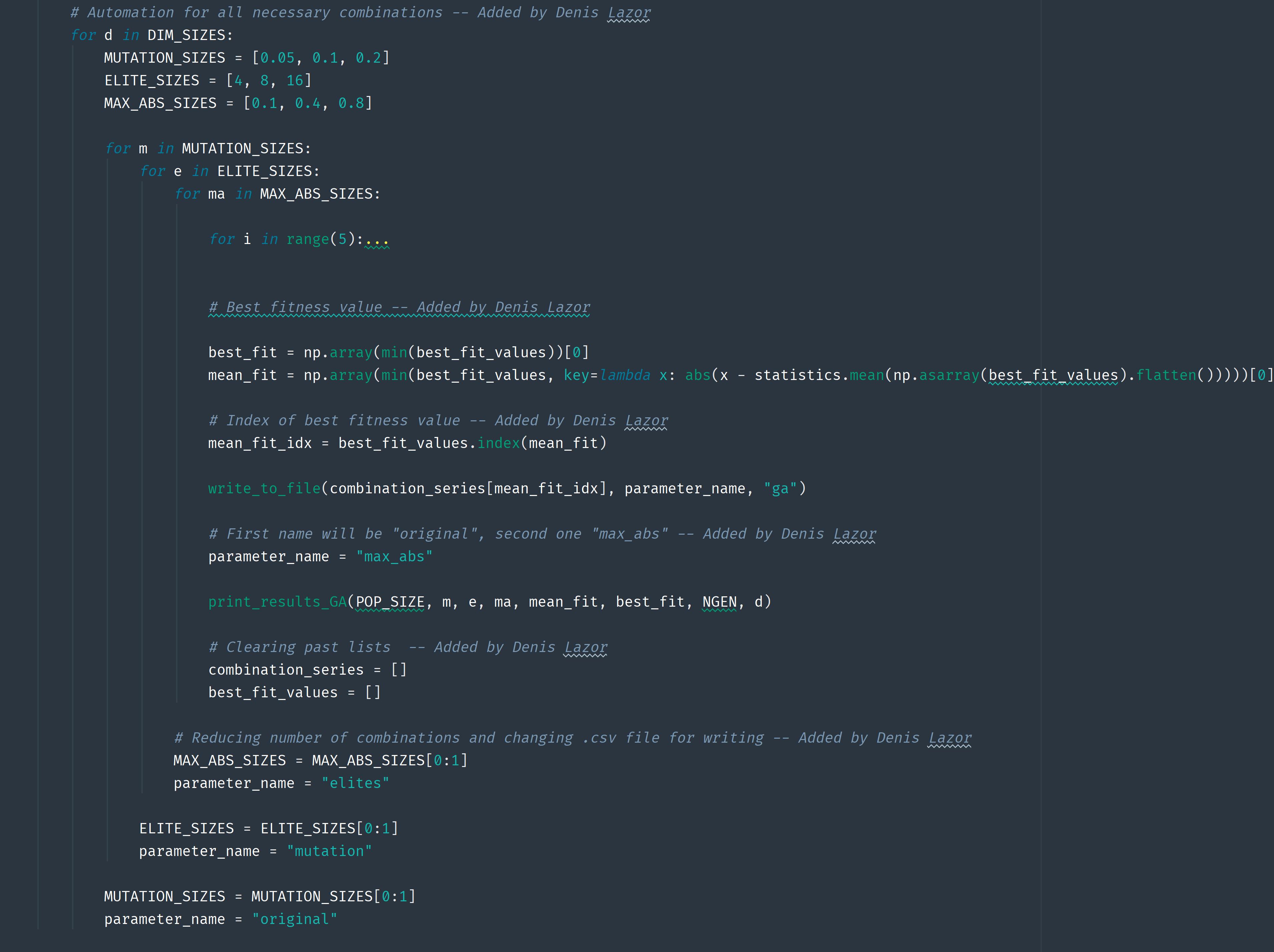
Slika 2.6.3. Funkcija za brisanje i pisanje sadržaja csv datoteka

Na slici 2.6.4. je prikazan dio *btnStartGA\_Click()* funkcije koji predstavlja definiranje potrebnih varijabli, provjeru i brisanje sadržaja .csv datoteka, postavljanje broja generacija i ime datoteke za prvi zapis('parameter\_name').



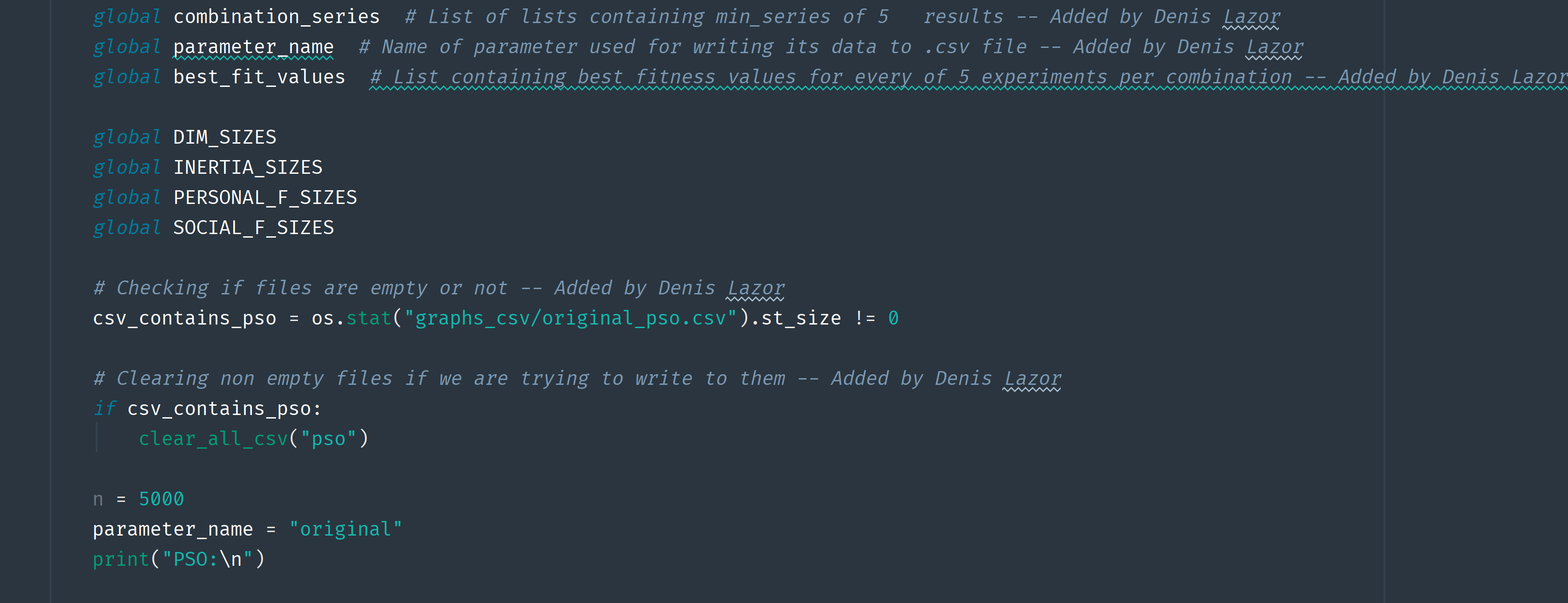
Slika 2.6.5. Dio btnStartGA\_Click() funkcije

Drugi dio iste funkcije je prikazan na slici 2.6.6. Svaki pokus se ponavlja 5 puta za svaku kombinaciju. Odrađeni su samo slučajevi u kojima se jedan parametar mijenja, a ostali su fiksni i tako za svaki parametar. Na kraju se polja parametara skraćuju na vrijednost prvog člana radi držanja fiksne vrijednosti dok se sljedeći parametar mijenja no kada se dimenzija Rastriginove funkcije promijeni, polja se ponovo inicijaliziraju radi ponavljanja istog postupka. Na kraju svakih 5 iteracija traži se najbolja i srednja fitness vrijednost u polju fitness vrijednosti od 5 pokusa (najbolja od svakog pokusa) . Zatim se traži indeks srednje fitness vrijednosti kako bi se iz polja koji sadrži vektore kretanja vrijednosti najbolje fitness funkcije ovisno o generacijama za svaki pokus izvukao onaj vektor s srednjom fitness vrijednošću. Zatim se ona zapisuje u .csv datoteku i na ekran se ispisuju rezultati te se mijenja ime parametra za zapis u datoteku('max\_abs' string se svaki put 'pregazi' nakon izlaska iz unutarnje petlje tako da se greškom ne zapisuje u njega kada ne treba). Te se na kraju briše sadržaj potrebnih globalnih polja. '…' predstavlja originalni kod funkcije.

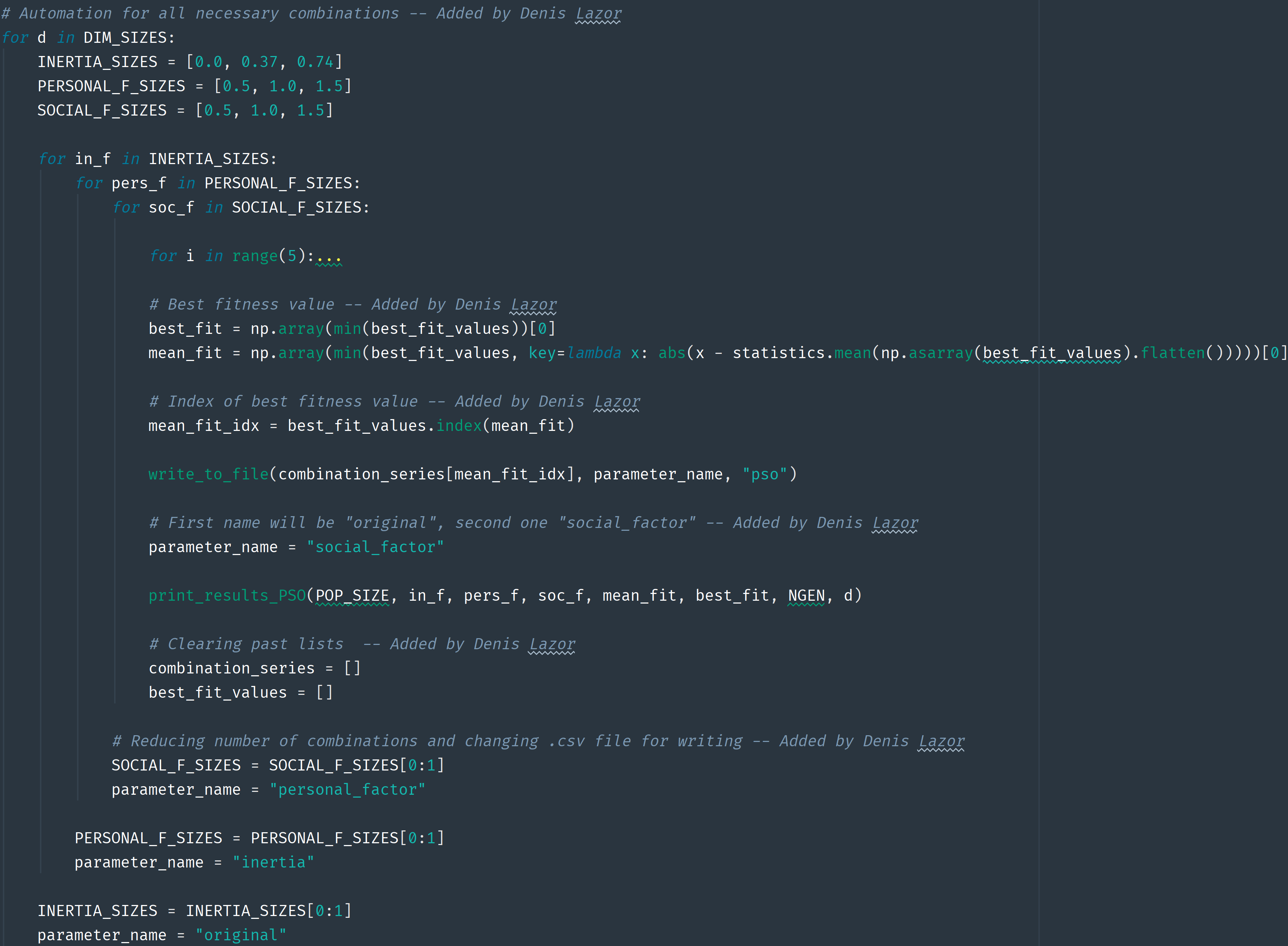


Slika 2.6.6. Dio btnStartGA\_Click() funkcije

Na slici 2.6.7. i 2.6.8. je prikazan isti postupak za automatsko prolaženje kombinacija, ali sada koristeći PSO algoritam i njegove parametre.

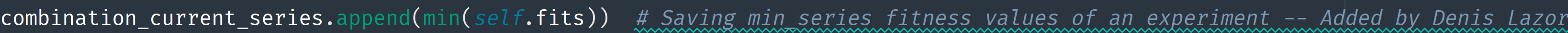


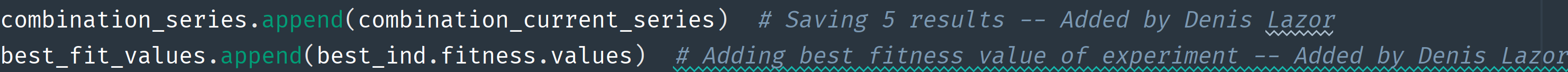
Slika 2.6.7. Dio btnStartPSO\_Click() funkcije



Slika 2.6.8. Dio btnStartPSO\_Click() funkcije

Spremanje vektora kretanja fitness funkcije i najboljih fitness vrijednosti pokusa se kao i u prethodnim vježbama odvija unutar *Evolve()* funkcije. Razlog spremanja u drugo polje su poteškoće pri pokušaju rada s *QLineSeries* objektima.





Slika 2.6.9. Dio evolveGA/PSO() funkcije

.

.

# **REZULTATI**

Mjerenje je za svaku kombinaciju provedeno 5 puta sa sljedećim parametrima :

GA:

• Maksimalna apsolutna mutacija: 0.1, 0.4, 0.8

• Mutacija: 5%, 10%, 20%

• Broj elitnih članova: 4, 8, 16

PSO:

• Inercija : 0.00, 0.37, 0.74

• Osobni faktor: 0.5, 1.0, 1.5

• Socijalni faktor: 0.5, 1.0, 1.5

Fiksni za sve eksperimente: Populacija(100), Broj generacija(5000)

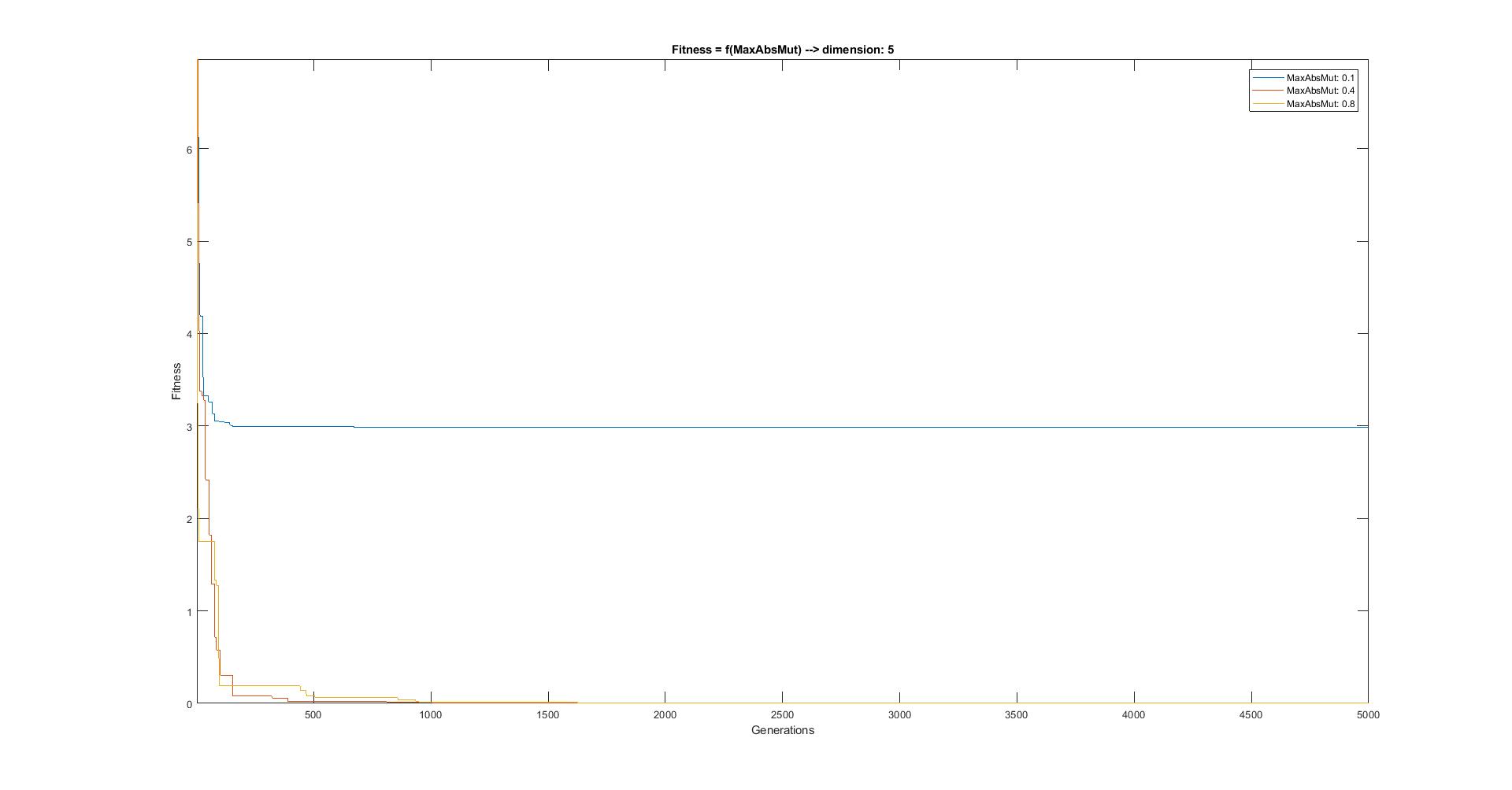
Dok se jedan parametar mijenja ostali su na minimalnim fiksnim vrijednostima.

## **Genetski algoritam**

### **Ovisnost o maksimalnoj apsolutnoj mutaciji**

Tablica 1 Ovisnost o maksimalnoj apsolutnoj mutaciji 5-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Broj elitnih članova** | 4 | | |
| **% Postotak mutacije** | 5 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | 0.4 | 0.8 |
| **Srednja vrijednost fitness funkcije** | 2.98496 | 0.00056 | 0.00208 |
| **Najbolje rješenje** | 0.99503 | 0.00037 | 0.00046 |



Slika 3.1.1.1. Prikaz ovisnosti o maksimalnoj apsolutnoj mutaciji 5-D

Tablica 2 Ovisnost o maksimalnoj apsolutnoj mutaciji 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Broj elitnih članova** | 4 | | |
| **% Postotak mutacije** | 5 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | 0.4 | 0.8 |
| **Srednja vrijednost fitness funkcije** | 0.99499 | 0.00053 | 0.00306 |
| **Najbolje rješenje** | 0.00008 | 0.000099 | 0.00018 |

Slika na kojoj se prikazuje snimka zaslona

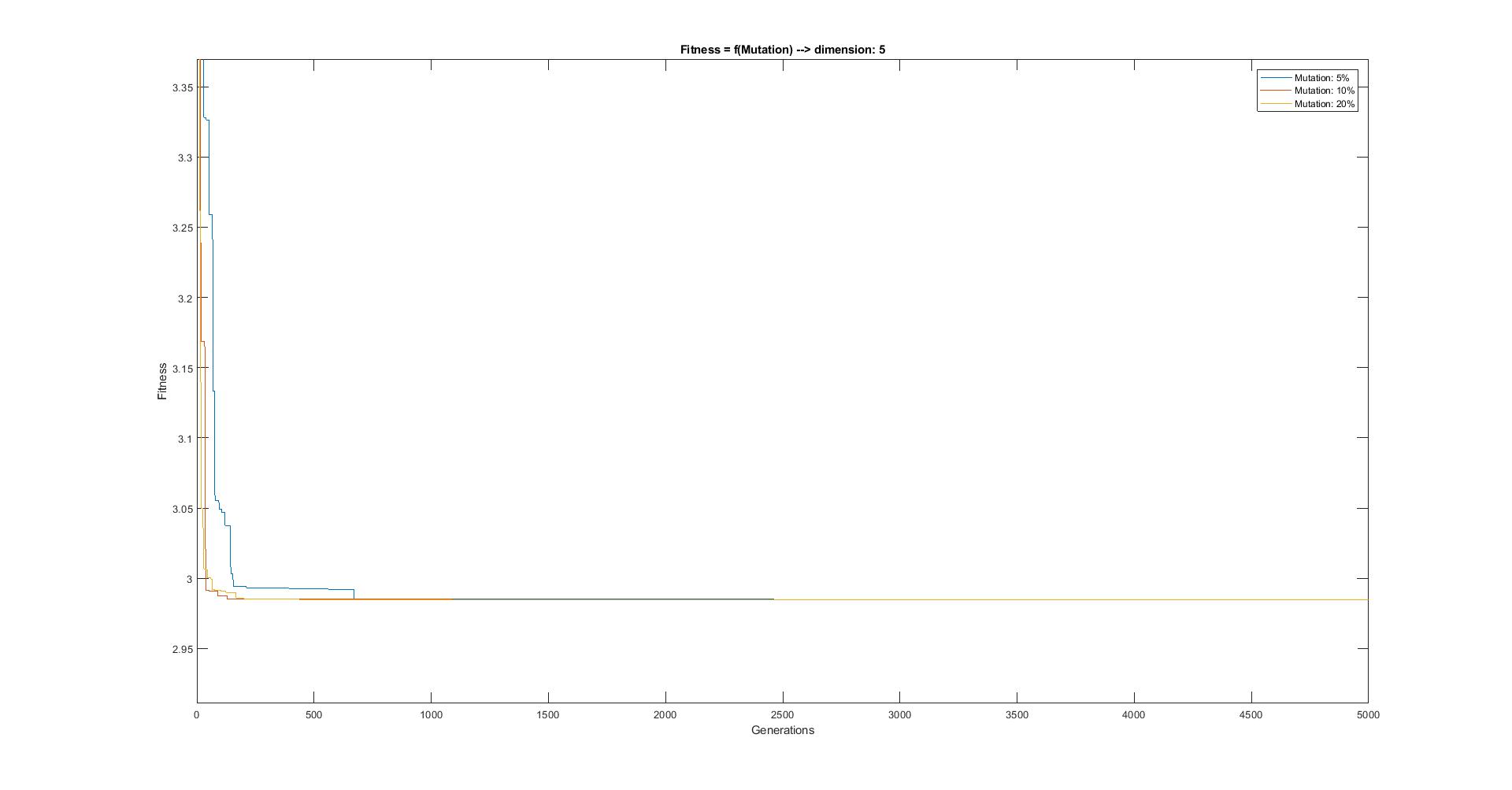
Opis je automatski generiran

Slika 3.1.1.2. Prikaz ovisnosti o maksimalnoj apsolutnoj mutaciji 10-D

### **Ovisnost o postotku mutacije**

Tablica 3 Ovisnost o postotku mutacije 5-D

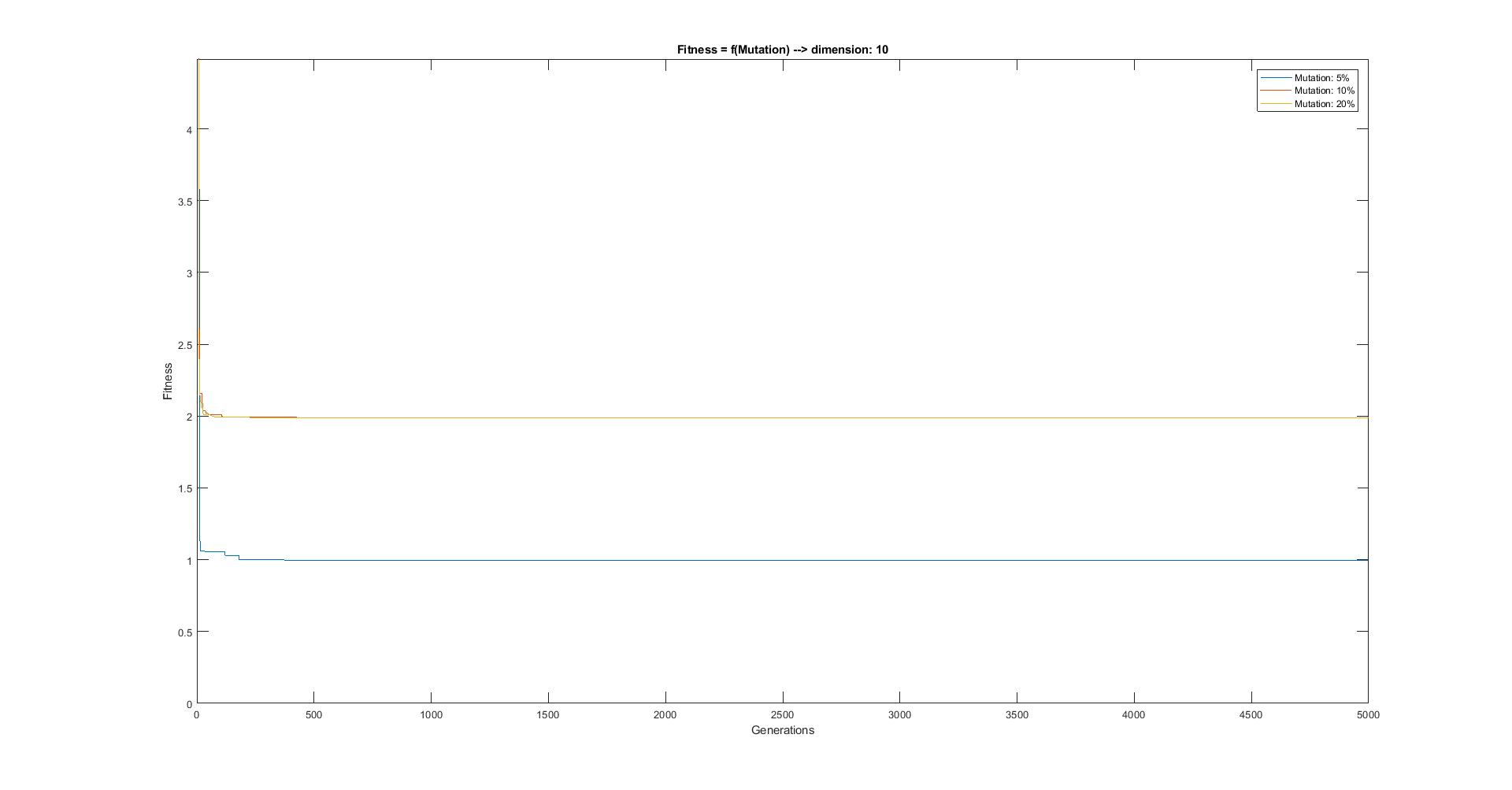
|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Broj elitnih članova** | 4 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | | |
| **% Postotak mutacije** | 5 | 10 | 20 |
| **Srednja vrijednost fitness funkcije** | 2.98496 | 2.98488 | 2.98488 |
| **Najbolje rješenje** | 0.99503 | 1.98993 | 2.75216 |



Slika 3.1.2.1. Prikaz ovisnosti o postotku mutacije 5-D

Tablica 4 Ovisnost o postotku mutacije 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Broj elitnih članova** | 4 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | | |
| **% Postotak mutacije** | 5 | 10 | 20 |
| **Srednja vrijednost fitness funkcije** | 0.99499 | 1.98993 | 1.98992 |
| **Najbolje rješenje** | 0.00008 | 0.000005 | 0.99496 |



Slika 3.1.2.2. Prikaz ovisnosti o postotku mutacije 10-D

### **Ovisnost o broju elitnih članova**

Tablica 5 Ovisnost o broju elitnih članova 5-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | | |
| **% Postotak mutacije** | 5 | | |
| **Broj elitnih članova** | 4 | 8 | 16 |
| **Srednja vrijednost fitness funkcije** | 2.98496 | 1.98995 | 2.98489 |
| **Najbolje rješenje** | 0.99503 | 0.99496 | 0.99496 |

Slika na kojoj se prikazuje čamac, voda, prijenosnik, velik

Opis je automatski generiran

Slika 3.1.3.1. Prikaz ovisnosti o broju elitnih članova 5-D

Tablica 6 Ovisnost o broju elitnih članova 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Maksimalna apsolutna mutacija** | 0.1 | | |
| **% Postotak mutacije** | 5 | | |
| **Broj elitnih članova** | 4 | 8 | 16 |
| **Srednja vrijednost fitness funkcije** | 0.99499 | 1.99000 | 1.98995 |
| **Najbolje rješenje** | 0.00008 | 0.99498 | 0.99498 |

Slika na kojoj se prikazuje prijenosnik, voda, čamac, računalo

Opis je automatski generiran

Slika 3.1.3.2. Prikaz ovisnosti o broju elitnih članova 10-D

## **Algoritam roja čestica**

### **Ovisnost o inerciji**

Tablica 7 Ovisnost o inerciji 5-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Osobni faktor** | 0.5 | | |
| **Socijalni faktor** | 0.5 | | |
| **Inercija** | 0 | 0.37 | 0.74 |
| **Srednja vrijednost fitness funkcije** | 5.99019 | 2.00693 | 3.97984 |
| **Najbolje rješenje** | 3.00048 | 1.98992 | 1.98992 |

Slika na kojoj se prikazuje snimka zaslona, muškarac

Opis je automatski generiran

Slika 3.2.1.1. Prikaz ovisnosti o inerciji 5-D

Tablica 8 Ovisnost o inerciji 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Osobni faktor** | 0.5 | | |
| **Socijalni faktor** | 0.5 | | |
| **Inercija** | 0 | 0.37 | 0.74 |
| **Srednja vrijednost fitness funkcije** | 3.68376 | 3.97984 | 4.97479 |
| **Najbolje rješenje** | 2.02726 | 1.98992 | 2.98488 |

Slika na kojoj se prikazuje snimka zaslona, voda, muškarac, čamac

Opis je automatski generiran Slika 3.2.1.2. Prikaz ovisnosti o inerciji 5-D

### **Ovisnost o osobnom faktoru**

Tablica 9 Ovisnost o osobnom faktoru 5-D

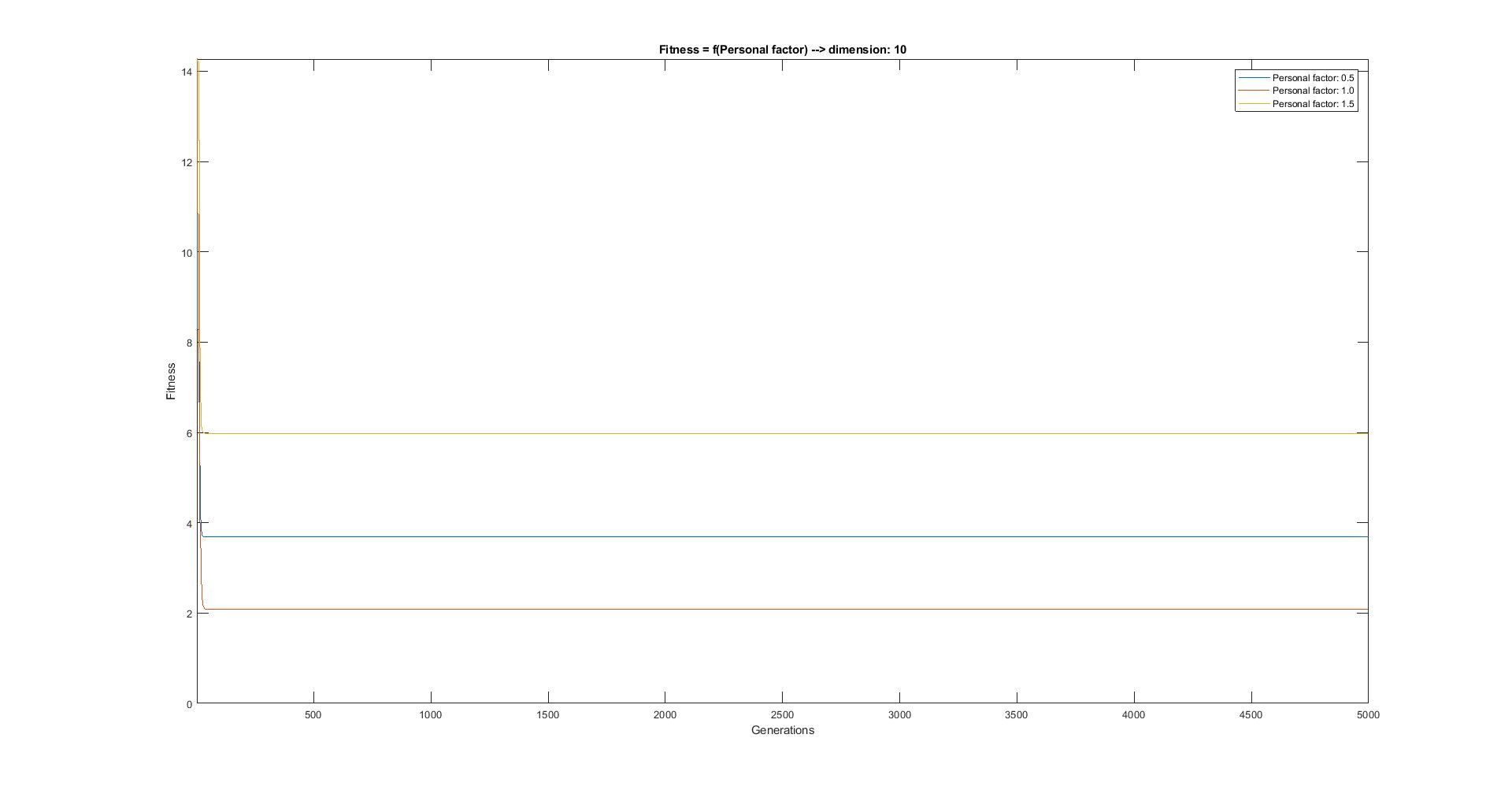
|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Inercija** | 0.0 | | |
| **Socijalni faktor** | 0.5 | | |
| **Osobni faktor** | 0.5 | 1.0 | 1.5 |
| **Srednja vrijednost fitness funkcije** | 5.99019 | 6.14521 | 2.00693 |
| **Najbolje rješenje** | 3.00048 | 1.36107 | 1.98992 |

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran Slika 3.2.2.1. Prikaz ovisnosti o osobnom faktoru 5-D

Tablica 10 Ovisnost o osobnom faktoru 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Inercija** | 0.0 | | |
| **Socijalni faktor** | 0.5 | | |
| **Osobni faktor** | 0.5 | 1.0 | 1.5 |
| **Srednja vrijednost fitness funkcije** | 3.68376 | 2.08073 | 5.97823 |
| **Najbolje rješenje** | 2.02726 | 1.13136 | 1.99032 |

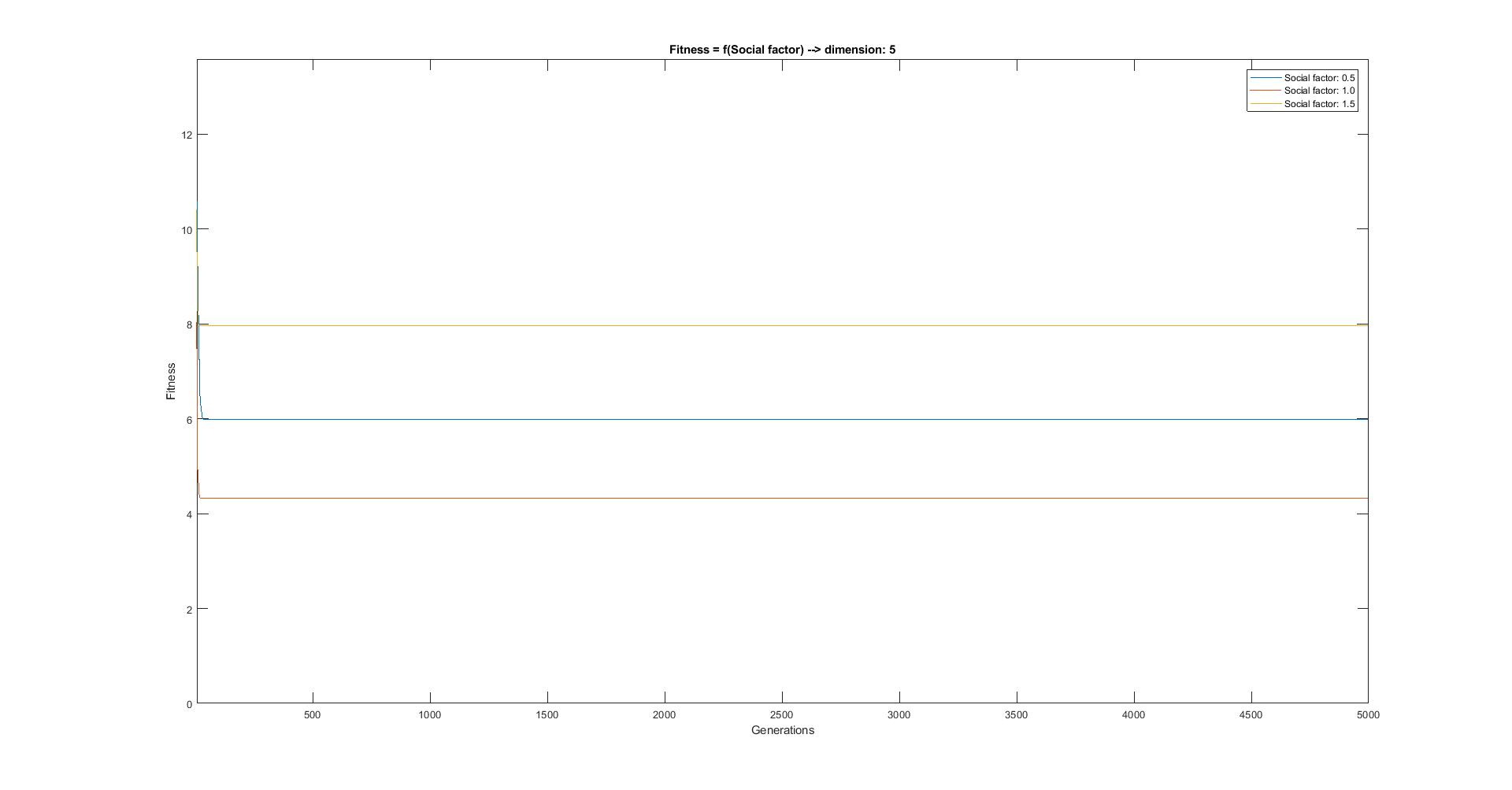


Slika 3.2.2.2.. Prikaz ovisnosti o osobnom faktoru 5-D

### **Ovisnost o socijalnom faktoru**

Tablica 11 Ovisnost o socijalnom faktoru 5-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 5 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Inercija** | 0.0 | | |
| **Osobni faktor** | 0.5 | | |
| **Socijalni faktor** | 0.5 | 1.0 | 1.5 |
| **Srednja vrijednost fitness funkcije** | 5.99019 | 4.32776 | 7.96162 |
| **Najbolje rješenje** | 3.00048 | 4.01696 | 3.97984 |

 Slika 3.2.3.1. Prikaz ovisnosti o socijalnom faktoru 5-D

Tablica 12 Ovisnost o socijalnom faktoru 10-D

|  |  |  |  |
| --- | --- | --- | --- |
| **Broj dimenzija** | 10 | | |
| **Broj generacija** | 5000 | | |
| **Veličina populacije** | 100 | | |
| **Inercija** | 0.0 | | |
| **Osobni faktor** | 0.5 | | |
| **Socijalni faktor** | 0.5 | 1.0 | 1.5 |
| **Srednja vrijednost fitness funkcije** | 3.68376 | 7.04651 | 8.95641 |
| **Najbolje rješenje** | 2.02726 | 3.09964 | 1.98992 |

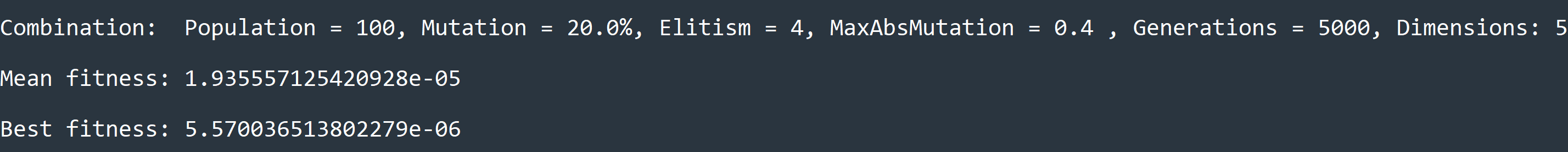
Slika na kojoj se prikazuje snimka zaslona, sjedenje, voda, ptica

Opis je automatski generiran Slika 3.2.3.2. Prikaz ovisnosti o socijalnom faktoru 5-D

# **ZAKLJUČAK**

Na temelju provedenih mjerenja može se izvesti zaključak o ovisnosti pojedinog parametra na konvergenciju rješenja genetskog algoritma i algoritma roja čestica.

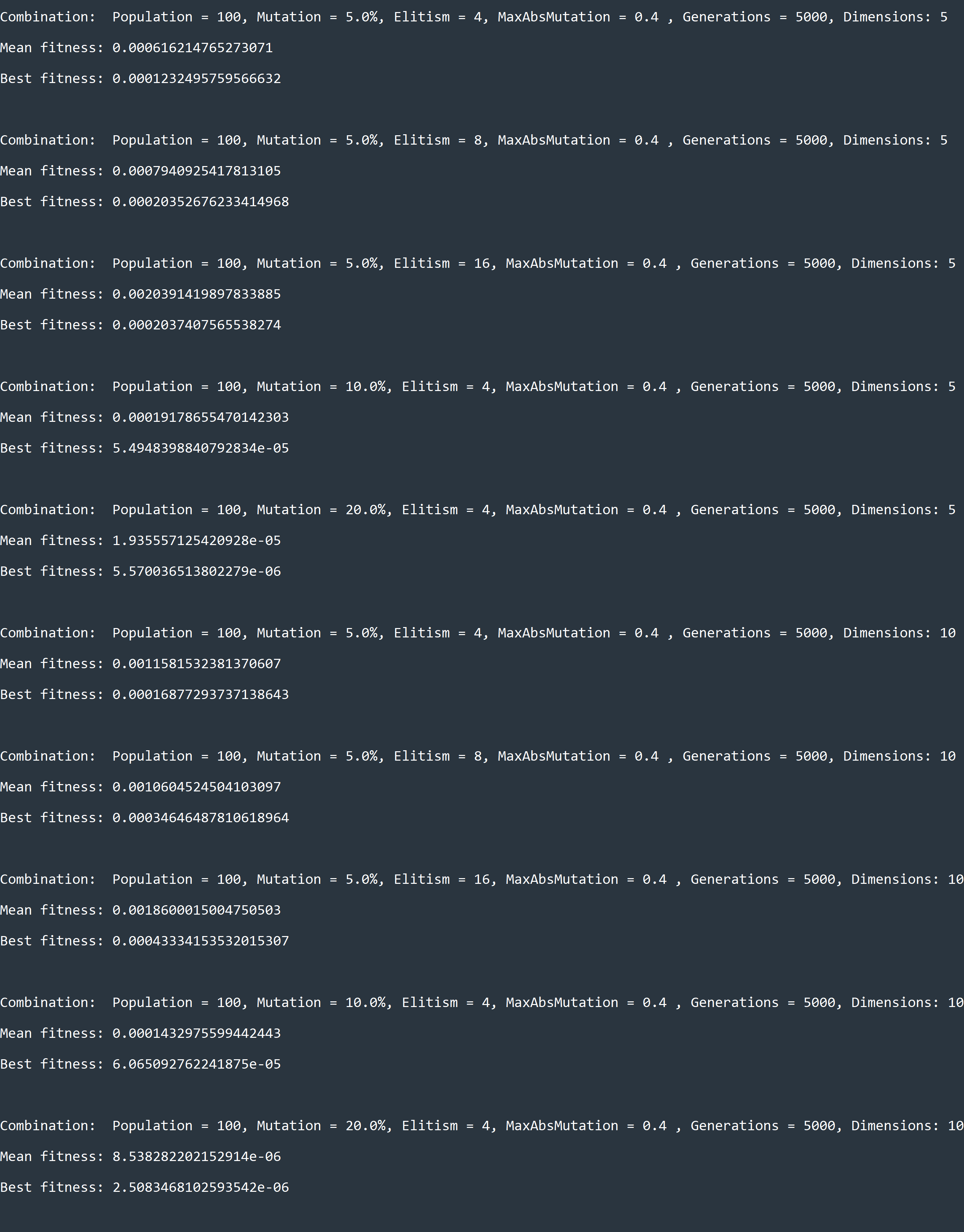
Općenito je genetski algoritam dao bolje rezultate od algoritma roja čestica s time da su jedini zadovoljavajući rezultati bili za genetski algoritam sa većom maksimalnom apsolutnom mutacijom. Na grafovima se vidi da algoritmi imaju običaj zapeti u lokalnom minimumu, a mutacija kod GA pomaže izaći iz tog minimuma kako bi algoritam dospio što bliže globalnom minimumu. Jedan o mogućih problema zapinjanja PSO algoritma u lokalnom minimumu je držanje inercije na vrijednosti 0 za većinu kombinacija. Bez inercije algoritam nema običaj izlaziti iz lokalnog područja pretraživanja dok velika vrijednost pridonosi mogućnosti 'prelijetanja' globalnog minimuma. Stoga su provedeni dodani eksperimenti sa inercijom od 0.74. Prosječna rješenja su i dalje bila slična, dok su najbolja rješenja pokusa bila bolja zato jer sada algoritam može lakše doći do područja globalnog minimuma, ali ga isto tako može i preskočiti. Algoritam je čak jednom i pogodio sam lokalni minimum( fitness = 0) , ali se to dešava jako rijetko i bit optimizacijskih algoritama je da daju konstantno dobre rezultate za neki skup parametara. Male vrijednosti osobnog faktora pridonosi mogućnosti prilagodbe ostatku roja te zapinjanju u lokalnom minimumu. Male vrijednosti socijalnog faktora dovode do nedostatka komunikacije između čestica tako da idu neovisno jedna o drugoj što poništava samu ideju ovog algoritma. Ako bi oba faktora bila na vrijednosti 0, čestice bi letile do granice prostora pretraživanja trenutnom brzinom nakon čega bi teško pronašle globalni minimum. Iz grafova se vidi da najbolje rezultate daju srednje vrijednosti oba faktora kada podjednako ovisi i o svojim položajima i o položajima ostalih čestica što se s veličinom prostora pretraživanja može prilagoditi. Što se tiče genetskog algoritma, optimalan broj elita za manje dimenzije je 8 dok je za veće 4. Ovisnost o postotku mutacije je podjednak s time da je razlog tomu držanje maksimalne apsolutne mutacije na minimumu. Zbog čega su i tu provedena dodatna mjerenja gdje su sve kombinacije ponovljene sa maksimalnoj apsolutnoj mutaciji jednakoj 0.4.

Iz tih rezultata se vidi da elitizam ne utječe previše na rezultate dok veliki postotak mutacija daje nabolje rezultate i svi su rezultati prilično dobri ( ispod 1). Dakle sam postotak mutacije i apsolutna mutacija imaju veliki utjecaj na ovakav tip problema gdje algoritmi imaju običaj zapeti u lokalnim ekstremima pa pomoću mutacija izlaze van. Kod PSO algoritama se to izvodi većom inercijom, ali tada se puno teže pogodi sam globalni minimum pa PSO zna puno češće preskočiti globalni minimum nego ga pogoditi. PSO općenito brže konvergira rješenju, ali je to rješenje najčešće lošije od rješenja GA algoritma.

Na problemu ovakve prirode GA daje bolje rezultate od PSO. PSO radi bolje na problemima kontinuirane prirode.

Prilog:

Ponovljeni eksperiment GA:



PSO s ispisanim jedinkama:

