Descriere\_Problema1

* Programul primeste ca parametrii: populatia(adica numarul de indivizi ai unei populatii), numarul de generatii si probabilitatea de incrucisare
* Programul contine o clasa Individ unde este retinuta o solutia sub forma x1, x2, f(x1,x2), un modul unde se afiseaza datele in fisier, un modul cu functiile utile rezolvarii problemei:
  + - Prima functie calculeaza f(x1,x2) unde x1 si x2 sunt dati ca parametru functiei.
    - A doua functie allege 2 indivizi random din populatie si-l returneaza pe cel mai bun.
    - A treia functie efectueaza o mutatie asupra unui individ cu o anumita probabilitate. Se alege random un numar din intervalul in care se afla x1 respectiv x2 si se inlocuieste x1/x2 cu numarul ales.
    - A patra functie primeste 2 indivizi si rezulta un inidivid prin incurcisarea celor 2.
    - Functia initializare populatie initializeaza random o populatie.
    - Ultima functie calculeaza minimul si maximul dintr-o populatie.

Mai avem un modul cu algoritmul:

1. Se genereaza automat n indivizi random pentru prima generatie. Pentru fiecare individ se alege aleator un alt individ, cei 2 se incurciseaza de unde rezulta un copil. Daca copilul este mai bun ca si individul initial se inlocuieste individul cu copilul. Se face o mutatie asupra copilului si se procedeazaz analog cu copilul si mutatia. Din fiecare generatie se alege minimul si maximul si se adauga intr un vector.

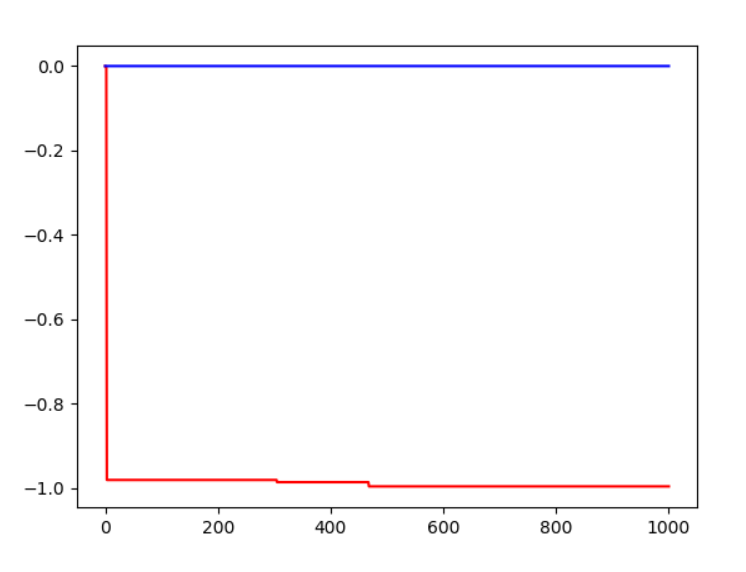
Ultimul modul este main-ul care colecteaza de la tastatura parametrii.

Pentru functia (13) care trebuie minimizata s-au obtinut urmatoarele date:

Minimul functiei este : 😊 -1 😊

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Numar rulari | Marime  Populatie | Numar  Generatii | Probabilitate  Mutatie | Best Solutie | Avg solutie |
| 10 | 500 | 500 | 0.5 | -0.99993 | -0.98270 |
| 0.7 | -0.99934 | -0.98110 |
| 400 | 0.6 | -0.99977 | -0.98152 |
| 600 | 0.4 | -0.99997 | -0.98986 |
| 700 | 500 | 0.5 | -0.99998 | -0.98769 |
| 400 | 0.5 | -0.99922 | -0.98410 |
| 600 | 0.5 | -0.99995 | -0.99357 |
| 1000 | 500 | 0.8 | -0.99977 | -0.98110 |
| 600 | 0.6 | -0.99939 | -0.99432 |
| 1000 | 0.5 | -0.99999 | -0.99498 |

* Datele pentru care s-au facut observatiile din tabel se gasesc in fisierul Date1.txt
* Cu cat crestem populatia si numarul de generatii cu atat se obtin solutii mai optime dar si timpul este mai lung. Acest algoritm este mai putin optim decat cel obtinut cu PSO. Coeficientul de mutatie este optim sa fie in jur de 0.5 ca sa existe un echilibru intre exporare si exploatare. Din tabel se observa ca generatiile crescute conduc la un rezultat mai bun decat populatia crescuta.
* O imbunatatire a algoritmului este de a nu pelca cu solutii random, de a pleca cu solutii obtinute de unul din algoritmii de HC, TS, SN.



Se oberva ca dupa 500 de iteratii algoritmul gaseste foarte greu o solutie mai optima sau posibil, s-a blocat intr-un minim local. O imbunatatiere ar fi de a parasi whil-ul atunci cand dupa aprox. 200 de generatii nu se mai imbunatateste solutia optima.