Vizualizacija algoritma optimizacije rojem čestica

Ivan Rep

21. siječnja 2022.

1 Algoritam optimizacije rojem čestica

Algoritam iterativno pokušava poboljšati kandidatska rješenje u odnosu na mjeru kvalitete tj. danu funkciju $f : \mathbb{R}^n \to \mathbb{R}$. Kandidatska rješenja $\mathbf{p}_i$ predstavljaju populaciju, a ta se populacija miče kroz prostor pretraživanja na temelju trenutne pozicije $\mathbf{x}_i$ i trenutne brzine $\mathbf{v}_i$. Na svako novo rješenje utječe najbolje rješenje te čestice i najbolje rješenje cijelog roja $\mathbf{g}$.

Pseudokod algoritma dan je u nastavku:

Algorithm 1: Optimizacija rojem čestica

za svaku česticu i:
  inicijaliziraj poziciju čestice i na vrijednost $\mathbf{x}_i \sim U(b_{lo}, b_{up})$
  inicijaliziraj najbolju poziciju čestice i na početnu poziciju: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  ako $f(\mathbf{p}_i) < f(\mathbf{g})$:
    ažuriraj najbolju poziciju roja: $\mathbf{g} \leftarrow \mathbf{p}_i$
  inicijaliziraj brzinu čestice p: $\mathbf{v}_i \sim U(-|b_{up} - b_{lo}|, |b_{up} - b_{lo}|)$
dok nema konvergencije:
  za svaku česticu i:
    za svaku dimenziju d:
      generiraj nasumične brojeve: $r_p, r_g \sim U(0, 1)$
      ažuriraj brzinu čestice i: $\mathbf{v}_{i,d} \leftarrow w\mathbf{v}_{i,d} + \phi_p r_p(\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g r_g(\mathbf{g}_d - \mathbf{x}_{i,d})$
    ažuriraj poziciju čestice i: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
    ako $f(\mathbf{x}_i) < f(\mathbf{p}_i)$:
      ažuriraj najbolju poziciju čestice: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
      ako $f(\mathbf{p}_i) < f(\mathbf{g})$:
        ažuriraj najbolju poziciju roja: $\mathbf{g} \leftarrow \mathbf{p}_i$

2 Opis razvijenog programa

Razvijeni program provodi optimizaciju rojem čestica nad zadanom funkcijom f. Prvo se provodi algoritam i zabilježe se pozicije čestica tijekom algoritma. Zatim se ove pozicije tretiraju kao kontrolne točke aproksimacijske uniformne B-spline kubne krivulje te se generiraju koordinate za njenu animaciju i za trag koji čestica ostavlja. Aproksimiraju se izokonture dane funkcije te se i one prikazuju na platnu. Program je ostvaren korištenjem programskog jezika Python i biblioteka NumPy, SymPy, SciPy, Colour i standardnih biblioteka. Za prikaz animacije na platnu korištene su biblioteke PyGame i PyOpenGL.

3 Upute za pokretanje

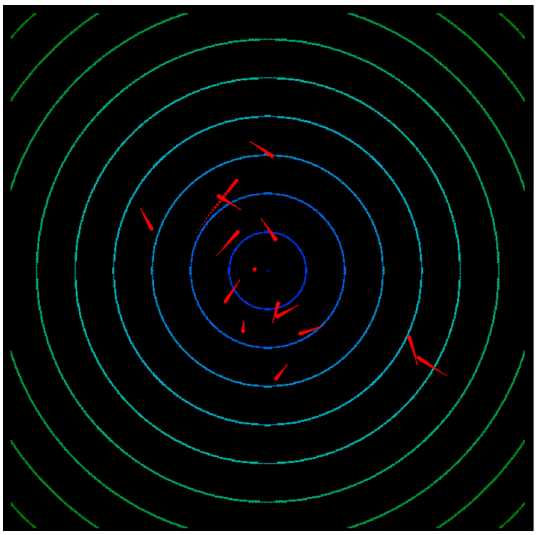
Program se pokreće kroz komandnu liniju. Prvo je potrebno pozicionirati se u direktorij gdje se nalazi datoteka *lab3.py*. Program se pokreće na sljedeći način:

```
python lab3.py --<zastavica1> <vrijednost1> --<zastavica2> <vrijednost2>...
```

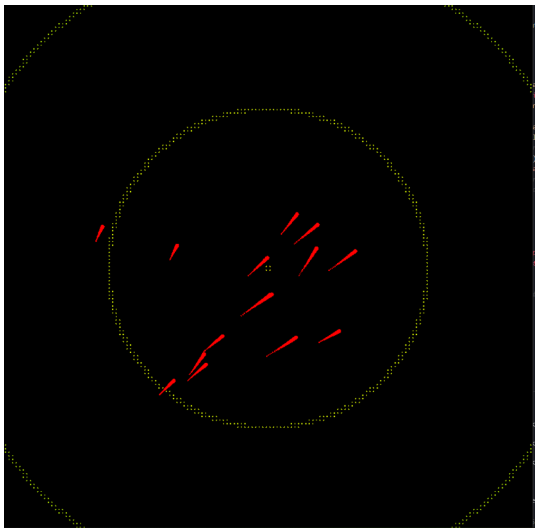
Program je moguće pokrenuti bez ikakvih zastavica jer sve varijable imaju pretpostavljene vrijednosti. Moguće zastavice su sljedeće:

- `--f` - funkcija koja se optimira, pretpostavljena vrijednost: `'(x**2 + y**2)**0.5'`
- `--pop_size` - veličina populacije, pretpostavljena vrijednost: `15`
- `--w` - konstanta w iz pseudokoda, pretpostavljena vrijednost: `0.5`
- `--fi_p` - konstanta ϕ_p iz pseudokoda, pretpostavljena vrijednost: `2`
- `--fi_g` - konstanta ϕ_g iz pseudokoda, pretpostavljena vrijednost: `2`
- `--b_lo` - konstanta b_{lo} iz pseudokoda, pretpostavljena vrijednost: `-20`
- `--b_up` - konstanta b_{up} iz pseudokoda, pretpostavljena vrijednost: `20`
- `--err` - aritmetička sredina pogreške trenutnih pozicija, služi kao uvjet zaustavljanja, pretpostavljena vrijednost: `10**-2`
- `--max_iter` - maksimalan broj koraka algoritma, služi kao uvjet zaustavljanja, pretpostavljena vrijednost: `10**4`
- `--speed_div` - nazivnik Δt vrijednosti koja se koristi pri računanju krivulje, pretpostavljena vrijednost: `10`
- `--speed_iter` - predstavlja konstantu kojom se množi brzina prilikom računanja nove pozicije čestice, pretpostavljena vrijednost: `0.5`
- `--p_size` - veličina prikaza čestice, pretpostavljena vrijednost: `5`
- `--trace_len` - dužina traga koji ostavlja čestica, pretpostavljena vrijednost: `20`
- `--num_c` - broj izokontura koje će se izračunati, ne moraju nužno sve biti prikazane, pretpostavljena vrijednost: `10`
- `--diff` - razlike u z vrijednosti izokontura, pretpostavljena vrijednost: `3`
- `--rtol` - relativna tolerancija, služi za određivanje jesu li koordinate točaka dovoljno blizu da bi bile prikazane u izokonturi, pretpostavljena vrijednost: `5e-4`
- `--atol` - apsolutna tolerancija, služi za određivanje jesu li koordinate točaka dovoljno blizu da bi bile prikazane u izokonturi, pretpostavljena vrijednost: `5e-2`
- `--iso_prec` - faktor koji pomaže pri izračunu izokontura, veće vrijednosti daju preciznije rezultate, pretpostavljena vrijednost: `15`
- `--iso_c_start` - početna boja u gradijentu boje za izokonture, pretpostavljena vrijednost: `'blue'`
- `--iso_c_finish` - konačna boja u gradijentu boje za izokonture, pretpostavljena vrijednost: `'green'`
- `--width` - širina prozora, pretpostavljena vrijednost: `800`
- `--height` - visina prozora, pretpostavljena vrijednost: `800`

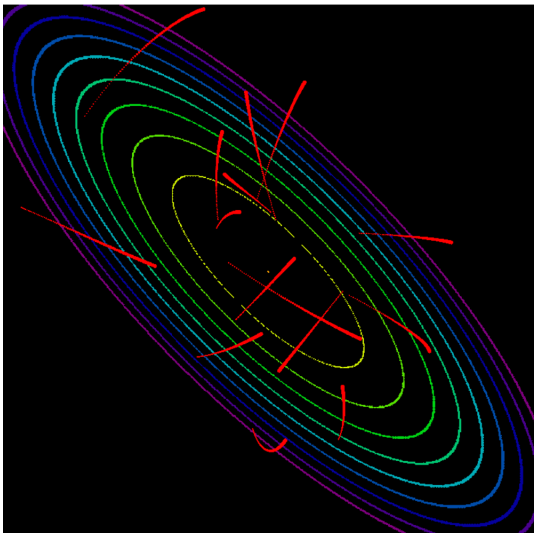
4 Primjeri



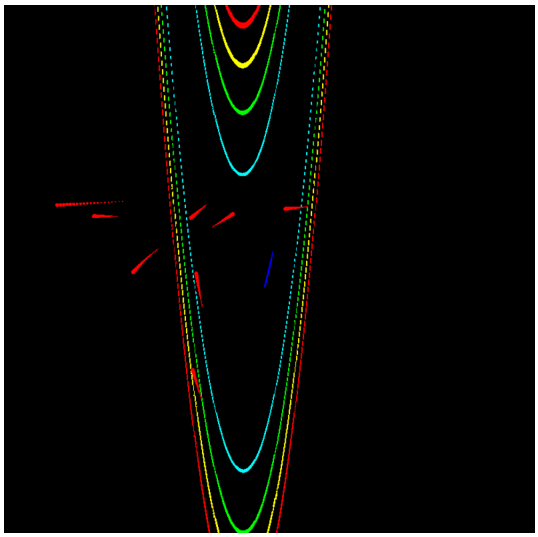
(a) `python lab3.py --f '(x**2 + y**2)**0.5'`



(b) `python lab3.py --iso_c_start 'yellow' --b_lo -5 --b_up -3`



(c) `python lab3.py --f '(x + 2*y - 7)**2 + (2*x + y - 5)**2' --rtol 1e-2 --atol 1e-2 --diff 100 --trace_len 70 --iso_c_start 'orange' --iso_c_finish 'purple'`



(d) `python lab3.py --pop_size 8 --diff 400 --num_c 5 --f '(2-x)**2 + 3*(y-x**2)**2' --atol 1.2e-1 --rtol 1.8e-2 --iso_c_finish 'red' --iso_prec 20`