24-10-24

Date / /
Page

Particle Swarm Optimization for function optimization

Particle Swarm Optimization is inspired by the social behaviour of birds flocking or fish schooling. PSO is used to find optimal solutions by iteratively improving a candidate solution with regard to a given measure of quality.

Algorithm

Step 1: Randomly initialize Swarm population of N particles X_i ($i = 1, 2 \ldots, n$)

Step 2: Select hyperparameter value
w, c1, c2

Step 3: For iter in range(max_iter):
For i in range (N):

a. Compute new velocity of ith particle

```
swarm[i].velocity =
        w * swarm[i].velocity +
        r1 * c1 * (swarm[i].best_pos -
            swarm[i].position) +
        r2 * c2 (best_pos_swarm -
            swarm[i].position)
```

b. Compute new position of ith particle using its new velocity

```
swarm[i].position += swarm[i].velocity
```

c. If position is not in range [minx, maxx] then clip it

```
if swarm[i].position < minx :
    swarm[i].position = minx
elif swarm[i].position > maxx :
    swarm[i].position = maxx
```

d. Update new best of the particles and new best of swarm
if swa Insensitive to scaling of design variables. swarm[i].fitness < swarm[i].bestFitness:
swarm[i].bestFitness = swarm[i].fitness
swarm[i].bestPos = swarm[i].pos

if swarm[i].fitness < best_fitness_swa:
best_fitness_swarm = swarm[i].fitness
best_pos_swarm = swarm[i].position
End-for
End-for

Step 4 Return best particle for swarm.

Output

Best Position : [-7.22517407 0e-13
2.0770816 4e-13]

Best Fitness : 5.65169263204 4056e-25