import random

import math

import copy

import sys

# Fitness functions

def fitness\_rastrigin(position):

return sum((xi \* xi) - (10 \* math.cos(2 \* math.pi \* xi)) + 10 for xi in position)

def fitness\_sphere(position):

return sum(xi \* xi for xi in position)

# Particle class

class Particle:

def \_\_init\_\_(self, fitness, dim, minx, maxx, seed):

self.rnd = random.Random(seed)

self.position = [(maxx - minx) \* self.rnd.random() + minx for \_ in range(dim)]

self.velocity = [(maxx - minx) \* self.rnd.random() + minx for \_ in range(dim)]

self.best\_part\_pos = self.position[:]

self.fitness = fitness(self.position)

self.best\_part\_fitnessVal = self.fitness

# PSO function

def pso(fitness, max\_iter, n, dim, minx, maxx):

w, c1, c2 = 0.729, 1.49445, 1.49445

rnd = random.Random(0)

swarm = [Particle(fitness, dim, minx, maxx, i) for i in range(n)]

best\_swarm\_pos, best\_swarm\_fitnessVal = [0.0] \* dim, sys.float\_info.max

for p in swarm:

if p.fitness < best\_swarm\_fitnessVal:

best\_swarm\_fitnessVal = p.fitness

best\_swarm\_pos = p.position[:]

for Iter in range(max\_iter):

if Iter % 10 == 0 and Iter > 1:

print(f"Iter = {Iter} best fitness = {best\_swarm\_fitnessVal:.3f}")

for p in swarm:

for k in range(dim):

r1, r2 = rnd.random(), rnd.random()

p.velocity[k] = w \* p.velocity[k] + c1 \* r1 \* (p.best\_part\_pos[k] - p.position[k]) + c2 \* r2 \* (best\_swarm\_pos[k] - p.position[k])

p.velocity[k] = max(min(p.velocity[k], maxx), minx)

p.position = [p.position[k] + p.velocity[k] for k in range(dim)]

p.fitness = fitness(p.position)

if p.fitness < p.best\_part\_fitnessVal:

p.best\_part\_fitnessVal = p.fitness

p.best\_part\_pos = p.position[:]

if p.fitness < best\_swarm\_fitnessVal:

best\_swarm\_fitnessVal = p.fitness

best\_swarm\_pos = p.position[:]

return best\_swarm\_pos

# Driver for Rastrigin function

def run\_pso(fitness, dim, minx, maxx):

print(f"Goal is to minimize the function in {dim} variables")

print(f"Function has known min = 0.0 at ({', '.join(['0'] \* (dim - 1))}, 0)")

num\_particles, max\_iter = 50, 100

best\_position = pso(fitness, max\_iter, num\_particles, dim, minx, maxx)

print(f"Best solution found: {', '.join([f'{x:.6f}' for x in best\_position])}")

print(f"Fitness of best solution = {fitness(best\_position):.6f}\n")

# Run PSO for Rastrigin and Sphere functions

print("\nBegin PSO for Rastrigin function\n")

run\_pso(fitness\_rastrigin, 3, -10.0, 10.0)

print("\nBegin PSO for Sphere function\n")

run\_pso(fitness\_sphere, 3, -10.0, 10.0)