Particle Swarm Optimization

Yamini Kota
MTech - SPML

1 Introduction

Particle Swarm Optimization is an iterative method to find the global optima of a function in the search space. It uses information about the local best known position and the global best position to update the position.

2 Basic Idea

PSO algorithm works by having a *Population* or a *Swarm* of candidate solutions, called *Particles*. Each particle moves in the search space, hence has a **Velocity** and remembers it's **Personal Best Position**, where it had the least function value. The particles in the swarm can communicate to decide the **Global Best Position**, where the function value is the least among all the particles in the swarm.

The movement of a particle is decided by its personal best position as well as the global best position using a simple mathematical formula. This is repeated until convergence or maximum number of iterations.

3 Algorithm and PsuedoCode

Let f:$R^n \rightarrow R$ be a function to be minimized over the domain S. Let N be the population size. Each particle has the position $x_i \epsilon R^n$ and a velocity $v_i \epsilon R^n$. Let p_i be the i^{th} particle's best known position and g be the global best known position.
The values b_{lo} and b_{up} represent the bounds on the search space S. Termination criteria could be maximum number of iterations or convergence of the function value to its global minimum. The parameters ω,ω_p and ω_g are user selected, which control the flow of the PSO algorithm.

Basic PSO Algorithm:

---

for each particle $i = 1, 2, ..., N$ **do**
  Initialize the particle's position with a uniformly distributed random vector: $x_i \sim U(b_{\text{lo}}, b_{\text{up}})$
  Initialize particle's best known position to its initial position: $p_i \leftarrow x_i$
  if $f(p_i) < f(g)$ **then**
    Update the swarm's best known position: $g \leftarrow p_i$
  end if
  Initialize the particle's velocity: $v_i \sim U(-|b_{\text{up}} - b_{\text{lo}}|, |b_{\text{up}} - b_{\text{lo}}|)$
end for
while a termination criterion is not met **do**
  for each particle $i = 1, 2, ..., N$ **do**
    for each dimension $d = 1, 2, ..., n$ **do**
      Pick random numbers: $r_p, r_g \sim U(0, 1)$
      Update the particle's velocity:

$$v_{\text{i,d}} \leftarrow \omega * v_{\text{i,d}} + \phi_p * r_p * (p_{\text{i,d}} - x_{\text{i,d}}) + \phi_g * r_g * (g_d - x_{\text{i,d}})$$

    end for
    Update the particle's position: $x_i \leftarrow x_i + v_i$
    if $f(x_i) < f(p_i)$ **then**
      Update the particle's best known position: $p_i \leftarrow x_i$
      if $f(p_i) < f(g)$ **then**
        Update the swarm's best known position: $g \leftarrow p_i$
      end if
    end if
  end for
end while

---

4 Different Objective Functions

PSO works sufficiently well in finding the global minimas for different functions. Some of the functions are:

1. Ackley Function

$$f(x, y) = -20 \exp\Big(-0.2\sqrt{0.5(x^2 + y^2)}\Big) - \exp\Big(0.5\big(\cos 2\pi x + \cos 2\pi y\big)\Big) + \exp(1) + 20$$

2. Beale Function

$$f(x, y) = \big(1.5 - x + xy\big)^2 + \big(2.25 - x + xy^2\big)^2 + \big(2.625 - x + xy^3\big)^2$$

3. Booth Function

$$f(x,y) = \left(x + 2y - 7\right)^2 + \left(2x + y - 5\right)^2$$

4. Cross-in-Tray Function

$$f(x,y) = -0.0001\left(\left|\sin\left(x\right)\sin\left(y\right)\exp\left(\left|100 - \frac{\sqrt{x^2+y^2}}{\pi}\right|\right)\right| + 1\right)^{0.1}$$

5. Easom Function

$$f(x,y) = -\cos\left(x\right)\cos\left(y\right)\exp\left(-\left(x-\pi\right)^2 - \left(y-\pi\right)^2\right)$$

6. Goldstein-Price Function

$$f(x,y) = \left[1 + \left(x+y+1\right)^2\left(19 - 14x + 3x^2 - 14y + 6xy + 3y^2\right)\right]$$
$$\times\left[30 + \left(2x-3y\right)^2\left(18 - 32x + 12x^2 + 45y - 36xy + 27y^2\right)\right]$$

7. Himmelblau Function

$$f(x,y) = \left(x^2 + y - 11\right)^2 + \left(x + y^2 - 7\right)^2$$

8. Holder Table Function

$$f(x,y) = -\left|\sin\left(x\right)\cos\left(y\right)\exp\left(\left|1 - \frac{\sqrt{x^2+y^2}}{\pi}\right|\right)\right|$$

9. Matyas Function

$$f(x,y) = 0.26\left(x^2 + y^2\right) - 0.48xy$$

10. Schaffer-N.2 Function

$$f(x,y) = 0.5 + \frac{\sin^2\left(x^2 - y^2\right) - 0.5}{\left[1 + 0.001\left(x^2+y^2\right)\right]^2}$$

11. Three-Hump Camel Function

$$f(x,y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$$

12. Eggholder Function

$$f(x,y) = -\left(y+47\right)\sin\left(\sqrt{\left|y + \frac{x}{2} + 47\right|}\right) - x\sin\left(\sqrt{\left|x - \left(y+47\right)\right|}\right)$$

5 Application: MLP Training

Use of PSO algorithm instead of Back Propagation for obtaining the Weights and Biases for the Neural Networks gives a considerably good accuracy. (Tested the PSO algorithm for SLFN with the IRIS dataset.)

References

[1] PPT on Particle Swarm Optimization - https://www.researchgate.net/profile/Sreenivas_Sremath_Tirumala/post/How_does_the_Particle_Swarm_Algorithm_works/attachment/59d63a4ec49f478072ea68d6/AS%3A273728099815424%401442273272370/download/bic_pso.ppt

[2] Wikipedia: Particle swarm optimization - https://en.wikipedia.org/wiki/Particle_swarm_optimization

[3] Optimization Test Functions - http://www.sfu.ca/~ssurjano/optimization.html

[4] MLP training using PSO - https://pyswarms.readthedocs.io/en/latest/examples/usecases/train_neural_network.html