Particle Swarm Optimization

Caden Hewlett

2024-09-10

Introduction and Terminology

Particle Swarm Optimization (PSO) is an evolutionary algorithm (EA) inspired by the schooling of fish or the flocking of birds. The algorithm optimizes an objective function by iteratively improving a candidate solution with respect to a given measure or quantity. It is a gradient-free technique, meaning that the objective function does not need to be differentiable.

The algorithm is initialized with a population of N candidate solutions called "particles." We henceforth denote the population of particles as $\mathfrak{X}_t = \{\mathbf{X}_t^{(1)}, \mathbf{X}_t^{(2)}, \ldots, \mathbf{X}_t^{(N)}\}$, where $t \in \mathbb{N}$ is the iteration of the algorithm. Each individual particle $\mathbf{X}_t^{(i)}$ has a position in the search space, for $i \in [1, N]$. For example, if the objective function f maps $\mathbb{R}^d \mapsto \mathbb{R}$, then each particle $\mathbf{X}_t^{(i)}$ is a d-dimensional vector of positions in the search space, denoted $\mathbf{X}_t^{(i)} = \langle x_1^{(i)}, \ldots, x_d^{(i)} \rangle_t$. In addition to positions, each particle has a vector of velocities $\mathbf{V}_t^{(i)}$ in d-dimensional space that evolves with each iteration. It should be noted that the objective function f is evaluated on the positions $f(\mathbf{X}_t^{(i)})$, and the change in positions between generations $\mathbf{X}_t^{(i)} \to \mathbf{X}_{t+1}^{(i)}$ is moderated by the velocity vector $\mathbf{V}_t^{(i)} = \langle v_1^{(i)}, \ldots, v_d^{(i)} \rangle_t$. Each particle has its own position and velocity. The velocity $\mathbf{V}_t^{(i)}$ evolves between generations $\mathbf{V}_t^{(i)} \to \mathbf{V}_{t+1}^{(i)}$ by the *velocity update equation.* The position of a vector between generations is determined by adding the updated velocity to the current position, i.e.

$$\mathbf{X}_{t+1}^{(i)} = \mathbf{X}_t^{(i)} + \mathbf{V}_{t+1}^{(i)}, \text{ for } i \in [1, N],\, t \in \mathbb{Z}^+$$

Where $\mathbf{X}_t^{(i)}$ is the current position of particle i in generation t, $\mathbf{X}_{t+1}^{(i)}$ is the updated position at iteration $t+1$, and $\mathbf{V}_{t+1}^{(i)}$ is the updated velocity that controls the change in position.

Velocity Update Equation

At each iteration, the fitness of a particle with respect to the position $\mathbf{X}_t^{(i)}$ is computed. Two key metrics are recorded from the particle fitness $f(\mathbf{X}_t^{(i)})$.

Personal Best

Each particle has a "personal best" as of iteration t, denoted $\mathbf{p}_t^{(i)}$. The personal best is the position vector of particle i that yields the smallest result from the objective function, recalling that PSO minimizes f. Mathematically, we denote the personal best as:

$$\mathbf{p}_t^{(i)} = \underset{\ell \in [1,t]}{\operatorname{argmin}}\big(f(\mathbf{X}_\ell^{(i)})\big), \text{ for } i \in [1, N], t \in \mathbb{Z}^+$$

Where $\mathbf{p}_t^{(i)}$ is the position that has produced the lowest objective function value up to iteration t. Each particle i maintains its own personal best $\mathbf{p}_t^{(i)}$ at each time step t.

Global Best

In addition to the personal best, the velocity of the particles are influenced by the "global best", denoted $\mathbf{g}_t$, which is the best position found across all particles in the swarm up to iteration t. The global best is the position vector across all N particles in $\mathfrak{X}_t$ that yields the smallest result from the objective function. It is defined as follows:

$$\mathbf{g}_t = \underset{j \in [1,N]}{\operatorname{argmin}}\big(f(\mathbf{p}_t^{(j)})\big), \text{ for } t \in \mathbb{Z}^+$$

Where $\mathbf{g}_t$ represents the unique global best position found among the personal bests $\mathbf{p}_t^{(j)}$ of all particles j in the swarm $\mathfrak{X}_t$ at iteration t.

Velocity Update

Using the global best $\mathbf{g}_t$ and the personal best $\mathbf{p}_t^{(i)}$, the velocity update equation for the i-th particle in the swarm at time t is given as follows.

$$\mathbf{V}_{t+1}^{(i)} = w\mathbf{V}_t^{(i)} + r_1 c_1\big(\mathbf{p}_t^{(i)} - \mathbf{X}_t^{(i)}\big) + r_2 c_2\big(\mathbf{g}_t - \mathbf{X}_t^{(i)}\big)$$
$$\text{For } \; c_1, c_2, w \in \mathbb{R}^+ \; \text{ and } \; r_1, r_2 \overset{\text{iid}}{\sim} \text{Uniform}(0,1)$$

Here, c_1 and c_2 are the acceleration coefficients and w is the inertia weight. The first acceleration coefficient c_1 controls controls the influence of the particle's distance from its personal best $\mathbf{p}_t^{(i)}$ on its next velocity. Similarly, the second acceleration coefficient c_2 moderates the impact of the particle's distance from the global best $\mathbf{g}_t$ on the next velocity. The inertia weight w controls the degree to which the current velocity $\mathbf{V}_t^{(i)}$ on the updated velocity $\mathbf{V}_{t+1}^{(i)}$. A high value of w results in faster-moving particles, promoting exploration of the search space. Conversely, a low w will yield slower-moving particles, promoting exploitation by encouraging the particle to refine the current search area. In the above, w is assumed to be constant; however, many implementations include a scheduled linear or exponential decay in w proportional to iteration t so that the algorithm exploits more as time progresses. By default, $w = \big(2\log(2)\big)^{-1}$ and $c_1 = c_2 = \frac{1}{2}\log(2)$, recommended by empirical studies such as (Kennedy and Eberhart 1995) and (Shi and Eberhart 1998).

Fully-Informed Particle Swarm

Informant-Based Particle Swarm Optimization (FIPS) is a variant of the traditional PSO algorithm that enhances how particles share information within the swarm. FIPS modifies the velocity update equation by allowing each particle to be influenced by multiple "informants" - a selected subset of other particles—rather than relying solely on the global best $\mathbf{g}_t$.

In `psoptim`, the informants are selected randomly and without replacement. Each particle $\mathbf{X}_t^{(i)}$ has its own sampled set of informants. Further, the acceleration $c = \frac{1}{2}\log(2)$ is constant across all particles. The number of informants is constant for each particle, and is dictated by a hyper-parameter ϱ and the swarm size N. The velocity update equation is given as follows,

$$\mathbf{V}_{t+1}^{(i)} = w\mathbf{V}_t^{(i)} + cr_i\big(\mathbf{p}_t^{(i)} - \mathbf{X}_t^{(i)}\big) + c\sum_{j \in \mathcal{I}_i}\big(\mathbf{p}_t^{(j)} - \mathbf{X}_t^{(i)}\big)$$

Where $\mathcal{I}_i$ is a set of size $\varrho N - 1$ containing the indices of the informants of the i-th particle.

$$\mathcal{I}_i = \big\{j \in \{1, 2, \dots N\} \setminus \{i\} \;\big|\; j \text{ is sampled without replacement}, |\mathcal{I}_i| = \lceil \varrho N \rfloor\big\}$$

The hyper-parameter ϱ controls the proportion of the total population that act as an informant for a given particle. It can be tuned independently; however, by default it is defined as follows:

$$\varrho = 1 - \Big(1 - \frac{1}{N}\Big)^d$$

Where N is the swarm size, and d is the number of dimensions of each particle, i.e., the dimensionality of the search space.

Constriction

In some implementations, a "constriction" denoted χ is applied to the velocities. Introduced in (Clerc and Kennedy 2002), the constriction factor helps to stabilize particle velocities, ensuring that they do not increase uncontrollably over time. Further, the constriction factor helps in guiding particles to converge towards a global optimum. It reduces the risk of particles diverging, which can happen if velocities are too large.

The velocity update4 including constriction is given as:

$$\mathbf{V}_{t+1}^{(i)} = \chi\Big(\mathbf{V}_t^{(i)} + r_1 c_1\big(\mathbf{p}_t^{(i)} - \mathbf{X}_t^{(i)}\big) + r_2 c_2\big(\mathbf{g}_t - \mathbf{X}_t^{(i)}\big)\Big)$$

Where $\chi \in \mathbb{R}^+$ is defined as:

$$\chi = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \text{ where } \phi = c_1 + c_2$$

Where, usually, $\phi \approx 4.1$ (Clerc and Kennedy 2002). Much like inertia weight w, the constriction factor χ can have scheduled temporal decay as a function of iteration t.

Survivability

One new implementation I will try is to implement a "survivability" quotient for each particle, similar to those present in genetic algorithms like GENOUD.

In general, the probability that each parameter vector $\mathbf{X}_t^{(i)}$ is selected to be reproduced in the generation $t+1$ is a function of its rank; namely:

$$\mathbb{P}\big(\mathbf{X}_t^{(i)} \in \mathfrak{X}_{t+1}\big) = a + (1-a)\cdot\left(1 - \left(\frac{R_t^{(i)} - 1}{N-1}\right)^p\right) \text{ where } R_t^{(i)} = \text{rank}\big(f(\mathbf{X}_t^{(i)})\big), \text{ and } a \in \mathbb{R} \subseteq [0,1]$$

Where $Q \in (0,1)$ is a tuning parameter which is fixed to $Q = 0.5$ in the R implementation and $R_t^{(i)} \in \{1, 2, \ldots, N\}$ is the rank of the code string as determined by the objective function and the direction of optimization. For example, if solving a minimization problem, $R_t^{(i)} = 1$ for the code string whose variables evaluated in the objective function are the smallest compared to all other individuals. It should be noted that an exception to the above geometric selection probability above is made when $R_t^{(i)} = 1$ for $i \in [1, N]$. In this case, the individual $\mathbf{X}_t^{(i)}$ is guaranteed to be brought to the next generation.

```
N = 500
swarm <- initialize_swarm(N, -2, 2)
objective_function <- function(x) { sum(x^2) }
fitness <- evaluate_fitness(swarm$X, objective_function)
trials <- rbinom(N, size = 1, prob = rank_map(fitness)) == 1
fit_df <- data.frame(
   fitness = fitness,
   index  = 1:N
 )
plot(x = fit_df$index, y = fit_df$fitness, main = "Random Swarm, Red Indicates Eliminated Points")
points(x = fit_df[!trials, "index"], y = fit_df[!trials, "fitness"], col = 'red')
```

Time-Varying Acceleration Coefficients and EPSO

There are also strategies wherein the coefficients w, c_1, etc. vary with the episode t relative to the maximal number of iterations T. One such model blends the idea of time-varying coefficients (TVAC) with evolution-

ary particle swarm optimization (EPSO),to define the time-dependent coefficients and velocity equations:

$$\mathbf{V}_{t+1}^{(i)} = w_{0t}^{(i)}\mathbf{V}_t^{(i)} + w_{1t}^{(i)}\left(\mathbf{p}_t^{(i)} - \mathbf{X}_t^{(i)}\right) + w_{2t}^{(i)}\left(\mathbf{g}_t - \mathbf{X}_t^{(i)}\right) + w_{3t}^{(i)}\left(\mathbf{p}_t^{(j\neq i)} - \mathbf{X}_t^{(i)}\right)$$

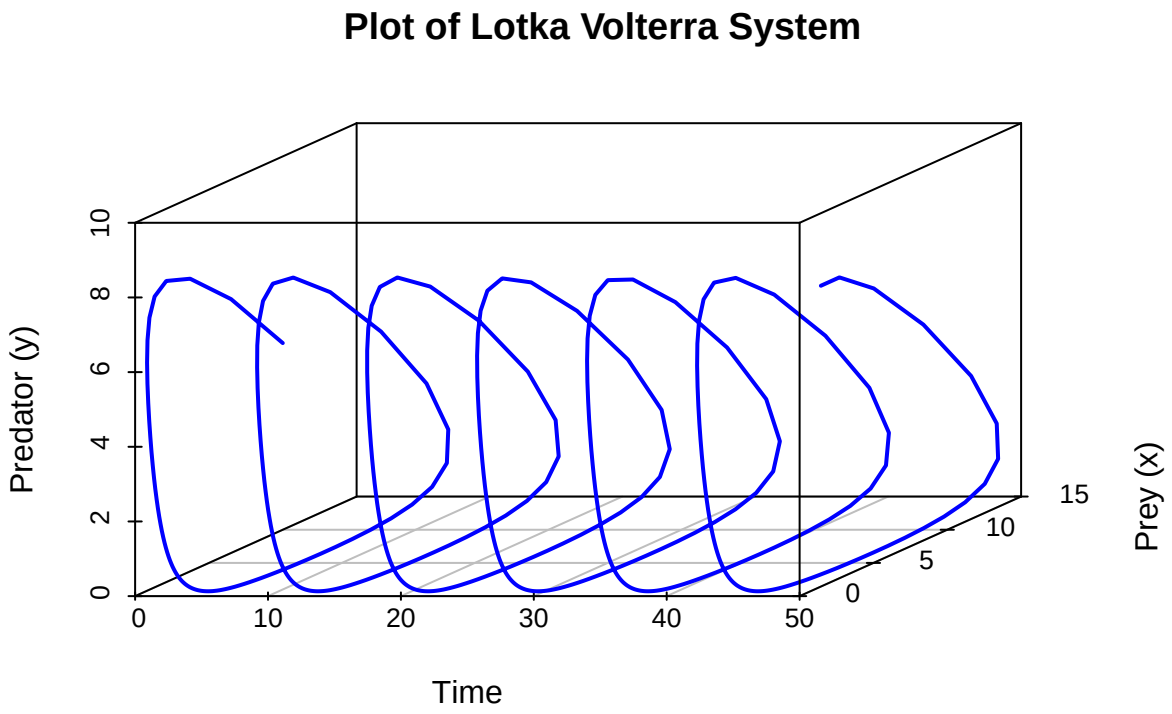
$$\text{Where } w_{0t}^{(i)} = w + \tau \cdot r_0 \text{ and } w_{kt} = c_k + \tau \cdot r_k, \text{ for } k \in [1,3]$$

Predator-Prey Dynamics I also want to explore predator-prey dynamics in terms of survivability and regeneration of particles. I will (tentatively) model the system with the Lotka–Volterra equations, which are differential equations given as follows:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \alpha x - \beta xy$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = -\gamma y + \delta xy$$

Here, t represents time; The prey's parameters, α and β, describe, respectively, the maximum prey per capita growth rate, and the effect of the presence of predators on the prey death rate. The predator's parameters, γ, δ , respectively describe the predator's per capita death rate, and the effect of the presence of prey on the predator's growth rate.



Sources

Clerc, Maurice, and James Kennedy. 2002. "The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space." *IEEE Transactions on Evolutionary Computation* 6 (1): 58–73.

Kennedy, James, and Russell C Eberhart. 1995. "Particle Swarm Optimization." In *Proceedings of ICNN'95-International Conference on Neural Networks*, 4:1942–48. IEEE.

Shi, Yuhui, and Russell C Eberhart. 1998. "A Modified Particle Swarm Optimizer." In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*, 69–73. IEEE.