Swarm Intelligence for Function Optimisation

May 2024

Contents

Abstract

Evolutionary algorithms are very useful meta-optimization techniques and they are useful for solving hard problems. one of the variations of those algorithms is swarm intelligence as they are inspired by the swarms of ants, bees, etc. Swarm intelligence has many applications such as Engineering Design Optimization, Data Mining and Machine Learning, Scheduling, and Routing Problems [6]. This report is a discussion of the performance of the artificial bee colony and particle swarm optimization using three benchmark functions to know what algorithm is best for optimization problems. the convergence rate and the average best solutions are the metrics used for this comparison. ABC has been better than PSO across these metrics and it can be concluded that it is more suitable for optimization.

1 Members

- omar tamer Mohammed 20210598
- omar ayman 20210597
- ahmed esmail 20210023
- ahmed mostfa 20210121
- ahmed mohammed 20210098
- ahmed maqboul 20210111

2 Introduction

swarm intelligence is the observation and study of natural systems and how they collaborate as a self-organized system to solve problems like ants, bees, fish, colonies, etc. Many algorithms that imitate these ideas have been developed in recent years and comparisons of how they do in different problems are a must. One area of interest is how these algorithms are used to optimize nonlinear and Many Local minimum functions. In this report, a comparison of performance between artificial bee colony **(ABC)** and particle swarm optimization **(PSO)** has been made using three benchmark functions.

3 Defining The Problem

classifying a problem has many approaches, in this report, a black-box view of the system has been used. this view divided any system into three components: inputs - model - outputs. the appearance of those individual parts helps in determining the category of the problem. the model is the function itself and it is known as the desired output. the only unknown part is the input that leads to the desired output. that leads to defining the problem as an **(optimisation problem)** and as the functions are defined on a set of boundaries, it is a **(Constrained optimisation problem)**.

4 Test Functions

Three test functions have been used to assess the performance of the two algorithms:

1. **Ackley Function:** The Ackley function is given by:

$$f(\mathbf{x}) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$$

   where $\mathbf{x}$ represents a vector of n dimensions.

2. **Rastrigin Function:** The Rastrigin function is given by:

$$f(\mathbf{x}) = An + \sum_{i=1}^{n}(x_i^2 - A\cos(2\pi x_i))$$

   where A is a constant parameter and $\mathbf{x}$ represents a vector of n dimensions.

3. **Sphere Function:** The Sphere function is given by:

$$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$$

   where $\mathbf{x}$ represents a vector of n dimensions.

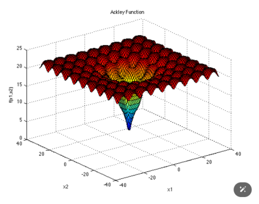

Figure 1: Ackley Function

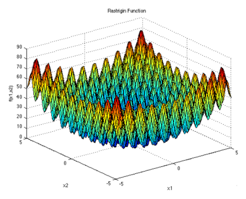

Figure 2: Rastrigin Function

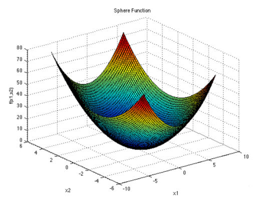

Figure 3: Sphere Function

5 Algorithms

5.1 Atrificial Bee Colony

is a meta-heuristic algorithm founded by Karaboga in 2005. ABC was founded by studying the honey bee swarms and how the bees found the best food sources.

the honey bee swarms are divided into three components: food sources - employed foragers - and unemployed foragers.[4]

1. food sources :

   is the quality of the source(flower) as nectar for the bees. in our case, it is the value of the objective function.

2. employed foragers :

   they are associated with only one food source that they are exploited at and they share the location and the quality of the food source with the other bees.

3. unemployed foragers :

   they are always searching for food sources and there are two types of them: scouts who search around the nest and onlookers who wait for information from the employed foragers to work on the best-founded food source.

the model of the algorithm has taken three parts from the honey bee swarm. the employed bees are half of the population. they are assigned to each solution meaning that the number of solutions is equal to the number of employed bees. the other half is onlooker bees who exhaust the best solutions. The employed bee whose food source(solution) has been exhausted by the bees becomes a scout.

the ABC works with the fitness function, not the objective function and it is defined by

$$\text{fitness} = \begin{cases} \frac{1}{1+\text{objective value}} & \text{if objective value} > 0 \\ 1 + |\text{objective value}| & \text{if objective value} < 0 \end{cases}$$

to update the solutions, ABC used greedy selection

$$x_{\text{current}} = \begin{cases} x_{\text{new}} & \text{if fitness\_new} > \text{fitness\_current} \\ x_{\text{current}} & \text{otherwise} \end{cases}$$

$$\text{objective\_current} = \begin{cases} \text{objective\_new} & \text{if fitness\_new} > \text{fitness\_current} \\ \text{objective\_current} & \text{otherwise} \end{cases}$$

5.1.1 employee phase

each employee tries to generate a new solution by choosing a random index in its current solution and then choose another random employee and update the solution by this formula

$x^{(j)}_{\text{new}} = x^{(j)}_{\text{current}} + \phi \cdot (x^{(j)}_{\text{current}} - x^{(j)}_{\text{p}})$

where

$x^{(j)}_{\text{new}}$ represents the new value of the index j.

ϕ is a random number between 0 and 1.

$x^{(j)}_{\text{current}}$ is the current value of the index j.

$x^{(j)}_{\text{p}}$ is the value of the index chosen from a random employee.

after that, we check the limits of the new solution, to make sure that the solution is feasible, then we calculate the fitness and the greedy selection. a trial number is kept and is increased after each iteration the solution fails to generate a new solution, this trial number will used in the scouting phase.

Algorithm 1 employee bee phase

```
for $i = 1tosize$
randomly selects a partner p such that i does not equal p
randomly selects a variable j and modifies the j th variable
$x^{(j)}_{\text{new}} = x^{(j)}_{\text{current}} + \phi \cdot (x^{(j)}_{\text{current}} - x^{(j)}_{\text{p}})$
bound $x^{(j)}_{\text{new}}$
evaluates the fitness and the objective function
do the greedy selection if false increase trial by 1
```

5.1.2 on looker phase

in onlooker phase , a probability function used to make a decision of exploiting a solution

$prob_{\text{i}} = 0.9 * \left(\frac{fit_{\text{i}}}{max(fit)}\right) + 0.1$

Algorithm 2 onlooker bee phase

```
set $m = 0$ and $n = 1$
while $m < N_{\text{p}}$
Generate random number $r$
if $r < prob_{\text{n}}$
randomly selects a partner $p$ such that $n$ does not equal p
randomly selects a variable $j$ and modifies the $j^{(th)}$ variable
bound $x^{(j)}_{\text{new}}$
evaluates the fitness and the objective function
Accept $x_{\text{new}}$, if $fit_{\text{new}} > fit_{\text{n}}$ and set trial to zero , else increase trial by 1
$m = m + 1$
end
$n = n + 1$
reset $n = 1$ if the value of $n$ greater than $N_{\text{p}}$
end
```

the idea is that in each iteration a random number r is calculated and if it is less than the probability, the onlooker will exploit the solution and try to generate a new solution from it using the same equation in the employee phase. better solutions can be chosen more than once and every onlooker exploits only one solution and the next onlooker should search after the last one. in the pseudo-code the m denoted the onlooker index variable while n denoted the index variable of the solutions(employed bees).

5.1.3 scout phase

in the scout phase, an input attribute called limit is used to compare the limit with the trial number of each solution. if the trial is bigger than the limit then the solution is removed and another random solution will replace it and the trial number of the new solution will be zero. the limit recommended to be $N_\mathrm{p} * D$ where D is the dimension number. [1]

Algorithm 3 scout phase

```
identify the food source $K$ whose trial greater than the limit
replace $x_\mathrm{k} = lb + (ub - lb) * r$
evaluates the fitness and the objective function
```

now we can add the whole three parts in the unified pseudo code for the ABC

Algorithm 4 ABC Algorithm

```
initializes a random population P
evaluates the fitness and the objective function
set trial equal to zero for all the food sources
for $t = 1$ to $T$
perform **employee bee phase** on the all food sources
determines the probability of each food source
perform **onlooker bee phase** to generate $N_\mathrm{p}$ food sources
memorize the best food sources
if a trial of any food greater than the limit
perform **scout phase**
end
```

6 particle swarm optimization (PSO)

the algorithm mimics the swarm theory of birds and fish and was invented by Eberhart and Kennedy [2]. each solution is represented by a particle. the idea of the algorithm that the particles are collaborating to find the optimal solution. each particle knows that the best solution has been found by the whole population and the best solution has been found individually. the location and velocity are the variations operators. each particle is a real vector denoted by $\bar{x} \in R^n$ and another perturbation vector (velocity vector) denoted by $\bar{v} \in R^n$ and $\bar{b}$ the personal best. so the i^{th} solution is denoted by $\left\langle \bar{x_i}, \bar{v_i}, \bar{b_i} \right\rangle$. each iteration the solution moved to a better place by updating the three components the updated solution denoted by $\left\langle \bar{x_i}', \bar{v_i}', \bar{b_i}' \right\rangle$

and the update process by these equations: [3]
$\bar{x_i}' = \bar{x_i} + \bar{v_i}'$
$\bar{v_i}' = w * \bar{v_i}' + \phi_1 * U_1 * (\bar{b_i} - \bar{x_i}) + \phi_2 * U_2 * (\bar{c} - \bar{x_i})$

the updated velocity vector is the weighted sum of three vectors one is the previous v and it is used to prevent the solution from changing its original velocity completely, the second vector $(\bar{b_i} - \bar{x_i})$ is the subtraction of the current position and the best position that has happened by the particle and it is used as memory for the previous best position.the last vector $(\bar{c} - \bar{x_i})$ where $\bar{c}$ is the global best is the subtraction from the global best to the current position. w and ϕ_i are the weights (w is called the inertia, ϕ_1 is the learning rate for the personal influence and ϕ_2 is the learning rate for the social influence), while U_1 and U_2 are random numbers drawn from the uniform distribution, and finally

$$\bar{b_i}' = \begin{cases} \bar{x_i}' & \text{if } F(\bar{x_i}') < F(\bar{b_i}') \\ \bar{b_i}' & \text{if } otherwise \end{cases}$$

Algorithm 5 PSO Algorithm

```
$t = 1$ and $T$ := number of iterations
initialize random swarm
evaluates the fitness of the swarm
while $t <= T$ DO
update $\bar{b_i}$ and find $\bar{c}$ for each particle $i$
update $\bar{x_i}$ and find $\bar{v_i}$ for each particle $i$
evaluates the new solutions and adds them to the next generation
end while
```

7 Experiments and performance criteria

to compare the ABC and PSO , the algorithms have been run for 30 times with different random seeds to make sure that initialization is random and to the recover the solutions easily with same parameters in each run. they have been compared on three test functions.

two performances criterion have been used :

7.0.1 convergence rate and plot

in each run , a plot of the fitness against the iteration was made and then taking an average over the 30 runs to make the plot more accurate. the algorithm with higher slope has a higher convergence rate.

7.0.2 accuracy and average

in each run , the best solution has been found is recorded after specific number of iterations. this process was repeated for 30 runs then taking average of all the runs to make the best solution more accurate and robust.as each run may have different best solution , a standard deviation is calculated for each 30 run.

8 Analysis and Results

the parameters settings used for ABC are

- colony size = 100
- number of employee bees S_N = number of on lookers = 50
- dimensions = 2
- limit = 100 $(dimenstions * S_N)$
- number of iterations = 100

the parameters settings used for PSO are

- $\phi_1 = 2$
- $\phi_2 = 2$
- dimensions = 2
- $w = 0.7$
- particle size = 50
- number of iterations = 100

8.0.1 convergence rate analysis

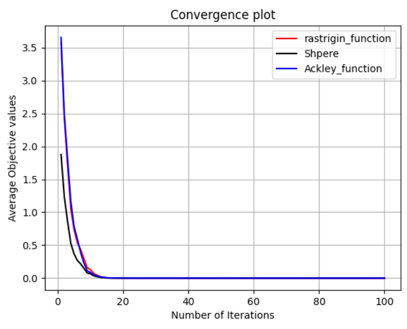

Figure 4: convergence plot of ABC

as shown in fig 4 and 5 , the convergence speed of ABC is a lot better than the PSO . it takes nearly 10 iterations for ABC to converge while PSO takes 40 iterations to converge.

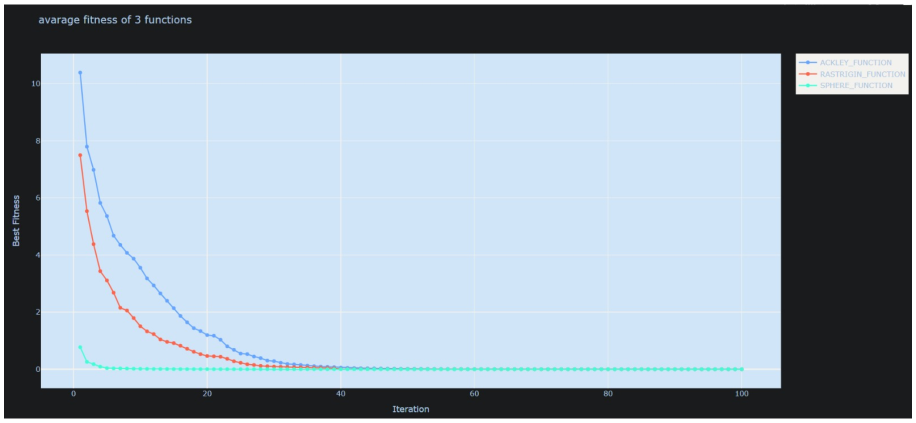

Figure 5: convergence plot of PSO

8.0.2 accuracy analysis

Table 1: best solution comparison of ABC and PSO algorithms on various functions after 100 iterations.

function	ABC	PSO
rastrigin_function	-0.00000008, 0.00000001	(-2.126e-06, -2.361e-07)
sphere_function	-0.00000003, 0.00000000	(8.210e-07, -4.634e-07)
Ackley_function	-0.00000002, 0.00000001	(2.422e-06, -1.957e-06)

from the table 1 the ABC outperformed the PSO by at least a factor of 10 in each function from the functions above.

9 parameter tuning of ABC

ABC has three control parameters which are the number of food sources, limit, and the number of iterations. in this report, the first two will be investigated by changing one parameter and making the other constants.

9.1 number of food sources

Table 2: best solution using colony size 100

function	ABC	std of ABC
rastrigin_function	-0.00000008, 0.00000001	0.00000076, 0.00000105
sphere_function	-0.00000003, 0.00000000	0.00000054, 0.00000075
Ackley_function	-0.00000002, 0.00000001	0.00000046, 0.00000068

Table 3: best solution using colony size 500

function	ABC	std of ABC
rastrigin_function	0.00000004, 0.00000009	0.00000055, 0.00000050
sphere_ function	0.00000002, 0.00000005	0.00000039, 0.00000035
Ackley_function	-0.00000001, 0.00000003	0.00000037, 0.00000032

Table 4: best solution using colony size 200

function	ABC	std of ABC
rastrigin_function	0.00000028, 0.00000029	0.00000147, 0.00000109
sphere_ function	0.00000014, 0.00000015	0.00000105, 0.00000078
Ackley_function	0.00000011, 0.00000009	0.00000086, 0.00000065

three different sizes of colonies are used 100, 200, and 500 while making the other parameters like the first experiment. the results are summarized in the three tables 2, 3 , 4. the colony size is responsible for exploitation while the limit is responsible for exploration. increasing the size from 100 to 200 makes the exploration factor has the lead but as the limit is still fixed that makes solutions achieve the limit more than the first trial this leads to the solutions decreasing a little bit in quality and having more standard deviation across the 30 runs. when increasing the size to 500, the exploitation factor takes the lead from the exploration. this helps the algorithm to search more in the best places although the limit is still fixed as seen in the table the std deviation is less than the size of 100 and the solutions are better.

9.1 limit size

as said before, the limit is responsible for the exploration. four limits are investigated 10, 50, 100, and 200 while the colony size is fixed to 100. four plots of convergence rate are shown down:

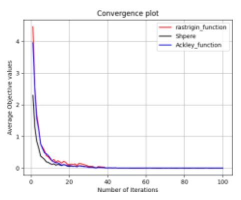

Figure 6: limit = 10

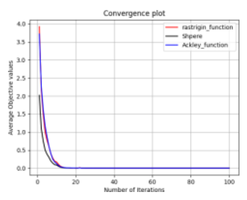

Figure 7: limit = 50

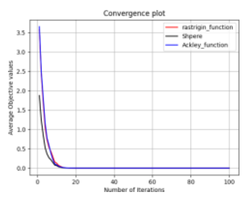

Figure 8: limit = 100

from the plots, it can be seen that when the limit is less than 100 or more than 100 the plots have a zig-zag shape. this is because in those cases one of the factors is taking the lead whether it is the exploration or exploitation and this is the reason the solutions in any iteration are not stable and different across

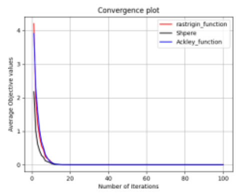

Figure 9: limit = 200

the runs. when the limit = 100 the graph is smooth and that is because there is a balance between the factors. this was the recommendation from the papers, the limit = size of the employees times the dimensions.

10 parameter tuning of PSO

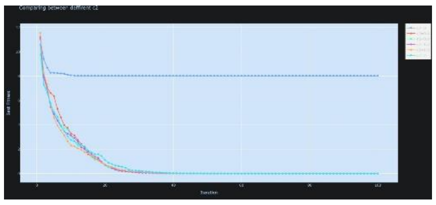

Figure10.1

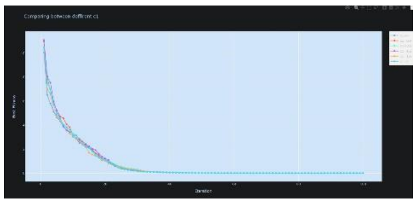

Figure10.2

Here in figure 10.1 the c2 when it becomes 0 Each particles Track only the Local Best position for that particle that means it explore the search space more and as we see he stuck in a local optimal solution .

In Figure 10.2, here when c1 becomes 0 Each Particle track only the Global best solution for the swarm it means it may be stuck in a local optimal solution if it has more local optimal points so it exploit the best solutions in particles .

As a rule of thumb , the sum of c1 and c2 should be 4 to maintain exploration and exploitation.

11 Development platform

all the implementations are written using Python language and the numpy library in Visual Studio code as the text editor. matplotlib is used for making the visualizations and tkinter is used for designing the GUI.

12 conclusions

in conclusion, the ABC has a better convergence rate and better solution in 100 iterations for the three benchmark functions, because of its ability to maintain diversity in the solution space and its robustness across the parameter settings while the PSO has a lack of diversity and is very sensitive to the parameters [5]. PSO is easy to implement while ABC may require more computational resources.

References

[1] Bahriye Akay and Dervis Karaboga. Parameter tuning for the artificial bee colony algorithm. In *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems: First International Conference,ICCCI 2009, Wroc-law, Poland, October 5-7, 2009. Proceedings 1*, pages 608–619. Springer, 2009.

[2] Russell Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the sixth international symposium onmicro machine and human science*, pages 39–43. Ieee, 1995.

[3] Meetu Jain, Vibha Saihjpal, Narinder Singh, and Satya Bir Singh. An overview of variants and advancements of pso algorithm. *Applied Sciences*, 12(17):8392, 2022.

[4] Dervis Karaboga et al. An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer . . . , 2005.

[5] Federico Marini and Beata Walczak. Particle swarm optimization (pso). a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 149:153–165, 2015.

[6] Dr Akhilesh A Mirza Samiulla Beg. A comprehensive study of artificial bee colony (abc) algorithms and its applications. 2019.